IN PARTNERSHIP WITH:
Université Rennes 1

Activity Report 2018

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
A2.2.4. - Parallel architectures
A2.2.5. - Run-time systems
A2.2.6. - GPGPU, FPGA...
A2.2.7. - Adaptive compilation
A2.2.8. - Code generation
A2.2.9. - Security by 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. Team, Visitors, External Collaborators

Research Scientists
Erven Rohou [Team leader, Inria, Senior Researcher, HDR]
Sylvain Collange [Inria, Researcher]
Byron Hawkins [Inria, Starting Research Position, from Dec 2018]
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
Imen Fassi [Univ de Rennes I]
Byron Hawkins [Inria, until Nov 2018]
Stefanos Skalistis [Univ de Rennes I]
Anita Tino [Inria, until Jul 2018]

PhD Students
Arif Ali Ana-Pparakkal [Inria, until Jan 2018]
Arthur Blanleuil [Univ de Rennes I, from Oct 2018]
Rabab Bouziane [Inria]
Niloofar Charmchi [Inria]
Kleovoulos Kalaitzidis [Inria]
Kévin Le Bon [Inria, from Sep 2018]
Viet Anh Nguyen [Univ de Rennes I, until Jan 2018]
Daniel Rodrigues Carvalho [Inria]
Benjamin Rouxel [Univ de Rennes I, until Nov 2018]
Bahram Yarahmadi [Inria, from Feb 2018]

Technical staff
Loïc Besnard [CNRS]
Nicolas Kiss [Inria, until Nov 2018]
Alexandre Kouyoumdjian [Inria, from Mar 2018]
Imane Lasri [Inria, until Jun 2018]
Kévin Le Bon [Apprentice, until Sep 2018]

Interns
Victor Careil [École normale supérieure de Rennes, from May 2018 until Jul 2018]
Pierre Le Luron [Univ de Rennes I, from May 2018 until Jul 2018]
Pierre Le Meur [Univ de Rennes I, from Apr 2018 until Aug 2018]

Administrative Assistant
Virginie Desroches [Inria]

Visiting Scientists
Caio de Lima [Universidade Federal de Minas Gerais, until Apr 2018]
Marcos Siraichi [Universidade Federal de Minas Gerais, from Jul 2018 until Oct 2018]
Marcos Siraichi [Universidade Federal de Minas Gerais, until Mar 2018]

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 micro-architectures. 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.

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

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 micro-architecture.

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.

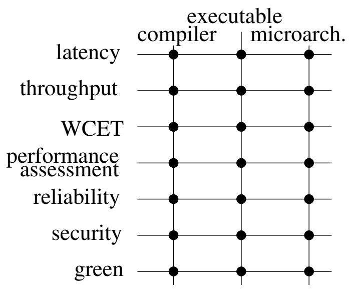

Figure 1. Perimeter of Research Objectives

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 workloads 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 micro-architecture, 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 the micro-architecture 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 micro-architecture and compilation techniques, we will be able to significantly improve efficiency, robustness and coverage of security mechanisms, as well as to partner with field experts to design innovative solutions.

2.1.9. Green Computing – Power Concerns

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 thus 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 evolution of the semiconductor industry predicts an exponential growth of the number of permanent faults [45]. 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-Silicon, 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 a 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 impact 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 increased 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 micro-architecture (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 continue to be relevant in addressing 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 new optimization opportunities and develops tailored compilation techniques for 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.

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

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 transparently [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 (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 [12] 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 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 micro-architecture

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 micro-architecture 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 recent years they have been recognized as world leaders in branch prediction [19][9] and in cache prefetching [7] and they have revived the forgotten concept of value prediction [11][10]. 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 cores;
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) [7]. The first research objective 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 research objective 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 benefits 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 [13]. It offers new possibilities 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 processors. For the two past decades, implementing ever wider issue superscalar processors has been challenging. The objective of our research on the execution core is to explore (and revisit) directions that 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 architectures [8]. Symmetric clustered organization allows to benefit from a simpler bypass network, but induce large complexity on the issue queue. One remarkable finding of our study [8] 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. [15]. 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 micro-architecture with a cluster dedicated to the execution of urgent instructions and a second cluster executing the non-urgent instructions. The micro-architecture 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 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 [17].

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 [11][10][19][9]. First, very high accuracy can be enforced at reasonable cost in coverage and minimal complexity [11]. 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 [10]. 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 [44] 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 micro-architectures 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 compiler levels.

Architecture.
The architecture convergence between Single Instruction Multiple Threads (SIMT) GPUs and multicore processors that we have been pursuing [5] 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.

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 analyze and/or control 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 [45].

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 kinds 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 effects.

Considering the memory hierarchy, we have already studied [4] 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.

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

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 [48] random number generator, and code obfuscating techniques (the obfuscating just-in-time compiler [43], or thread-based control flow mangling [46]).

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 in 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 will 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 impact 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. Highlights of the Year

5.1. Highlights of the Year

5.1.1. Awards

André Seznec won the three tracks of the 1st Championship of Value Prediction with the EVES predictor.

Arthur Perais, former PhD student in the project-team, and André Seznec received the best paper award at the conference ACM PACT 2018 for their paper “Cost Effective Speculation with the Omnipredictor”.

BEST PAPERS AWARDS:

[39]
A. SEZNEC. *Exploring value prediction with the EVES predictor*, in "CVP-1 2018 - 1st Championship Value Prediction", Los Angeles, United States, June 2018, pp. 1-6, https://hal.inria.fr/hal-01888864

[35]
A. PERAIS, A. SEZNEC. *Cost Effective Speculation with the Omnipredictor*, in "International conference on Parallel Architectures and Compilation Techniques (PACT ’18),", Limassol, Cyprus, November 2018 [*DOI :* 10.1145/3243176.3243208], https://hal.inria.fr/hal-01888884

6. New Software and Platforms

6.1. 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.

ATMI (Analytical model of Temperature in MIcroprocessors) is an ad hoc temperature model 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/

6.2. HEPTANE

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

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.
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/

6.3. tiptop

KEYWORDS: Instructions - Cycles - Cache - CPU - Performance - HPC - Branch predictor
SCIENTIFIC DESCRIPTION: Tiptop is a new simple and flexible user-level tool that collects hardware counter data on Linux platforms (version 2.6.31+) and displays them in a way simple to the Linux "top" utility. The goal is to make the collection of performance and bottleneck data as simple as possible, including simple installation and usage. No privilege is required, any user can run tiptop.

Tiptop is written in C. It can take advantage of libncurses when available for pseudo-graphic display. 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.

Current version is 2.3.1, released October 2017. Tiptop has been integrated in major Linux distributions, such as Fedora, Debian, Ubuntu.
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

6.4. 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/pacap/software/padrone

6.5. 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.

FUNCTIONAL DESCRIPTION: If-memo is a linker-based technique for enabling software memorizing of any dynamically linked pure function by function interception. Typically, this framework is useful to intercept the computationally expensive pure functions - the transcendental functions from the math library. 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/pacap/software/if-memo/

6.6. Simty

KEYWORDS: GPU - Softcore - FPGA - SIMT - Multi-threading - RISC-V
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

6.7. Barra

KEYWORDS: GPU - GPGPU - Tesla ISA - Debug - Computer architecture - Performance - Profiling - Simulator - HPC - CUDA
SCIENTIFIC DESCRIPTION: Research on throughput-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: Version 0.5.10 introduces: Timing model, Tesla-like architecture model, Fermi-like architecture model, New per-PC control-flow divergence management, Support for Simultaneous branch and warp interweaving, Support for 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/

6.8. Memoization

KEYWORDS: Optimization - Pure function - Memoization

FUNCTIONAL DESCRIPTION: Memoization is a technique used at runtime that consists in caching results of pure functions and retrieving them instead of computing it when the arguments repeat. It can be applied to C and C++ programs. To be memoized, the interface of a pure function (or a method) must verify the following properties: (1) the function/method has at most four arguments of same type T, (2) the function/method returns a data of type T, (3) T is either 'double', 'float', or 'int'.

The memoization operation of a function/method is controlled by several parameters: the size of the internal table (number of entries), the replacement policy to be used in case of index conflict (whether the value of the table must be replaced or not), an approximation threshold that allows to not distinguish very close values). It is also possible to initialize the table with the content of a file, and to save the content of the table to a file at the end of the execution (the data may be used as input for a future execution).

- Participants: Loïc Besnard, Imane Lasri and Erven Rohou
- Contact: Loïc Besnard

6.9. FiPlib

KEYWORDS: Compilation - Approximate computing - Fixed-point representation
FUNCTIONAL DESCRIPTION: FiPlib is a C++ library that provides type definition and conversion operations for computations in fixed-point representation. Basic arithmetic as well as logical operations are transparently supported thanks to operator overloading. FiPlib also provides optimized implementations of the transcendental math functions of libm. For convenient integration, FiPlib is released as C++ header files only. Optionally, FiPlib can detect overflows and compute errors compared to floating point representation.

- Participants: Pierre Le Meur, Imane Lasri and Erven Rohou
- Contact: Erven Rohou

6.10. sigmask

KEYWORDS: Compilation - Side-channel - Masking - Security - Embedded systems
SCIENTIFIC DESCRIPTION: Sigmask is a compiler plugin based on the LLVM infrastructure that automatically protects secret information in programs, such as encryption keys, against side-channel attacks. The programmer annotates their source code to highlight variables containing sensitive data. The compiler automatically analyzes the program and computes all memory locations potentially derived from the secret. It then applies a masking scheme to avoid information leakage. Sigmask provides several schemes: OSDM (Orthogonal Direct Sum Masking), IP (Inner Product) Masking, and simple random bit masking. The programmer may also provide their own masking scheme through a well-defined API.

- Participants: Nicolas Kiss, Damien Hardy and Erven Rohou
- Contact: Erven Rohou

7. New Results

7.1. Compilation and Optimization

Participants: Arif Ali Ana-Pparakkal, Loïc Besnard, Rabab Bouziane, Sylvain Collange, Byron Hawkins, Imane Lasri, Kévin Le Bon, Erven Rohou.

7.1.1. Optimization in the Presence of NVRAM

Participants: Rabab Bouziane, Erven Rohou, Bahram Yarahmadi.

Energy-efficiency has become one major challenge in both embedded and high-performance computing. Different approaches have been investigated to solve the challenge, e.g., heterogeneous multicore, system runtime and device-level power management, or deployment of emerging non-volatile memories (NVMs), such as Spin-Transfer Torque RAM (STT-RAM), which inherently have quasi-null leakage. This enables to reduce the static power consumption, which tends to become dominant in modern systems. The usage of NVM in memory hierarchy comes however at the cost of expensive write operations in terms of latency and energy.

7.1.1.1. Silent-Stores

We propose [31] 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.

7.1.1.2. Variable Retention Time

In order to mitigate expensive writes, we leverage the notion of *δ-worst-case execution time* (δ-WCET), which consists of partial WCET estimates [32]. From program analysis, δ-WCETs are determined and used to safely allocate data to NVM memory banks with variable data retention times. The δ-WCET analysis computes the WCET between any two locations in a function code, i.e., between basic blocks or instructions. Our approach is validated on the Mälardalen benchmark suite and significant memory dynamic energy reductions (up to 80 %, and 66 % on average) are reported.

This research is done in collaboration with Abdoulaye Gamatié at LIRMM (Montpellier) within the context the ANR project CONTINUUM. Results are detailed in the PhD thesis document of Rabab Bouziane, defended in December 2018 [20].

7.1.1.3. Efficient checkpointing for intermittently-powered systems

Future internet of things (IoT) ultra low-power micro controllers do not have any battery. Instead they harvest energy from the environment such as solar or radio and store it to a capacitor. However, one of the unique problems with these energy harvesting devices is the unstable energy supply which causes frequent power failures during the execution of the program. As a result, a program may not be able to terminate with one power cycle. A solution to this problem consists in using non-volatile memory (NVM) such as FLASH or Ferroelectric RAM (FRAM) and checkpointing the volatile state of the program to the non-volatile memory regularly. The program can resume its execution when the power is back. However, checkpointing regularly at runtime has overhead, and poses some memory inconsistency problems [47]. By statically analyzing the program binary code, we propose to insert checkpoints in the proper places in the program to decrease the amount of checkpointing overhead at runtime. Also, a compiler can give hints to runtime system about whether to do the checkpoint or not. Concerning the reduction of checkpointing overhead, we are analyzing the binary code of the program for estimating the energy of each section of the program. We rely on Heptane, which is originally designed for estimating worst-case execution time. However, by giving energy cost to each ISA instruction, we can estimate the energy consumption of the sections of the program for processors like MSP430 or Arm-cortex m0+ which are typical low-power embedded processors. With this information, we insert checkpoints into the LLVM IR and also apply optimizations in order to have better performance and energy efficiency at runtime.

This research is done within the context of the project IPL ZEP.

7.1.2. Dynamic Binary Optimization

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

Modern hardware features can boost the performance of an application, but software vendors are often limited to the lowest common denominator to maintain compatibility with the spectrum of processors used by their clients. Given more detailed information about the hardware features, a compiler can generate more efficient code, but even if the exact CPU model is known, manufacturer confidentiality policies leave substantial uncertainty about precise performance characteristics. In addition, the activity of other programs colocated in the same runtime environment can have a dramatic effect on application performance. For example, if a shared CPU cache is being heavily used by other programs, memory access latencies may be orders of magnitude longer than those recorded during an isolated profiling session, and instruction scheduling based on such profiles may lose its anticipated advantages. Program input can also drastically change the efficiency of statically compiled code, yet in many cases is subject to total uncertainty until the moment the input arrives during program execution. We have developed FITTCHOOSER [30] to defer optimization of a program's most processor-intensive functions until execution time. FITTCHOOSER begins by profiling the application to determine the performance characteristics that are in effect for the present execution, then generates a set of candidate variations and dynamically links them in succession to empirically measure which of them performs best. The underlying binary instrumentation framework Padrone allows for selective transformation of the program without otherwise modifying its structure or interfering with the flow of execution, making it possible for FITTCHOOSER to minimize the overhead of its dynamic optimization process. Our experimental evaluation demonstrates up to 19 % speedup on a selection of programs from the SPEC CPU 2006 and PolyBench suites while introducing less than 1 % overhead. The FITTCHOOSER prototype achieves these gains with a minimal repertoire of optimization techniques taken from the static compiler itself, which not only testifies to the effectiveness of dynamic optimization, but also suggests that further gains can be achieved by expanding FITTCHOOSER's repertoire of program transformations to include more diverse and more advanced techniques.

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

Nowadays almost every device has parallel architecture, hence parallelization is almost always desirable. However, parallelizing legacy running programs is very challenging. That is due to the fact that usually source code is not available, and runtime parallelization is challenging. Also, detecting parallelizable code is difficult, due to possible dependencies and different execution paths that are undecidable statically. Therefore, speculation is a typical approach whereby wrongly parallelized code is detected and rolled back at runtime. We proposed [27] utilizing processes to implement speculative parallelization using on-stack replacement, allowing for generally simple and portable design where forking a new process enters the speculative state, and killing a faulty process simply performs the roll back operation. While the cost of such operations are high, the approach is promising for cases where the parallel section is long and dependency issues are rare. Also, our proposed system performs speculative parallelization on binary code at runtime, without the need for source code, restarting the program or special hardware support. Initial experiments show about 2× to 3× speedup for speculative execution over serial, when three fourth of loop iterations are parallelizable. Maximum speculation overhead over pure parallel execution is measured at 5.8 %.

This research was partially done within the context of the project PHC IMHOTEP.

7.1.3. Autotuning

Participants: Loïc Besnard, Imane Lasri, Pierre Le Meur, Erven Rohou.

The ANTAREX project relies on a Domain Specific Language LARA [5] of the Clava environment [6]. This DSL is based on Aspect Oriented Programming concepts to allow applications to enforce extra functional properties such as energy-efficiency and performance and to optimize Quality of Service in an adaptive way. The DSL approach allows the definition of energy-efficiency, performance, and adaptivity strategies as well as their enforcement at runtime through application autotuning and resource and power management [28], [29].

In this context, this year we have integrated in Clava some technologies: the memoization, the precision tuning and the loop splitting compilation.

[5] https://web.fe.up.pt/~specs/projects/lara/doku.php
[6] http://specs.fe.up.pt/tools/clava

7.1.3.1. Memoization

The concept of memoization essentially involves saving the results of functions together with their inputs so that when the input repeats, the result is taken from a look-up table. This technique, whose objective is to improve sequential performance, has been implemented for C and C++ languages. The support library of this technology allows in particular flexibility for the table management. This work has been submitted for publication in the Elsevier journal SoftwareX. The support library is available at https://gforge.inria.fr/projects/memoization (registered with APP under number IDDN.FR.001.250029.000.S.P.2018.000.10800)

7.1.3.2. Precision tuning

The developed aspects on the type precision consist in the parametrization of the applications in terms of types. Indeed, error-tolerating applications are increasingly common in the emerging field of real-time HPC. Thus, recent works investigated the use of customized precision in HPC as a way to provide a breakthrough in power and performance. This parametrization allows to test easily and quickly different type representations (such as `double`, `float`, `fixed-point`).

7.1.3.3. Loop splitting

The loop splitting technique takes advantage of long running loops to explore the impact of several optimization sequences at once, thus reducing the number of necessary runs. We rely on a variant of loop peeling which splits a loop into into several loops, with the same body, but a subset of the iteration space. New loops execute consecutive chunks of the original loop. We then apply different optimization sequences on each loop independently. Timers around each chunk observe the performance of each fragment. This technique may be generalized to combine compiler options and different implementations of a function called in a loop. It is useful when, for example, the profiling of the application shows that a function is critical in term of time of execution. In this case, the user must try to find the best implementation of their algorithm.

This research is done within the context of the ANTAREX FET HPC collaborative project. The software is being registered with APP.

7.1.4. Hardware/Software JIT Compiler

Participant: Erven Rohou.

In order to provide dynamic adaptation of the performance/energy trade-off, systems today rely on heterogeneous multi-core architectures (different micro-architectures on a chip). These systems are limited to single-ISA approaches to enable transparent migration between the different cores. To offer more trade-offs, we can integrate statically scheduled micro-architecture and use Dynamic Binary Translation (DBT) for task migration. However, in a system where performance and energy consumption are a prime concern, the translation overhead has to be kept as low as possible. We propose Hybrid-DBT [26], an open-source, hardware accelerated DBT system targeting VLIW cores. Three different hardware accelerators have been designed to speed-up critical steps of the translation process. Experimental study shows that the accelerated steps are two orders of magnitude faster than their software equivalent. The impact on the total execution time of applications and the quality of generated binaries are also measured.

Our proposed DBT framework targets the RISC-V ISA, for which both OoO and in-order implementations exist. Our experimental results [37] show that our approach can lead to best-case performance and energy efficiency when compared against static VLIW configurations.

This work is part of the PhD of Simon Rokicki [22], co-advised by Erven Rohou.

7.1.5. Qubit allocation for quantum circuit compilers

Participants: Sylvain Collange, Marcos Siraichi, Victor Careil.

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. In collaboration with Vinícius Fernandes dos Santos, Fernando Pereira and Marcos Yukio Siraichi at UFMG, 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 was presented at the Code Generation and Optimization (CGO) conference [40].

7.2. Processor Architecture

Participants: Sylvain Collange, Niloofar Charmchi, Kleovoulos Kalaitzidis, Pierre Michaud, Daniel Rodrigues Carvalho, André Seznec, Anita Tino.

7.2.1. Value prediction

Participants: Kleovoulos Kalaitzidis, André Seznec.

For the 1st Championship on Value Prediction (CVP1), we have explored the performance limits of value prediction for small value predictors (8KB and 32KB) in the context of a processor assuming a large instruction window (256-entry ROB), a perfect branch predictor, fetching 16 instructions per cycle, an unlimited number of functional units, but a large value misprediction penalty with a complete pipeline flush at commit on a value misprediction

Our proposition EVES, for Enhanced VTAGE Enhanced Stride, combines two predictor components which do not use on the result of the last occurrence of the instruction to compute the prediction. We use an enhanced version of the VTAGE predictor [11], E-VTAGE. Second, we propose an enhanced version of the stride predictor, E-Stride. E-Stride computes the prediction from the last committed occurrence of the instruction and the number of speculative inflight occurrences of the instruction in the pipeline. The prediction flowing out from E-Stride or E-VTAGE is used only when its confidence is high. A major contribution of this study is the algorithm to assign confidence to predictions depending on the expected benefit/loss from the prediction.

The EVES predictor won the three tracks of CVP1 [39].

7.2.2. Compressed caches

Participants: Daniel Rodrigues Carvalho, Niloofar Charmchi, André Seznec.

Recent advances in research on compressed caches make them an attractive design point for effective hardware implementation for last-level caches. For instance, the yet another compressed cache (YACC) layout [14] leverages both spatial and compression factor localities to pack compressed contiguous memory blocks from a 4-block super-block in a single cache block location. YACC requires less than 2 % extra storage over a conventional uncompressed cache. Performance of LLC is also highly dependent on its cache block replacement management. This includes allocation and bypass decision on a miss as well as replacement target selection which is guided by priority insertion policy on allocation and priority promotion policy on a hit. YACC uses the same cache layout as a conventional set-associative uncompressed cache Therefore the LLC cache management policies that were introduced during the past decade can be transposed to YACC. However, YACC features super-block tags instead of block tags. For uncompressed block, these super-block tags can be used to monitor the reuse behavior of blocks from the same super-block. We introduce the First In Then First Use Bypass (FITFUB) allocation policy for YACC. With FITFUB, a missing uncompressed block that belongs to a super-block that is already partially valid in the cache is not stored in the cache on its first use, but only on its first reuse if any. FITFUB can be associated with any priority insertion/promotion policy. YACC+FITFUB with compression turned off, achieves an average 6.5%/8% additional performance over a conventional LLC, for single-core/multi-core workloads, respectively. When compression is enabled, the performance benefits associated with compression and FITFUB are almost additive reaching 12.7%/17%. This leads us to call this design the Synergistic cache layout for Reuse and Compression (SRC). SRC reaches the performance benefit that would be obtained with a 4× larger cache, but with less than 2 % extra storage [34].

7.2.3. The Omnipredictor

Participant: André Seznec.

Modern superscalar processors heavily rely on out-of-order and speculative execution to achieve high performance. The conditional branch predictor, the indirect branch predictor and the memory dependency predictor are among the key structures that enable efficient speculative out-of-order execution. Therefore, processors implement these three predictors as distinct hardware components. In [35] we propose the omnipredictor that predicts conditional branches, memory dependencies and indirect branches at state-of-the-art accuracies without paying the hardware cost of the memory dependency predictor and the indirect jump predictor. We first show that the TAGE prediction scheme based on global branch history can be used to concurrently predict both branch directions and memory dependencies. Thus, we unify these two predictors within a regular TAGE conditional branch predictor whose prediction is interpreted according to the type of the instruction accessing the predictor. Memory dependency prediction is provided at almost no hardware overhead. We further show that the TAGE conditional predictor can be used to accurately predict indirect branches through using TAGE entries as pointers to Branch Target Buffer entries. Indirect target prediction can be blended into the conditional predictor along with memory dependency prediction, forming the omnipredictor.

7.2.4. Branch prediction

Participant: Pierre Michaud.

The branch predictor is the keystone of modern superscalar micro-architectures. The TAGE predictor, introduced by André Seznec and Pierre Michaud in 2006, is the most storage-efficient conditional branch predictor known today [16]. Although TAGE is very accurate, it does not exploit its input information perfectly, as significant prediction accuracy improvements are obtained by complementing TAGE with a perceptron-based *statistical corrector* using the same input information [18]. The statistical corrector, even small, makes the whole predictor more complex. We proposed an alternative TAGE-like predictor, called BATAGE, making statistical correction superfluous. BATAGE has the same global structure as TAGE but uses a different tagged-entry format and different prediction and update algorithms. The main reason for TAGE needing statistical correction is the *cold-counter* problem, that is, the fact that recently created tagged entries contain little branch history. To solve the cold-counter problem, we replaced the up-down counter in the tagged entry with two counters counting the *taken* and *not-taken* occurrences separately, and we introduced Bayesian confidence estimation based on Laplace's rule of succession. We also introduced a method called *Controlled Allocation Throttling* for adjusting the rate of creation of tagged entries dynamically. The resulting predictor, BATAGE, obviates the need for external statistical correction [25].

7.2.5. 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 published in the Journal of Parallel and Distributed Computing [6].

7.2.6. *Toward out-of-order SIMT micro-architecture*

Participants: Sylvain Collange, Anita Tino.

Prior work highlights the continued importance of maintaining adequate sequential performance within throughput-oriented cores [49]. 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.

7.3. WCET estimation and optimization

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

7.3.1. *WCET estimation for many core processors*

Participants: Imen Fassi, Damien Hardy, Viet Anh Nguyen, Isabelle Puaut, Benjamin Rouxel, Stefanos Skalistis.

7.3.1.1. 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 is 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 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. An implementation on the Kalray MPPA machine allowed to identify implementation-related overheads. All results are described in the PhD thesis document of Viet Anh Nguyen [21], defended in February 2018.

This research is part of the PIA Capacités project.

7.3.1.2. Shared resource contentions and WCET estimation

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.

The work along this direction is part of the PhD thesis of Benjamin Rouxel, defended in December 2018. The new results in 2018 concern scheduling/mapping of parallel applications on multi-core systems using ScratchPad Memories (SPMs). We have recently proposed techniques that jointly select SPM contents off-line, in such a way that the cost of SPM loading/unloading is hidden. Communications are fragmented to augment hiding possibilities. Experimental results show the effectiveness of the proposed techniques on streaming applications and synthetic task-graphs. The overlapping of communications with computations allows the length of generated schedules to be reduced by 4 % in average on streaming applications, and by 8 % in average (with maximum of 16 % for both test cases) for synthetic task graphs. We further show on a case study that generated schedules can be implemented with low overhead on a predictable multi-core architecture (Kalray MPPA) [23].

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

Parallel architectures are nowadays not only 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. The Argo toolchain operates on Scilab and XCoS inputs, and targets ScratchPad Memory (SPM)-based multi-cores. Data-layout and loop transformations play a key role in this flow as they improve SPM efficiency and reduce the number of accesses to shared main memory.

In our most recent work [33], we study how these transformations impact WCET estimates of sequential codes. We demonstrate that they can bring significant improvements of WCET estimates (up to 2.7$\times$) provided that the WCET analysis process is guided with automatically generated flow annotations obtained using polyhedral counting techniques.

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

7.3.2. WCET estimation and optimizing compilers

Participants: Imen Fassi, Isabelle Puaut.

Compiler optimizations, although reducing the execution times of programs, raise issues in static WCET estimation techniques and tools. Flow facts, such as loop bounds, may not be automatically found by static WCET analysis tools after aggressive code optimizations. In this work [36], we explore the use of iterative compilation (WCET-directed program optimization to explore the optimization space), with the objective to (i) allow flow facts to be automatically found and (ii) select optimizations that result in the lowest WCET estimates. We also explore to which extent code outlining helps, by allowing the selection of different optimization options for different code snippets of the application.

7.3.3. Partial WCET

Participants: Rabab Bouziane, Erven Rohou.

Computing the worst-case execution time (WCET) of tasks is important for real-time system design. The industry and research communities have developed a wealth of techniques to compute relevant WCET approximations. Traditionally, WCETs are estimated at the granularity of a function (or task). We propose an approach to estimate partial WCET (δ-WCET), i.e., the worst-case execution time between two locations in a function, such as basic blocks or instructions. Our technique [41] is derived from the well-known implicit path enumeration technique. It takes into account both the control flow graph and the architecture (pipeline and cache hierarchy). Some useful applications of such δ-WCETs are motivated in this paper.

This research is part of the ANR Continuum project.

7.4. Security

Participants: Damien Hardy, Byron Hawkins, Nicolas Kiss, Kévin Le Bon, Erven Rohou.

7.4.1. Compiler-based automation of side-channel countermeasures

Masking is a popular protection against side-channel analysis exploiting the power consumption or electromagnetic radiations. Besides the many schemes based on simple Boolean encoding, some alternative schemes such as Orthogonal Direct Sum Masking (ODSM) or Inner Product Masking (IP) aim to provide more security, reduce the entropy or combine masking with fault detection. The practical implementation of those schemes is done manually at assembly or source-code level, some of them even stay purely theoretical. We propose a compiler extension to automatically apply different masking schemes for block cipher algorithms. We introduce a generic approach to describe the schemes and we manage to insert three of them at compile-time on an AES implementation. A practical side-channel analysis as well as fault injections have been performed on an Arm microcontroller to assess the correctness of the code inserted.

The resulting compiler plugin (sigmask) is registered with APP under number IDDN.FR.001.490003.000.S.P.2018.000.10000

This research was done within the context of the project ANR CHIST-ERA SECODE.

7.4.2. Program protection through dynamic binary rewriting

Programs written in languages such as C and C++ are prone to memory corruptions because of the manual management of the memory from the programmer. Even today, memory corruptions are among the most dangerous vulnerabilites. According to the MITRE ranking, these bugs are considered one of the top three most dangerous software vulnerabilites.

Thanks to our library Padrone, we are able to instrument the execution of a program with a minimal overhead, making it possible to move or add code in the target process during its execution. We showed that we can change the address of a function at runtime, thus presenting a moving target to an attacker, and making attacks more difficult.

Many security policies have been developed to protect programs. One of them, the Control-Flow Integrity (CFI) ensures the control-flow of the program cannot be altered, preventing the execution of malicious code. Unfortunately, implementations of precise CFI impose a consequent overhead in performance, due to the instrumentation of the execution of the program. We work on building a solution that is able to adapt its protection level to the situation. Adapting the protection level allows us to reduce even further the overhead in performance when the protection is not needed.

8. Bilateral Contracts and Grants with Industry

8.1. Bilateral Grants with Industry

8.1.1. Intel research grant INTEL2016-11174

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

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

9. Partnerships and Cooperations

9.1. Regional Initiatives

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

9.2. National Initiatives

9.2.1. Capacités: Projet "Investissement d'Avenir" (1/11/14 – 31/01/2018)

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

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.

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

Participants: Erven Rohou, Bahram Yarahmadi.

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 computing 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 [42]. The main application target is Internet of Things (IoT).

9.2.3. ANR Continuum (2015–2019)

Participants: Rabab Bouziane, Erven Rohou.

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.

9.2.4. Hybrid SIMD architectures (2018–2019)

Participants: Sylvain Collange, Alexandre Kouyoumdjian, Erven Rohou.

The project objective is to define new parallel computer architectures that offer high parallel performance on high-regularity workloads while keeping the flexibility to run more irregular parallel workloads. inspired by both GPU and SIMD or vector architectures.

This project is funded by the French Ministry of Armed Forces (*Ministère des Armées*).

9.2.5. DGA/PEC ARMOUR (2018–2021)

Participants: Kévin Le Bon, Erven Rohou.

ARMOUR (dynAmic binaRy optiMizatiOn cyber-secURity) aims at improving the security of computing systems at the software level. Our contribution will be twofold: (1) identify vulnerabilities in existing software, and (2) develop adaptive countermeasure mechanisms against attacks. We will rely on dynamic binary rewriting (DBR) which consists in observing a program and modifying its binary representation in memory while it runs. DBR does not require the source code of the programs it manipulates, making it convenient for commercial and legacy applications. We will study the feasibility of an adaptive security agent that monitors target applications and deploys (or removes) countermeasures based on dynamic conditions. Lightweight monitoring is appropriate when the threat condition is low, heavy countermeasures will be dynamically woven into the code when an attack is detected. Vulnerability analysis will be based on advanced fuzzing. DBR makes it possible to monitor and modify deeply embedded variables, inaccessible to traditional monitoring systems, and also to detect unexpected/suspicious values taken by variables and act before the application crashes.

ARMOUR is funded by DGA (*Direction Générale de l'Armement*) and PEC (*Pôle d'Excellence Cyber*).

9.3. European Initiatives

9.3.1. FP7 & H2020 Projects

9.3.1.1. ANTAREX

Participants: Loïc Besnard, Imane Lasri, Erven Rohou.

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

Program: H2020

Duration: September 2015 – November 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.

9.3.1.2. ARGO

Participants: Imen Fassi, Damien Hardy, Isabelle Puaut.

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

Program: H2020

Type: RIA

Duration: Jan 2016 – Mar 2019

Coordinator: Karlsruhe Institut für Technologie (Germany)

Université de Rennes 1 contact: Steven Derrien

Partners:

Karlsruher Institut für Technologie (Germany)

SCILAB enterprises SAS (France)

Université de Rennes 1 (France)

Technologiko Ekpaideftiko Idryma (TEI) Dytikis Elladas (Greece)

Absint GmbH (Germany)

Deutsches Zentrum für Luft- und Raumfahrt EV (Germany)

Fraunhofer (Germany)

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.

9.3.1.3. 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.

9.3.1.4. Eurolab-4-HPC

Participant: Erven Rohou.

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

Program: H2020

Duration: September 2018 – September 2020

Coordinator: Chalmers Tekniska Hoegskola AB (Sweden)

Partners:

Barcelona Supercomputing Center - Centro Nacional de Supercomputacion (Spain)

Chalmers Tekniska Hoegskola (Sweden)

Foundation for Research and Technology Hellas (Greece)

Universität Stuttgart (Germany)

The University of Manchester (United Kingdom)

Inria (France)

Universität Augsburg (Germany)

ETH Zürich (Switzerland)

École Polytechnique Federale de Lausanne (Switzerland)

Technion - Israel Institute of Technology (Israel)

The University of Edinburgh (United Kingdom)

Rheinisch-Westfaelische Technische Hochschule Aachen (Germany)

Universiteit Gent (Belgium)

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 long-term 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.

9.4. International Initiatives

9.4.1. ANR CHIST-ERA SECODE 2016–2018

Participants: Damien Hardy, Nicolas Kiss, Erven Rohou.

Title: SECODE – Secure Codes to Thwart Cyber-Physical Attacks

CHIST-ERA - RTCPS

Duration: January 2016 – December 2018

Coordinator: Télécom Paris Tech (France)

Partners:

Télécom Paris Tech (France)

Inria (France)

Université Paris 8 (France)

Sabancı Üniversitesi (Turkey)

Université Catholique de Louvain (Belgium)

Inria contact: 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.

9.5. International Research Visitors

9.5.1. Visits of International Scientists

9.5.1.1. Internships

Caio de Lima and Marcos Siraichi, both from Universidade Federal de Minas Gerais (Brazil), visited PACAP for internships:

- Caio de Lima: Jan 9 – Apr 5;
- Marcos Siraichi: Dec 15 2017 – Mar 3 and Jul 16 – Oct 13.

9.5.2. Visits to International Teams

André Seznec visited Intel Microprocessor Research Labs at Bangalore (India) from 24th to 28th of September.

10. Dissemination

10.1. Promoting Scientific Activities

10.1.1. Scientific Events Organisation

10.1.1.1. General Chair, Scientific Chair

Isabelle Puaut is General Chair of the 2018 IEEE Real-Time Systems Symposium (RTSS), held in Nashville, Tennessee (USA) in December 2018.

10.1.2. Scientific Events Selection

10.1.2.1. Member of the Conference Program Committees

- Sylvain Collange was PC member of DATE 2018 and Compas'2018.
- Pierre Michaud was a member of the program committees of the ICCD 2018 and HPCA 2019 conferences.
- Isabelle Puaut was a member of the program committee of the Euromicro Conference on Real Time Systems (ECRTS) 2018 and 2019, IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS) 2018.
- Isabelle Puaut was a member of the program committee of the “Real-Time and (Networked) Embedded Systems” track of IEEE ETFA 2019.
- Isabelle Puaut was a member of the program committee of the 26th International Conference on Real-Time Networks and Systems, held in Poitiers, October 2018.
- Isabelle Puaut was a member of the program committee of the 18th Workshop on Worst-Case Execution Time Analysis (WCET 2018), held in Barcelona, Spain, July 2018.

- Erven Rohou was a member of the program committee of the International Symposium on Code Generation and Optimization (CGO) 2019.
- Erven Rohou was a member of the program committee of the following international workshops: Euro-EDUPAR, ANDARE, REV-A.
- André Seznec was a member of the ICCD 2018 program committee.

10.1.2.2. Reviewer

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

10.1.3. Journal

10.1.3.1. Member of the Editorial Boards

- Isabelle Puaut is Associate Editor of IEEE Transactions on Computers (IEEE TC) and Springer International Journal of Time-Critical Computing Systems.
- André Seznec is a member of the editorial boards of IEEE Micro and ACM Transactions on Architecture and Compiler Optimization.

10.1.3.2. Reviewer - Reviewing Activities

Members of PACAP routinely review submissions to numerous international journals.

10.1.4. Invited Talks

Members of the PACAP team were invited to present their activity at the RISC-V seminar organized by CEA Grenoble.

André Seznec was an invited speaker at the PER' 18 workshop in Gothenburg, Sweden, May 2018.

André Seznec was an invited speaker at the WOS'18 workshop in Rennes December 2018.

10.1.5. 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).

Isabelle Puaut is member of the steering committee of the Euromicro Conference on Real Time Systems (ECRTS).

10.1.6. Research Administration

Sylvain Collange is a member of the CUMIR (*Commission "Utilisateurs des moyens informatiques Recherche"*).

Isabelle Puaut is member of the Research Council (*Commission Recherche*) of the Université de Rennes 1. 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 Université de Rennes 1).

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

10.2. Teaching - Supervision - Juries

10.2.1. Teaching

- Licence: D. Hardy, Real-time systems, 68 hours, L3, Université de Rennes 1, France
- Master: D. Hardy, Operating systems, 53 hours, M1, Université de Rennes 1, France
- Master: I. Puaut, Operating systems: concepts and system programming under Linux (SEL), 75 hours, M1, Université de Rennes 1, France

- Master: I. Puaut, Operating systems kernels (NOY), 30 hours, M1, Université de Rennes 1, France
- Master: I. Puaut, Real-time systems, 55 hours, M1, Université de Rennes 1, France
- Master: I. Puaut, Research-oriented student project, 24 hours, M1, Université de Rennes 1, France
- Master: I. Puaut, Optimizing and Parallelizing Compilers (OPC), 9 hours, M2, Université de Rennes 1, France
- Master: I. Puaut, Writing of scientific publications, 9 hours, M2 and PhD students, Université de Rennes 1, France
- Master: A. Seznec, Advanced Design and Architectures, 12 hours, M2 SIF, Université de Rennes 1.
- Master: S. Collange, Parallel Programming, 22 hours, M1, Université de Rennes 1, France
- Master: S. Collange, GPU programming, 32 hours, M2, ESIR, France
- Master: S. Collange, Advanced computer architecture, 4 hours, M2, Université de Rennes 1, France
- Master: S. Collange, Advanced CUDA programming, 8 hours, M2, Sorbonne Universités, France

10.2.2. Supervision

PhD: Viet Anh Nguyen, Worst-Case Execution Time (WCET) Estimation for Many-core Architectures, Université de Rennes 1, Feb 2018, advisors I. Puaut (50 %) and D. Hardy (50 %)

Benjamin Rouxel, Code optimizations for WCET calculation on many-core platforms, Dec 2018, advisors I. Puaut (70 %) and S. Derrien from the CAIRN group (30 %).

PhD : Rabab Bouziane, Software-level Analysis and Optimization to Mitigate the Cost of Write Operations on Non-Volatile Memories, Université de Rennes 1, Dec 2018, advisors E. Rohou (70 %) et A. Gamatié from LIRMM (30 %)

PhD : Simon Rokicki, Accélération matérielle pour la traduction dynamique de programmes binaires, Université de Rennes 1, 17 Dec 2018, advisors S. Derrien from CAIRN (70 %) et E. Rohou (30 %)

PhD in progress : Kévin Le Bon, Dynamic Binary Analysis and Optimization for Cyber-Security, started Dec 2018, advisors E. Rohou (30 %), G. Hiet from CIDRE (35 %), F. Tronel from CIDRE (35 %)

PhD in progress : Bahram Yarahmadi, Compiler Optimizations and Worst-Case Energy Consumption, started Feb 2018, advisor E. Rohou

PhD in progress: Arif Ali Ana-Pparakkal, *Dynamic Function Specialization*, Université de Rennes 1, started Feb 2015, advisor E. Rohou

PhD in progress: Kleovoulos Kalaitzidis, Ultrawide Issue Superscalar Processors, Université de Rennes 1, started Dec 2016, advisor A. Seznec

PhD in progress: Niloofar Charmchi, Hardware prefetching and related issues, Université de Rennes 1, started Jan 2017, advisor A. Seznec and S. Collange

PhD in progress: Daniel Rodrigues Carvalho, Towards a compressed memory hierarchy, Université de Rennes 1, started Oct 2017, advisor A. Seznec

10.2.3. Juries

Isabelle Puaut was a member of the following hiring committees (comités de sélection):

- assistant professor position at Université de Rennes 1 on cybersecurity.
- assistant professor at Université de Bretagne Occidentale on cybersecurity in real-time systems
- assistant professor at Université de Nantes on software and hardware for embedded systems

Erven Rohou was a *special expert* for admittance of A. Jimborean as Associate Professor (Swedish *docent*) in Computer Science (Uppsala, Sweden).

Isabelle Puaut was a member of the following committees:

- Mohamed Said Mosli Bouksiaa, Performance variation considered helpful, Université de Paris Saclay, April 2018 (jury member)
- Risat Mahmud Pathan, Design and analysis of real-time parallel and distributed systems. Chalmers University of Technology, Sweden, Oavlönad Docent, Swedish equivalent to HdR, external reviewer.

Erven Rohou was a member of the following committees:

- Antoine Faravelon, Acceleration of memory accesses in Dynamic Binary translation, Université Grenoble Alpes, Oct 2018 (reviewer).

10.3. Popularization

10.3.1. Internal or external Inria responsibilities

Erven Rohou is "correspondant scientifique des relations internationales" for Inria Rennes Bretagne Atlantique. As such he is a member of the Inria COST GTRI (Groupe de Travail "Relations Internationales").

10.3.2. Articles and contents

PACAP contributed to Inria's white book on cybersecurity.

10.3.3. Education

Erven Rohou was invited to present the life of a researcher in computer science to middle school students (*Collège de Cesson-Sévigné)*

10.3.4. Interventions

We welcomed a student, grade of 3e (middle-school), for her 3-day observation stay to discover the daily life of researchers in computer science.

11. 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. 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

[5] 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

[6] S. KALATHINGAL, S. COLLANGE, B. SWAMY, A. SEZNEC. *DITVA: Dynamic Inter-Thread Vectorization Architecture*, in "Journal of Parallel and Distributed Computing", October 2018, pp. 1-32 [*DOI :* 10.1016/J.JPDC.2017.11.006], https://hal.archives-ouvertes.fr/hal-01655904

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

[8] 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

[9] 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

[10] 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

[11] 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

[12] 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

[13] 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

[14] S. SARDASHTI, A. SEZNEC, D. A. WOOD. *Yet Another Compressed Cache: a Low Cost Yet Effective Compressed Cache*, in "ACM Transactions on Architecture and Code Optimization", September 2016, 25 p. , https://hal.inria.fr/hal-01354248

[15] 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

[16] 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

[17] 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

[18] 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, December 2011, https://hal.inria.fr/hal-00639193

[19] 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

Publications of the year

Doctoral Dissertations and Habilitation Theses

[20] R. BOUZIANE. *Software-level Analysis and Optimization to Mitigate the Cost of Write Operations on Non-Volatile Memories*, Université de Rennes 1 [UR1], December 2018, https://hal.inria.fr/tel-01954076

[21] V. A. NGUYEN. *Cache-conscious Off-Line Real-Time Scheduling for Multi-Core Platforms: Algorithms and Implementation*, Université de Rennes 1 [UR1], February 2018, https://hal.inria.fr/tel-01933422

[22] S. ROKICKI. *Hardware Accelerated Dynamic Binary Translation*, Université de Rennes 1 [UR1], December 2018, https://hal.archives-ouvertes.fr/tel-01959136

[23] B. ROUXEL. *Minimising communication costs impact when scheduling real-time applications on multi-core architectures*, Université de Rennes 1, December 2018, https://hal.inria.fr/tel-01945456

Articles in International Peer-Reviewed Journals

[24] S. KALATHINGAL, S. COLLANGE, B. SWAMY, A. SEZNEC. *DITVA: Dynamic Inter-Thread Vectorization Architecture*, in "Journal of Parallel and Distributed Computing", October 2018, pp. 1-32 [*DOI :* 10.1016/J.JPDC.2017.11.006], https://hal.archives-ouvertes.fr/hal-01655904

[25] P. MICHAUD. *An Alternative TAGE-like Conditional Branch Predictor*, in "ACM Transactions on Architecture and Code Optimization", May 2018, vol. 15, n^{o} 3, pp. 1-24 [*DOI :* 10.1145/3226098], https://hal.inria.fr/hal-01799442

[26] S. ROKICKI, E. ROHOU, S. DERRIEN. *Hybrid-DBT: Hardware/Software Dynamic Binary Translation Targeting VLIW*, in "IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems", August 2018, pp. 1-14 [*DOI :* 10.1109/TCAD.2018.2864288], https://hal.archives-ouvertes.fr/hal-01856163

[27] M. YUSUF, A. EL-MAHDY, E. ROHOU. *Runtime, Speculative On-Stack Parallelization of For-Loops in Binary Programs*, in "IEEE Letters of the Computer Society", October 2018, pp. 1-4 [*DOI :* 10.1109/LOCS.2018.2872454], https://hal.inria.fr/hal-01890719

Invited Conferences

[28] C. SILVANO, G. AGOSTA, A. BARTOLINI, A. R. BECCARI, L. BENINI, L. BESNARD, J. BISPO, R. CMAR, J. M. R. CARDOSO, C. CAVAZZONI, S. CHERUBIN, D. GADIOLI, M. GOLASOWSKI, I. LASRI, J. MARTINOVIČ, G. PALERMO, P. PINTO, E. ROHOU, N. SANNA, K. SLANINOVÁ, E. VITALI. *ANTAREX: A DSL-based Approach to Adaptively Optimizing and Enforcing Extra-Functional Properties in High Performance Computing*, in "Euromicro DSD/SEEA 2018", Prague, Czech Republic, August 2018, pp. 1-8, https://hal.inria.fr/hal-01890152

[29] C. Silvano, G. Palermo, G. Agosta, A. H. Ashouri, D. Gadioli, S. Cherubin, E. Vitali, L. Benini, A. Bartolini, D. Cesarini, J. Cardoso, J. Bispo, P. Pinto, R. Nobre, E. Rohou, L. Besnard, I. Lasri, N. Sanna, C. Cavazzoni, R. Cmar, J. Martinovič, K. Slaninová, M. Golasowski, A. R. Beccari, C. Manelfi. *Autotuning and Adaptivity in Energy Efficient HPC Systems: The ANTAREX Toolbox*, in "CF 2018 - 15th ACM International Conference on Computing Frontiers", Ischia, Italy, ACM, May 2018, pp. 270-275 [*DOI :* 10.1145/3203217.3205338], https://hal.inria.fr/hal-01932706

International Conferences with Proceedings

[30] A. A. Ap, K. Le Bon, B. Hawkins, E. Rohou. *FITTCHOOSER: A Dynamic Feedback-Based Fittest Optimization Chooser*, in "HPCS 2018 - 16th International Conference on High Performance Computing & Simulation - Special Session on Compiler Architecture, Design and Optimization", Orléans, France, July 2018, pp. 1-8, https://hal.inria.fr/hal-01808658

[31] R. Bouziane, E. Rohou, A. Gamatié. *Compile-Time Silent-Store Elimination for Energy Efficiency: an Analytic Evaluation for Non-Volatile Cache Memory*, in "RAPIDO: Rapid Simulation and Performance Evaluation", Manchester, United Kingdom, ACM, January 2018, pp. 1-8 [*DOI :* 10.1145/3180665.3180666], https://hal.inria.fr/hal-01660686

[32] R. Bouziane, E. Rohou, A. Gamatié. *Energy-Efficient Memory Mappings based on Partial WCET Analysis and Multi-Retention Time STT-RAM*, in "RTNS: Real-Time Networks and Systems", Poitiers, France, October 2018, pp. 148-158 [*DOI :* 10.1145/3273905.3273908], https://hal.inria.fr/hal-01871320

[33] T. Lefeuvre, E. K. Kasnakli, I. Fassi, I. Puaut, C. Cullmann, S. Derrien, G. Gebhard. *Using Polyhedral Techniques to Tighten WCET Estimates of Optimized Code: A Case Study with Array Contraction*, in "DATE 2018 - Design Automation and Test Europe", Dresden, Germany, IEEE, March 2018, pp. 925-930 [*DOI :* 10.23919/DATE.2018.8342142], https://hal.inria.fr/hal-01815499

[34] B. Panda, A. Seznec. *Synergistic Cache Layout For Reuse and Compression*, in "PACT '18 - International conference on Parallel Architectures and Compilation Techniques", Limassol, Cyprus, November 2018, pp. 1-13 [*DOI :* 10.1145/3243176.3243178], https://hal.inria.fr/hal-01888880

[35] *Best Paper*
A. Perais, A. Seznec. *Cost Effective Speculation with the Omnipredictor*, in "International conference on Parallel Architectures and Compilation Techniques (PACT '18),", Limassol, Cyprus, November 2018 [*DOI :* 10.1145/3243176.3243208], https://hal.inria.fr/hal-01888884.

[36] I. Puaut, M. Dardaillon, C. Cullmann, G. Gebhard, S. Derrien. *Fine-Grain Iterative Compilation for WCET Estimation*, in "WCET 2018 - 18th International Workshop on Worst-Case Execution Time Analysis", Barcelona, Spain, July 2018, pp. 1-12 [*DOI :* 10.4230/OASIcs.WCET.2018.9], https://hal.inria.fr/hal-01889944

[37] S. Rokicki, E. Rohou, S. Derrien. *Supporting Runtime Reconfigurable VLIWs Cores Through Dynamic Binary Translation*, in "DATE 2018 - IEEE/ACM Design, Automation & Test in Europe Conference & Exhibition", Dresden, Germany, IEEE, March 2018, pp. 1009-1014 [*DOI :* 10.23919/DATE.2018.8342160], https://hal.archives-ouvertes.fr/hal-01653110

[38] S. Rokicki, E. Rohou, S. Derrien. *Aggressive Memory Speculation in HW/SW Co-Designed Machines*, in "DATE 2019 - IEEE/ACM Design, Automation and Test in Europe", Florence, Italy, March 2019, https://hal.archives-ouvertes.fr/hal-01941876

[39] *Best Paper*
A. Seznec. *Exploring value prediction with the EVES predictor*, in "CVP-1 2018 - 1st Championship Value Prediction", Los Angeles, United States, June 2018, pp. 1-6, https://hal.inria.fr/hal-01888864.

[40] M. Y. Siraichi, V. F. d. Santos, S. Collange, F. M. Quintão 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

Conferences without Proceedings

[41] R. Bouziane, E. Rohou, A. Gamatié. *Partial Worst-Case Execution Time Analysis*, in "ComPAS: Conférence en Parallélisme, Architecture et Système", Toulouse, France, July 2018, pp. 1-8, https://hal.inria.fr/hal-01803006

Other Publications

[42] G. Berthou, A. Carer, H.-P. Charles, S. Derrien, K. Marquet, I. Miro-Panades, D. Pala, I. Puaut, F. Rastello, T. Risset, E. Rohou, G. Salagnac, O. Sentieys, B. Yarahmadi. *The Inria ZEP project: NVRAM and Harvesting for Zero Power Computations*, March 2018, 10th Annual Non-Volatile Memories Workshop (NVMW), Poster, https://hal.inria.fr/hal-01941766

References in notes

[43] 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

[44] 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

[45] S. Nassif, N. Mehta, Y. Cao. *A resilience roadmap*, in "Design, Automation Test in Europe Conference Exhibition (DATE), 2010", March 2010, pp. 1011-1016

[46] 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

[47] B. Ransford, B. Lucia. *Nonvolatile memory is a broken time machine*, in "Proceedings of the workshop on Memory Systems Performance and Correctness", ACM, 2014, 5 p.

[48] 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

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