IN PARTNERSHIP WITH:
Université Rennes 1

Activity Report 2017

Project-Team PACAP

Pushing Architecture and Compilation for Application Performance

IN COLLABORATION WITH: Institut de recherche en informatique et systèmes aléatoires (IRISA)

RESEARCH CENTER
Rennes - Bretagne-Atlantique

THEME
Architecture, Languages and Compilation

Table of contents

Project-Team PACAP

Creation of the Project-Team: 2016 July 01

Keywords:

Computer Science and Digital Science:

A1.1. - Architectures
A1.1.1. - Multicore, Manycore
A1.1.2. - Hardware accelerators (GPGPU, FPGA, etc.)
A1.1.3. - Memory models
A1.1.4. - High performance computing
A1.1.5. - Exascale
A1.1.9. - Fault tolerant systems
A1.1.10. - Reconfigurable architectures
A1.1.11. - Quantum architectures
A1.6. - Green Computing
A2.2. - Compilation
A2.2.1. - Static analysis
A2.2.2. - Memory models
A2.2.3. - Run-time systems
A2.2.4. - Parallel architectures
A2.2.5. - GPGPU, FPGA, etc.
A2.2.6. - Adaptive compilation
A2.3.1. - Embedded systems
A2.3.3. - Real-time systems
A4.2. - Correcting codes
A4.4. - Security of equipment and software
A8.9. - Performance evaluation
A8.10. - Computer arithmetic

Other Research Topics and Application Domains:

B1. - Life sciences
B2. - Health
B3. - Environment and planet
B4. - Energy
B5. - Industry of the future
B6. - IT and telecom
B7. - Transport and logistics
B8. - Smart Cities and Territories
B9. - Society and Knowledge

1. Personnel

Research Scientists

Erven Rohou [Team leader, Inria, Senior Researcher, HDR]
Sylvain Collange [Inria, Researcher]
Pierre Michaud [Inria, Researcher]
André Seznec [Inria, Senior Researcher, HDR]

Faculty Members
Damien Hardy [Univ de Rennes I, Associate Professor]
Isabelle Puaut [Univ de Rennes I, Professor, HDR]

Post-Doctoral Fellows
Fernando Endo [Inria, until May 2017]
Imen Fassi [Univ de Rennes I, from Oct 2017]
Byron Hawkins [Inria, from Sep 2017]
Biswabandan Panda [Inria, until May 2017]
Stefanos Skalistis [Univ de Rennes I, from Dec 2017]

PhD Students
Arif Ali Ana-Pparakkal [Inria]
Rabab Bouziane [Inria]
Niloofar Charmchi [Inria]
Kleovoulos Kalaitzidis [Inria]
Viet Anh Nguyen [Univ de Rennes I]
Daniel Rodrigues Carvalho [Inria, from Oct 2017]
Benjamin Rouxel [Univ de Rennes I]

Technical staff
Loïc Besnard [CNRS SED, Senior Research Engineer, seconded at 40 %]
Nicolas Kiss [Inria]
Imane Lasri [Inria]
Sébastien Martinez [Univ de Rennes I, until Jun 2017]
Kévin Le Bon [Inria, from Sep 2017]

Intern
Anita Tino [Inria, until Apr 2017]

Administrative Assistant
Virginie Desroches [Inria]

Visiting Scientists
Stefano Cherubin [Politecnico di Milano, from Mar 2017 until Apr 2017]
Andrei Rimsa Alvares [Centro Federal de Educação Tecnológica de Minas Gerais, until Feb 2017]
Marcos Siraichi [Universidade Federal de Minas Gerais, from Dec 2017]

2. Overall Objectives

2.1. Overall Objectives

2.1.1. Long-Term Goal

In brief, the long-term goal of the PACAP project-team is about *performance*, that is: how fast programs run. We intend to contribute to the ongoing race for exponentially increasing performance and for performance guarantees.

Traditionally, the term "performance" is understood as "how much time is needed to complete execution". *Latency*-oriented techniques focus on minimizing the average-case execution time (ACET). We are also interested in other definitions of performance. *Throughput*-oriented techniques are concerned with how many units of computations can be completed per unit of time. This is more relevant on manycores and GPUs where many computing nodes are available, and latency is less critical. Finally, we also study worst-case execution time (WCET), which is extremely important for critical real-time systems where designers must guarantee that deadlines are met, in any situation.

Given the complexity of current systems, simply assessing their performance has become a non-trivial task which we also plan to tackle.

We occasionally consider other metrics related to performance, such as power efficiency, total energy, overall complexity, and real-time response guarantee. Our ultimate goal is to propose solutions that make computing systems more efficient, taking into account current and envisioned applications, compilers, runtimes, operating systems, and microarchitectures. And since increased performance often comes at the expense of another metric, identifying the related trade-offs is of interest to PACAP.

The previous decade witnessed the end of the "magically" increasing clock frequency and the introduction of commodity multicore processors. PACAP will likely experience the end of Moore's law [1], and the generalization of commodity heterogeneous manycore processors. This impacts how performance is increased and how it can be guaranteed. It is also a time where exogenous parameters should be promoted to first-class citizens:

1. the existence of faults, whose impact is becoming increasingly important when the photo-lithography feature size decreases;
2. the need for security at all levels of computing systems;
3. *green* computing, or the growing concern of power consumption.

2.1.2. Approach

We strive to address performance in a way as transparent as possible for users. For example, instead of proposing any new language, we consider existing applications (written for example in standard C), and we develop compiler optimizations that immediately benefit programmers; we propose microarchitectural features as opposed to changes in processor instruction sets; we analyze and re-optimize binary programs automatically, without any user intervention.

The perimeter of research directions proposed for the PACAP project-team derive from the intersection of two axes: on the one hand, our high-level research objectives, derived from the overall panorama of computing systems, on the other hand the existing expertise and background of the team members on key technology (see illustration on Figure 1). Note that it does not imply that we will systematically explore all intersecting points of the figure, yet all correspond to a sensible research direction. These lists are neither exhaustive, nor final. Operating systems in particular constitute a promising operating point for several of the issues we plan to tackle. Other aspects will likely emerge during the lifespan of the project-team.

2.1.3. Latency-oriented Computing

Improving the ACET of general purpose systems has been the "core business" of PACAP's ancestors (CAPS and ALF) for two decades. We plan to pursue this line of research, acting at all levels: compilation, dynamic optimizations, and microarchitecture.

2.1.4. Throughput-Oriented Computing

The goal is to maximize the performance-to-power ratio. We will leverage the execution model of throughput-oriented architectures (such as GPUs) and extend it towards general purpose systems. To address the memory wall issue, we will consider bandwidth saving techniques, such as cache and memory compression.

[1] Moore's law states that the number of transistors in a circuit doubles (approximately) every two years.

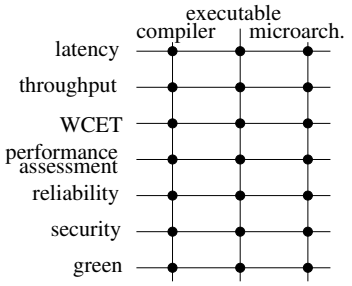

Figure 1. Perimeter of Research Objectives

2.1.5. Real-Time Systems – WCET

Designers of real-time systems must provide an upper bound of the worst-case execution time of the tasks within their systems. By definition this bound must be safe (i.e. greater than any possible execution time). To be useful, WCET estimates have to be as tight as possible. The process of obtaining a WCET bound consists in analyzing a binary executable, modeling the hardware, and then maximizing an objective function that takes into account all possible flows of execution and their respective execution times. Our research will consider the following directions:

1. better modeling of hardware to either improve tightness, or handle more complex hardware (e.g. multicores);
2. eliminate unfeasible paths from the analysis;
3. consider probabilistic approaches where WCET estimates are provided with a confidence level.

2.1.6. Performance Assessment

Moore's law drives the complexity of processor micro-architectures, which impacts all other layers: hypervisors, operating systems, compilers and applications follow similar trends. While a small category of experts is able to comprehend (parts of) the behavior of the system, the vast majority of users are only exposed to – and interested in – the bottom line: how fast their applications are actually running. In the presence of virtual machines and cloud computing, multi-programmed workload add yet another degree of non-determinism to the measure of performance. We plan to research how application performance can be characterized and presented to a final user: behavior of the microarchitecture, relevant metrics, possibly visual rendering. Targeting our own community, we also research techniques appropriate for fast and accurate ways to simulate future architectures, including heterogeneous designs, such as latency/throughput platforms.

Once diagnosed, the way bottlenecks are addressed depends on the level of expertise of users. Experts can typically be left with a diagnostic as they probably know better how to fix the issue. Less knowledgeable users must be guided to a better solution. We plan to rely on iterative compilation to generate multiple versions of critical code regions, to be used in various runtime conditions. To avoid the code bloat resulting from multiversioning, we will leverage split-compilation to embed code generation "recipes" to be applied just-in-time, or even at rutime thanks to dynamic binary translation. Finally, we will explore the applicability of auto-tuning, where programmers expose which parameters of their code can be modified to generate alternate versions of the program (for example trading energy consumption for quality of service) and let a global orchestrator make decisions.

2.1.7. Dealing with Faults – Reliability

Semiconductor technology evolution suggests that permanent failure rates will increase dramatically with scaling. While well-known approaches, such as error correcting codes, exist to recover from failures and

provide fault-free chips, the exponential growth of the number of faults will make them unaffordable in the future. Consequently, other approaches like fine-grained disabling and reconfiguration of hardware elements (e.g. individual functional units or cache blocks) will become economically necessary. This fine-grained disabling will degrade performance compared to a fault-free execution. This evolution impacts performance (both ACET and WCET). We plan to address this evolution, and propose new techniques, which can be developed at any level. For example, at microarchitecture level, one might consider designing part of a cache in an older technology to guarantee a minimum level of performance; at compile-time, one might generate redundant code for critical sections; at run-time, one can monitor faults and apply corrective measures to the software, or hardware. Solutions involving multiple levels are also very promising.

2.1.8. Dealing with Attacks – Security

Computer systems are under constant attack, from young hackers trying to show their skills, to "professional" criminals stealing credit card information, and even government agencies with virtually unlimited resources. A vast amount of techniques have been proposed in the literature to circumvent attacks. Many of them cause significant slowdowns due to additional checks and countermeasures. Thanks to our expertise in microarchitecture and compilation techniques, we will be able to significantly improve efficiency, robustness and coverage of security mechanism, as well as to partner with field experts to design innovative solutions.

2.1.9. Green Computing

Power consumption has become a major concern of computing systems, at all form factors, ranging from energy-scavenging sensors for IoT, to battery powered embedded systems and laptops, and up to supercomputers operating in the tens of megawatts. Execution time and energy are often related optimization goals. Optimizing for performance under a given power cap, however, introduces new challenges. It also turns out that technologists introduce new solutions (e.g. magnetic RAM) which, in turn, result in new trade-offs and optimization opportunities.

3. Research Program

3.1. Motivation

Our research program is naturally driven by the evolution of our ecosystem. Relevant recent changes can be classified in the following categories: technological constraints, evolving community, and domain constraints. We hereby summarize these evolutions.

3.1.1. Technological constraints

Until recently, binary compatibility guaranteed portability of programs, while increased clock frequency and improved micro-architecture provided increased performance. However, in the last decade, advances in technology and micro-architecture started translating into more parallelism instead. Technology roadmaps even predict the feasibility of thousands of cores on a chip by 2020. Hundreds are already commercially available. Since the vast majority of applications are still sequential, or contain significant sequential sections, such a trend put an end to the automatic performance improvement enjoyed by developers and users. Many research groups consequently focused on parallel architectures and compiling for parallelism.

Still, the performance of applications will ultimately be driven by the performance of the sequential part. Despite a number of advances (some of them contributed by members of the team), sequential tasks are still a major performance bottleneck. Addressing it is still on the agenda of the proposed PACAP project-team.

In addition, due to power constraints, only part of the billions of transistors of a microprocessor can be operated at any given time (the *dark silicon* paradigm). A sensible approach consists in specializing parts of the silicon area to provide dedicated accelerators (not run simultaneously). This results in diverse and heterogeneous processor cores. Application and compiler designers are now confronted with a moving target, challenging portability and jeopardizing performance.

Finally, we live in a world where billions of sensors, actuators, and computers play a crucial role in our life: flight control, nuclear plant management, defense systems, banking, or health care. These systems must be reliable, despite the fact that they are subject to faults (for example due to aging, charged particle hit, or random noise). Faults will soon become the new *de facto* standard. The evolutions of the semiconductor industry predict an exponential growth of the number of permanent faults [56]. Reliability considerations usually degrade performance. We will propose solutions to mitigate this impact (for example by limiting overheads to critical sections).

Note on technology.
Technology also progresses at a fast pace. We do not propose to pursue any research on technology *per se*. Recently proposed paradigms (non-Si, brain-inspired) have received lots of attention from the research community. We do *not* intend to invest in those paradigms, but we will continue to investigate compilation and architecture for more conventional programming paradigms. Still, several technological shifts may have consequences for us, and we will closely monitor their developments, they include for example non-volatile memory (impacts security, makes writes longer than loads), 3D-stacking (impacts bandwidth), and photonics (impacts latencies and connection network).

3.1.2. Evolving community

The PACAP project-team tackles performance-related issues, for conventional programming paradigms. In fact, programming complex environments is no longer the exclusive domain of experts in compilation and architecture. A large community now develops applications for a wide range of targets, including mobile "apps", cloud, multicore or heterogeneous processors.

This also includes domain scientists (in biology, medicine, but also social sciences) who started relying heavily on computational resources, gathering huge amounts of data, and requiring considerable amount of processing to analyze them. Our research is motivated by the growing discrepancy between on the one hand the complexity of the workloads and the computing systems, and on the other hand the expanding community of developers at large, with limited expertise to optimize and to map efficiently computations to compute nodes.

3.1.3. Domain constraints

Mobile, embedded systems have become ubiquitous. Many of them have real-time constraints. For this class of systems, correctness implies not only producing the correct result, but also doing so within specified deadlines. In the presence of heterogeneous, complex and highly dynamic systems, producing *tight* (i.e. useful) upper bound to the worst-case execution time has become extremely challenging. Our research will aim at improving the tightness as well as enlarging the set of features that can be safely analyzed.

The ever growing dependence of our economy on computing systems also implies that security has become of utmost importance. Many systems are under constant attacks from intruders. Protection has a cost also in terms of performance. We plan to leverage our background to contribute solutions that minimize this impact.

Note on Applications Domains.
PACAP works on fundamental technologies for computer science: processor architecture, performance-oriented compilation and guaranteed response time for real-time. The research results may have impacts on any application domain that requires high performance execution (telecommunication, multimedia, biology, health, engineering, environment...), but also on many embedded applications that exhibit other constraints such as power consumption, code size and guaranteed response time.

We strive to extract from active domains the fundamental characteristics that are relevant to our research. For example, *big data* is of interest to PACAP because it relates to the study of hardware/software mechanisms to efficiently transfer huge amounts of data to the computing nodes. Similarly, the *Internet of Things* is of interest because it has implications in terms of ultra low power consumption.

3.2. Research Objectives

Processor micro-architecture and compilation have been at the core of the research carried by the members of the project teams for two decades, with undeniable contributions. They continue to be the foundation of PACAP.

Heterogeneity and diversity of processor architectures now require new techniques to guarantee that the hardware is satisfactorily exploited by the software. One of our goals is to devise new static compilation techniques (cf. Section 3.2.1), but also build upon iterative [1] and split [2] compilation to continuously adapt software to its environment (Section 3.2.2). Dynamic binary optimization will also play a key role in delivering adapting software and delivering performance.

The end of Moore's law and Dennard's scaling [2] offer an exciting window of opportunity, where performance improvements will no longer derive from additional transistor budget or increased clock frequency, but rather come from breakthroughs in microarchitecture (Section 3.2.3). Reconciling CPU and GPU designs (Section 3.2.4) is one of our objectives.

Heterogeneity and multicores are also major obstacles to determining tight worst-case execution times of real-time systems (Section 3.2.5), which we plan to tackle.

Finally, we also describe how we plan to address transversal aspects such as reliability (Section 3.2.6), power efficiency (Section 3.2.7), and security (Section 3.2.8).

3.2.1. Static Compilation

Static compilation techniques continues to be relevant to address the characteristics of emerging hardware technologies, such as non-volatile memories, 3D-stacking, or novel communication technologies. These techniques expose new characteristics to the software layers. As an example, non-volatile memories typically have asymmetric read-write latencies (writes are much longer than reads) and different power consumption profiles. PACAP studies the new optimization opportunities and develop tailored compilation techniques for the upcoming compute nodes. New technologies may also be coupled with traditional solutions to offer new trade-offs. We study how programs can adequately exploit the specific features of the proposed heterogeneous compute nodes.

We propose to build upon iterative compilation [1] to explore how applications perform on different configurations. When possible, Pareto points are related to application characteristics. The best configuration, however, may actually depend on runtime information, such as input data, dynamic events, or properties that are available only at runtime. Unfortunately a runtime system has little time and means to determine the best configuration. For these reasons, we also leverage split-compilation [2]: the idea consists in pre-computing alternatives, and embedding in the program enough information to assist and drive a runtime system towards to the best solution.

3.2.2. Software Adaptation

More than ever, software needs to adapt to its environment. In most cases, this environment remains unknown until runtime. This is already the case when one deploys an application to a cloud, or an "app" to mobile devices. The dilemma is the following: for maximum portability, developers should target the most general device; but for performance they would like to exploit the most recent and advanced hardware features. JIT compilers can handle the situation to some extent, but binary deployment requires dynamic binary rewriting. Our work has shown how SIMD instructions can be upgraded from SSE to AVX [3]. Many more opportunities will appear with diverse and heterogeneous processors, featuring various kinds of accelerators.

On shared hardware, the environment is also defined by other applications competing for the same computational resources. It becomes increasingly important to adapt to changing runtime conditions, such as the contention of the cache memories, available bandwidth, or hardware faults. Fortunately, optimizing at runtime is also an opportunity, because this is the first time the program is visible as a whole: executable and libraries

[2] According to Dennard scaling, as transistors get smaller the power density remains constant, and the consumed power remains proportional to the area.

(including library versions). Optimizers may also rely on dynamic information, such as actual input data, parameter values, etc. We have already developed a software platform [16] to analyze and optimize programs at runtime, and we started working on automatic dynamic parallelization of sequential code, and dynamic specialization.

We started addressing some of these challenges in ongoing projects such as Nano2017 PSAIC Collaborative research program with STMicroelectronics, as well as within the Inria Project Lab MULTICORE. The starting H2020 FET HPC project ANTAREX also addresses these challenges from the energy perspective. We further leverage our platform and initial results to address other adaptation opportunities. Efficient software adaptation requires expertise from all domains tackled by PACAP, and strong interaction between all team members is expected.

3.2.3. Research directions in uniprocessor microarchitecture

Achieving high single-thread performance remains a major challenge even in the multicore era (Amdahl's law). The members of the PACAP project-team have been conducting research in uniprocessor microarchitecture research for about 20 years covering major topics including caches, instruction front-end, branch prediction, out-of-order core pipeline, and value prediction. In particular, in the recent years they have been recognized as world leaders in branch prediction [22][11] and in cache prefetching [9] and they have revived the forgotten concept of value prediction [14][13]. This research was supported by the ERC Advanced grant DAL (2011-2016) and also by Intel. We pursue research on achieving ultimate unicore performance. Below are several non-orthogonal directions that we have identified for mid-term research:

1. management of the memory hierarchy (particularly the hardware prefetching);
2. practical design of very wide issue execution core;
3. speculative execution.

Memory design issues:
Performance of many applications is highly impacted by the memory hierarchy behavior. The interactions between the different components in the memory hierarchy and the out-of-order execution engine have high impact on performance.

The last *Data Prefetching Contest* held with ISCA 2015 has illustrated that achieving high prefetching efficiency is still a challenge for wide-issue superscalar processors, particularly those featuring a very large instruction window. The large instruction window enables an implicit data prefetcher. The interaction between this implicit hardware prefetcher and the explicit hardware prefetcher is still relatively mysterious as illustrated by Pierre Michaud's BO prefetcher (winner of DPC2) [9]. The first objective of the research is to better understand how the implicit prefetching enabled by the large instruction window interacts with the L2 prefetcher and then to understand how explicit prefetching on the L1 also interacts with the L2 prefetcher.

The second objective of the research is related to the interaction of prefetching and virtual/physical memory. On real hardware, prefetching is stopped by page frontiers. The interaction between TLB prefetching (and on which level) and cache prefetching must be analyzed.

The prefetcher is not the only actor in the hierarchy that must be carefully controlled. Significant benefit can also be achieved through careful management of memory access bandwidth, particularly the management of spatial locality on memory accesses, both for reads and writes. The exploitation of this locality is traditionally handled in the memory controller. However, it could be better handled if larger temporal granularity was available. Finally, we also intend to continue to explore the promising avenue of compressed caches. In particular we recently proposed the skewed compressed cache [17]. It offers new possibility for efficient compression schemes.

Ultra wide-issue superscalar.
To effectively leverage memory level parallelism, one requires huge out-of-order execution structures as well as very wide issue superscalar processor. For the two past decades, implementing ever wider issue superscalar processor has been challenging. The objective of our research on the execution core is to explore (and revisit) directions to allow the design of a very wide-issue (8-to-16 way) out-of-order execution core while mastering its complexity (silicon area, hardware logic complexity, power/energy consumption).

The first direction that we are exploring is the use of clustered architecture [10]. Symmetric clustered organization allows to benefit from simpler bypass network, but induce large complexity on the issue queue. One remarkable finding of our study [10] is that, when considering two large clusters (e.g. 8-wide) steering large groups of consecutive instructions (e.g. 64 μops) to the same cluster is quite efficient. This opens opportunities to limit the complexity of the issue queues (monitoring fewer buses) and register files (fewer ports and physical registers) in the clusters, since not all results have to be forwarded to the other cluster.

The second direction that we are exploring is associated with the approach that we developed with Sembrant et al. [18]. It reduces the number of instructions waiting in the instruction queues for the applications benefiting from very large instruction windows. Instructions are dynamically classified as ready (independent from any long latency instruction) or non-ready, and as urgent (part of a dependency chain leading to a long latency instruction) or non-urgent. Non-ready non-urgent instructions can be delayed until the long latency instruction has been executed; this allows to reduce the pressure on the issue queue. This proposition opens the opportunity to consider an asymmetric microarchitecture with a cluster dedicated to the execution of urgent instructions and a second cluster executing the non-urgent instructions. The microarchitecture of this second cluster could be optimized to reduce complexity and power consumption (smaller instruction queue, less aggressive scheduling...)

Speculative execution.
Out-of-order (OoO) execution relies on speculative execution that requires predictions of all sorts: branch, memory dependency, value...

The PACAP members have been major actors of the branch prediction research for the last 20 years; and their proposals have influenced the design of most of the hardware branch predictors in current microprocessors. We will continue to steadily explore new branch predictor designs as for instance [20].

In speculative execution, we have recently revisited value prediction (VP) which was a hot research topic between 1996 and 2002. However it was considered until recently that value prediction would lead to a huge increase in complexity and power consumption in every stage of the pipeline. Fortunately, we have recently shown that complexity usually introduced by value prediction in the OoO engine can be overcome [14][13][22][11]. First, very high accuracy can be enforced at reasonable cost in coverage and minimal complexity [14]. Thus, both prediction validation and recovery by squashing can be done outside the out-of-order engine, at commit time. Furthermore, we propose a new pipeline organization, EOLE ({Early | Out-of-order | Late} Execution), that leverages VP with validation at commit to execute many instructions outside the OoO core, in-order [13]. With EOLE, the issue-width in OoO core can be reduced without sacrificing performance, thus benefiting the performance of VP without a significant cost in silicon area and/or energy. In the near future, we will explore new avenues related to value prediction. These directions include register equality prediction and compatibility of value prediction with weak memory models in multiprocessors.

3.2.4. Towards heterogeneous single-ISA CPU-GPU architectures

Heterogeneous single-ISA architectures have been proposed in the literature during the 2000's [55] and are now widely used in the industry (ARM big.LITTLE, NVIDIA 4+1...) as a way to improve power-efficiency in mobile processors. These architectures include multiple cores whose respective microarchitectures offer different trade-offs between performance and energy efficiency, or between latency and throughput, while offering the same interface to software. Dynamic task migration policies leverage the heterogeneity of the platform by using the most suitable core for each application, or even each phase of processing. However, these works only tune cores by changing their complexity. Energy-optimized cores are either identical cores implemented in a low-power process technology, or simplified in-order superscalar cores, which are far from state-of-the-art throughput-oriented architectures such as GPUs.

We investigate the convergence of CPU and GPU at both architecture and compilation levels.

Architecture.

The architecture convergence between Single Instruction Multiple Threads (SIMT) GPUs and multicore processors that we have been pursuing [7] opens the way for heterogeneous architectures including latency-optimized superscalar cores and throughput-optimized GPU-style cores, which all share the same instruction set. Using SIMT cores in place of superscalar cores will enable the highest energy efficiency on regular sections of applications. As with existing single-ISA heterogeneous architectures, task migration will not necessitate any software rewrite and will accelerate existing applications.

Compilers for emerging heterogeneous architectures.
Single-ISA CPU+GPU architectures will provide the necessary substrate to enable efficient heterogeneous processing. However, it will also introduce substantial challenges at the software and firmware level. Task placement and migration will require advanced policies that leverage both static information at compile time and dynamic information at run-time. We are tackling the heterogeneous task scheduling problem at the compiler level. As a first step, we are prototyping scheduling algorithms on existing multiple-ISA CPU+GPU architectures like NVIDIA Tegra X1.

3.2.5. Real-time systems

Safety-critical systems (e.g. avionics, medical devices, automotive...) have so far used simple unicore hardware systems as a way to control their predictability, in order to meet timing constraints. Still, many critical embedded systems have increasing demand in computing power, and simple unicore processors are not sufficient anymore. General-purpose multicore processors are not suitable for safety-critical real-time systems, because they include complex micro-architectural elements (cache hierarchies, branch, stride and value predictors) meant to improve average-case performance, and for which worst-case performance is difficult to predict. The prerequisite for calculating tight WCET is a deterministic hardware system that avoids dynamic, time-unpredictable calculations at run-time.

Even for multi and manycore systems designed with time-predictability in mind (Kalray MPPA manycore architecture [3], or the Recore manycore hardware [4]) calculating WCETs is still challenging. The following two challenges will be addressed in the mid-term:

1. definition of methods to estimate WCETs tightly on manycores, that smartly analyzes and/or controls shared resources such as buses, NoCs or caches;
2. methods to improve the programmability of real-time applications through automatic parallelization and optimizations from model-based designs.

3.2.6. Fault Tolerance

Technology trends suggest that, in tomorrow's computing world, failures will become commonplace due to many factors, and the expected probability of failure will increase with scaling. While well-known approaches, such as error correcting codes, exist to recover from failures and provide fault-free chips, the exponential growth of the number of faults will make them unaffordable in the future. Consequently, other approaches such as fine-grained disabling and reconfiguration of hardware elements (e.g. individual functional units or cache blocks) will become economically necessary. We are going to enter a new era: functionally correct chips with variable performance among chips and throughout their lifetime [56].

Transient and permanent faults may be detected by similar techniques, but correcting them generally involves different approaches. We are primarily interested in permanent faults, even though we do not necessarily disregard transient faults (e.g. the TMR approach in the next paragraph addresses both kind of faults).

CPU.
Permanent faults can occur anywhere in the processor. The performance implications of faulty cells vary depending on how the array is used in a processor. Most of micro-architectural work aiming at assessing the performance implications of permanently faulty cells relies on simulations with random fault-maps. These studies are, therefore, limited by the fault-maps they use that may not be representative for the average and distributed performance. They also do not consider aging effect.

[3] http://www.kalrayinc.com
[4] http://www.recoresystems.com/

Considering the memory hierarchy, we have already studied [5] the impact of permanent faults on the average and worst-case performance based on analytical models. We will extend these models to cover other components and other designs, and to analyze the interaction between faulty components.

For identified critical hardware structures, such as the memory hierarchy, we will propose protection mechanisms by for instance using larger cells, or even by selecting a different array organization to mitigate the impact of faults.

Another approach to deal with faults is to introduce redundancy at the code level. We propose to consider static compilation techniques focusing on existing hardware. As an example, we plan to leverage SIMD extensions of current instruction sets to introduce redundancy in scalar code at minimum cost. With these instructions, it will be possible to protect the execution from both soft errors by using TMR (triple modular redundancy) with voters in the code itself, and permanent faults without the need of extra hardware support to deconfigure faulty functional units.

Reconfigurable Computing.
In collaboration with the CAIRN project-team, we propose to construct Coarse Grain Reconfigurable Architectures (CGRA) from a sea of basic arithmetic and memory elements organized into clusters and connected through a hierarchical interconnection network. These clusters of basic arithmetic operators (e.g. 8-bit arithmetic and logic units) would be able to be seamlessly configured to various accuracy and data types to adapt the consumed energy to application requirements taking advantage of approximate computations. We propose to add new kinds of error detection (and sometimes correction) directly at the operator level by taking advantage of the massive redundancy of the array. As an example, errors can be tracked and detected in a complex sequence of double floating-point operations by using a reduced-precision version of the same processing.

Such reconfigurable blocks will be driven by compilation techniques, in charge of computing checkpoints, detecting faults, and replaying computations when needed.

Dynamic compilation techniques will help better exploit faulty hardware, by allocating data and computations on correct resources. In case of permanent faults, we will provide a mechanism to reconfigure the hardware, for example by reducing the issue width of VLIW processors implemented in CGRA. Dynamic code generation (JIT compiler) will re-generate code for the new configuration, guaranteeing portability and optimal exploitation of the hardware.

3.2.7. Power efficiency

PACAP addresses power-efficiency at several levels. First, we design static and split compilation techniques to contribute to the race for Exascale computing (the general goal is to reach 10^{18} FLOP/s at less than 20 MW). Second, we focus on high-performance low-power embedded compute nodes. Within the ANR project Continuum, in collaboration with architecture and technology experts from LIRMM and the SME Cortus, we research new static and dynamic compilation techniques that fully exploit emerging memory and NoC technologies. Finally, in collaboration with the CAIRN project-team, we investigate the synergy of reconfigurable computing and dynamic code generation.

Green and heterogeneous high-performance computing.
Concerning HPC systems, our approach consists in mapping, runtime managing and autotuning applications for green and heterogeneous High-Performance Computing systems up to the Exascale level. One key innovation of the proposed approach consists of introducing a separation of concerns (where self-adaptivity and energy efficient strategies are specified aside to application functionalities) promoted by the definition of a Domain Specific Language (DSL) inspired by aspect-oriented programming concepts for heterogeneous systems. The new DSL will be introduced for expressing adaptivity/energy/performance strategies and to enforce at runtime application autotuning and resource and power management. The goal is to support the parallelism, scalability and adaptability of a dynamic workload by exploiting the full system capabilities (including energy management) for emerging large-scale and extreme-scale systems, while reducing the Total Cost of Ownership (TCO) for companies and public organizations.

High-performance low-power embedded compute nodes.

We will address the design of next generation energy-efficient high-performance embedded compute nodes. It focuses at the same time on software, architecture and emerging memory and communication technologies in order to synergistically exploit their corresponding features. The approach of the project is organized around three complementary topics: 1) compilation techniques; 2) multicore architectures; 3) emerging memory and communication technologies. PACAP will focus on the compilation aspects, taking as input the software-visible characteristics of the proposed emerging technology, and making the best possible use of the new features (non-volatility, density, endurance, low-power).

Hardware Accelerated JIT Compilation.
Reconfigurable hardware offers the opportunity to limit power consumption by dynamically adjusting the number of available resources to the requirements of the running software. In particular, VLIW processors can adjust the number of available issue lanes. Unfortunately, changing the processor width often requires recompiling the application, and VLIW processors are highly dependent of the quality of the compilation, mainly because of the instruction scheduling phase performed by the compiler. Another challenge lies in the high constraints of the embedded system: the energy and execution time overhead due to the JIT compilation must be carefully kept under control.

We started exploring ways to reduce the cost of JIT compilation targeting VLIW-based heterogeneous many-core systems. Our approach relies on a hardware/software JIT compiler framework. While basic optimizations and JIT management are performed in software, the compilation back-end is implemented by means of specialized hardware. This back-end involves both instruction scheduling and register allocation, which are known to be the most time-consuming stages of such a compiler.

3.2.8. Security

Security is a mandatory concern of any modern computing system. Various threat models have led to a multitude of protection solutions. Members of PACAP already contributed, thanks to the HAVEGE [59] random number generator, and code obfuscating techniques (the obfuscating just-in-time compiler [54], or thread-based control flow mangling [58]).

We partner with security experts who can provide intuition, know-how and expertise, in particular in defining threat models, and assessing the quality of the solutions. Our background in compilation and architecture helps design more efficient and less expensive protection mechanisms.

We already have ongoing research directions related to security. SECODE (Secure Codes to Thwart Cyber-physical Attacks) is a project started January 2016, in collaboration with security experts from Télécom Paris Tech, Paris 8, Université Catholique de Louvain (Belgium), and University of Sabancı (Turkey). We also plan to partner with the Inria/CentraleSupelec CIDRE project-team to design a tainting technique based on a just-in-time compiler.

Compiler-based data protection.
We specify and design error correction codes suitable for an efficient protection of sensitive information in the context of Internet of Things (IoT) and connected objects. We partner with experts in security and codes to prototype a platform that demonstrates resilient software. PACAP's expertise is key to select and tune the protection mechanisms developed within the project, and to propose safe, yet cost-effective solutions from an implementation point of view.

JIT-based tainting.
Dynamic information flow control (DIFC, also known as *tainting*) is used to detect intrusions and to identify vulnerabilities. It consists in attaching metadata (called *taints* or *labels*) to information containers, and to propagate the taints when particular operations are applied to the containers: reads, writes, etc. The goal is then to guarantee that confidential information is never used to generate data sent to an untrusted container; conversely, data produced by untrusted entities cannot be used to update sensitive data.

The containers can be of various granularities: fine-grain approaches can deal with single variables, coarser-grain approaches consider a file as a whole. The CIDRE project-team has developed several DIFC monitors. kBlare is coarse-grain monitor in the Linux kernel. JBlare is a fine-grain monitor for Java applications. Fine-grain monitors provide a better precision at the cost of a significant overhead in execution time.

Combining the expertise of CIDRE in DIFC with our expertise in JIT compilation willl help design hybrid approaches. An initial static analysis of the program prior to installation or execution will feed information to a dynamic analyzer that propagates taints during just-in-time compilation.

4. Application Domains

4.1. Any computer usage

The PACAP team is working on the fundamental technologies for computer science: processor architecture, performance-oriented compilation and guaranteed response time for real-time. The research results may have impacts on any application domain that requires high performance execution (telecommunication, multimedia, biology, health, engineering, environment...), but also on many embedded applications that exhibit other constraints such as power consumption, code size and guaranteed response time. Our research activity implies the development of software prototypes.

5. New Software and Platforms

5.1. ATMI

KEYWORDS: Analytic model - Chip design - Temperature
SCIENTIFIC DESCRIPTION: Research on temperature-aware computer architecture requires a chip temperature model. General purpose models based on classical numerical methods like finite differences or finite elements are not appropriate for such research, because they are generally too slow for modeling the time-varying thermal behavior of a processing chip.

We have developed an ad hoc temperature model, ATMI (Analytical model of Temperature in MIcroprocessors), for studying thermal behaviors over a time scale ranging from microseconds to several minutes. ATMI is based on an explicit solution to the heat equation and on the principle of superposition. ATMI can model any power density map that can be described as a superposition of rectangle sources, which is appropriate for modeling the microarchitectural units of a microprocessor.
FUNCTIONAL DESCRIPTION: ATMI is a library for modelling steady-state and time-varying temperature in microprocessors. ATMI uses a simplified representation of microprocessor packaging.

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

5.2. HEPTANE

KEYWORDS: IPET - WCET - Performance - Real time - Static analysis - Worst Case Execution Time
SCIENTIFIC DESCRIPTION: WCET estimation

Status: Registered with APP (Agence de Protection des Programmes). Available under GNU General Public License v3, with number IDDN.FR.001.510039.000.S.P.2003.000.10600.

The aim of Heptane is to produce upper bounds of the execution times of applications. It is targeted at applications with hard real-time requirements (automotive, railway, aerospace domains). Heptane computes WCETs using static analysis at the binary code level. It includes static analyses of microarchitectural elements such as caches and cache hierarchies.

For more information, please contact Damien Hardy or Isabelle Puaut.
FUNCTIONAL DESCRIPTION: In a hard real-time system, it is essential to comply with timing constraints, and Worst Case Execution Time (WCET) in particular. Timing analysis is performed at two levels: analysis of the WCET for each task in isolation taking account of the hardware architecture, and schedulability analysis of all the tasks in the system. Heptane is a static WCET analyser designed to address the first issue.

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

5.3. tiptop

KEYWORDS: Instructions - Cycles - Cache - CPU - Performance - HPC - Branch predictor
SCIENTIFIC DESCRIPTION: Tiptop is written in C. It can take advantage of libncurses when available for pseudo-graphic display.

Performance, hardware counters, analysis tool.

Status: Registered with APP (Agence de Protection des Programmes). Available under GNU General Public License v2, with number IDDN.FR.001.450006.000.S.P.2011.000.10800. Current version is 2.3.1, released October 2017.

Tiptop has been integrated in major Linux distributions, such as Fedora, Debian, Ubuntu.

Tiptop is a new simple and flexible user-level tool that collects hardware counter data on Linux platforms (version 2.6.31+). The goal is to make the collection of performance and bottleneck data as simple as possible, including simple installation and usage. In particular, we stress the following points.

Installation is only a matter of compiling the source code. No patching of the Linux kernel is needed, and no special-purpose module needs to be loaded.

No privilege is required, any user can run tiptop
FUNCTIONAL DESCRIPTION: Today's microprocessors have become extremely complex. To better understand the multitude of internal events, manufacturers have integrated many monitoring counters. Tiptop can be used to collect and display the values from these performance counters very easily. Tiptop may be of interest to anyone who wants to optimise the performance of their HPC applications.

- Participant: Erven Rohou
- Contact: Erven Rohou
- URL: http://tiptop.gforge.inria.fr

5.4. PADRONE

KEYWORDS: Legacy code - Optimization - Performance analysis - Dynamic Optimization
FUNCTIONAL DESCRIPTION: Padrone is new platform for dynamic binary analysis and optimization. It provides an API to help clients design and develop analysis and optimization tools for binary executables. Padrone attaches to running applications, only needing the executable binary in memory. No source code or debug information is needed. No application restart is needed either. This is especially interesting for legacy or commercial applications, but also in the context of cloud deployment, where actual hardware is unknown, and other applications competing for hardware resources can vary. The profiling overhead is minimum.

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

5.5. If-memo

KEYWORD: Performance
SCIENTIFIC DESCRIPTION: We propose a linker based technique for enabling software memorizing of any dynamically linked pure function by function interception and we illustrate our framework using a set of computationally expensive pure functions - the transcendental functions. Our technique does not need the availability of source code and thus can even be applied to commercial applications as well as applications with legacy codes. As far as users are concerned, enabling memoization is as simple as setting an environment variable. Our framework does not make any specific assumptions about the underlying architecture or compiler too-chains, and can work with a variety of current architectures.

- Participants: Arjun Suresh and Erven Rohou
- Contact: Erven Rohou
- URL: https://team.inria.fr/alf/software/if-memo/

5.6. Simty

KEYWORDS: RISC-V - Multi-threading - SIMT - FPGA - Softcore - GPU
FUNCTIONAL DESCRIPTION: Simty is a massively multi-threaded processor core that dynamically assembles SIMD instructions from scalar multi-thread code. It runs the RISC-V (RV32-I) instruction set. Unlike existing SIMD or SIMT processors like GPUs, Simty takes binaries compiled for general-purpose processors without any instruction set extension or compiler changes. Simty is described in synthesizable VHDL.

- Author: Sylvain Collange
- Contact: Sylvain Collange
- URL: https://gforge.inria.fr/projects/simty

5.7. Barra

KEYWORDS: Performance - Computer architecture - Debug - Tesla ISA - GPU - Profiling - CUDA - HPC - Simulator - GPGPU
SCIENTIFIC DESCRIPTION: Research on throughout-oriented architectures demands accurate and representative models of GPU architectures in order to be able to evaluate new architectural ideas, explore design spaces and characterize applications. The Barra project is a simulator of the NVIDIA Tesla GPU architecture.

Barra builds upon knowledge acquired through micro-benchmarking, in order to provide a baseline model representative of industry practice. The simulator provides detailed statistics to identify optimization opportunities and is fully customizable to experiment ideas of architectural modifications. Barra incorporates both a functional model and a cycle-level performance model.
FUNCTIONAL DESCRIPTION: Barra is a Graphics Processing Unit (GPU) architecture simulator. It simulates NVIDIA CUDA programs at the assembly language level. Barra is a tool for research on computer architecture, and can also be used to debug, profile and optimize CUDA programs at the lowest level.

RELEASE FUNCTIONAL DESCRIPTION: Timing model Tesla-like architecture model Fermi-like architecture model New per-PC control-flow divergence management Simultaneous branch and warp interweaving Affine vector cache

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

6. New Results

6.1. Compiler, vectorization, interpretation

Participants: Erven Rohou, André Seznec, Sylvain Collange, Rabab Bouziane, Arif Ali Ana-Pparakkal, Stefano Cherubin, Byron Hawkins, Arif Ali Ana-Pparakkal, Imane Lasri, Kévin Le Bon.

6.1.1. Improving sequential performance through memoization

Participants: Erven Rohou, Imane Lasri, André Seznec.

Many applications perform repetitive computations, even when properly programmed and optimized. Performance can be improved by caching results of pure functions, and retrieving them instead of recomputing a result (a technique called memoization).

We previously proposed [23] a simple technique for enabling software memoization of any dynamically linked pure function and we illustrate our framework using a set of computationally expensive pure functions – the transcendental functions.

A restriction of the proposed framework was that memoization was restricted only to dynamically linked functions and the functions must be determined beforehand. We extended this work, and we propose function memoization using a compile-time technique thus extending the scope of memoization to user defined functions as well as making it transparently applicable to any dynamically linked functions. Our compile-time technique allows static linking of memoization code and this increases the benefit due to memoization by leveraging the inlining capability for the memoization wrapper. Our compile-time analysis can also handle functions with pointer parameters , and we handle constants more efficiently. Instruction set support can also be considered, and we propose associated hardware leading to additional performance gain.

This work was presented at the Compiler Construction Conference 2017 [50]. It is also described in the PhD thesis of Arjun Suresh [24].

6.1.2. Optimization in the Presence of NVRAM

Participants: Erven Rohou, Rabab Bouziane.

Beyond the fact of generating machine code, compilers play a critical role in delivering high performance, and more recently high energy efficiency. For decades, the memory technology of target systems has consisted in SRAM at cache level, and DRAM for main memory. Emerging non-volatile memories (NVMs) open up new opportunities, along with new design challenges. In particular, the asymmetric cost of read/write accesses calls for adjusting existing techniques in order to efficiently exploit NVMs. In addition, this technology makes it possible to design memories with cheaper accesses at the cost of lower data retention times. These features can be exploited at compile time to derive better data mappings according to the application and data retention characteristics. We reviewed a number of compile-time analysis and optimization techniques, and how they could apply to systems in presence of NVMs [37]. In particular, we consider the case of the reduction of the number of writes, and the analysis of variables lifetime for memory bank assignment of program variables.

Concerning the reduction of writes, we propose a fast evaluation of NVM integration at cache level, together with a compile-time approach for mitigating the penalty incurred by the high write latency of STT-RAM. We implement a code optimization in LLVM for reducing so-called *silent stores*, i.e., store instruction instances that write to memory values that were already present there. This makes our optimization portable over any architecture supporting LLVM. Then, we assess the possible benefit of such an optimization on the Rodinia benchmark suite through an analytic approach based on parameters extracted from the literature devoted to NVMs. This makes it possible to rapidly analyze the impact of NVMs on memory energy consumption. Reported results show up to 42 % energy gain when considering STT-RAM caches. This work is accepted for publication at RAPIDO'18 [38].

This research is done in collaboration with Abdoulaye Gamatié at LIRMM (Montpellier) within the context the the ANR project CONTINUUM.

6.1.3. Dynamic Binary Optimization

Participants: Erven Rohou, Arif Ali Ana-Pparakkal, Kévin Le Bon, Byron Hawkins.

6.1.3.1. Dynamic Function Specialization

Participants: Erven Rohou, Arif Ali Ana-Pparakkal, Kévin Le Bon.

Compilers can do better optimization with the knowledge of run-time behavior of the program. *Function specialization* is a compilation technique that consists in optimizing the body of a function for specific values of an argument. Different versions of a function are created to deal with the most frequent values of the arguments, as well as the default case. Compilers can do a better optimization with the knowledge of run-time behaviour of the program. Static compilers, however, can hardly predict the exact value/behaviour of arguments, and even profiling collected during previous runs is never guaranteed to capture future behaviour. We propose a dynamic function specialization technique, that captures the actual values of arguments during execution of the program and, when profitable, creates specialized versions and include them at runtime. Our approach relies on dynamic binary rewriting. We present [36] the principles and implementation details of our technique, analyze sources of overhead, and present our results.

This research is done within the context of the Nano 2017 PSAIC collaborative project.

6.1.3.2. Runtime Vectorization of Binary Programs

Participant: Erven Rohou.

In many cases, applications are not optimized for the hardware on which they run. Several reasons contribute to this unsatisfying situation, such as legacy code, commercial code distributed in binary form, or deployment on compute farms. In fact, backward compatibility of ISA guarantees only the functionality, not the best exploitation of the hardware. In this work, we focus on maximizing the CPU efficiency for the SIMD extensions.

We previously proposed [3] a binary-to-binary optimization framework where loops vectorized for an older version of the processor SIMD extension are automatically converted to a newer one. It is a lightweight mechanism that does not include a vectorizer, but instead leverages what a static vectorizer previously did. We showed that many loops compiled for x86 SSE can be dynamically converted to the more recent and more powerful AVX; as well as, how correctness is maintained with regards to challenges such as data dependencies and reductions. We obtained speedups in line with those of a native compiler targeting AVX.

We now focus on runtime vectorization of loops in binary codes that were not originally vectorized [29]. For this purpose, we use open source frameworks that we have tuned and integrated to

1. dynamically lift the x86 binary into the Intermediate Representation form of the LLVM compiler,
2. abstract hot loops in the polyhedral model,
3. use the power of this mathematical framework to vectorize them,
4. and finally compile them back into executable form using the LLVM Just-In-Time compiler.

In most cases, the obtained speedups are close to the number of elements that can be simultaneously processed by the SIMD unit. The re-vectorizer and auto-vectorizer are implemented inside a dynamic optimization platform; it is completely transparent to the user, does not require any rewriting of the binaries, and operates during program execution.

This work is done in collaboration with Philippe Clauss (Inria CAMUS), it is part of the PhD work of Nabil Hallou [26].

6.1.4. Hardware/Software JIT Compiler

Participant: Erven Rohou.

Dynamic Binary Translation (DBT) is often used in hardware/software co-design to take advantage of an architecture model while using binaries from another one. The co-development of the DBT engine and of the execution architecture leads to architecture with special support to these mechanisms. We proposed [46] a hardware accelerated dynamic binary translation where the first steps of the DBT process are fully accelerated in hardware. Results showed that using our hardware accelerators leads to a speed-up of 8× and a cost in energy 18× lower, compared with an equivalent software approach.

Single ISA-Heterogeneous multi-cores such as the ARM big.LITTLE have proven to be an attractive solution to explore different energy/performance trade-offs. Such architectures combine Out of Order cores with smaller in-order ones to offer different power/energy profiles. They however do not really exploit the characteristics of workloads (compute-intensive vs. control dominated). In our recent work, we propose to enrich these architectures with runtime configurable VLIW cores, which are very efficient at compute-intensive kernels. To preserve the single ISA programming model, we resort to Dynamic Binary Translation, and use this technique to enable dynamic code specialization for Runtime Reconfigurable VLIWs cores. Our proposed DBT framework targets the RISC-V ISA, for which both OoO and in-order implementations exist. Our experimental results show that our approach can lead to best-case performance and energy efficiency when compared against static VLIW configurations.

This work has been accepted for publication at DATE 2018 [53].

This research is done in collaboration with Steven Derrien and Simon Rokicki from the CAIRN team.

6.1.5. Customized Precision Computing

Participants: Erven Rohou, Stefano Cherubin, Imane Lasri.

Error-tolerating applications are increasingly common in the emerging field of real-time HPC. Proposals have been made at the hardware level to take advantage of inherent perceptual limitations, redundant data, or reduced precision input, as well as to reduce system costs or improve power efficiency. At the same time, works on floating-point to fixed-point conversion tools allow us to trade-off the algorithm exactness for a more efficient implementation. In this work [39], we aim at leveraging existing, HPC-oriented hardware architectures, while including in the precision tuning an adaptive selection of floating-and fixed-point arithmetic. Our proposed solution takes advantage of the application domain knowledge of the programmers by involving them in the first step of the interaction chain. We rely on annotations written by the programmer on the input file to know which variables of a computational kernel should be converted to fixed-point. The second stage replaces the floating-point variables in the kernel with fixed-point equivalents. It also adds to the original source code the utility functions to perform data type conversions from floating-point to fixed-point, and vice versa. The output of the second stage is a new version of the kernel source code which exploits fixed-point computation instead of floating-point computation. As opposed to typical custom-width hardware designs, we only rely on the standard 16-bit, 32-bit and 64-bit types. We also explore the impact of the fixed-point representation on auto-vectorization. We discuss the effect of our solution in terms of time-to-solutions, error and energy-to-solution.

This is done within the context of the ANTAREX project in collaboration with Stefano Cherubin, and Giovanni Agosta from Politecnico di Milano, and Olivier Sentieys from the CAIRN team.

6.1.6. SPMD Function Call Re-Vectorization

Participant: Sylvain Collange.

SPMD programming languages for SIMD hardware such as C for CUDA, OpenCL or ISPC have contributed to increase the programmability of SIMD accelerators and graphics processing units. However, SPMD languages still lack the flexibility offered by low-level SIMD programming on explicit vectors. To close this expressiveness gap while preserving the SPMD abstraction, we introduce the notion of Function Call Re-Vectorization (CREV). CREV allows changing the dimension of vectorization during the execution of an SPMD kernel, and exposes it as a nested parallel kernel call. CREV affords a programmability close to dynamic parallelism, a feature that allows the invocation of kernels from inside kernels, but at much lower cost. We defined a formal semantics of CREV, and implemented it on the ISPC compiler. To validate our

idea, we have used CREV to implement some classic algorithms, including string matching, depth first search and Bellman-Ford, with minimum effort. These algorithms, once compiled by ISPC to Intel-based vector instructions, are as fast as state-of-the-art implementations, yet much simpler. As an example, our straightforward implementation of string matching beats the Knuth-Morris-Pratt algorithm by 12%. This work was presented at the ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP) 2017 [45].

This work was done in collaboration with Rubens Emilio and Fernando Pereira at UFMG, as part of the Inria PROSPIEL Associate Team.

6.1.7. Qubit allocation for quantum circuit compilers

Participant: Sylvain Collange.

Quantum computing hardware is becoming a reality. For instance, IBM Research makes a quantum processor available in the cloud to the general public. The possibility of programming an actual quantum device has elicited much enthusiasm. Yet, quantum programming still lacks the compiler support that modern programming languages enjoy today. To use universal quantum computers like IBM's, programmers must design low-level circuits. In particular, they must map logical qubits into physical qubits that need to obey connectivity constraints. This task resembles the early days of programming, in which software was built in machine languages. We have formally introduced the qubit allocation problem and provided an exact solution to it. This optimal algorithm deals with the simple quantum machinery available today; however, it cannot scale up to the more complex architectures scheduled to appear. Thus, we also provide a heuristic solution to qubit allocation, which is faster than the current solutions already implemented to deal with this problem.

This paper is accepted for publication at the Code Generation and Optimization (CGO) conference [49].

This work was done in collaboration with Vinícius Fernandes dos Santos, Fernando Pereira and Marcos Yukio Siraichi at UFMG, Brazil.

6.2. Processor Architecture

Participants: Pierre Michaud, Sylvain Collange, Erven Rohou, André Seznec, Biswabandan Panda, Fernando Endo, Kleovoulos Kalaitzidis, Daniel Rodrigues Carvalho, Anita Tino.

Processor, cache, locality, memory hierarchy, branch prediction, multicore, power, temperature

6.2.1. Microarchitecture

6.2.1.1. Bayesian TAGE predictors

Participant: Pierre Michaud.

The TAGE conditional branch predictor, introduced by André Seznec and Pierre Michaud in 2006, is the most storage-efficient branch predictor known today [19]. André Seznec has won the last four branch prediction championships, each time with a TAGE-based predictor. However, since 2006, the improvements in prediction accuracy have been relatively modest and were mostly obtained at the cost of increased hardware complexity. In particular, André Seznec added a Statistical Corrector to TAGE to correct some of its deficiencies [21]. This may be an indication that our understanding of TAGE is not complete and that further accuracy gains are waiting to be discovered. The problem tackled by the statistical corrector is that of cold counters: a TAGE-like predictor constantly allocate new entries, erasing the branch history information stored in the up-down counters of the overwritten entries. TAGE mitigates this problem by using the confidence level of the up-down counter and a meta-predictor. However, fundamentally, the information on the degree of coldness of the up-down counter is not available in TAGE. Therefore we propose to replace the up-down counter with a dual-counter counting separately taken and not-taken occurrences. Replacing the up-down counter with a dual-counter requires to redefine prediction confidence estimation. We found that a Bayesian formula, namely Laplace's rule of succession, provides effective confidence estimation. We also discovered a method, based on the dual-counter, for reducing the number of allocations. By combining these new findings, we devised a new TAGE-like predictor called BATAGE, more accurate than TAGE, making external statistical correction superfluous. As of December 2017, this work is in the process of being submitted to a journal.

6.2.1.2. Interactions Between Value Prediction and Compiler Optimizations
Participants: André Seznec, Fernando Endo.

Increasing instruction-level parallelism is regaining attractiveness within the microprocessor industry. The EOLE microarchitecture [13] and D-VTAGE value predictor [14] were recently introduced to solve practical issues of value prediction (VP). In particular, they remove the most significant difficulties that forbade an effective VP hardware. In [28], we present a detailed evaluation of the potential of VP in the context of EOLE/D-VTAGE and different compiler options. Our study shows that if no single general rule always applies – more optimization might sometimes leads to more performance – unoptimized codes often gets a large benefit from the prediction of redundant loads.

6.2.1.3. Prefetch Management on Multicore Systems
Participants: André Seznec, Biswabandan Panda.

In multi-core systems, an application's prefetcher can interfere with the memory requests of other applications using the shared resources, such as last level cache and memory bandwidth. Towards this end, we propose a solution to manage prefetching in multi-core systems [32]. In particular, we make two fundamental observations: First, a strong positive correlation exists between the accuracy of a prefetcher and the amount of prefetch requests it generates relative to an application's total (demand and prefetch) requests. Second, a strong positive correlation exists between the ratio of total prefetch to demand requests and the ratio of average last level cache miss service times of demand to prefetch requests. In [32], we propose Band-pass prefetching a simple and low-overhead mechanism to effectively manage prefetchers in multi-core systems that builds on those two observations. Our solution consists of local and global prefetcher aggressiveness control components, which altogether, control the flow of prefetch requests between a range of prefetch to demand requests ratios.

6.2.1.4. Managing Shared Last Level Caches in Large Multicores
Participant: André Seznec.

Multi-core processors employ shared Last Level Caches (LLC). This trend continuewith large multi-core processors (16 cores and beyond) as well. At the same time, the associativity of LLC tends to remain in the order of sixteen. Consequently, with large multicore processors, the number of applications or threads that share the LLC becomes larger than the associativity of the cache itself. LLC management policies have been extensively studied for small scale multi-cores (4 to 8 cores) and associativity degree in the 16 range. However, the impact of LLC management on large multi-cores is essentially unknown, in particular when the associativity degree is smaller than the number of applications. In [33], we introduce Adaptive Discrete and deprioritized Application PrioriTization (ADAPT), an LLC management policy addressing the large multi-cores where the LLC associativity degree is smaller than the number of applications. ADAPT builds on the use of the Footprint-number metric. We propose a monitoring mechanism that dynamically samples cache sets to estimate the Footprint-number of applications and classifies them into discrete (distinct and more than two) priority buckets. The cache replacement policy leverages this classification and assigns priorities to cache lines of applications during cache replacement operations. We further find that deprioritizing certain applications during cache replacement is beneficial to the overall performance.

6.2.1.5. Augmenting superscalar architecture for efficient many-thread parallel execution
Participants: Sylvain Collange, André Seznec.

Threads of Single-Program Multiple-Data (SPMD) applications often exhibit very similar control flows, i.e. they execute the same instructions on different data. We propose the Dynamic Inter-Thread Vectorization Architecture (DITVA) to leverage this implicit data-level parallelism in SPMD applications by assembling dynamic vector instructions at runtime. DITVA extends an in-order SMT processor with SIMD units with an inter-thread vectorization execution mode. In this mode, multiple scalar threads running in lockstep share a single instruction stream and their respective instruction instances are aggregated into SIMD instructions. To balance thread-and data-level parallelism, threads are statically grouped into fixed-size independently scheduled warps. DITVA leverages existing SIMD units and maintains binary compatibility with existing CPU

architectures. Our evaluation on the SPMD applications from the PARSEC and Rodinia OpenMP benchmarks shows that a 4-warp × 4-lane 4-issue DITVA architecture with a realistic bank-interleaved cache achieves 1.55× higher performance than a 4-thread 4-issue SMT architecture with AVX instructions while fetching and issuing 51 % fewer instructions, achieving an overall 24 % energy reduction.

This work has been accepted for publication in the Journal of Parallel and Distributed Computing [30]. It was done in collaboaration with Sajith Kalathingal and Bharath Swamy from Intel Bangalore (India).

6.2.1.6. Generalizing the SIMT execution model to general-purpose instruction sets

Participant: Sylvain Collange.

The *Single Instruction, Multiple Threads* (SIMT) execution model as implemented in NVIDIA Graphics Processing Units (GPUs) associates a multi-thread programming model with an SIMD execution model [57]. It combines the simplicity of scalar code from the programmer's and compiler's perspective with the efficiency of SIMD execution units at the hardware level. However, current SIMT architectures demand specific instruction sets. In particular, they need specific branch instructions to manage thread divergence and convergence. Thus, SIMT GPUs have remained incompatible with traditional general-purpose CPU instruction sets.

We designed Simty, an SIMT processor proof of concept that lifts the instruction set incompatibility between CPUs and GPUs. Simty is a massively multi-threaded processor core that dynamically assembles SIMD instructions from scalar multi-thread code. It runs the RISC-V (RV32-I) instruction set. Unlike existing SIMD or SIMT processors like GPUs, Simty takes binaries compiled for general-purpose processors without any instruction set extension or compiler changes. Simty is described in synthesizable RTL. A FPGA prototype validates its scaling up to 2048 threads per core with 32-wide SIMD units.

The Simty architecture was presented at the First Workshop on Computer Architecture Research with RISC-V (CARRV 2017) [40].

Both conventional and generalized SIMT architectures like Simty use hardware or software mechanisms to keep track of control-flow divergence and convergence among threads. A new class of such algorithms is gaining popularity in the literature in the last few years. We presented a new classification of these techniques based on their common characteristic, namely traversals of the control-flow graph based on lists of paths. We compared the implementation cost on an FPGA of path lists and per-thread program counters within the Simty processor. The sorted list enables significantly better scaling starting from 8 threads per warp.

This work was presented in French in Conférence d'informatique en Parallélisme, Architecture et Système (ComPAS) [51] and is available in English as a technical report [52].

6.2.1.7. Toward out-of-order SIMT microarchitecture

Participants: Sylvain Collange, Anita Tino.

Prior work highlights the continued importance of maintaining adequate sequential performance within throughput-oriented cores [60]. Out-of-order superscalar architectures as used in high-performance CPU cores can meet such demand for single-thread performance. However, GPU architectures based on SIMT have been limited so far to in-order execution because of a major scientific obstacle: the partial dependencies between instructions that SIMT execution induces thwart register renaming. This ongoing project is seeking to generalize out-of-order execution to SIMT architectures. In particular, we revisit register renaming techniques originally proposed for predicate conversion to support partial register updates efficiently. Out-of-order dynamic vectorization holds the promise to close the CPU-GPU design space by enabling low-latency, high-throughput design points.

6.3. WCET estimation and optimization

Participants: Isabelle Puaut, Damien Hardy, Viet Anh Nguyen, Benjamin Rouxel, Sébastien Martinez, Erven Rohou, Imen Fassi, Loïc Besnard, Stefanos Skalistis.

6.3.1. WCET estimation for many core processors

Participants: Viet Anh Nguyen, Damien Hardy, Sébastien Martinez, Isabelle Puaut, Benjamin Rouxel.

6.3.1.1. Optimization of WCETs by considering the effects of local caches

The overall goal of this research is to define WCET estimation methods for parallel applications running on many-core architectures, such as the Kalray MPPA machine.

Some approaches to reach this goal have been proposed, but they assume the mapping of parallel applications on cores already done. Unfortunately, on architectures with caches, task mapping requires a priori known WCETs for tasks, which in turn requires knowing task mapping (i.e., co-located tasks, co-running tasks) to have tight WCET bounds. Therefore, scheduling parallel applications and estimating their WCET introduce a chicken and egg situation.

We have addressed this issue by developing both optimal and heuristic techniques for solving the scheduling problem, whose objective is to minimize the WCET of a parallel application. Our proposed static partitioned non-preemptive mapping strategies address the effect of local caches to tighten the estimated WCET of the parallel application. Experimental results obtained on real and synthetic parallel applications show that co-locating tasks that reuse code and data improves the WCET by 11 % on average for the optimal method and by 9 % on average for the heuristic method [35].

This research is part of the PIA Capacités project.

6.3.1.2. Accounting for shared resource contentions to minimize WCETs

Accurate WCET analysis for multi-cores is known to be challenging, because of concurrent accesses to shared resources, such as communication through busses or Networks on Chips (NoC). Since it is impossible in general to guarantee the absence of resource conflicts during execution, current WCET techniques either produce pessimistic WCET estimates or constrain the execution to enforce the absence of conflicts, at the price of a significant hardware under-utilization. In addition, the large majority of existing works consider that the platform workload consists of independent tasks. As parallel programming is the most promising solution to improve performance, we envision that within only a few years from now, real-time workloads will evolve toward parallel programs. The WCET behavior of such programs is challenging to analyze because they consist of *dependent* tasks interacting through complex synchronization/communication mechanisms.

In a first work (thesis of Benjamin Rouxel), we proposed techniques that account for interferences to access shared ressources, in order to minimize the WCET of parallel applications. An optimal and a heuristic method are proposed to map and schedule tasks on multi-cores. These methods take the structure of applications (synchronizations/communications) into consideration to tightly identify shared resource interferences and consequently tighten WCET estimates. Our heuristic improves by 19% the overall WCET compared to a worst-case contention baseline [47], [31].

In a second study [44], we have studied the gain that could be obtained on an initially produced time-triggered non-preemptive schedule, by the introduction of slack time, in order to avoid interference between tasks. The introduction of slack time is performed using an optimal technique using Integer Linear Programming (ILP), to evaluate how much at best can be gained. Experimental results using synthetic task graphs and a Kalray-like architecture with round-robin bus arbitration show that avoiding contention reduces WCETs, albeit by a small percentage. The highest reductions are observed on applications with the highest memory demand, and when the application is scheduled on the highest number of cores.

This work is performed in cooperation with Steven Derrien from the CAIRN research group and is part of the ARGO H2020 project.

6.3.1.3. WCET-Aware Parallelization of Model-Based Applications for Multi-Cores

Parallel architectures are nowadays no longer confined to the domain of high performance computing, they are also increasingly used in embedded time-critical systems.

The ongoing ARGO H2020 project provides a programming paradigm and associated tool flow to exploit the full potential of architectures in terms of development productivity, time-to-market, exploitation of the platform computing power and guaranteed real-time performance. In [41] we give an overview of the objectives of ARGO and explore the challenges introduced by our approach.

6.3.2. WCET estimation tool and benchmarks

Participants: Damien Hardy, Isabelle Puaut, Benjamin Rouxel, Loïc Besnard.

Estimation of worst-case execution times (WCETs) is required to validate the temporal behavior of hard real time systems. Heptane is an open-source software program that estimates upper bounds of execution times on MIPS and ARM v7 architectures, offered to the WCET estimation community to experiment new WCET estimation techniques. The software architecture of Heptane was designed to be as modular and extensible as possible to facilitate the integration of new approaches. In [42], we present the current status of Heptane, give information on the analyses it implements, as well as how to use it and extend it.

We all had quite a time to find non-proprietary architecture-independent exploitable parallel benchmarks for Worst-Case Execution Time (WCET) estimation and real-time scheduling. However, there is no consensus on a parallel benchmark suite, when compared to the single-core era and the Mälardalen benchmark suite. In [48] we bridge part of this gap, by presenting a collection of benchmarks with the following good properties: (i) easily analyzable by static WCET estimation tools (written in structured C language, in particular neither goto nor dynamic memory allocation, containing flow information such as loop bounds); (ii) independent from any particular run-time system (MPI, OpenMP) or real-time operating system. Each benchmark is composed of the C source code of its tasks, and an XML description describing the structure of the application (tasks and amount of data exchanged between them when applicable). Each benchmark can be integrated in a full end-to-end empirical method validation protocol on multi-core architecture. This proposed collection of benchmarks is derived from the well known StreamIT benchmark suite and will be integrated in the TACleBench suite in a near future.

6.4. Security

Participants: Erven Rohou, Damien Hardy, Nicolas Kiss.

Physical attacks represent a very important threat in the context of embedded systems: these attacks try to recover cryptographic keys by exploiting the physical behavior of the device. They can either be passive (e.g. by monitoring the power consumption of the device) or active (e.g. by injecting errors to reveal or deduce sensitive data).

One family of countermeasures to protect against those passive attacks (also known as *side-channel* attacks) is called masking. The principle is to “hide” data with masks so that internal values used in computations can not be predicted with the behavior observed. We modified the LLVM compiler (version 3.8) to automatically insert masking coutermeasures into the code at compile-time. Our modification works at intermediate level (IR level), this way we can perform low-level transformations (e.g. memory allocation, instructions replacement) while covering most of the architectures used in the embedded world.

The main innovation of this work is the generic approach used for the transformation and thus, the ability to easily change the masking scheme without modifying the compiler internal code. We introduced a way to describe in high-level language (C/C++) the masking operations independently in what we call “primitives”. With this technique, we implemented “Boolean Masking” and we tested the efficiency on an embedded implementation of AES. After measuring the electromagnetic emissions of 20,000 executions, we performed a Correlation Power Analysis (CPA) and results have shown that the countermeasure is correctly applied. Hence, it is not possible anymore to recover the cryptographic key with this type of attack.

This work is done in the context of the SECODE CHIST-ERA project.

7. Bilateral Contracts and Grants with Industry

7.1. Bilateral Contracts with Industry

7.1.1. Nano 2017 PSAIC

Participants: Arif Ali Ana-Pparakkal, Erven Rohou.

Nano 2017 PSAIC is a collaborative R&D program involving Inria and STMicroelectronics. The PSAIC (Performance and Size Auto-tuning through Iterative Compilation) project concerns the automation of program optimization through the combination of several tools and techniques such as: compiler optimization, profiling, trace analysis, iterative optimization and binary analysis/rewriting. For any given application, the objective is to devise through a fully automated process a compiler profile optimized for performance and code size. For this purpose, we are developing instrumentation techniques that can be focused and specialized to a specific part of the application aimed to be monitored.

The project involves the Inria teams PACAP, AriC, CAMUS and CORSE. PACAP contributes program analyses at the binary level, as well as binary transformations. We will also study the synergy between static (compiler-level) and dynamic (run-time) analyses.

7.2. Bilateral Grants with Industry

7.2.1. Intel research grant INTEL2014-8957

Participants: André Seznec, Biswabandan Panda, Fernando Endo.

Intel is supporting the research of the PACAP project-team on “Mixing branch and value prediction to enable high sequential performance”.

7.2.2. Intel research grant INTEL2016-11174

Participants: André Seznec, Pierre Michaud, Kleovoulos Kalaitzidis, Niloofar Charmchi.

Intel is supporting the research of the PACAP project-team on “Design tradeoffs for extreme cores”.

8. Partnerships and Cooperations

8.1. Regional Initiatives

8.1.1. Britanny region fellowship

Participants: Niloofar Charmchi, André Seznec.

The Britanny Region is partially funding a Ph.D. fellowship for Niloofar Charmchi on the topic “Hardware prefetching and related issues”.

8.2. National Initiatives

8.2.1. Capacités: Projet “Investissement d’Avenir”, 1/11/14 to 31/01/2018

Participants: Damien Hardy, Isabelle Puaut, Viet Anh Nguyen, Sébastien Martinez.

The project objective is to develop a hardware and software platform based on manycore architectures, and to demonstrate the relevance of these manycore architectures (and more specifically the Kalray manycore) for several industrial applications. The Kalray MPPA manycore architecture is currently the only one able to meet the needs of embedded systems simultaneously requiring high performance, lower power consumption, and the ability to meet the requirements of critical systems (low latency I/O, deterministic processing times, and dependability).

The project partners are Kalray (lead), Airbus, Open-Wide, Safran Sagem, IS2T, Real Time at Work, Dassault Aviation, Eurocopter, MBDA, ProbaYes, IRIT, Onera, Verimag, Inria, Irisa, Tima and Armines.

8.2.2. Zero Power Computing Systems (ZEP): Inria Project Lab, 2017–2020

Participant: Erven Rohou.

This proposal addresses the issue of designing tiny wireless, batteryless, computing objects, harvesting energy in the environment. The energy level harvested being very low, very frequent energy shortages are expected. In order for the new system to maintain a consistent state, it will be based on a new architecture embedding non-volatile RAM (NVRAM). In order to benefit from the hardware innovations related to energy harvesting and NVRAM, software mechanisms will be designed. On the one hand, a compilation pass will compute a worst-case energy consumption. On the other hand, dedicated runtime mechanisms will allow:

1. to manage efficiently and correctly the NVRAM-based hardware architecture;
2. to use energy intelligently, by using the worst-case energy consumption.

The ZEP project gathers four Inria teams that have a scientific background in architecture, compilation, operating systems together with the CEA Lialp and Lisan laboratories of CEA LETI & LIST. The main application target is Internet of Things (IoT).

8.2.3. ANR Continuum 2015–2019

Participants: Erven Rohou, Rabab Bouziane.

The CONTINUUM project aims to address the energy-efficiency challenge in future computing systems by investigating a design continuum for compute nodes, which seamlessly goes from software to technology levels via hardware architecture. Power saving opportunities exist at each of these levels, but the real measurable gains will come from the synergistic focus on all these levels as considered in this project. Then, a cross-disciplinary collaboration is promoted between computer science and microelectronics, to achieve two main breakthroughs: i) combination of state-of-the-art heterogeneous adaptive embedded multicore architectures with emerging communication and memory technologies and, ii) power-aware dynamic compilation techniques that suitably match such a platform.

Continuum started on Oct 1st 2015. Partners are LIRMM and Cortus SAS.

8.2.4. ANR W-SEPT 2012-2017

Participants: Isabelle Puaut, Erven Rohou.

Critical embedded systems are generally composed of repetitive tasks that must meet drastic timing constraints, such as termination deadlines. Providing an upper bound of the worst-case execution time (WCET) of such tasks at design time is thus necessary to prove the correctness of the system. Static WCET estimation methods, although safe, may produce largely over-estimated values. The objective of the project is to produce tighter WCET estimates by discovering and transforming flow information at all levels of the software design process, from high level-design models (e.g. Scade, Simulink) down to binary code.

The ANR W-SEPT project partners are Verimag Grenoble, IRIT Toulouse, Inria Rennes. A case study is provided by Continental Toulouse.

8.3. European Initiatives

8.3.1. FP7 & H2020 Projects

8.3.1.1. ANTAREX

Participants: Erven Rohou, Imane Lasri.

Title: Auto-Tuning and Adaptivity appRoach for Energy efficient exascale HPC Systems

Program: H2020

Duration: September 2015 - September 2018

Coordinator: Politecnico di Milano, Italy (POLIMI)

Partners:

Consorzio Interuniversitario Cineca (Italy)
Dompé Farmaceutici Spa (Italy)
Eidgenoessische Technische Hochschule Zürich (Switzerland)
Vysoka Skola Banska - Technicka Univerzita Ostrava (Czech Republic)
Politecnico di Milano (Italy)
Sygic As (Slovakia)
Universidade do Porto (Portugal)

Inria contact: Erven Rohou

Energy-efficient heterogeneous supercomputing architectures need to be coupled with a radically new software stack capable of exploiting the benefits offered by the heterogeneity at all the different levels (supercomputer, job, node) to meet the scalability and energy efficiency required by Exascale supercomputers. ANTAREX will solve these challenging problems by proposing a disruptive holistic approach spanning all the decision layers composing the supercomputer software stack and exploiting effectively the full system capabilities (including heterogeneity and energy management). The main goal of the ANTAREX project is to provide a breakthrough approach to express application self-adaptivity at design-time and to runtime manage and autotune applications for green and heterogenous High Performance Computing (HPC) systems up to the Exascale level.

8.3.1.2. Eurolab-4-HPC

Participant: André Seznec.

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

Program: H2020

Duration: September 2015 - September 2017

Coordinator: Chalmers Tekniska Hoegskola AB

Partners:

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

Inria contact: Albert Cohen (Inria Paris)

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 oriented communities in Europe, working across all layers of the system stack and, at the same time, fueling new industries in HPC.

8.3.1.3. ARGO

Participants: Isabelle Puaut, Damien Hardy, Imen Fassi.

Title: Argo: WCET-Aware Parallelization of Model-Based Applications for Heterogeneous Parallel Systems

Program: H2020

Type: RIA

Duration: Jan 2016 - Dec 2018

Coordinator: Karlsruher Institut für Technologie (KIT)

Université Rennes I contact: Steven Derrien

Partners:

Karlsruher Institut für Technologie (KIT)

SCILAB enterprises SAS

Recore Systems BV

Université de Rennes I

Technologiko Ekpaideftiko Idryma (TEI) Dytikis Elladas

Absint GmbH

Deutsches Zentrum für Luft - und Raumfahrt EV

Fraunhofer

Increasing performance and reducing costs, while maintaining safety levels and programmability are the key demands for embedded and cyber-physical systems in European domains, e.g. aerospace, automation, and automotive. For many applications, the necessary performance with low energy consumption can only be provided by customized computing platforms based on heterogeneous many-core architectures. However, their parallel programming with time-critical embedded applications suffers from a complex toolchain and programming process. Argo (WCET-Aware PaRallelization of Model-Based Applications for HeteroGeneOus Parallel Systems) will address this challenge with a holistic approach for programming heterogeneous multi- and many-core architectures using automatic parallelization of model-based real-time applications. Argo will enhance WCET-aware automatic parallelization by a crosslayer programming approach combining automatic tool-based and user-guided parallelization to reduce the need for expertise in programming parallel heterogeneous architectures. The Argo approach will be assessed and demonstrated by prototyping comprehensive time-critical applications from both aerospace and industrial automation domains on customized heterogeneous many-core platforms.

Argo also involves Steven Derrien and Angeliki Kritikakou from the CAIRN team.

8.3.1.4. HiPEAC4 NoE

Participants: Pierre Michaud, Erven Rohou, André Seznec.

P. Michaud, A. Seznec and E. Rohou are members of the European Network of Excellence HiPEAC4.

HiPEAC4 addresses the design and implementation of high-performance commodity computing devices in the 10+ year horizon, covering both the processor design, the optimizing compiler infrastructure, and the evaluation of upcoming applications made possible by the increased computing power of future devices.

8.4. International Initiatives

8.4.1. ANR CHIST-ERA SECODE 2016-2018

Participants: Nicolas Kiss, Damien Hardy, Erven Rohou.

In this project, we specify and design error correction codes suitable for an efficient protection of sensitive information in the context of Internet of Things (IoT) and connected objects. Such codes mitigate passive attacks, like memory disclosure, and active attacks, like stack smashing. The innovation of this project is to leverage these codes for protecting against both cyber and physical attacks. The main advantage is a full coverage of attacks of the connected embedded systems, which is considered as a smart connected device and also a physical device. The outcome of the project is first a method to generate and execute cyber-resilient software, and second to protect data and its manipulation from physical threats like side-channel attacks. Theses results are demonstrated by using a smart sensor application with hardened embedded firmware and tamper-proof hardware platform.

Partners are Télécom Paris Tech, Université Paris 8, Sabancı Üniversitesi (Turkey), and Université Catholique de Louvain (Belgium).

8.4.2. PHC IMHOTEP

Participant: Erven Rohou.

Title: Thoth – An Automatic Dynamic Binary Parallelisation System

International Partner (Institution - Laboratory - Researcher):

Egypt-Japan University of Science and Technology - Prof. Ahmed ElMahdy.

Dates: 2016–2017

With the current global trend towards utilizing cloud computing and smart devices, executing the same application across becomes a necessity. Moreover, parallelism is now abundant with various forms that include thread- and data-parallel execution models. Such diversity in ISA and explicit parallelism makes software development cost prohibitive, especially for natively optimized binaries. This project leverages dynamic binary translation technology to provide for exploiting the underlying parallel resources without the need of having the source code of the application. In particular the project integrates low overhead dynamic profiling, novel OSR parallel de-optimization and a retargetable parallelization modules to allow for dynamic parallelization of binaries.

8.4.3. Inria Associate Teams Not Involved in an Inria International Labs

8.4.3.1. PROSPIEL

Participant: Sylvain Collange.

Title: Profiling and specialization for locality

International Partner (Institution - Laboratory - Researcher):

Universidade Federal de Minas Gerais (Brazil) - DCC - Fernando Magno Quintão Pereira

Start year: 2015

See also: https://team.inria.fr/pacap/prospiel/

The PROSPIEL project aims at optimizing parallel applications for high performance on new throughput-oriented architectures: GPUs and many-core processors. Traditionally, code optimization is driven by a program analysis performed either statically at compile-time, or dynamically at run-time. Static program analysis is fully reliable but often over-conservative. Dynamic analysis provides more accurate data, but faces strong execution time constraints and does not provide any guarantee. By combining profiling-guided specialization of parallel programs with runtime checks for correctness, PROSPIEL seeks to capture the advantages of both static analysis and dynamic analysis. The project relies on the polytope model, a mathematical representation for parallel loops, as a theoretical foundation. It focuses on analyzing and optimizing performance aspects that become increasingly critical on modern parallel computer architectures: locality and regularity.

8.5. International Research Visitors

Prof. Ahmed ElMahdy, from the Egypt-Japan University of Science and Technology (E-JUST), Alexandria, Egypt, visited PACAP for two weeks in September, in the context of the project PHC IMHOTEP.

8.5.1. Visits of International Scientists

8.5.1.1. Internships

Stefano Cherubin, PhD student at Politecnico di Milano for one month in Mar 2017, within the context of the ANTAREX H2020 project.

Andrei Rimsa Alvares, PhD at UFMG and Assistant Professor at CEFET-MG, 1 month from January 6 to February 5, 2017, PROSPIEL Associate Team.

9. Dissemination

9.1. Promoting Scientific Activities

9.1.1. Scientific Events Organisation

9.1.1.1. Member of the Organizing Committees

A. Seznec is member of the ACM/IEEE ISCA symposium steering committee.

9.1.2. Scientific Events Selection

9.1.2.1. Chair of Conference Program Committees

Isabelle Puaut is Program Chair of the 2017 IEEE Real-Time Systems Symposium (RTSS), held in Paris, France. She is general Chair of RTSS 2018, to be held in Nashville, Tennessee.

9.1.2.2. Member of the Conference Program Committees

Sylvain Collange was PC member of Compas'2017.

Pierre Michaud was a member of the program committees of the ISCA 2017 and MICRO 2017 conferences.

Isabelle Puaut is member of the program committees of the Euromicro Conference on Real Time Systems (ECRTS) 2017 and 2018, the IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS) 2017 and 2018.

André Seznec is a member of IEEE Micro 2018 Top Picks selection committee.

André Seznec was a member of the IEEE Micro 2017 and SAMOS 2017 conference program committee.

9.1.2.3. Reviewer

Members of PACAP routinely review submissions to numerous international conferences and events.

9.1.3. Journal

9.1.3.1. Member of the Editorial Boards

Isabelle Puaut is Associate Editor of IEEE Transactions on Computers (IEEE TC).

André Seznec is a member of the editorial boards of IEEE Micro and ACM Transactions on Architecture and Compiler Optimization.

9.1.3.2. Reviewer - Reviewing Activities

Members of PACAP routinely review submissions to numerous international journals.

9.1.4. Invited Talks

Erven Rohou was invited to the seminar "WCET meets compilation". He presented an invited talk.

9.1.5. Leadership within the Scientific Community

Isabelle Puaut is member of the steering committee of RTNS (Real-Time Networks and Systems).

Isabelle Puaut is member of the steering committee of the Worst Case Execution Time (WCET) workshop, held in conjunction with the Euromicro Conference on Real Time Systems (ECRTS).

9.1.6. Research Administration

Isabelle Puaut is member of the Research Council (Commission Recherche) of the University of Rennes I. She is member of the working group "Habilitation à Diriger des Recherches".

Isabelle Puaut is member of the board of directors (Conseil d'Administration) of ISTIC (computer science and electrical engineering departement of University of Rennes I).

Erven Rohou is a member of the Inria CDT (Commission du Développement Technologique).

As "correspondant scientifique des relations internationales" for Inria Rennes Bretagne Atlantique, Erven Rohou is a member of the Inria COST GTRI (Groupe de Travail "Relations Internationales" du Comité d'Orientation Scientifique et Technologique).

André Seznec is an elected member of the Administration Council of Inria.

9.2. Teaching - Supervision - Juries

9.2.1. Teaching

Licence: Damien Hardy, Linux and C programming, 22 hours, L3, Université de Rennes I, France

Licence: Damien Hardy, Real-time systems, 46 hours, L3, Université de Rennes I, France

Master: Damien Hardy, Operating systems, 60 hours, M2, Université de Rennes I, France

Master: Isabelle Puaut, Operating systems: concepts and system programming under Linux (SEL), 75 hours, M1, Université Rennes I, France

Master: Isabelle Puaut, Operating systems internals (NOY), 55 hours, M1, Université Rennes I, France

Master: Isabelle Puaut, Optimizing and Parallelizing Compilers (OPC), 6 hours, M2, Université Rennes I, France

Master: Isabelle Puaut, Real-time systems, 48 hours, M1, Université Rennes I, France

Master: Isabelle Puaut, Writing of scientific publications, 20 hours, M2/PhD, Université Rennes I, France

Master: Sylvain Collange, Parallel Programming, 22 hours, M1, Université Rennes I, France

Master: Sylvain Collange, GPU programming, 32 hours, M2, ESIR, France

Master: Sylvain Collange, Advanced computer architecture, 4 hours, M2, Université Rennes I, France

Master: Sylvain Collange, Advanced CUDA programming, 8 hours, M2, Université Pierre et Marie Curie Paris 6, France

Master: André Seznec, Advanced Architectures, 8 hours, M2, Université de Rennes I, France

9.2.2. Supervision

PhD: Nabil Hallou, *Runtime Optimization of Binary Through Vectorization Transformations*, Université Rennes I, Dec 2017, co-advisors E. Rohou and P. Clauss (EPI Camus Inria Strasbourg)

PhD: Andrea Mondelli, *Revisiting Wide Superscalar Microarchitecture*, Université Rennes I, Sep 2017, co-advisors P. Michaud and A. Seznec

PhD in progress: Rabab Bouziane, *Compilation techniques to exploit novel low-power architecture and technology solutions*, Université Rennes I, started Nov 2015, advisor E. Rohou and Abdoulaye Gamatié (LIRMM, Montpellier)

PhD in progress, Viet Anh Nguyen, *Worst-Case Execution Time (WCET) Estimation for Many-core Architectures*, Université Rennes I, started Jan 2015, co-advisors I. Puaut and D. Hardy.

PhD in progress, Benjamin Rouxel, *Code optimizations for WCET calculation on many-core platforms*, started Oct 2015, co-advisors I. Puaut (70 %) and S. Derrien (30 %) from the CAIRN group.

PhD in progress: Arif Ali Ana-Pparakkal, *Dynamic Function Specialization*, Université Rennes I, started Feb 2015, advisor E. Rohou

PhD in progress: Simon Rokicki, *Compilation dynamique hybride logiciel/matériel*, Université Rennes I, started Sep 2015, co-advisors E. Rohou and Steven Derrien (CAIRN)

PhD in progress: Kalitzidis Kleovoulos, *Ultrawide Issue Superscalar Processors*, Université Rennes I, started Dec 2016, advisor A. Seznec

PhD in progress: Niloofar Charmchi, *Hardware prefetching and related issues*, Université Rennes I, started Jan 2017, advisor A. Seznec

PhD in progress: Daniel Rodriguez Carvalho, *Towards a compressed memory hierarchy*, Université Rennes I, started Oct 2017, advisor A. Seznec

9.2.3. Juries

Isabelle Puaut was a member of the following committees:

- Antoine Blin, Vers une utilisation efficace des processeurs multi-cœurs dans les systèmes embarqués à criticités multiples, Université Pierre et Marie Curie (UPMC), January 2017 (reviewer)
- Laure Abdallah, Worst-case delay analysis of core-to-IO flows over many-cores architectures, Université de Toulouse, April 2017 (reviewer)
- Stefanos Skalistis, Efficient adaptive hard real-time multi-processor systems, École Polytechnique Fédérale de Lausanne (EPFL), October 2017 (reviewer)
- Fabrice Guet, Étude de l'application de la théorie des valeurs extrêmes pour l'estimation fiable et robuste du pire temps d'exécution probabiliste, Université de Toulouse, December 2017 (examiner)
- Nabil Hallou, Runtime optimization of binary through vectorization transformations, Université de Rennes I, December 2017 (examiner)
- Quentin Perret, Predictable execution on many-core processors, Université de Toulouse, April 2017 (examiner)
- Soukayna Msirdi, Modular Avionics Software Integration on Multi-Core COTS, Université de Toulouse, July 2017 (reviewer)

Erven Rohou was a member of the following committees:

- Nabil Hallou, Runtime optimization of binary through vectorization transformations, Université de Rennes I, December 2017.
- Laurent Georget, Suivi de flux d'information correct pour les systèmes d'exploitation Linux, Université de Rennes I, September 2017
- Mohammed Boussaa, Automatic Non-Functional Testing and Tuning of Configurable Generators, Université de Rennes I, September 2017.
- Thierno Barry, Sécurisation à la compilation de logiciels contre les attaques en fautes, Université de Lyon, CEA Grenoble, November 2017.
- Shixiong Xu, Data Layout Oriented Compilation Techniques in Vectorization for Multi-/Many-cores, Trinity College, Dublin, Ireland, June 2017.

Erven Rohou was an external expert in the recruitment committee of Alexandra Jimborean, Uppsala Universitet, Sweden.

Erven Rohou was a member of the selection committee of Université de Paris-Est Marne-la-Vallée.

9.3. Popularization

Nicolas Kiss, Damien Hardy and Erven Rohou presented a poster at the “European Cyber Week”, organized by the “Pôle d’Excellence Cyber”.

Erven Rohou presented a poster at the Teratec Café, describing the ANTAREX H2020 project.

10. Bibliography

Major publications by the team in recent years

[1] F. BODIN, T. KISUKI, P. M. W. KNIJNENBURG, M. F. P. O’BOYLE, E. ROHOU. *Iterative Compilation in a Non-Linear Optimisation Space*, in "Workshop on Profile and Feedback-Directed Compilation (FDO-1), in conjunction with PACT ’98", October 1998

[2] A. COHEN, E. ROHOU. *Processor Virtualization and Split Compilation for Heterogeneous Multicore Embedded Systems*, in "DAC", June 2010, pp. 102–107

[3] N. HALLOU, E. ROHOU, P. CLAUSS, A. KETTERLIN. *Dynamic Re-Vectorization of Binary Code*, in "SAMOS", July 2015, https://hal.inria.fr/hal-01155207

[4] D. HARDY, I. PUAUT. *Static probabilistic Worst Case Execution Time Estimation for architectures with Faulty Instruction Caches*, in "21st International Conference on Real-Time Networks and Systems", Sophia Antipolis, France, October 2013 [*DOI :* 10.1145/2516821.2516842], https://hal.inria.fr/hal-00862604

[5] D. HARDY, I. SIDERIS, N. LADAS, Y. SAZEIDES. *The performance vulnerability of architectural and non-architectural arrays to permanent faults*, in "MICRO 45", Vancouver, Canada, December 2012, https://hal.inria.fr/hal-00747488

[6] D. HARDY, I. SIDERIS, A. SAIDI, Y. SAZEIDES. *EETCO: A tool to estimate and explore the implications of datacenter design choices on the tco and the environmental impact*, in "Workshop on Energy-efficient Computing for a Sustainable World in conjunction with the 44th Annual IEEE/ACM International Symposium on Microarchitecture (Micro-44)", 2011

[7] S. KALATHINGAL, S. COLLANGE, B. NARASIMHA SWAMY, A. SEZNEC. *Dynamic Inter-Thread Vectorization Architecture: extracting DLP from TLP*, in "International Symposium on Computer Architecture and High-Performance Computing (SBAC-PAD)", Los Angeles, United States, October 2016, https://hal.inria.fr/hal-01356202

[8] M.-K. LEE, P. MICHAUD, J. S. SIM, D. NYANG. *A simple proof of optimality for the MIN cache replacement policy*, in "Information Processing Letters", September 2015, 3 p. [*DOI :* 10.1016/J.IPL.2015.09.004], https://hal.inria.fr/hal-01199424

[9] P. MICHAUD. *A Best-Offset Prefetcher* ***Champion***, in "2nd Data Prefetching Championship", Portland, OR, USA, June 2015, https://hal.inria.fr/hal-01165600

[10] P. MICHAUD, A. MONDELLI, A. SEZNEC. *Revisiting Clustered Microarchitecture for Future Superscalar Cores: A Case for Wide Issue Clusters*, in "ACM Transactions on Architecture and Code Optimization (TACO) ", August 2015, vol. 13, n^{o} 3, 22 p. [*DOI :* 10.1145/2800787], https://hal.inria.fr/hal-01193178

[11] P. MICHAUD, A. SEZNEC. *Pushing the branch predictability limits with the multi-poTAGE+SC predictor : **Champion in the unlimited category***, in "4th JILP Workshop on Computer Architecture Competitions (JWAC-4): Championship Branch Prediction (CBP-4)", Minneapolis, United States, June 2014, https://hal.archives-ouvertes.fr/hal-01087719

[12] A. PERAIS. *Increasing the performance of superscalar processors through value prediction*, Université Rennes 1, September 2015, https://tel.archives-ouvertes.fr/tel-01282474

[13] A. PERAIS, A. SEZNEC. *EOLE: Paving the Way for an Effective Implementation of Value Prediction*, in "International Symposium on Computer Architecture", Minneapolis, MN, United States, ACM/IEEE, June 2014, vol. 42, pp. 481 - 492 [*DOI :* 10.1109/ISCA.2014.6853205], https://hal.inria.fr/hal-01088130

[14] A. PERAIS, A. SEZNEC. *Practical data value speculation for future high-end processors*, in "International Symposium on High Performance Computer Architecture", Orlando, FL, United States, IEEE, February 2014, pp. 428 - 439 [*DOI :* 10.1109/HPCA.2014.6835952], https://hal.inria.fr/hal-01088116

[15] A. PERAIS, A. SEZNEC. *EOLE: Toward a Practical Implementation of Value Prediction*, in "IEEE Micro", June 2015, vol. 35, n^{o} 3, pp. 114 - 124 [*DOI :* 10.1109/MM.2015.45], https://hal.inria.fr/hal-01193287

[16] E. RIOU, E. ROHOU, P. CLAUSS, N. HALLOU, A. KETTERLIN. *PADRONE: a Platform for Online Profiling, Analysis, and Optimization*, in "Dynamic Compilation Everywhere", Vienna, Austria, January 2014

[17] S. SARDASHTI, A. SEZNEC, D. A. WOOD. *Skewed Compressed Caches*, in "47th Annual IEEE/ACM International Symposium on Microarchitecture, 2014", Minneapolis, United States, December 2014, https://hal.inria.fr/hal-01088050

[18] A. SEMBRANT, T. CARLSON, E. HAGERSTEN, D. BLACK-SHAFFER, A. PERAIS, A. SEZNEC, P. MICHAUD. *Long Term Parking (LTP): Criticality-aware Resource Allocation in OOO Processors*, in "International Symposium on Microarchitecture, Micro 2015", Honolulu, United States, Proceeding of the International Symposium on Microarchitecture, Micro 2015, ACM, December 2015, https://hal.inria.fr/hal-01225019

[19] A. SEZNEC, P. MICHAUD. *A case for (partially)-tagged geometric history length predictors*, in "Journal of Instruction Level Parallelism", April 2006, http://www.jilp.org/vol8

[20] A. SEZNEC, J. SAN MIGUEL, J. ALBERICIO. *The Inner Most Loop Iteration counter: a new dimension in branch history* , in "48th International Symposium On Microarchitecture", Honolulu, United States, ACM, December 2015, 11 p. , https://hal.inria.fr/hal-01208347

[21] A. SEZNEC. *A New Case for the TAGE Branch Predictor*, in "MICRO 2011 : The 44th Annual IEEE/ACM International Symposium on Microarchitecture, 2011", Porto Allegre, Brazil, ACM (editor), ACM-IEEE, December 2011, https://hal.inria.fr/hal-00639193

[22] A. SEZNEC. *TAGE-SC-L Branch Predictors: **Champion in 32Kbits and 256 Kbits category***, in "JILP - Championship Branch Prediction", Minneapolis, United States, June 2014, https://hal.inria.fr/hal-01086920

[23] A. SURESH, B. NARASIMHA SWAMY, E. ROHOU, A. SEZNEC. *Intercepting Functions for Memoization: A Case Study Using Transcendental Functions*, in "ACM Transactions on Architecture and Code Optimization (TACO) ", July 2015, vol. 12, n^{o} 2, 23 p. [*DOI :* 10.1145/2751559], https://hal.inria.fr/hal-01178085

[24] A. SURESH. *Intercepting functions for memoization*, Université Rennes 1, May 2016, https://tel.archives-ouvertes.fr/tel-01410539

[25] D. D. C. TEIXEIRA, S. COLLANGE, F. M. Q. PEREIRA. *Fusion of calling sites*, in "International Symposium on Computer Architecture and High-Performance Computing (SBAC-PAD)", Florianópolis, Santa Catarina, Brazil, October 2015 [*DOI :* 10.1109/SBAC-PAD.2015.16], https://hal.archives-ouvertes.fr/hal-01410221

Publications of the year

Doctoral Dissertations and Habilitation Theses

[26] N. HALLOU. *Runtime Optimization of Binary Through Vectorization Transformations*, Université de Rennes 1 [UR1], December 2017, https://hal.inria.fr/tel-01672263

[27] A. MONDELLI. *Revisiting Wide Superscalar Microarchitecture*, Université de Rennes 1, September 2017, https://hal.inria.fr/tel-01597752

Articles in International Peer-Reviewed Journals

[28] F. ENDO, A. PERAIS, A. SEZNEC. *On the Interactions Between Value Prediction and Compiler Optimizations in the Context of EOLE*, in "ACM Transactions on Architecture and Code Optimization", June 2017, https://hal.inria.fr/hal-01519869

[29] N. HALLOU, E. ROHOU, P. CLAUSS. *Runtime Vectorization Transformations of Binary Code*, in "International Journal of Parallel Programming", June 2017, vol. 8, n^{o} 6, pp. 1536 - 1565 [*DOI :* 10.1007/s10766-016-0480-z], https://hal.inria.fr/hal-01593216

[30] S. KALATHINGAL, S. COLLANGE, B. SWAMY, A. SEZNEC. *DITVA: Dynamic Inter-Thread Vectorization Architecture*, in "Journal of Parallel and Distributed Computing", 2017, pp. 1-32, forthcoming [*DOI :* 10.1016/J.JPDC.2017.11.006], https://hal.archives-ouvertes.fr/hal-01655904

[31] B. ROUXEL, S. DERRIEN, I. PUAUT. *Tightening Contention Delays While Scheduling Parallel Applications on Multi-core Architectures*, in "ACM Transactions on Embedded Computing Systems (TECS)", October 2017, vol. 16, n^{o} 5s, pp. 1 - 20 [*DOI :* 10.1145/3126496], https://hal.archives-ouvertes.fr/hal-01655383

[32] A. SRIDHARAN, B. PANDA, A. SEZNEC. *A Band-pass Prefetching : An Effective Prefetch Management Mechanism using Prefetch-fraction Metric in Multi-core Systems*, in "ACM Transactions on Architecture and Code Optimization", June 2017, https://hal.inria.fr/hal-01519648

[33] A. SRIDHARAN, A. SEZNEC. *Dynamic and Discrete Cache Insertion Policies for Managing Shared Last Level Caches in Large Multicores*, in "Journal of Parallel and Distributed Computing", May 2017, vol. 106, pp. 215–226 [*DOI :* 10.1016/J.JPDC.2017.02.004], https://hal.inria.fr/hal-01519650

Invited Conferences

[34] C. SILVANO, A. BARTOLINI, A. BECCARI, C. MANELFI, C. CAVAZZONI, D. GADIOLI, E. ROHOU, G. PALERMO, G. AGOSTA, J. MARTINOVIČ, J. BISPO, J. M. P. CARDOSO, J. BARBOSA, K. SLANINOVÁ, L. BENINI, M. PALKOVIČ, N. SANNA, P. PINTO, R. CMAR, R. NOBRE, S. CHERUBIN. *The ANTAREX Tool Flow for Monitoring and Autotuning Energy Efficient HPC Systems*, in "SAMOS 2017 - International

Conference on Embedded Computer Systems: Architecture, Modeling and Simulation", Pythagorion, Greece, July 2017, https://hal.inria.fr/hal-01615945

International Conferences with Proceedings

[35] V. A. ANH NGUYEN, D. HARDY, I. PUAUT. *Cache-conscious offline real-time task scheduling for multi-core processors*, in "29th Euromicro Conference on Real-Time Systems (ECRTS17)", Dubrovnik, Croatia, Euromicro Conference on Real-Time Systems, June 2017 [*DOI :* 10.4230/LIPICS.ECRTS.2017.14], http://hal.upmc.fr/hal-01590421

[36] A. A. AP, E. ROHOU. *Dynamic Function Specialization*, in "International Conference on Embedded Computer Systems: Architectures, MOdeling and Simulation", Pythagorion, Samos, Greece, July 2017, https://hal.inria.fr/hal-01597880

[37] R. BOUZIANE, E. ROHOU, A. GAMATIÉ. *How Could Compile-Time Program Analysis help Leveraging Emerging NVM Features?*, in "EDIS 2017 - First international conference on Embedded & Distributed Systems", Oran, Algeria, December 2017, pp. 1-6, https://hal.inria.fr/hal-01655195

[38] R. BOUZIANE, E. ROHOU, A. GAMATIÉ. *Compile-Time Silent-Store Elimination for Energy Efficiency: an Analytic Evaluation for Non-Volatile Cache Memory*, in "RAPIDO 2018 - 10th Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools", Manchester, United Kingdom, HiPEAC, January 2018, pp. 1-8, https://hal.inria.fr/hal-01660686

[39] S. CHERUBIN, G. AGOSTA, I. LASRI, E. ROHOU, O. SENTIEYS. *Implications of Reduced-Precision Computations in HPC: Performance, Energy and Error*, in "International Conference on Parallel Computing (ParCo)", Bologna, Italy, September 2017, https://hal.inria.fr/hal-01633790

[40] S. COLLANGE. *Simty: generalized SIMT execution on RISC-V*, in "First Workshop on Computer Architecture Research with RISC-V (CARRV 2017)", Boston, United States, First Workshop on Computer Architecture Research with RISC-V, October 2017, 6 p. , https://hal.inria.fr/hal-01622208

[41] S. DERRIEN, I. PUAUT, P. ALEFRAGIS, M. BEDNARA, H. BUCHER, C. DAVID, Y. DEBRAY, U. DURAK, I. FASSI, C. FERDINAND, D. HARDY, A. KRITIKAKOU, G. RAUWERDA, S. REDER, M. SICKS, T. STRIPF, K. SUNESEN, T. TER BRAAK, N. VOROS, J. †. BECKER. *WCET-aware parallelization of model-based applications for multi-cores: The ARGO approach*, in "Design Automation and Test in Europe (DATE), 2017", Lausanne, Switzerland, March 2017, pp. 286 - 289 [*DOI :* 10.23919/DATE.2017.7927000], http://hal.upmc.fr/hal-01590418

[42] D. HARDY, B. ROUXEL, I. PUAUT. *The Heptane Static Worst-Case Execution Time Estimation Tool*, in "17th International Workshop on Worst-Case Execution Time Analysis (WCET 2017)", Dubrovnik, Croatia, International Workshop on Worst-Case Execution Time Analysis, June 2017, vol. 8, 12 p. [*DOI :* 10.4230/OASICS.WCET.2017.8], http://hal.upmc.fr/hal-01590444

[43] C. MAIZA, P. RAYMOND, C. PARENT-VIGOUROUX, A. BONENFANT, F. CARRIER, H. CASSÉ, P. CUENOT, D. CLARAZ, N. HALBWACHS, E. JAHIER, H. LI, M. DE MICHIEL, V. MUSSOT, I. PUAUT, C. ROCHANGE, E. ROHOU, J. RUIZ, P. SOTIN, W.-T. SU. *The W-SEPT Project: Towards Semantic-Aware WCET Estimation*, in "17th International Workshop on Worst-Case Execution Time Analysis (WCET 2017)", Dubrovnik, Croatia, International Workshop on Worst-Case Execution Time Analysis, June 2017, 13 p. [*DOI :* 10.4230/OASICS.WCET.2017.9], http://hal.upmc.fr/hal-01590442

[44] S. MARTINEZ, D. HARDY, I. PUAUT. *Quantifying WCET reduction of parallel applications by introducing slack time to limit resource contention*, in "International Conference on Real-Time Networks and Systems (RTNS), 2017", Grenoble, France, International Conference on Real-Time Networks and Systems, October 2017 [*DOI :* 10.475/123_4], http://hal.upmc.fr/hal-01590532

[45] R. E. A. MOREIRA, S. COLLANGE, F. M. Q. PEREIRA. *Function Call Re-Vectorization*, in "ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP)", Austin, Texas, United States, February 2017, https://hal.archives-ouvertes.fr/hal-01410186

[46] S. ROKICKI, E. ROHOU, S. DERRIEN. *Hardware-Accelerated Dynamic Binary Translation*, in "IEEE/ACM Design, Automation & Test in Europe Conference & Exhibition (DATE)", Lausanne, Switzerland, March 2017, https://hal.inria.fr/hal-01423639

[47] B. ROUXEL, S. DERRIEN, I. PUAUT. *Tightening contention delays while scheduling parallel applications on multi-core architectures*, in "International Conference on Embedded Software (EMSOFT), 2017", Seoul, South Korea, International Conference on Embedded Software, October 2017, 20 p. [*DOI :* 10.1145/3126496], http://hal.upmc.fr/hal-01590508

[48] B. ROUXEL, I. PUAUT. *STR2RTS: Refactored StreamIT benchmarks into statically analyzable parallel benchmarks for WCET estimation & real-time scheduling*, in "17th International Workshop on Worst-Case Execution Time Analysis (WCET 2017)", Dubrovnik, Croatia, June 2017 [*DOI :* 10.4230/OASICS.WCET.2017.1], http://hal.upmc.fr/hal-01590446

[49] M. Y. SIRAICHI, V. F. D. SANTOS, S. COLLANGE, F. M. Q. PEREIRA. *Qubit Allocation*, in "CGO 2018 - International Symposium on Code Generation and Optimization", Vienna, Austria, February 2018, pp. 1-12 [*DOI :* 10.1145/3168822], https://hal.archives-ouvertes.fr/hal-01655951

[50] A. SURESH, E. ROHOU, A. SEZNEC. *Compile-Time Function Memoization*, in "26th International Conference on Compiler Construction", Austin, United States, February 2017, https://hal.inria.fr/hal-01423811

National Conferences with Proceedings

[51] S. COLLANGE, N. BRUNIE. *Parcours par liste de chemins : une nouvelle classe de mécanismes de suivi de flot SIMT*, in "Conférence d'informatique en Parallélisme, Architecture et Système (ComPAS)", Sophia Antipolis, France, June 2017, https://hal.inria.fr/hal-01522901

Research Reports

[52] S. COLLANGE, N. BRUNIE. *Path list traversal: a new class of SIMT flow tracking mechanisms*, Inria Rennes - Bretagne Atlantique, June 2017, n° RR-9073, https://hal.inria.fr/hal-01533085

Other Publications

[53] S. ROKICKI, E. ROHOU, S. DERRIEN. *Supporting Runtime Reconfigurable VLIWs Cores Through Dynamic Binary Translation*, December 2017, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01653110

References in notes

[54] M. HATABA, A. EL-MAHDY, E. ROHOU. *OJIT: A Novel Obfuscation Approach Using Standard Just-In-Time Compiler Transformations*, in "International Workshop on Dynamic Compilation Everywhere", January 2015

[55] R. KUMAR, D. M. TULLSEN, N. P. JOUPPI, P. RANGANATHAN. *Heterogeneous chip multiprocessors*, in "IEEE Computer", nov. 2005, vol. 38, n^{o} 11, pp. 32–38

[56] S. NASSIF, N. MEHTA, Y. CAO. *A resilience roadmap*, in "Design, Automation Test in Europe Conference Exhibition (DATE), 2010", March 2010, pp. 1011-1016

[57] J. NICKOLLS, W. J. DALLY. *The GPU computing era*, in "Micro, IEEE", 2010, vol. 30, n^{o} 2, pp. 56–69

[58] R. OMAR, A. EL-MAHDY, E. ROHOU. *Arbitrary control-flow embedding into multiple threads for obfuscation: a preliminary complexity and performance analysis*, in "Proceedings of the 2nd international workshop on Security in cloud computing", ACM, 2014, pp. 51–58

[59] A. SEZNEC, N. SENDRIER. *HAVEGE: A user-level software heuristic for generating empirically strong random numbers*, in "ACM Transactions on Modeling and Computer Simulation (TOMACS)", 2003, vol. 13, n^{o} 4, pp. 334–346

[60] H. WONG, T. M. AAMODT. *The Performance Potential for Single Application Heterogeneous Systems*, in "8th Workshop on Duplicating, Deconstructing, and Debunking", 2009