RESEARCH CENTRE

Bordeaux - Sud-Ouest

IN PARTNERSHIP WITH:

Institut Polytechnique de Bordeaux,
Université de Bordeaux

2021
ACTIVITY REPORT

Project-Team
STORM

STatic Optimizations, Runtime Methods

IN COLLABORATION WITH: Laboratoire Bordelais de Recherche en Informatique (LaBRI)

DOMAIN

Networks, Systems and Services, Distributed Computing

THEME

Distributed and High Performance Computing

Contents

Project-Team STORM

Creation of the Project-Team: 2017 July 01

Keywords

Computer sciences and digital sciences

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

A2.1.7. – Distributed programming

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...

Other research topics and application domains

B2.2.1. – Cardiovascular and respiratory diseases

B3.2. – Climate and meteorology

B3.3.1. – Earth and subsoil

B3.4.1. – Natural risks

B4.2. – Nuclear Energy Production

B5.2.3. – Aviation

B5.2.4. – Aerospace

B6.2.2. – Radio technology

B6.2.3. – Satellite technology

B6.2.4. – Optic technology

B9.2.3. – Video games

1 Team members, visitors, external collaborators

Research Scientists

- Olivier Aumage [Inria, Researcher, HDR]
- Scott Baden [Inria, Chair, from Oct 2021 until Nov 2021]
- Amina Guermouche [Inria, Advanced Research Position, from May 2021 until Jul 2021]
- Laércio Lima Pilla [CNRS, Researcher]
- Mihail Popov [Inria, Starting Faculty Position]
- Emmanuelle Saillard [Inria, Researcher]

Faculty Members

- Denis Barthou [Team leader, Institut National Polytechnique de Bordeaux, Professor, HDR]
- Marie-Christine Counilh [Univ de Bordeaux, Associate Professor]
- Raymond Namyst [Univ de Bordeaux, Professor, HDR]
- Samuel Thibault [Univ de Bordeaux, Professor, HDR]
- Pierre-André Wacrenier [Univ de Bordeaux, Associate Professor]

Post-Doctoral Fellow

- Adrien Cassagne [Bordeaux INP, until Aug 2021, ATER]

PhD Students

- Celia Ait Kaci Tassadit [Bull]
- Paul Beziau [CEA, until December 2021]
- Baptiste Coye [UBISOFT, CIFRE]
- Idriss Daoudi [Inria, until Sep 2021]
- Maxime Gonthier [Inria, from Sep 2021]
- Romain Lion [Inria]
- Gwenole Lucas [Inria]
- Van Man Nguyen [CEA]

Technical Staff

- Nathalie Furmento [CNRS, Engineer, Permanent position]
- Amina Guermouche [Univ de Bordeaux, Engineer, from Aug 2021]
- Kun He [Inria, Engineer]
- Mariem Makni [Inria, Engineer, until Jul 2021]
- Chiheb Sakka [Inria, Engineer]
- Bastien Tagliaro [Inria, Engineer, from Oct 2021]

Interns and Apprentices

- Vincent Alba [Inria, from May 2021 until Jul 2021]
- Edgar Baucher [Inria, from May 2021 until Jul 2021]
- Charly Castes [Inria, from Feb 2021 until Aug 2021]
- Mael Keryell [Inria, from Feb 2021 until Jun 2021]
- Mustapha Regragui [Inria, from May 2021 until Sep 2021]
- Pierre Antoine Rouby [Inria, from May 2021 until Jul 2021]
- Lana Scravaglieri [Inria, from May 2021 until Aug 2021]

Administrative Assistant

- Sabrina Duthil [Inria]

External Collaborators

- Scott Baden [Université de Californie, until Sep 2021]
- Hugo Brunie [Université de Californie, until Mar 2021]
- Jean-Marie Couteyen [Airbus]
- Amina Guermouche [Telecom SudParis, until Apr 2021]

2 Overall objectives

Runtime systems successfully support the complexity and heterogeneity of modern architectures thanks to their dynamic task management. Compiler optimizations and analyses are aggressive in iterative compilation frameworks, suitable for library generations or domain specific languages (DSL), in particular for linear algebra methods. To alleviate the difficulties for programming heterogeneous and parallel machines, we believe it is necessary to provide inputs with richer semantics to runtime and compiler alike, and in particular by combining both approaches.

This general objective is declined into three sub-objectives, the first concerning the expression of parallelism itself, the second the optimization and adaptation of this parallelism by compilers and runtimes and the third concerning the necessary user feed back, either as debugging or simulation results, to better understand the first two steps.

1. Expressing parallelism: As shown in the following figure, we propose to work on parallelism expression through Domain Specific Languages, PGAS languages, C++ enhanced with libraries or even pragmas able to capture the essence of the algorithms used through usual parallel languages such as SyCL, OpenMP and through high performance libraries. The language richer semantics will be driven by applications, with the idea to capture at the algorithmic level the parallelism of the problem and perform dynamic data layout adaptation, parallel and algorithmic optimizations. The principle here is to capture a higher level of semantics, enabling users to express not only parallelism but also different algorithms.
2. Optimizing and adapting parallelism: The goal is to address the evolving hardware, by providing mechanisms to efficiently run the same code on different architectures. This implies to adapt parallelism to the architecture by either changing the granularity of the work or by adjusting the execution parameters. We rely on the use of existing parallel libraries and their composition, and more generally on the separation of concern between the description of tasks, that represent semantic units of work, and the tasks to be executed by the different processing units. Splitting or coarsening moldable tasks, generating code for these tasks, and exploring runtime parameters (e.g., frequency, vectorization, prefetching, scheduling) is part of this work.

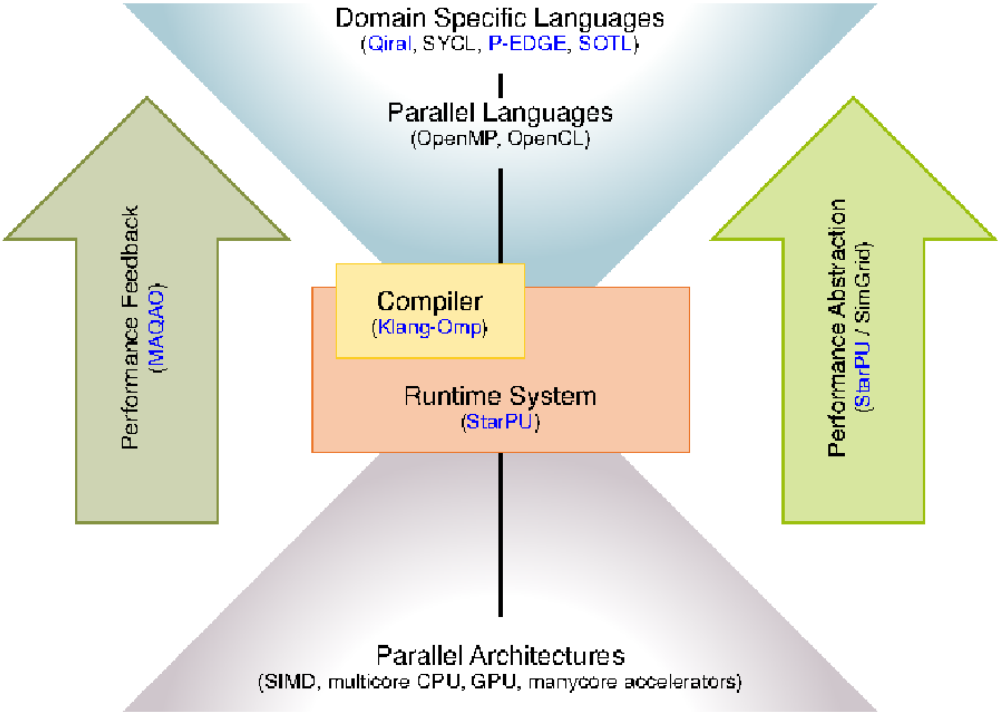

Figure 1: STORM Big Picture

3. Finally, the abstraction we advocate for requires to propose a feed back loop. This feed back has two objectives: to make users better understand their application and how to change the expression of parallelism if necessary, but also to propose an abstracted model for the machine. This allows to develop and formalize the compilation, scheduling techniques on a model, not too far from the real machine. Here, simulation techniques are a way to abstract the complexity of the architecture while preserving essential metrics.

3 Research program

3.1 Parallel Computing and Architectures

Following the current trends of the evolution of HPC systems architectures, it is expected that future Exascale systems (i.e. Sustaining 10^{18} flops) will have millions of cores. Although the exact architectural details and trade-offs of such systems are still unclear, it is anticipated that an overall concurrency level of $O(10^9)$ threads/tasks will probably be required to feed all computing units while hiding memory latencies. It will obviously be a challenge for many applications to scale to that level, making the underlying system sound like "embarrassingly parallel hardware."

From the programming point of view, it becomes a matter of being able to expose extreme parallelism within applications to feed the underlying computing units. However, this increase in the number of cores also comes with architectural constraints that actual hardware evolution prefigures: computing units will feature extra-wide SIMD and SIMT units that will require aggressive code vectorization or "SIMDization", systems will become hybrid by mixing traditional CPUs and accelerators units, possibly on the same chip as the AMD APU solution, the amount of memory per computing unit is constantly decreasing, new levels of memory will appear, with explicit or implicit consistency management, etc. As a result, upcoming extreme-scale system will not only require unprecedented amount of parallelism to be efficiently exploited, but they will also require that applications generate adaptive parallelism capable to map tasks over heterogeneous computing units.

The current situation is already alarming, since European HPC end-users are forced to invest in a difficult and time-consuming process of tuning and optimizing their applications to reach most of current supercomputers' performance. It will go even worse with the emergence of new parallel architectures (tightly integrated accelerators and cores, high vectorization capabilities, etc.) featuring unprecedented degree of parallelism that only too few experts will be able to exploit efficiently. As highlighted by the ETP4HPC initiative, existing programming models and tools won't be able to cope with such a level of heterogeneity, complexity and number of computing units, which may prevent many new application opportunities and new science advances to emerge.

The same conclusion arises from a non-HPC perspective, for single node embedded parallel architectures, combining heterogeneous multicores, such as the ARM big.LITTLE processor and accelerators such as GPUs or DSPs. The need and difficulty to write programs able to run on various parallel heterogeneous architectures has led to initiatives such as HSA, focusing on making it easier to program heterogeneous computing devices. The growing complexity of hardware is a limiting factor to the emergence of new usages relying on new technology.

3.2 Scientific and Societal Stakes

In the HPC context, simulation is already considered as a third pillar of science with experiments and theory. Additional computing power means more scientific results, and the possibility to open new fields of simulation requiring more performance, such as multi-scale, multi-physics simulations. Many scientific domains able to take advantage of Exascale computers, these "Grand Challenges" cover large panels of science, from seismic, climate, molecular dynamics, theoretical and astrophysics physics... Besides, more widespread compute intensive applications are also able to take advantage of the performance increase at the node level. For embedded systems, there is still an on-going trend where dedicated hardware is progressively replaced by off-the-shelf components, adding more adaptability and lowering the cost of devices. For instance, Error Correcting Codes in cell phones are still hardware chips, but new software and adaptative solutions relying on low power multicores are also explored for antenna. New usages are also appearing, relying on the fact that large computing capacities are becoming more affordable

and widespread. This is the case for instance with Deep Neural Networks where the training phase can be done on supercomputers and then used in embedded mobile systems. Even though the computing capacities required for such applications are in general a different scale from HPC infrastructures, there is still a need in the future for high performance computing applications.

However, the outcome of new scientific results and the development of new usages for these systems will be hindered by the complexity and high level of expertise required to tap the performance offered by future parallel heterogeneous architectures. Maintenance and evolution of parallel codes are also limited in the case of hand-tuned optimization for a particular machine, and this advocates for a higher and more automatic approach.

3.3 Towards More Abstraction

As emphasized by initiatives such as the European Exascale Software Initiative (EESI), the European Technology Platform for High Performance Computing (ETP4HPC), or the International Exascale Software Initiative (IESP), the HPC community needs new programming APIs and languages for expressing heterogeneous massive parallelism in a way that provides an abstraction of the system architecture and promotes high performance and efficiency. The same conclusion holds for mobile, embedded applications that require performance on heterogeneous systems.

This crucial challenge given by the evolution of parallel architectures therefore comes from this need to make high performance accessible to the largest number of developers, abstracting away architectural details providing some kind of performance portability, and provided a high level feed-back allowing the user to correct and tune the code. Disruptive uses of the new technology and groundbreaking new scientific results will not come from code optimization or task scheduling, but they require the design of new algorithms that require the technology to be tamed in order to reach unprecedented levels of performance.

Runtime systems and numerical libraries are part of the answer, since they may be seen as building blocks optimized by experts and used as-is by application developers. The first purpose of runtime systems is indeed to provide *abstraction*. Runtime systems offer a uniform programming interface for a specific subset of hardware or low-level software entities (e.g., POSIX-thread implementations). They are designed as thin user-level software layers that complement the basic, general purpose functions provided by the operating system calls. Applications then target these uniform programming interfaces in a portable manner. Low-level, hardware dependent details are hidden inside runtime systems. The adaptation of runtime systems is commonly handled through drivers. The abstraction provided by runtime systems thus enables portability. Abstraction alone is however not enough to provide portability of performance, as it does nothing to leverage low-level-specific features to get increased performance and does nothing to help the user tune his code. Consequently, the second role of runtime systems is to *optimize* abstract application requests by dynamically mapping them onto low-level requests and resources as efficiently as possible. This mapping process makes use of scheduling algorithms and heuristics to decide the best actions to take for a given metric and the application state at a given point in its execution time. This allows applications to readily benefit from available underlying low-level capabilities to their full extent without breaking their portability. Thus, optimization together with abstraction allows runtime systems to offer portability of performance. Numerical libraries provide sets of highly optimized kernels for a given field (dense or sparse linear algebra, tensor products, etc.) either in an autonomous fashion or using an underlying runtime system.

Application domains cannot resort to libraries for all codes however, computation patterns such as stencils are a representative example of such difficulty. The compiler technology plays here a central role, in managing high level semantics, either through templates, domain specific languages or annotations. Compiler optimizations, and the same applies for runtime optimizations, are limited by the level of semantics they manage and the optimization space they explore. Providing part of the algorithmic knowledge of an application, and finding ways to explore a larger space of optimization would lead to more opportunities to adapt parallelism, memory structures, and is a way to leverage the evolving hardware. Compilers and runtime play a crucial role in the future of high performance applications, by defining the input language for users, and optimizing/transforming it into high performance code. Adapting the parallelism and its orchestration according to the inputs, to energy, to faults, managing heterogeneous memory, better define and select appropriate dynamic scheduling methods, are among

the current works of the STORM team.

The results of the team research in 2021 reflect this focus. Results presented in Sections 8.19, 8.2, 8.7 correspond to efforts for higher abstractions through C++ or PGAS, and for decoupling algorithmics from parallel optimizations. Static and dynamic optimizations are presented in 8.8, 8.9, 8.1, 8.5, 8.3 , 8.4, as well as 8.6 for a more efficient exploration. Section 8.7 correspond to application of previously developped techniques and tools for vectorization Results described in Sections 8.8 and 8.9 provide feedback information, through visualization and error detection for parallel executions. The work described in Sections 8.13, 8.10, 8.20, 8.11 and 8.17 focus in particular on StarPU and its development in order to better abstract architecture, resilience and optimizations. The works described in Sections 8.15 and 8.16 correspond to scheduling methods for lightweight tasks or repetitive task graphs.

Finally, Section 8.18 present an on-going effort on improving the Chameleon library and strengthening its relation with StarPU and the NewMadeleine communication library. They represent real-life applications for the runtime methods we develop. Section 8.20 presents application to bigdata application, and 8.7 to cardiac simulation.

4 Application domains

4.1 Application domains benefiting from HPC

The application domains of this research are the following:

- Bioinformatics
- Environment, in particular CO_2 capture (see Exa2PRO, 10.2.1)
- Health and heart disease analysis (see EXACARD 10.3.1 and Microcard project projects 10.2.1)
- Software infrastructures for Telecommunications (see AFF3CT, 8.19)
- Aeronautics (collaboration with Airbus, J.-M. Couteyen)
- Video games (collaboration with Ubisoft, see 9.1)

4.2 Application in High performance computing/Big Data

Most of the research of the team has application in the domain of software infrastructure for HPC and compute intensive applications.

5 Social and environmental responsibility

5.1 Impact of research results

The research performed in the context of the EXA2PRO project improves the performance of a CO_2 capture appplication, which consequently allows to improve the performance of CO_2 capture material and process.

6 Highlights of the year

- Mihail Popov, Inria researcher (Inria Starting Faculty Position) arrived in the team in Jan. 2021.
- Laercio Lima Pilla, CNRS researcher arrived in the team in Jan. 2021.
- Samuel Thibault has been promoted Professor of the University of Bordeaux.
- ETP4HPC white papers "Task-Based Performance Portability in HPC", [online version].

7 New software and platforms

In the team, we focus essentially on software platforms.

7.1 New software

7.1.1 Chameleon

Keywords: Runtime system, Task-based algorithm, Dense linear algebra, HPC, Task scheduling

Scientific Description: Chameleon is part of the MORSE (Matrices Over Runtime Systems @ Exascale) project. The overall objective is to develop robust linear algebra libraries relying on innovative runtime systems that can fully benefit from the potential of those future large-scale complex machines.

We expect advances in three directions based first on strong and closed interactions between the runtime and numerical linear algebra communities. This initial activity will then naturally expand to more focused but still joint research in both fields.

1. Fine interaction between linear algebra and runtime systems. On parallel machines, HPC applications need to take care of data movement and consistency, which can be either explicitly managed at the level of the application itself or delegated to a runtime system. We adopt the latter approach in order to better keep up with hardware trends whose complexity is growing exponentially. One major task in this project is to define a proper interface between HPC applications and runtime systems in order to maximize productivity and expressivity. As mentioned in the next section, a widely used approach consists in abstracting the application as a DAG that the runtime system is in charge of scheduling. Scheduling such a DAG over a set of heterogeneous processing units introduces a lot of new challenges, such as predicting accurately the execution time of each type of task over each kind of unit, minimizing data transfers between memory banks, performing data prefetching, etc. Expected advances: In a nutshell, a new runtime system API will be designed to allow applications to provide scheduling hints to the runtime system and to get real-time feedback about the consequences of scheduling decisions.

2. Runtime systems. A runtime environment is an intermediate layer between the system and the application. It provides low-level functionality not provided by the system (such as scheduling or management of the heterogeneity) and high-level features (such as performance portability). In the framework of this proposal, we will work on the scalability of runtime environment. To achieve scalability it is required to avoid all centralization. Here, the main problem is the scheduling of the tasks. In many task-based runtime environments the scheduler is centralized and becomes a bottleneck as soon as too many cores are involved. It is therefore required to distribute the scheduling decision or to compute a data distribution that impose the mapping of task using, for instance the so-called “owner-compute” rule. Expected advances: We will design runtime systems that enable an efficient and scalable use of thousands of distributed multicore nodes enhanced with accelerators.

3. Linear algebra. Because of its central position in HPC and of the well understood structure of its algorithms, dense linear algebra has often pioneered new challenges that HPC had to face. Again, dense linear algebra has been in the vanguard of the new era of petascale computing with the design of new algorithms that can efficiently run on a multicore node with GPU accelerators. These algorithms are called “communication-avoiding” since they have been redesigned to limit the amount of communication between processing units (and between the different levels of memory hierarchy). They are expressed through Direct Acyclic Graphs (DAG) of fine-grained tasks that are dynamically scheduled. Expected advances: First, we plan to investigate the impact of these principles in the case of sparse applications (whose algorithms are slightly more complicated but often rely on dense kernels). Furthermore, both in the dense and sparse cases, the scalability on thousands of nodes is still limited, new numerical approaches need to be found. We will specifically design sparse hybrid direct/iterative methods that represent a promising approach.

Overall end point. The overall goal of the MORSE associate team is to enable advanced numerical algorithms to be executed on a scalable unified runtime system for exploiting the full potential of future exascale machines.

Functional Description: Chameleon is a dense linear algebra software relying on sequential task-based algorithms where sub-tasks of the overall algorithms are submitted to a Runtime system. A Runtime system such as StarPU is able to manage automatically data transfers between not shared memory area (CPUs-GPUs, distributed nodes). This kind of implementation paradigm allows to design high performing linear algebra algorithms on very different type of architecture: laptop, many-core nodes, CPUs-GPUs, multiple nodes. For example, Chameleon is able to perform a Cholesky factorization (double-precision) at 80 TFlop/s on a dense matrix of order 400 000 (i.e. 4 min 30 s).

Release Contributions: Chameleon includes the following features:

- BLAS 3, LAPACK one-sided and LAPACK norms tile algorithms - Support QUARK and StarPU runtime systems and PaRSEC since 2018 - Exploitation of homogeneous and heterogeneous platforms through the use of BLAS/LAPACK CPU kernels and cuBLAS/MAGMA CUDA kernels - Exploitation of clusters of interconnected nodes with distributed memory (using OpenMPI)

URL: https://gitlab.inria.fr/solverstack/chameleon

Contact: Emmanuel Agullo

Participants: Cédric Castagnede, Samuel Thibault, Emmanuel Agullo, Florent Pruvost, Mathieu Faverge

Partners: Innovative Computing Laboratory (ICL), King Abdullha University of Science and Technology, University of Colorado Denver

7.1.2 KStar

Name: The KStar OpenMP Compiler

Keywords: Source-to-source compiler, OpenMP, Task scheduling, Compilers, Data parallelism

Functional Description: The KStar software is a source-to-source OpenMP compiler for languages C and C++. The KStar compiler translates OpenMP directives and constructs into API calls from the StarPU runtime system or the XKaapi runtime system. The KStar compiler is virtually fully compliant with OpenMP 3.0 constructs. The KStar compiler supports OpenMP 4.0 dependent tasks and accelerated targets.

URL: https://gitlab.inria.fr/kstar/kastors

Publications: hal-01517153, hal-01372022, hal-01081974

Contact: Olivier Aumage

Participants: Nathalie Furmento, Olivier Aumage, Philippe Virouleau, Samuel Thibault

7.1.3 AFF3CT

Name: A Fast Forward Error Correction Toolbox

Keywords: High-Performance Computing, Signal processing, Error Correction Code

Functional Description: AFF3CT proposes high performance Error Correction algorithms for Polar, Turbo, LDPC, RSC (Recursive Systematic Convolutional), Repetition and RA (Repeat and Accumulate) codes. These signal processing codes can be parameterized in order to optimize some given metrics, such as Bit Error Rate, Bandwidth, Latency, ...using simulation. For the designers of such signal processing chain, AFF3CT proposes also high performance building blocks so to develop new algorithms. AFF3CT compiles with many compilers and runs on Windows, Mac OS X, Linux environments and has been optimized for x86 (SSE, AVX instruction sets) and ARM architectures (NEON instruction set).

News of the Year: The AFF3CT toolbox was successfully used to develop a the software implementation of real- time DVB-S2 transceiver. For this purpose, USRP modules were combined with multicore and SIMD CPUs. Thus some components are directly from the AFF3CT library and others such as the synchronization functions have been added. The transceiver code is portable on x86 and ARM architectures.

URL: https://aff3ct.github.io/

Publications: hal-02358306, hal-01965629, hal-01977885, hal-01203105, hal-01363980, hal-01363975, hal-01987848, hal-01965633

Authors: Adrien Cassagne, Bertrand Le Gal, Camille Leroux, Denis Barthou, Olivier Aumage

Contact: Denis Barthou

Partner: IMS

7.1.4 VITE

Name: Visual Trace Explorer

Keywords: Visualization, Execution trace

Functional Description: ViTE is a trace explorer. It is a tool made to visualize execution traces of large parallel programs. It supports Pajé, a trace format created by Inria Grenoble, and OTF and OTF2 formats, developed by the University of Dresden and allows the programmer a simpler way to analyse, debug and/or profile large parallel applications.

URL: https://solverstack.gitlabpages.inria.fr/vite/

Contact: Mathieu Faverge

Participant: Mathieu Faverge

7.1.5 PARCOACH

Name: PARallel Control flow Anomaly CHecker

Keywords: High-Performance Computing, Program verification, Debug, MPI, OpenMP, Compilation

Scientific Description: PARCOACH verifies programs in two steps. First, it statically verifies applications with a data- and control-flow analysis and outlines execution paths leading to potential deadlocks. The code is then instrumented, displaying an error and synchronously interrupting all processes if the actual scheduling leads to a deadlock situation.

Functional Description: Supercomputing plays an important role in several innovative fields, speeding up prototyping or validating scientific theories. However, supercomputers are evolving rapidly with now millions of processing units, posing the questions of their programmability. Despite the emergence of more widespread and functional parallel programming models, developing correct and effective parallel applications still remains a complex task. As current scientific applications mainly rely on the Message Passing Interface (MPI) parallel programming model, new hardwares designed for Exascale with higher node-level parallelism clearly advocate for an MPI+X solutions with X a thread-based model such as OpenMP. But integrating two different programming models inside the same application can be error-prone leading to complex bugs - mostly detected unfortunately at runtime. PARallel COntrol flow Anomaly CHecker aims at helping developers in their debugging phase.

URL: https://parcoach.github.io/index.html

Publications: hal-00920901, hal-01078762, hal-01078759, hal-01252321, hal-01253204, hal-01199718, hal-01420655, hal-01937316, hal-02390025

Contact: Emmanuelle Saillard

Participants: Emmanuelle Saillard, Denis Barthou, Pierre Huchant

Partner: CEA

7.1.6 StarPU

Name: The StarPU Runtime System

Keywords: Multicore, GPU, Scheduling, HPC, Performance

Scientific Description: Traditional processors have reached architectural limits which heterogeneous multicore designs and hardware specialization (eg. coprocessors, accelerators, ...) intend to address. However, exploiting such machines introduces numerous challenging issues at all levels, ranging from programming models and compilers to the design of scalable hardware solutions. The design of efficient runtime systems for these architectures is a critical issue. StarPU typically makes it much easier for high performance libraries or compiler environments to exploit heterogeneous multicore machines possibly equipped with GPGPUs or Cell processors: rather than handling low-level issues, programmers may concentrate on algorithmic concerns.Portability is obtained by the means of a unified abstraction of the machine. StarPU offers a unified offloadable task abstraction named "codelet". Rather than rewriting the entire code, programmers can encapsulate existing functions within codelets. In case a codelet may run on heterogeneous architectures, it is possible to specify one function for each architectures (eg. one function for CUDA and one function for CPUs). StarPU takes care to schedule and execute those codelets as efficiently as possible over the entire machine. In order to relieve programmers from the burden of explicit data transfers, a high-level data management library enforces memory coherency over the machine: before a codelet starts (eg. on an accelerator), all its data are transparently made available on the compute resource.Given its expressive interface and portable scheduling policies, StarPU obtains portable performances by efficiently (and easily) using all computing resources at the same time. StarPU also takes advantage of the heterogeneous nature of a machine, for instance by using scheduling strategies based on auto-tuned performance models.

StarPU is a task programming library for hybrid architectures.

The application provides algorithms and constraints: - CPU/GPU implementations of tasks, - A graph of tasks, using either the StarPU's high level GCC plugin pragmas or StarPU's rich C API.

StarPU handles run-time concerns: - Task dependencies, - Optimized heterogeneous scheduling, - Optimized data transfers and replication between main memory and discrete memories, - Optimized cluster communications.

Rather than handling low-level scheduling and optimizing issues, programmers can concentrate on algorithmic concerns!

Functional Description: StarPU is a runtime system that offers support for heterogeneous multicore machines. While many efforts are devoted to design efficient computation kernels for those architectures (e.g. to implement BLAS kernels on GPUs), StarPU not only takes care of offloading such kernels (and implementing data coherency across the machine), but it also makes sure the kernels are executed as efficiently as possible.

URL: https://starpu.gitlabpages.inria.fr/

Publications: hal-02403109, hal-02421327, hal-02872765, hal-02914793, hal-02933803, hal-01473475, hal-01474556, tel-01538516, hal-01718280, hal-01618526, tel-01816341, hal-01410103, hal-01616632, hal-01353962, hal-01842038, hal-01181135, tel-01959127, hal-01355385, hal-01284004, hal-01502749, hal-01332774, hal-01372022, tel-01483666, hal-01147997, hal-01182746, hal-01120507, hal-01101045, hal-01081974, hal-01101054, hal-01011633, hal-01005765, hal-01283949, hal-00987094, hal-00978364, hal-00978602, hal-00992208, hal-00966862, hal-00925017, hal-00920915, hal-00824514, hal-00926144, hal-00773610, hal-01284235, hal-00853423, hal-00807033, tel-00948309, hal-00772742, hal-00725477,

hal-00773114, hal-00697020, hal-00776610, hal-01284136, inria-00550877, hal-00648480, hal-00661320, inria-00606200, hal-00654193, inria-00547614, hal-00643257, inria-00606195, hal-00803304, inria-00590670, tel-00777154, inria-00619654, inria-00523937, inria-00547616, inria-00467677, inria-00411581, inria-00421333, inria-00384363, inria-00378705, hal-01517153, tel-01162975, hal-01223573, hal-01361992, hal-01386174, hal-01409965, hal-02275363, hal-02296118

Contact: Olivier Aumage

Participants: Corentin Salingue, Andra Hugo, Benoît Lize, Cédric Augonnet, Cyril Roelandt, François Tessier, Jérôme Clet-Ortega, Ludovic Courtes, Ludovic Stordeur, Marc Sergent, Mehdi Juhoor, Nathalie Furmento, Nicolas Collin, Olivier Aumage, Pierre Wacrenier, Raymond Namyst, Samuel Thibault, Simon Archipoff, Xavier Lacoste, Terry Cojean, Yanis Khorsi, Philippe Virouleau, LoÏc Jouans, Leo Villeveygoux

7.1.7 somp

Name: SOMP

Keywords: Simulation, Task scheduling, OpenMP

Functional Description: sOMP is a simulator for task-based applications running on shared-memory architectures, utilizing the SimGrid framework. The aim is to predict the performance of applications on various machine designs while taking different memory models into account, using a trace from a sequential execution.

URL: https://gitlab.inria.fr/idaoudi/omps/-/wikis/home

Publications: hal-02933803, hal-03177026v2

Contact: Samuel Thibault

Participant: Idriss Daoudi

7.1.8 MIPP

Name: MyIntrinsics++

Keywords: SIMD, Vectorization, Instruction-level parallelism, C++, Portability, HPC, Embedded

Scientific Description: MIPP is a portable and Open-source wrapper (MIT license) for vector intrinsic functions (SIMD) written in C++11. It works for SSE, AVX, AVX-512 and ARM NEON (32-bit and 64-bit) instructions.

Functional Description: MIPP enables writing portable and yet highly optimized kernels to exploit the vector processing capabilities of modern processors. It encapsulates architecture specific SIMD intrinsics routine into a header-only abstract C++ API.

Release Contributions: Version for APP application without external contributions

News of the Year: [2021] Prototyping of RISC-V RVV vector intrinsic support.

URL: https://github.com/aff3ct/MIPP

Publications: hal-01888010, tel-03118420

Contact: Denis Barthou

Participants: Adrien Cassagne, Denis Barthou, Edgar Baucher, Olivier Aumage

Partners: INP Bordeaux, Université de Bordeaux

8 New results

8.1 MPI detach - Towards automatic asynchronous local completion

Participants: V.M. Nguyen, E. Saillard, D. Barthou.

When aiming for large-scale parallel computing, waiting time due to network latency, synchronization, and load imbalance are the primary opponents of high parallel efficiency. A common approach to hide latency with computation is the use of non-blocking communication. In the presence of a consistent load imbalance, synchronization cost is just the visible symptom of the load imbalance. Tasking approaches as in OpenMP, TBB, OmpSs, or C++20 coroutines promise to expose a higher degree of concurrency, which can be distributed on available execution units and significantly increase load balance. Available MPI non-blocking functionality does not integrate seamlessly into such tasking parallelization. In this work, we present a slim extension of the MPI interface to allow seamless integration of non-blocking communication with available concepts of asynchronous execution in OpenMP and C++. Using our concept allows to span task dependency graphs for asynchronous execution over the full distributed memory application. We furthermore investigate compile-time analysis necessary to transform an application using blocking MPI communication into an application integrating OpenMP tasks with our proposed MPI interface extension [9].

8.2 Code transformations for improving performance and productivity of PGAS applications

Participants: S. Baden, E. Saillard, D. Barthou, O. Aumage.

The PGAS model is an attractive means of treating irregular fine-grained communication on distributed memory systems, providing a global memory abstraction that supports low-overhead Remote Memory Access (RMA), direct access to memory located in remote address spaces. RMA performance benefits from hardware support generally provided by modern high performance communication networks, delivering low over- head communication needed in irregular applications such as Metagenomics.

The proposed research program will apply source-to-source transformation to PGAS code. The project will target the UPC++ library [21], a US Department of Energy Exascale Computing Project that the applicant lead for three years at Lawrence Berkeley National Laboratory (LBNL) in Berkeley, California (USA). Source-to-source transformations will be investigated, and a translator will be built that realizes the transformations. UPC++ is open-source and the LBNL development team actively supports the software. The goal of the project is to investigate a source-to-source translator for re- structuring UPC++ code to improve performance. Four transformations will be investigated, as described next: Localization; Communication aggregation; Overlap communication with computation; Adaptive push-pull algorithms.

This work is in the context of the International Inria Chair for Scott Baden.

8.3 Leveraging compiler analysis for NUMA and Prefetch optimization

Participants: M. Popov, E. Saillard.

Performance counter based characterization is currently used to optimize the complex search space of threads and data mapping along with prefetchers on NUMA systems. While optimizing such spaces provides significant performance gains, it requires to dynamically profile applications resulting in huge a characterization overhead. We started a collaboration with the University of Iowa to reduce this overhead by using insights from the compiler. We develop a new static analysis method that characterizes the

LLVM Intermediate Representation: it extracts information and exposes it to a deep learning network to optimize new unseen applications across the NUMA/prefetch space.

We demonstrated that the statically based optimizations achieve 80% of the gains compared to a dynamic approach but without any costly profiling. We further evaluated a hybrid model which predicts whether to use static or dynamic characterization. The hybrid model achieves similar gains as the dynamic model but only profiles 30% of the applications. These results are accepted for publication at IPDPS 2022. We further plan to extend this static characterization workflow for errors detection in parallel applications.

8.4 Generalizing NUMA and Prefetch optimization

Participants: O. Aumage, A. Guermouche, L. Lima Pilla, M. Popov, E. Saillard, L. Scravaglieri.

In addition to the NUMA/prefetch characterization, we also studied how such optimizations generalize. Applications behavior change depending on their inputs. Therefore, an optimal configuration (thread and data mapping along with prefetch) for a fixed input might not be optimal when we change the inputs. We quantified how changing the inputs impacts the performance gains compared to a native per-input optimization. We also studied the energy impact of NUMA/prefetch and characterized how cross-inputs optimizations also affect energy.

We observed that applications can lose more than 25% of the gains due to input changes. We also showed that energy is more affected by NUMA/prefetch than performance and must to be optimized separately. Optimizing performance and energy provide on average 1.6x and 4x gains respectively. Performance-based optimizations can lose up to 50% of the energy gains compared to native energy-based optimizations.

We plan to submit these results along with a machine learning model that predicts efficient performance and energy configurations across inputs using dynamic profiling. We also intend to extend this model with static characterization to predict behaviors at a reduced cost.

8.5 Extension of MIPP SIMD library to RISC-V

Participants: A. Cassagne, E. Baucher, D. Barthou, O. Aumage.

MIPP library 7.1.8 is a portable C++ header to write SIMD code on ARM (Neon) and Intel (SEE, AVX, AVX2, AVX512) architectures. We started to adapt MIPP to RISC-V with the V extension (for SIMD extension). The change is more important than just an adaptation of the instruction set, since the length of vectors in RISC-V-V can be changed and adapted during one execution. This has a large impact on the design of the way vector code is described. We developped a new version of MIPP, adapted for RISC-V and adopting the same design for ARM (Neon) and Intel architectures. This is still work in progress.

8.6 Selection of Legalization Algorithms using Deep Convolutional Neural Networks

Participants: L. Lima Pilla, M. Popov.

In the context of a collaboration with the Federal University of Santa Catarina, Brazil, and the University of Calgary, Canada, we have investigated ways to train and employ machine learning models to automate the selection of legalization algorithms for the physical design of integrated circuits. This automation provides better legalization solutions with greatly reduced training and legalization times [7].

8.7 Code optimization and generation for Cardiac simulation

Participants: C. Sakka, V. Alba, E. Saillard, D. Barthou, A. Guermouche, M.-C. Counilh.

In the context of the ANR Exacard project, we optimized the code Propag for the cardiac electro-physiology simulation. We used for this MIPP [23] for writing the different SIMD kernels. A port on GPU is in progress. Besides, during the internship of V.Alba, we optimized the code generator of Myokit, a framework with a DSL to specify different mathematical models for cardiac electro-physiology. The code generator takes a mathematical formulation of the code and transforms it into OpenCL. We modified the generator so that the kernels used StarPU and an automatic load balancing strategy was put in place in order to run on several (possibly heterogeneous) GPUs.

8.8 The MPI Bugs Initiative

Participants: M. Laurent, E. Saillard.

Ensuring the correctness of MPI programs becomes as challenging and important as achieving the best performance. Many tools have been proposed in the literature to detect incorrect usages of MPI in a given program. However, the limited set of code samples each tool provides and the lack of metadata stating the intent of each test make it difficult to assess the strengths and limitations of these tools. We have developped the MPI BUGS INITIATIVE (MBI), a complete collection of MPI codes to assess the status of MPI verification tools. We introduce a classification of MPI errors and provide correct and incorrect codes covering many MPI features and our categorization of errors. The resulting suite comprises 1,668 codes, each coming with a well-formatted header that clarifies the intent of each code and specifies how to execute and evaluate it. We evaluated the completeness of the MPI BUGS INITIATIVE against eight state- of-the-art MPI verification tools: Aislinn, CIVL, ISP, ITAC, Mc SimGrid, MPI-SV, MUST and PARCOACH [15].

8.9 Dynamic Data Race Detection for MPI-RMA Programs

Participants: C.T. Ait Kaci, E. Saillard, D. Barthou.

One-sided communications is a well known distributed programming paradigm for high performance computers, as its properties allows for a greater asynchronism and computation/communication overlap than classical message passing mechanisms. The Remote Memory Access interface of MPI (MPI-RMA) is an interface in which each process explicitly exposes an area of its local memory as accessible to other processes to provide asynchronous one-sided reads, writes and updates. While MPI-RMA is expected to greatly enhance the performance and permit efficient implementations on multiple platforms, it also comes with several challenges with respect to memory consistency. Developers must handle complex memory consistency models and complex programming semantics. The RMA-Analyzer is a new tool that detects memory consistency errors (also known as data races) during MPI-RMA program executions. It collects relevant MPI-RMA operations and load/store accesses during ex- ecution and performs an on-the-fly analysis to stop the program in case of a consistency violation [12].

8.10 Task scheduling with memory constraints

Participants: M. Gonthier, S. Thibault.

When dealing with larger and larger datasets processed by task-based applications, the amount of system memory may become too small to fit the working set, depending on the task scheduling order. In collaboration with the ROMA team, we have published theoretical modeling and analysis [5]. We have devised a new scheduling strategies which reorder tasks to strive for locality and thus reduce the amount of communications to be performed. It was shown to be more effective than the current state of the art, particularly in the most constrained cases. We published initial results in the Coloc workshop [14] and a complete version as research report [19], currently being submitted for journal publication. We have extended the results from mono-GPU to multi-GPU, and introduced another strategy which selects tasks dynamically with a locality-aware principle rather than computing a static ordering. This amply improved the scheduling cost. This is currently being submitted for conference publication.

8.11 Failure Tolerance for StarPU

Participants: R. Lion, S. Thibault.

Since supercomputers keep growing in terms of core numbers, the reliability decreases the same way. The project H2020 EXA2PRO and more precisely the PhD thesis of Romain Lion aimed to propose solutions for the failure tolerance problem, including StarPU. While exploring decades of research about the resilience techniques, we have identified properties in our runtime's paradigm that can be exploited in order to propose a solution with lower overhead than the generic existing ones. We have implemented a checkpointing solution in StarPU, and evaluated its overhead in terms of additional communications. We brought to light that we can build a synergy between the application-induced communications and the checkpointing-induced communications. This allows to keep the checkpoint overhead to a reasonable amount (less than 10% additional communication). We have reported the obtained insights in a Dagstuhl report [4]

8.12 Energy-aware task scheduling in StarPU

Participants: A. Guermouche, M. Makni, S. Thibault.

In the context of the EXA2PRO project and the visit of A. Guermouche, we have investigated the time/energy behavior of several dense linear algebra kernels. We have found that they can exhibit largely different compromises, which raised the question of revising task scheduling to take into account energy efficiency. We have improved StarPU's ability to integrate energy performance models, and integrated helpers for performing energy measurement even with the coarse-grain support provided by the hardware. We have shown that the energy/time Pareto front can be presented to the application user, to decide which compromise should be chosen.

8.13 FPGA support in StarPU

Participants: M. Makni, S. Thibault.

In the context of the EXA2PRO project we have integrated into StarPU the support for FPGA devices from the Maxeler manufacturer. Since such devices can directly stream data in/out from/to the main memory, we had to make StarPU more flexible on the provenance and destination of task data, so as to benefit from this capacity, and provide more flexibility to the task scheduler. We have completed the implementation with support for multiple FPGAs, and collaborated with CNRS to make their MetalWall high-capacitor simulation application leverage this dynamic runtime. This allows for flexible runtime decision of the specialization of the various FPGA devices.

8.14 Integration of a runtime system in an software stack aiming for exascale computing

Participants: O. Aumage, S. Thibault.

In the context of the EXA2PRO european project, the StarPU runtime system was integrated in the EXA2PRO software stack. The goal is to enhance programmability of the next-generation accelerator-based exascale platforms by integrating high-level software composition and skeleton programming, a dynamic runtime system, and technical debt management tools. We published a common paper [8] which describes the stack and shows the effectiveness of the approach on several application cases.

8.15 Scheduling iterative task graph for video games

Participants: B. Coye, D. Barthou, L. Lima Pilla, R. Namyst.

In the context of Baptiste Coye's PhD started in March 2020 in partnership with Ubisoft, Baptiste has studied the task graph and their scheduling from real games. The transition to a more modular architecture, with a central scheduler, is currently under study. There are opportunities for optimization steming from the fact that the task graph is repeatedely executed each timeframe, with few modifications from one frame to the next. The tasks have varying execution times, but differing only slightly from one frame to the other. Moreover, some tasks can be rescheduled from one time frame to the next. Taking into account these constraints the current research effort is on how to better define the tasks and their dependences and constraints, and then propose an incremental modification the task graph.

8.16 Task-based execution model for fine-grained tasks

Participants: C. Castes, E. Saillard, O. Aumage.

Sequential task-based programming models paired with advanced runtime systems allow the programmer to write a sequential algorithm independently of the hardware architecture in a productive and portable manner, and let a third party software layer —the runtime system—, deal with the burden of scheduling a correct, parallel execution of that algorithm to ensure performance. Developing algorithms that specifically require fine-grained tasks along this model is still considered prohibitive, however, due to per-task management overhead, forcing the programmer to resort to a less abstract, and hence more complex "task+X" model. We thus investigated the possibility to offer a tailored execution model, trading dynamic mapping for efficiency by using on a decentralized, conservative in-order execution of the task flow, while preserving the benefits of relying on the sequential task-based programming model. We proposed a formal specification of the execution model as well as a prototype implementation, which we assess on a shared-memory multicore architecture with several synthetic workloads. The results showed that under the condition of a proper task mapping supplied by the programmer, the pressure on the runtime system is significantly reduced and the execution of fine-grained task flows is much more efficient.

8.17 Hierarchical Tasks

Participants: N. Furmento, G. Lucas, R. Namyst, S. Thibault, P.A. Wacrenier.

Task-based runtimes such as StarPU use the Sequential Task Flow programming model which has the intrinsic limitation of supporting static task graphs only. This leads to potential submission overhead and to a static task graph which is not necessarily adapted for execution on heterogeneous systems.

A standard approach is to find a trade- off between the granularity needed by accelerator devices and the one required by CPU cores to achieve performance.

We have extended the STF model in StarPU to enable tasks subgraphs, also called hierarchical tasks, at runtime. This approach allows for a more dynamic task graph, by dynamically adapting the granularity to meet the optimal size of the targeted computing resources. We are working on an initial implementation that we evaluate on shared memory heterogeneous systems, using the Chameleon dense linear algebra library.

8.18 ADT Gordon

Participants: O. Aumage, N. Furmento, S. Thibault.

In collaboration with the HIEPACS and TADAAM Inria teams, we have strengthened the relations between the Chameleon linear algebra library from HIEPACS, our StarPU runtime scheduler, and the NewMadeleine high-performance communication library from TADAAM. More precisely, we have improved the interoperation between StarPU and NewMadeleine, to more carefully decide when NewMadeleine should proceed with communications. We have then introduced the notion of dynamic collective operations, which opportunistically introduce communication trees to balance the communication load. We have also evaluated the Chameleon + StarPU stack in the context of a biodiversity application of the PLEIADE team. Our stack proved to be able to process very large matrices (more than a million matrix side size), which was unachievable before, with reasonable processing time. We had to carefully integrate the I/O required for loading the matrices with the computation themselves.

8.19 High performance software defined radio with AFF3CT

Participants: A. Cassagne, D. Barthou, O. Aumage.

The AFF3CT library 7.1.3, developed jointly between IMS and the STORM team, which aims to model error correcting codes for numerical communications has been further improved in different ways. The automatic parallelization of the tasks describing the simulation of a whole chain of signal transmission has been designed, using a Domain Specific Language. This allows the development of Software Defined Radio and has been put to work on use case with Airbus. These results have been defended in Adrien Cassagne's PhD thesis [22].

A complete software DVB-S2 tranciever has been developped with success thanks to AFF3CT. For this purpose, USRP modules were combined with multicore and SIMD CPUs. Thus, some components are directly taken from the AFF3CT library and others such as the synchronization functions have been developped and added to the library. Experiment results show the performance but also the flexibility and the portability of the transceiver[13]

8.20 HPC Big Data Convergence

Participants: O. Aumage, N. Furmento, K. He, S. Thibault.

This work is partly done within the framework of the project hpc-scalable-ecosystem from région Nouvelle Aquitaine. It is a collaboration with members of the Hiepacs team and the LaBRI.

A Java interface for StarPU has been implemented and allows to execute Map Reduce applications on top of StarPU. We have made some preliminary experiments on Cornac, a big data application for visualising huge graphs.

We have also developed a new C++ library, called Yarn++, to allow the execution of HPC applications on Big Data clusters. We are doing initial evaluation with a FMM application written on top of StarPU on the PlaFRIM platform.

In the context of the HPC-BIGDATA IPL, a Python interface for StarPU has been started, to allow executing Python tasks on top of StarPU. This will allow to close the gap between the HPC and BigData communities by allowing the latter to directly execute their applications with the runtime system of the former. The challenge at stake is that the Python interpreter itself is not parallel, so data has to be transferred from one interpreter to another. We integrated the StarPU task-based interface within Python and its notion of asynchronous *Future*. We integrated the use of Python objects with task tasks. We added support for TCP/IP-based master-slave distributed execution.

Also in the context of the HPC-BIGDATA IPL, we have started integrating a machine-learning-based task scheduler (i.e. BigData for HPC), designed by Nathan Grinsztajn (from Lille) in the StarPU runtime system. The results which Nathan obtained in pure simulation were promising, the integration will allow to confirm them with actual applications and real execution.

In the context of the TEXTAROSSA project [11], and in collaboration with the TOPAL team, we have started using the StarPU runtime system to execute machine-learning applications on heterogeneous platforms (i.e. HPC for BigData). We ported the use of cuDNN with StarPU, and started to model the scheduling as a constraint system.

8.21 Simulation of OpenMP task based programs

Participants: I. Daoudi, S. Thibault.

A simulator for OpenMP task-based programs has been designed as part of Inria's IPL HAC-Specis project, and the PhD thesis of Idriss Daoudi. We have carefully modeled the memory architecture details of two very different platforms, and implemented a simulation of the cache effects of dense linear algebra. This allowed to obtain a good simulation accuracy [18]. Idriss wrote and defended his PhD thesis [17].

9 Bilateral contracts and grants with industry

9.1 Bilateral contracts with industry

Participants: Denis Barthou, Emmanuelle Saillard, Raymond Namyst.

- Contract with ATOS/Bull for the PhD CIFRE of Célia Ait Kaci (2019-2022),
- Contract with Ubisoft for the PhD CIFRE of Baptiste Coye (2020-2023),
- Contract with CEA for the PhD of Van Man Nguyen (2019-2022), P.Beziau (2018-2021) and other short contracts

10 Partnerships and cooperations

10.1 International initiatives

10.1.1 Associate Teams in the framework of an Inria International Lab or in the framework of an Inria International Program

COHPC

Participants: E. Saillard, D. Barthou.

Title: Correctness and Performance of HPC Applications

Duration: 2019 - 2022

Coordinator: Costin Iancu (cciancu@lbl.gov)

Partners: Lawrence Berkeley National Laboratory, USA.

Inria contact: Emmanuelle Saillard

Summary: This collaboration aims to develop methods and tools to aid developers with problems of correctness and performance in HPC applications for Exascale systems. There are several requirements for such tools: precision, scalability, heterogeneity and soundness. In order to improve developer productivity, we aim to build tools for guided code transformations (semi-automatic) using a combination of static and dynamic analysis. Static analysis techniques will enable soundness and scalability in execution time. Dynamic analysis techniques will enable precision, scalability in LoCs and heterogeneity for hybrid parallelism. A key aspect of the collaboration is to give precise feedback to developers in order to help them understand what happens in their applications and facilitate the debugging and optimization processes.

HPCProSol

Participants: L. Lima Pilla.

Title: HPC problems and solutions

Duration: 2021 - 2023

Coordinator: Carla Osthoff / Francieli Zanon Boito

Partners: LNCC, Petropolis, Brazil.

Inria contact: Francieli Zanon Boito (Tadaam team)

Summary: This project's main goal is to study and characterize the new HPC workload, represented by a set of scientific applications that are important to the LNCC because they are representative of its Santos Dumont machine's workload. This generated knowledge will guide the proposal of monitoring and profiling techniques for applications, and the design of new coordination mechanisms to arbitrate resources in HPC environments. We are interested in evaluating and improving individual applications' performance, but also on using this study to provide a better understanding of how performance is impacted by aspects such as interference. Moreover, we want to identify metrics that can be used to predict performance and deviations from the applications' expected behaviors, specially at run time.

10.1.2 Participation in other International Programs

PHC Germaine de Staël :

Participants: L. Lima Pilla, M. Popov.

Title: Broad Bundle of BEnchmARks for Scheduling in HPC, Big Data, and ML (3BEARS)

Duration: 2021 - 2022

Program: Partenariat Hubert Curien, CNRS

Coordinator: Laércio Lima Pilla

Partners: University of Basel, Switzerland

Summary: The goal of the project is to develop ways to co-design parallel applications and scheduling algorithms in order to achieve high performance and optimize resource utilization. Parallel applications nowadays are a mix of HPC, Big Data, and Machine Learning (ML) software. They show varied computational profiles, being compute-, data-, I/O-intensive, or a combination thereof. Because of the varied nature of their parallelism, their performance can degrade due to factors such as synchronization, management of parallelism, communication, and load imbalance. In this situation, scheduling has to be done with care to avoid causing new performance problems (e.g., fixing load imbalance may degrade communication performance). In this work, we concentrate explicitly on scheduling algorithms that minimize load imbalance and/or minimize communication costs. Our focus is the characterization of workloads represented by the mix of HPC, Big Data, and ML applications, in order to use them to test existing scheduling techniques and to enable the development of novel and more suitable scheduling techniques.

Inria International Chair

IIC BADEN Scott

Name of the chair: Code transformations for improving performance and productivity of PGAS applications

Institution of origin: U. of San Diego, California

Country: USA

Dates: From Wed Jan 01 2020 to Tue Dec 31 2024

Title: Code transformations for improving performance and productivity of PGAS applications

Summary: This research program focuses on source-to-source transformation to PGAS code. The project targets the UPC++ library [21], a US Department of Energy Exascale Computing Project that Scott Baden led for three years at Lawrence Berkeley National Laboratory (LBNL) in Berkeley, California (USA). UPC++ is open-source and the LBNL development team actively supports the software. The goal of the project is to investigate a source-to-source translator for restructuring UPC++ code to improve performance. Four transformations will be investigated: Localization; Communication aggregation; Overlap communication with computation and Adaptive push-pull algorithms.

10.2 European initiatives

10.2.1 FP7 & H2020 projects

EXA2PRO

Title: Enhancing Programmability and boosting Performance Portability for Exascale Computing Systems

Duration: 2018-2021

Coordinator: ICCS

Partners:

- CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE CNRS (France)
- ETHNIKO KENTRO EREVNAS KAI TECHNOLOGIKIS ANAPTYXIS (Greece)
- FORSCHUNGSZENTRUM JULICH GMBH (Germany)
- INSTITUTE OF COMMUNICATION AND COMPUTER SYSTEMS, ICCS (Greece)
- LINKOPINGS UNIVERSITET (Sweden)
- MAXELER TECHNOLOGIES LIMITED (UK)
- UNIVERSITE DE BORDEAUX (France)

Inria contact: Samuel Thibault

Summary: The vision of EXA2PRO is to develop a programming environment that will enable the productive deployment of highly parallel applications in exascale computing systems. EXA2PRO pro- gramming environment will integrate tools that will address significant exascale challenges. It will support a wide range of scientific applications, provide tools for improving source code quality, enable efficient exploitation of exascale systems' heterogeneity and integrate tools for data and memory management optimization. Additionally, it will provide various fault-tolerance mechanisms, both user-exposed and at runtime system level and performance monitoring features.

MICROCARD

Title: MICROCARD

Duration: 2021-2024

Coordinator: Mark Potse, U.Bordeaux

Partners:

- Karlsruhe Institute of Technology (Germany)
- Megware (UK)
- Orobix (Italy)
- Simula (Norway)
- U.Strasbourg (France)
- U.Bordeaux (France)
- U.della Svizzera Italiana (Italy)
- U.di Pavia (Italy)
- Zuse Institute, Berlin (Germany)

Inria contact: Mark Potse

Summary: The purpose of MICROCARD is to develop software that can model a heart cell by cell. Such software may run on large super computers. As a consequence, MICROCARD will develop algorithms that are tailored to the mathematical problem, to the size of the problem and to the design of future exascale architectures.

Textarossa

Title: Towards EXtreme scale Technologies and Accelerators for euROhpc hw/Sw Supercomputing Applications for exascale

Duration: 2021-2024

Coordinator: ENEA

Partners:

- Agenzia Nationale per le nuove tecnologie, ENEA (Italy),
- Fraunhofer institute (Germany),
- Consorzio interuniversitario nazionale (Italy),
- Inria (France),
- ATOS/Bull (France),
- E4 Computer engineering,
- Barcelona Supercomputing Center (Spain),
- Instytut Chemmi biorganicznej Polskiej (Poland),
- Istituto nazionale di fisica nucleare (Italy),
- Consiglio Nazionale delle Ricerche (Italy),
- In Quattro SRL (Italy)

Inria contact: Raymond Namyst, Olivier Beaumont

Summary: The project aims to tackle these gaps by applying a co-design approach to design and develop an efficient supercomputer system based on new hardware accelerators, innovative two-phase cooling equipment, advanced algorithms, methods and software products for traditional HPC domains as well as for emerging domains in high performance artificial intelligence and high performance data analytics

10.2.2 Other european programs/initiatives

PRACE project 6IP

Title: PRACE-6IP

Duration: 2019-2022

Coordinator: FORSCHUNGSZENTRUM JULICH GMBH

Partners: 29 organisms all around Europe (see web site for a complete list)

Inria contact: for Storm team, Olivier Aumage, Samuel Thibault

Summary: The objectives of PRACE-6IP are to build on and seamlessly continue the successes of PRACE and start new innovative and collaborative activities proposed by the consortium. These include: assisting the development of PRACE 2; strengthening the internationally recognized PRACE brand; continuing and extend advanced training which so far provided more than 36 400 person·training days; preparing strategies and best practices towards Exascale computing, work on forward-looking SW solutions; coordinating and enhancing the operation of the multi-tier HPC systems and services; and supporting users to exploit massively parallel systems and novel architectures. A high level Service Catalogue is provided.

10.3 National initiatives

ELCI The ELCI PIA project (Software Environment for HPC) aims to develop a new generation of software stack for supercomputers, numerical solvers, runtime and programming development environments for HPC simulation. The ELCI project also aims to validate this software stack by showing its capacity to offer improved scalability, resilience, security, modularity and abstraction on real applications. The coordinator is Bull, and the different partners are CEA, INRIA, SAFRAN, CERFACS, CNRS CORIA, CENAERO, ONERA, UVSQ, Kitware and AlgoTech.

10.3.1 ANR

ANR SOLHARIS SOLHARIS

- ANR PRCE 2019 Program, 2019 - 2023 (48 months)
- Identification: ANR-19-CE46-0009
- Coordinator: CNRS-IRIT-INPT
- Other partners: INRIA-LaBRI Bordeaux, INRIA-LIP Lyon, CEA/CESTA, Airbus CRT
- Abstract: SOLHARIS aims at achieving strong and weak scalability (i.e., the ability to solve problems of increasingly large size while making an effective use of the available computational resources) of sparse, direct solvers on large scale, distributed memory, heterogeneous computers. These solvers will rely on asynchronous task-based parallelism, rather than traditional and widely adopted message-passing and multithreading techniques; this paradigm will be implemented by means of modern runtime systems which have proven to be good tools for the development of scientific computing applications. The challenges that SOLHARIS will be confronted with are related to the complexity, irregularity and, to some extent, unpredictability of sparse, direct solvers, to the large scale, complexity and heterogeneity of supercomputing platforms and to the ever increasing performance gap between processing units, memories and interconnects. SOLHARIS will tackle these challenges with three, tightly combined research efforts. First, it will develop dense and sparse linear algebra algorithms that can achieve better scalability by means of higher concurrency and efficiency and lower memory consumption. Second, it will improve runtime systems with novel features that enhance their performance and scalability and extend their programming interface to allow for an efficient and portable implementation of scalable algorithms. Third, it will develop scheduling methods for achieving high perfor mance and scalability of both runtime systems and sparse direct solvers on large heterogeneous supercomputers.

ANR EXACARD

- AAPG ANR 2018 (42 months)
- Coordinator: Yves Coudière (Carmen) INRIA Bordeaux
- Abstract: Cardiac arrhythmia affect millions of patients and cause 300,000 deaths each year in Europe. Most of these arrhythmia are due to interaction between structural and electrophysiological changes in the heart muscle. A true understanding of these phenomena requires numerical simulations at a much finer resolution, and larger scale, than currently possible. Next-generation, heterogeneous, high-performance computing (HPC) systems provide the power for this. But the large scale of the computations pushes the limits of current runtime optimization systems, and together with task-based parallelism, prompts for the development of dedicated numerical methods and HPC runtime optimizations. With a consortium including specialists of these domains and cardiac modeling, we will investigate new task-based optimization techniques and numerical methods to utilize these systems for cardiac simulations at an unprecedented scale, and pave the way for future use cases.

10.3.2 IPL - Inria Project Lab

HAC-SPECIS (High-performance Application and Computers, Studying PErformance and Correctness In Simulation)

- Inria IPL 2016 - 2020 (48 months)
- Coordinator: Arnaud Legrand (team Polaris, Inria Rhône Alpes)

Since June 2016, the team is participating to the HAC-SPECIS Inria Project Lab (IPL). This national initiative aims at answering methodological needs of HPC application and runtime developers and allowing to study real HPC systems both from the correctness and performance point of view. To this end, it gathers experts from the HPC, formal verification and performance evaluation community.

HPC-BigData (High Performance Computing and Big Data)

- Inria IPL 2018 - 2022 (48 months)
- Coordinator: Bruno Raffin (team DataMove, Inria Rhône Alpes)

Since June 2018, the team is participating to the HPC-BigData Inria Project Lab (IPL). The goal of this HPC-BigData IPL is to gather teams from the HPC, Big Data and Machine Learning (ML) areas to work at the intersection between these domains. Research is organized along three main axes: high performance analytics for scientific computing applications, high performance analytics for big data applications, infrastructure and resource management.

EQIP (Engineering for Quantum Information Processors)

- Inria IPL 2020 - 2024 (48 months)
- Coordinator: Anthony Leverrier (team Cosmiq)

Since Nov 2020, the team is participating to the EQIP Inria Project Lab (IPL). The goal of this EQIP IPL is to build a quantum processor, develop the software stack needed to operate a large-scale quantum computer, and develop quantum solutions to overcome classical computers. The EQIP challenge requires expertise in (1) superconducting qubits, (2) simulation of quantum systems, (3) numerical methods, (4) control theory, (5) programming languages and formal methods, (6) compilation, (7) quantum error correction, (8) quantum emulation, (9) cryptography and cryptanalysis, (10) quantum algorithms, (11) high-performance computing and gathers teams in this domains.

11 Dissemination

Participants: All team members.

11.1 Promoting scientific activities

11.1.1 Scientific events: organisation

General chair, scientific chair Emmanuelle Saillard was general chair of the C3PO workshop.

11.1.2 Scientific events: selection

Chair of conference program committees Samuel Thibault was global chair of the *High-performance architectures and accelerators* topic of Euro-Par 2021. Emmanuelle Saillard was chair of the parallelism track of Compas 21.

Member of the conference program committees STORM members have participated as technical program committee members for the following conferences and workshops: ICCS 2021, HCW 2021, Correctness 2021, SC (Performance track and Programming Systems track), ISC 2021.

Reviewer STORM members have conducted wide reviewing activities for the following conferences and workshops: Euro-Par, SuperComputing, PEHC, IPDPS, PDP, ICCS, ISC, Correctness workshop.

11.1.3 Journal

Member of the editorial boards Samuel Thibault is Associate Editor for JPDC.

Reviewer - reviewing activities STORM members have conducted wide reviewing activities for the following journals: IEEE TPDS, IEEE TC, JPDC, CCPE, Electronic Letters.

11.1.4 Invited talks

- Olivier Aumage: Workshop IPDPS RADR, May 2021 (videoconference).
- Samuel Thibault: Huawei workshop, March 2021 (videoconference).
- Samuel Thibault: ISC 2021, July 2021 (videoconference).

11.1.5 Leadership within the scientific community

- Raymond Namyst is involved in the national "plan de relance" NUMEX on HPC for the following years.

11.1.6 Scientific expertise

- Olivier Aumage took part in the selection process of the ATOS–Joseph Fourier Prize 2021.
- Nathalie Furmento was part of the HCERES evaluation committee for the L3I laboratory in November 2021.
- Nathalie Furmento was member of four research engineers recruitment committees for Inria in December 2021.

11.1.7 Research administration

- Olivier Aumage: head of High Performance Runtime Systems Team, and elected member of LaBRI laboratory's Scientific Committee, since April 2021.
- Nathaie Furmento is a member of the CDT at Bordeaux-Sud Ouest Inria Research Center.
- Nathalie Furmento is an elected member of LaBRI's council, since January 2021.
- Samuel Thibault is a nominated member of LaBRI's council, since January 2021.
- Emmanuelle Saillard: Member of the Commission de délégation at Bordeaux-Sud-Ouest Inria Research Centre.

11.2 Teaching - Supervision - Juries

11.2.1 Teaching

- Licence: Emmanuelle Saillard, Introduction to research, 1HeTD, L3, University of Bordeaux.
- Engineering School: Emmanuelle Saillard, Introduction to Algorithms, 32HeCIx, L3, ENSEIRB-MATMECA.
- Engineering School: Emmanuelle Saillard, Tree Structure, 32HeCI, L3, ENSEIRB-MATMECA.
- Engineering School: Emmanuelle Saillard, Languages of parallelism, 12HeC, M2, ENSEIRB-MATMECA.
- Engineering School: Mihail Popov, Project C, 25HeC, L3, ENSEIRB-MATMECA.

- Master: Laércio Lima Pilla, Algorithms for High-Performance Computing Platforms, 12HeTD, M2, ENSEIRB-MATMECA and University of Bordeaux.
- Master: Laércio Lima Pilla, Scheduling and Runtime Systems, 27.75 HeTD, M2, University of Paris-Saclay.
- Engineering School: Adrien Cassagne, Microprocessors architecture, 20HeTD, L3, ENSEIRB-MATMECA.
- Engineering School: Romain Lion, System Programming, 18HeTD, M1, ENSEIRB-MATMECA.
- Licence: Marie-Christine Counilh, Introduction to Computer Science (64HeTD), Introduction to C programming (52HeTD), L1, University of Bordeaux.
- Master MIAGE: Marie-Christine Counilh, Object oriented programming in Java (30HeTD), M1, University of Bordeaux.
- Licence: Samuel Thibault is responsible for the computer science topic of the first university year.
- Licence: Samuel Thibault is responsible for the Licence Pro ADSILLH (Administration et Développeur de Systèmes Informatiques à base de Logiciels Libres et Hybrides).
- Licence: Samuel Thibault is responsible for the 1st year of the computer science Licence.
- Licence: Samuel Thibault, Introduction to Computer Science, 56HeTD, L1, University of Bordeaux.
- Licence: Samuel Thibault, Networking, 51HeTD, Licence Pro, University of Bordeaux.
- Master: Samuel Thibault, Operating Systems, 24HeTD, M1, University of Bordeaux.
- Master: Samuel Thibault, Software Security, 60HeTD, M1, University of Bordeaux.
- Master: Samuel Thibault, System Security, 20HeTD, M2, University of Bordeaux.
- Engineering School: Denis Barthou is the head of the computer science teaching department of ENSEIRB-MATMECA (300 students, 20 faculties, 120 external teachers).
- Engineering School: Denis Barthou, Architectures (L3), Parallel Architectures (M2), Procedural Generation for 3D Games (M2), C/Algorithm projects (L3).
- Licence, Pierre-André Wacrenier, is responsible for the 3rd year of the computer science Licence.
- Licence, Pierre-André Wacrenier, Introduction to Computer Science (64HeTD, L1), Systems Programming (64HeTD, L3), University of Bordeaux.
- Master, Pierre-André Wacrenier, Parallel Programming (64HeTD), University of Bordeaux.
- Raymond Namyst was involved in the introduction of Computer Science courses in the French High School (Lycée) scholar program. In particular, he was in charge of organizing a one-week condensed training session to 96 High School teachers on the following topics: Computer Architecture, Operating Systems, Networks and Robotics. The goal was to prepare then to teach computer science basics to students starting from September 2019, and to help them to prepare material for practice sessions.
- Denis Barthou was teacher for the previous courses, in Computer Architecture.
- Licence, Amina Guermouche, System Programming (32HeTD, L3)

11.2.2 Supervision

- Postdoc (ATER, 2020-2021): Adrien Cassagne, with Olivier Aumage, Denis Barthou, Christophe Jego, Camille Leroux.
- PhD Idriss Daoudi, oct. 2018 - sep. 2021, Samuel Thibault, Thierry Gautier.
- PhD in progress: Romain Lion, October 2018, Samuel Thibault.
- PhD in progress: Célia Ait Kaci, "Analysis and optimizations for partitioned global address space based HPC applications", 2019 - , supervised by Emmanuelle Saillard and Denis Barthou.
- PhD in progress: Baptiste Coye, "Dynamic scheduling of task graph by composition", supervised by Raymond Namyst and Denis Barthou.
- PhD in progress: Maxime Gonthier, "Memory-constrained Scheduling in a task-based model", october 2020 - , supervised by Loris Marchal and Samuel Thibault.
- PhD in progress: Van-Man Nguyen, "Automatic optimization of parallel applications using non-blocking communications", 2020 - , supervised by Emmanuelle Saillard and Denis Barthou.
- PhD in progress: Jean-François David, "Optimizing memory usage when training Deep Neural Networks", september 2021 - , supervised by Olivier Beaumont, Lionel Eyraud, Raymond Namyst, and Samuel Thibault.
- PhD in progress: Gwenolé Lucas, "Programming Heterogeneous Architectures using Divisible Tasks", supervised by Mathieu Faverge, Abdou Guermouche, Raymond Namyst and Pierre-André Wacrenier.
- Internship: Vincent Alba, May 2021 - Sept. 2021, Marie-Christine Counilh, Denis Barthou.
- Internship: Edgar Baucher, June 2021 - Jul. 2021, Denis Barthou, Adrien Cassagne.
- Internship: Mustafa Regragui, May 2021 - Sep. 2021, Laércio Lima Pilla, Baptiste Coye.
- Internship: Pierre-Antoine Rouby, May 2021 - Jul. 2021, Emmanuelle Saillard.
- Internship: Lana Scravaglieri, June 2021 - Aug. 2021, Mihail Popov.
- Engineer: Bruno Tessier, Inria engineer assigned to Jean Zay platform support team at IDRIS supercomputing center, since April 2021, Olivier Aumage.

11.2.3 Juries

- Olivier Aumage: PhD grants committee, Mathematics and Informatics Doctoral School (EDMI), University of Bordeaux.
- Emmanuelle Saillard: PhD of James Trotter (reviewer): High-performance finite element computations.
- Emmanuelle Saillard: Associate Professor position at the University of Rennes, Apr. and May 2021.
- Samuel Thibault: PhD of Alexis Lescouet: Memory management for operating systems and runtimes.
- Samuel Thibault: PhD of Benjamin Dauphin: Liveness Analysis Techniques and Run-time Environment for Memory Management of dataflow Applications.
- Samuel Thibault: PhD of Daniel Torres: Application-Based Fault Tolerance for Numerical Linear Algebra at Large Scale.
- Denis Barthou: reviewer for the PhD of Salwa Kobeissi, U. Strasbourg : “Speculative Rewriting of Recursive Programs as Loop Candidates for Efficient Parallelization and Optimization using an Inspector-Executor Mechanism”.

11.3 Popularization

11.3.1 Internal or external Inria responsibilities

- Emmanuelle Saillard is responsible of popularization activities for Inria Bordeaux Sud-Ouest.
- Emmanuelle Saillard : Organization of a meet-a-researcher day for ENS Lyon, November 2021.

11.3.2 Articles and contents

- ETP4HPC Whitepaper "Task-based Performance Portability in HPC": This paper proposed by Olivier Aumage, Paul Carpenter (BSC) and Siegfried Benkner (Univ. of Vienna), summarises how task abstraction, which first appeared in the 1990s and is already mainstream in HPC, should be the basis for a composable and dynamic performance-portable interface. It outlines the innovations that are required in the programming model and runtime layers, and highlights the need for a greater degree of trust among application developers in the ability of the underlying software layers to extract full performance. These steps will help realise the vision for performance portability across current and future architectures and problems.

11.3.3 Education

EasyPAP, developped in the team [6][24] is a framework designed to make learning parallel programming more accessible and attractive to students. A comprehensive set of tools allows them to quickly get visual feedback about the parallel behavior of their code, to analyze the locality of the computations, and to understand performance issues. It features a wide range of 2D computation kernels that the students are invited to parallelize using Pthreads, OpenMP, OpenCL or MPI. Execution of kernels can be interactively visualized, and powerful monitoring tools allow students to observe both the scheduling of computations and the assignment of 2D tiles to threads/processes. By focusing on algorithms and data distribution, students can experiment with diverse code variants and tune multiple parameters, resulting in richer problem exploration and faster progress towards efficient solutions.

11.3.4 Interventions

Popularization intervention in 2021:

- Emmanuelle Saillard : Circuit scientifique bordelais "hors les murs", October 2021, Excideuil.

12 Scientific production

12.1 Major publications

[1] O. Aumage. 'Instruments of Productivity for High Performance Computing'. Habilitation à diriger des recherches. Université de Bordeaux (UB), France, Dec. 2020. URL: https://hal.inria.fr/tel-03105625.

[2] A. Lasserre, R. Namyst and P.-A. Wacrenier. 'EASYPAP: a Framework for Learning Parallel Programming'. In: *Journal of Parallel and Distributed Computing* (2021). DOI: 10.1016/j.jpdc.2021.07.018. URL: https://hal.archives-ouvertes.fr/hal-03126887.

[3] L. Papadopoulos, D. Soudris, C. Kessler, A. Ernstsson, J. Ahlqvist, N. Vasilas, A. I. Papadopoulos, P. Seferlis, C. Prouveur, M. Haefele, S. Thibault, A. Salamanis, T. Ioakimidis and D. Kehagias. 'EXA2PRO: A Framework for High Development Productivity on Heterogeneous Computing Systems'. In: *IEEE Transactions on Parallel and Distributed Systems*. Special Section on Innovative R&D toward the Exascale Era (Aug. 2021). DOI: 10.1109/TPDS.2021.3104257. URL: https://hal.inria.fr/hal-03318644.

12.2 Publications of the year

International journals

[4] E. Agullo, M. Altenbernd, H. Anzt, L. Bautista-Gomez, T. Benacchio, L. Bonaventura, H.-J. Bungartz, S. Chatterjee, F. M. Ciorba, N. Debardeleben, D. Drzisga, S. Eibl, C. Engelmann, W. N. Gansterer, L. Giraud, D. Göddeke, M. Heisig, F. Jézéquel, N. Kohl, S. Xiaoye, R. Lion, M. Mehl, P. Mycek, M. Obersteiner, E. S. Quintana-Ortí, F. Rizzi, U. Rüde, M. Schulz, F. Fung, R. Speck, L. Stals, K. Teranishi, S. Thibault, D. Thönnes, A. Wagner and B. Wohlmuth. 'Resiliency in numerical algorithm design for extreme scale simulations'. In: *International Journal of High Performance Computing Applications* (20th Sept. 2021). URL: https://hal.inria.fr/hal-03348787.

[5] G. Bathie, L. Marchal, Y. Robert and S. Thibault. 'Dynamic DAG Scheduling Under Memory Constraints for Shared-Memory Platforms'. In: *International Journal of Networking and Computing* (Jan. 2021), pp. 1–29. DOI: 10.15803/ijnc.11.1_27. URL: https://hal.inria.fr/hal-03029847.

[6] A. Lasserre, R. Namyst and P.-A. Wacrenier. 'EASYPAP: a Framework for Learning Parallel Programming'. In: *Journal of Parallel and Distributed Computing* (2021). DOI: 10.1016/j.jpdc.2021.07.018. URL: https://hal.archives-ouvertes.fr/hal-03126887.

[7] R. Netto, S. Fabre, T. A. Fontana, V. Livramento, L. Lima Pilla, L. Behjat and J. L. Guntzel. 'Algorithm Selection Framework for Legalization Using Deep Convolutional Neural Networks and Transfer Learning'. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2021). DOI: 10.1109/TCAD.2021.3079126. URL: https://hal.archives-ouvertes.fr/hal-03245856.

[8] L. Papadopoulos, D. Soudris, C. Kessler, A. Ernstsson, J. Ahlqvist, N. Vasilas, A. I. Papadopoulos, P. Seferlis, C. Prouveur, M. Haefele, S. Thibault, A. Salamanis, T. Ioakimidis and D. Kehagias. 'EXA2PRO: A Framework for High Development Productivity on Heterogeneous Computing Systems'. In: *IEEE Transactions on Parallel and Distributed Systems*. Special Section on Innovative R&D toward the Exascale Era (Aug. 2021). DOI: 10.1109/TPDS.2021.3104257. URL: https://hal.inria.fr/hal-03318644.

[9] J. Protze, M.-A. Hermanns, M. S. Müller, V. M. Nguyen, J. Jaeger, E. Saillard, P. Carribault and D. Barthou. 'MPI detach — Towards automatic asynchronous local completion'. In: *Parallel Computing* 109 (Mar. 2022), p. 102859. DOI: 10.1016/j.parco.2021.102859. URL: https://hal-cea.archives-ouvertes.fr/cea-03537990.

[10] A. Santana, V. Freitas, M. Castro, L. Lima Pilla and J.-F. Méhaut. 'ARTful: A model for user-defined schedulers targeting multiple high-performance computing runtime systems'. In: *Software: Practice and Experience* (4th Apr. 2021). DOI: 10.1002/spe.2977. URL: https://hal.archives-ouvertes.fr/hal-02454426.

International peer-reviewed conferences

[11] G. Agosta, D. Cattaneo, W. Fornaciari, A. Galimberti, G. Massari, F. Reghenzani, F. Terraneo, D. Zoni, C. Brandolese, M. Celino et al. 'TEXTAROSSA: Towards EXtreme scale Technologies and Accelerators for euROhpc hw/Sw Supercomputing Applications for exascale'. In: DSD 2021 - 24th Euromicro Conference on Digital System Design. Palermo / Virtual, Italy, 1st Sept. 2021. URL: https://hal.inria.fr/hal-03329640.

[12] T. C. Aitkaci, M. Sergent, E. Saillard, D. Barthou and G. Papauré. 'Dynamic Data Race Detection for MPI-RMA Programs'. In: EuroMPI 2021 - European MPI Users's Group Meeting. Munich, Germany, 15th May 2021. DOI: 10.1145/1122445.1122456. URL: https://hal.archives-ouvertes.fr/hal-03374614.

[13] A. Cassagne, M. Leonardon, R. Tajan, C. Leroux, C. Jégo, O. Aumage and D. Barthou. 'A Flexible and Portable Real-time DVB-S2 Transceiver using Multicore and SIMD CPUs'. In: The 11th IEEE International Symposium on Topics in Coding (ISTC 2021). Montréal, Canada, 30th Aug. 2021. DOI: 10.1109/ISTC49272.2021.9594063. URL: https://hal.archives-ouvertes.fr/hal-03336450.

[14] M. Gonthier, L. Marchal and S. Thibault. 'Locality-Aware Scheduling of Independent Tasks for Runtime Systems'. In: COLOC - 5th workshop on data locality - 27th International European Conference on Parallel and Distributed Computing. Lisbon, Portugal: Springer, 30th Aug. 2021, pp. 1–12. URL: https://hal.archives-ouvertes.fr/hal-03290998.

[15] M. Laurent, E. Saillard and M. Quinson. 'The MPI BUGS INITIATIVE: a Framework for MPI Verification Tools Evaluation'. In: Correctness 2021: Fifth International Workshop on Software Correctness for HPC Applications. St. Louis, United States, 19th Nov. 2021, pp. 1–9. URL: https://hal.inria.fr/hal-03474762.

Edition (books, proceedings, special issue of a journal)

[16] O. Aumage, P. Carpenter and S. Benkner. *Task-Based Performance Portability in HPC: Maximising long-term investments in a fast evolving, complex and heterogeneous HPC landscape*. 6th Oct. 2021. DOI: 10.5281/zenodo.5549731. URL: https://hal.inria.fr/hal-03368013.

Doctoral dissertations and habilitation theses

[17] I. Daoudi. 'Performance Modelling and Simulation of OpenMP Applications'. Université de Bordeaux, 21st Sept. 2021. URL: https://tel.archives-ouvertes.fr/tel-03416335.

Reports & preprints

[18] I. Daoudi, S. Thibault and T. Gautier. *Draft: sOMP: NUMA and cache-aware simulations for task-based applications*. RR-9400. Inria, 22nd Mar. 2021, p. 25. URL: https://hal.inria.fr/hal-03177026.

[19] M. Gonthier, L. Marchal and S. Thibault. *Locality-Aware Scheduling of Independant Tasks for Runtime Systems*. RR-9394. Inria Grenoble -Rhône-Alpes, 2021, p. 21. URL: https://hal.inria.fr/hal-03144290.

12.3 Other

Softwares

[20] [SW] M. Belaoucha, C. Alias, D. Barthou and S. Touati, *FADAlib: an open source C++ library for fuzzy array dataflow analysis*, 25th Nov. 2021. LIC: GNU Lesser General Public License v3.0 or later. HAL: ⟨hal-03445991⟩, URL: https://hal.archives-ouvertes.fr/hal-03445991, SWHID: ⟨swh:1:dir:fc7481ee438316b9ce5b273ca894114bf658d3d9;origin=https://hal.archives-ouvertes.fr/hal-03445991;visit=swh:1:snp:518f2d28a2d2a1ad15ee2f630b40be3e24a0f8b1;anchor=swh:1:rel:488f5aa5aaa21fc92f24f0f7c9b571e56f1325ec;path=/⟩.

12.4 Cited publications

[21] J. Bachan, S. B. Baden, S. Hofmeyr, M. Jacquelin, A. Kamil, D. Bonachea, P. H. Hargrove and H. Ahmed. 'UPC++: A High-Performance Communication Framework for Asynchronous Computation'. In: *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. 2019, pp. 963–973. DOI: 10.1109/IPDPS.2019.00104.

[22] A. Cassagne. 'Optimization and parallelization methods for software-defined radio'. Theses. Université de Bordeaux, Dec. 2020. URL: https://tel.archives-ouvertes.fr/tel-03118420.

[23] A. Cassagne, O. Aumage, D. Barthou, C. Leroux and C. Jego. 'MIPP: a Portable C++ SIMD Wrapper and its use for Error Correction Coding in 5G Standard'. In: *The 4th Workshop on Programming Models for SIMD/Vector Processing (WPMVP 2018)*. Vienna, Austria: ACM Press, Feb. 2018. DOI: 10.1145/3178433.3178435. URL: https://hal.inria.fr/hal-01888010.

[24] A. Lasserre, R. Namyst and P.-A. Wacrenier. 'EASYPAP: a Framework for Learning Parallel Programming'. In: *2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. 2020, pp. 276–283. DOI: 10.1109/IPDPSW50202.2020.00059.