RESEARCH CENTRE

Bordeaux - Sud-Ouest

IN PARTNERSHIP WITH:

Institut Polytechnique de Bordeaux,
Université de Bordeaux

2020 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 Team: 2015 January 01, updated into 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]
- 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, Associate Professor, HDR]
- Pierre-André Wacrenier [Univ de Bordeaux, Associate Professor]

Post-Doctoral Fellow

- Pierre Huchant [Inria, until Jul 2020]

PhD Students

- Celia Ait Kaci Tassadit [Bull]
- Carsten Bruns [Univ Côte d'Azur, Sep 2020]
- Adrien Cassagne [Inria, until Sep 2020]
- Baptiste Coye [Inria]
- Idriss Daoudi [Inria]
- Romain Lion [Inria]
- Gwenole Lucas [Inria]
- Van Man Nguyen [CEA]

Technical Staff

- Nathalie Furmento [CNRS, Engineer]
- Kun He [Inria, Engineer]
- Alexis Juven [Inria, Engineer, until Nov 2020]
- Mariem Makni [Inria, Engineer]
- Chiheb Sakka [Inria, Engineer, from Oct 2020]

Interns and Apprentices

- Alexis Bandet [Inria, from Jun 2020 until Jul 2020]
- Luca Bourroux [Inria, from Jun 2020 until Jul 2020]
- Vincent Bridonneau [Univ de Bordeaux, from Feb 2020 until Aug 2020]
- Léo Counilh [Inria, from Mar 2020 un]
- Marc Duclusaud [Inria, until Jan 2020]
- Valentin Gaisset [Inria, from Jun 2020 until Jul 2020]
- Maxime Gonthier [Inria, from Apr 2020 until Jul 2020]
- Paul Grenet [Inria, Feb 2020]
- Ethan Leclerc [Inria, Feb 2020]
- Kentin Moultson [Inria, Mar 2020]
- Radjasouria Vinayagame [Inria, until Jan 2020]

Administrative Assistant

- Sabrina Duthil [Inria]

Visiting Scientists

- Amina Guermouche [Polytech Sorbonne, until Jul 2020]
- Georgios Tzanos [Université nationale et capodistrienne d'Athènes, until Apr 2020]

External Collaborators

- Scott Baden [Université de Californie, from Oct 2020]
- Hugo Brunie [Université de Californie, from Apr 2020]
- Adrien Cassagne [Univ de Bordeaux, from Oct 2020]
- Jean-Marie Couteyen [Airbus]
- Amina Guermouche [Polytech Sorbonne, from Aug 2020]

2 Overall objectives

A successful approach to deal with the complexity of modern architectures is centered around the use of runtime systems, to manage tasks dynamically, these runtime systems being either generic or specific to an application. Similarly, on the compiler side, optimizations and analyses are more aggressive in iterative compilation frameworks, suitable for library generations or DSL, in particular for linear algebra methods. To go beyond this state of the art and alleviate the difficulties for programming these 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 two sub-objectives, the first concerning the expression of parallelism itself, the second the optimization and adaptation of this parallelism by compilers and runtimes.

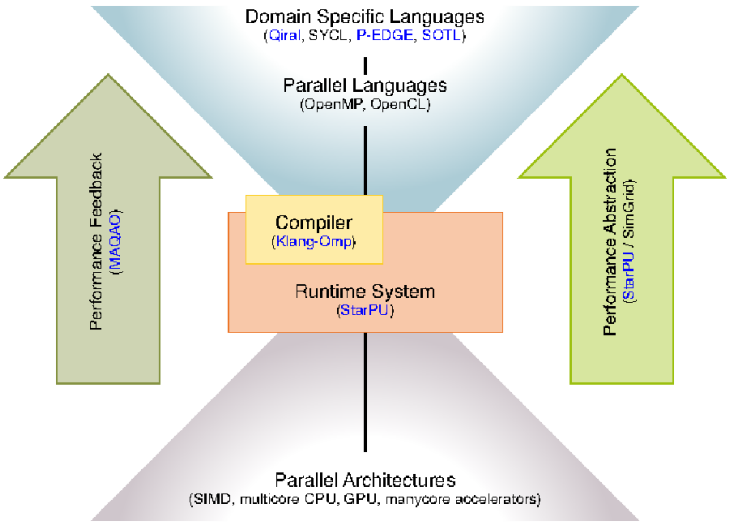

Figure 1: STORM Big Picture

- Expressing parallelism: As shown in the following figure, we propose to work on parallelism expression through Domain Specific Languages, able to capture the essence of the algorithms used through usual parallel languages such as OpenCL, OpenMP and through high performance libraries. The DSLs 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.

- Optimizing and adapting parallelism: The goal here is to leverage the necessary adaptation to evolving hardware, by providing mechanisms allowing users to run the same code on different architectures. This implies to adapt parallelism, in particular the granularity of the work, to the architecture. This relies on the use of existing parallel libraries and their composition, and more generally 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 scheduling them is part of this work.

  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 compiling, 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, embedded applications are also able to take advantage of these performance increase. 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 with the forthcoming 5G protocol, new software and adaptative solutions relying on low power multicores are also explored. 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. The same consideration applies for big data problems, of internet of things, where small sensors provide large amount of data that need to be processed in short amount of time. 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 mobile, embedded systems will be hindered by the complexity and high level of expertise required to tap the performance offered by future parallel heterogeneous architectures.

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 (e.g., OpenGL or DirectX are well-established examples of runtime systems dedicated to hardware-accelerated graphics) 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, FFT, 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. Providing part of the algorithmic knowledge of an application, for instance knowing that it computes a 5-point stencil and then performs a dot product, would lead to more opportunities to adapt parallelism, memory structures, and is a way to leverage the evolving hardware. Besides, with the need for automatic optimization comes the need for *feed-back* to the user, corresponding to the need to debug the code and also to understand what the runtime has performed. Here the compiler plays also a central role in the analysis of the code, and the instrumentation of the program given to the runtime.

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. The objective of STORM is to propose better interactions between compiler and runtime and more semantics for both approaches.

The results of the team on-going research in 2020 reflect this focus. Results presented in Sections 8.15, 8.11 and 8.10 correspond to efforts for higher abstractions through DSL or libraries, and decouple algorithmics from parallel optimizations. Results described in Sections 8.2 and 8.16 provide feed-back information, through visualization and errors detection for parallel executions. The work described in Sections 8.3, 8.4, 8.5,8.12, 8.6 and 8.13 focus in particular on StarPU and its development in order to better abstract architecture, resilience and optimizations. The work presented Section 8.1 aims to help developers with optimization.

Finally, Sections 8.14 and 8.17 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.

4 Application domains

4.1 Application domains benefiting from HPC

The application domains of this research are the following:

- Bioinformatics (see ADT Gordon 10.4.2)
- Environment, in particular CO_2 capture (see Exa2PRO, 10.3.1)
- Health and heart disease analysis (see EXACARD, 10.4.1)
- Software infrastructures for Telecommunications (see AFF3CT, 8.10, 10.4.2)
- Aeronautics (collaboration with Airbus, J.-M. Couteyen)

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 BigData (ANR SOLHAR 10.4.1, Inria ADT SwLoc and Gordon 10.4.2, IPL HAC-SPECIS and BigData 10.4.3, PIA project ELCI 10.4, H2020 projects INTERTWinE and Exa2Pro 10.3.1 and PRACE project PRACE5IP 10.3.1).

5 Social and environmental responsibility

5.1 Impact of research results

The research performed in the context of the EXA2PRO project (10.3.1) 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

6.1 Awards

Our paper that combines the theoretical memory-constrained scheduling [5], and the practical implementation, was selected as Outstanding paper of the APDCM 2020 workshop, and thus selected for publication in extended version in the International Journal on Networking and Computing [2].

7 New software and 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

Authors: Emmanuel Agullo, Mathieu Faverge, Samuel Thibault, Florent Pruvost

Contacts: Emmanuel Agullo, Florent Pruvost, Mathieu Faverge, Samuel Thibault

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 hwloc

Name: Hardware Locality

Keywords: NUMA, Multicore, GPU, Affinities, Open MPI, Topology, HPC, Locality

Functional Description: Hardware Locality (hwloc) is a library and set of tools aiming at discovering and exposing the topology of machines, including processors, cores, threads, shared caches, NUMA memory nodes and I/O devices. It builds a widely-portable abstraction of these resources and exposes it to applications so as to help them adapt their behavior to the hardware characteristics. They may consult the hierarchy of resources, their attributes, and bind task or memory on them.

hwloc targets many types of high-performance computing applications, from thread scheduling to placement of MPI processes. Most existing MPI implementations, several resource managers and task schedulers, and multiple other parallel libraries already use hwloc.

News of the Year: hwloc 2.1 brought support for modern multi-die processors and memory-side caches. It also enhanced memory locality in heterogeneous memory architecture (e.g. with non-volatile memory DIMMs). The visualization of many-core platforms was also improved by factorizing objects when many of them are identical.

URL: http://www.open-mpi.org/projects/hwloc/

Publications: inria-00429889, hal-00985096, hal-01183083, hal-01330194, hal-01400264, hal-01402755, hal-01644087, hal-02266285

Authors: Brice Goglin, Samuel Thibault

Contact: Brice Goglin

Participants: Brice Goglin, Valentin Hoyet

Partners: Open MPI consortium, Intel, AMD, IBM

7.1.3 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

Authors: Olivier Aumage, Yanis Khorsi, Philippe Virouleau, Pierrick Brunet, Samuel Pitoiset, Thierry Gautier

Contacts: Olivier Aumage, Thierry Gautier

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

7.1.4 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).

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

Contacts: Adrien Cassagne, Denis Barthou, Olivier Aumage, Camille Leroux, Christophe Jégo

Partner: IMS

7.1.5 SwLoc

Name: Software Contexts for Locality

Keywords: HPC, Locality, Contexts, Multicore, GPU

Functional Description: SwLoc is a library for flexible and generic partitioning of computing resources (processors, accelerators) to be able to co-excute confined parallel regions which can rely on different runtime systems (e.g. OpenMP, Intel TBB, StarPU, etc.). With all different hypervisor strategies, It is possible to adapt dynamically the computing resources of each context, in order to match each parallel region's need as closely as possible.

URL: http://swloc.gforge.inria.fr/web/

Contacts: Corentin Salingue, Raymond Namyst

7.1.6 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: http://vite.gforge.inria.fr/

Authors: Mathieu Faverge, Nicolas Richart, Cédric Augonnet

Contact: Mathieu Faverge

Participant: Mathieu Faverge

7.1.7 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

Authors: Emmanuelle Saillard, Denis Barthou, Pierre Huchant, Radjasouria Vinayagame

Contact: Emmanuelle Saillard

Participants: Emmanuelle Saillard, Denis Barthou, Pierre Huchant

Partner: CEA

7.1.8 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-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

Authors: Simon Archipoff, Cédric Augonnet, Olivier Aumage, Guillaume Beauchamp, William Braik, Bérenger Bramas, Alfredo Buttari, Adrien Cassagne, Arthur Chevalier, Jérôme Clet-Ortega, Terry Cojean, Nicolas Collin, Ludovic Courtès, Yann Courtois, Jean-Marie Couteyen, Vincent Danjean, Alexandre Denis, Lionel Eyraud-Dubois, Nathalie Furmento, Brice Goglin, David Antonio Gomez Jauregui, Sylvain Henry, Andra Hugo, Mehdi Juhoor, Thibaud Lambert, Erwan Leria, Xavier Lacoste, Mathieu Lirzin, Benoît Lize, Benjamin Lorendeau, Antoine Lucas, Brice Mortier, Stojce Nakov, Raymond Namyst, Lucas Leandro Nesi, Joris Pablo, Damien Pasqualinotto, Samuel Pitoiset, Nguyen Quôc-Dinh, Cyril Roelandt, Anthony Roy, Chiheb Sakka, Corentin Salingue, Lucas Schnorr, Marc Sergent, Anthony Simonet, Luka Stanisic, Ludovic Stordeur, Guillaume Sylvand, Francois Tessier, Samuel Thibault, Leo Villeveygoux, Pierre-André Wacrenier

Contacts: Samuel Thibault, Nathalie Furmento, Olivier Aumage

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

8 New results

8.1 Automatic Code Motion to Extend MPI Nonblocking Overlap Window

One possible approach to reduce MPI communication overheads is to overlap communications with computations. MPI allows this solution through its nonblocking communication mode: a nonblocking communication is composed of an initialization and a completion call. It is then possible to overlap the communication by inserting computations between these two calls. The use of nonblocking collective calls is however still marginal and adds a new layer of complexity. As part of Van Man Nguyen PhD thesis, we have developed an automatic static optimization that (i) transforms blocking MPI communications into their nonblocking counterparts and (ii) performs extensive code motion to increase the size of overlapping intervals between initialization and completion calls [10].

8.2 Static MPI Nonblocking and Persistent Communication Validation

MPI nonblocking and persistent communications are an important part of the MPI standard. Because of their design, nonblocking and persistent operations ask for extra care when using them, and especially the handling of arguments given to them. It is the responsibility of the user when inserting procedure calls to abide to the MPI standard rules. We have developped an extension of PARCOACH static analysis to detect misuse of MPI nonblocking and persistent communications. The new analysis adds the detection of four new error classes related to these types of communications: Missing wait, unmatched wait, request overwriting and buffer data race. This work is part of Van Man Nguyen PhD thesis [11].

8.3 Performance monitoring and Steering Framework

Two frameworks were developed within the context of the project H2020 EXA2PRO to offer performance monitoring and steering APIs into the StarPU runtime system, to be targeted by external tools.

The performance monitoring framework enables StarPU to export performance counters in a generic, safe, extensible way, to give external tools access to internal metrics and statistics, such as the peak number of tasks in the dependence waiting queue, the cumulated execution time by worker thread, and the number of ready tasks of a given kind waiting for an execution slot.

The performance steering framework defines a set of runtime-actionable knobs that can be used to steer the execution of an application on top of StarPU from an external tool, with similar properties of genericity, safety and extensibility as the performance monitoring framework.

8.4 Task scheduling with memory constraints

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 devised a new scheduling strategy which reorders 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.

8.5 Leveraging compiler analysis for task scheduling

Polyhedral analysis of task graph submission loops allow to get at compilation time a representation of the task graph, and perform insightful analyses thanks to the obtained overview of the whole task graph. Task scheduling heuristics, on the other hand, usually keep only a limited view over the task graph, to avoid prohibitive algorithmic costs. We have started to collaborate with the CASH Inria team to transfer some of the insights of the compiler to the compiler. We have notably made the compiler automatically compute a cut of the task graph below which the availability parallelism is lower than the capacities of the target hardware. The scheduler can then at that point switch between a heuristic which privileges task acceleration, and a heuristic which privileges the critical path. Only preliminary results have been obtained so far.

8.6 Failure Tolerance for StarPU

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 aims 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) [8].

8.7 Energy-aware task scheduling in StarPU

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.

8.8 FPGA support in StarPU

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.

8.9 Scheduling iterative task graph for video games

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.

8.10 AFF3CT

The AFF3CT library 7.1.4, 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 [16].

8.11 Matlab API for AFF3CT

As part of the AFF3CT development action, an API compatible with the Matlab mathematical software was designed on top of the AFF3CT fast forward error correction toolbox to allow the library to be used in a high-level manner, directly from the Matlab environment. Due to the relatively large number of classes exported by AFF3CT, an automatized process was designed to let AFF3CT classes be wrapped adequately for Matlab's MEX interface to external libraries.

8.12 HPC Big Data Convergence

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.

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 paved the road to achieve this while avoiding data copies.

8.13 Hierarchical Tasks

We are continuing our work, on the partitioning of the data and the prioritization of task graphs to optimize the use of resources of a machine. Hierarchical tasks allow a better control over the submission of an application's task graph by allowing to dynamically adapt the granularity of the calculations. In the ANR project Solharis, hierarchical tasks are proposed as a solution for a better management of dynamic task graphs. We have continued to explore new solutions for maximizing the performance of hierarchical tasks.

8.14 ADT Gordon

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.

8.15 StarPU in Julia

Julia is a modern language for parallelism and simulation that aims to ease the effort for developing high performance codes. In this context, we carry on the development of a StarPU binding inside Julia. It is possible to launch StarPU tasks inside Julia, either given as libraries, or described in Julia directly. The tasks described in Julia are compiled into either source OpenMP code or CUDA code. We improved further the support of StarPU in Julia. On different benchmarks running on multicore only, the performance obtained of Julia + StarPU is similar to the performance of C StarPU and similar to a pure C code calling Intel MKL library.

8.16 Simulation of OpenMP task based programs

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. These results have been published in [6].

8.17 OpenMP enabled version of Chameleon

An OpenMP enabled version of the Chameleon linear algebra library was designed within the context of European Project PRACE-5IP. This enables the Chameleon library to be available on any platform for which an OpenMP compiler is installed, without any requirement for third party task-based runtime

systems. A preliminary support of the OpenMP port for heterogeneous, accelerated platform was also designed as part of this work.

9 Bilateral contracts and grants with industry

9.1 Bilateral contracts with industry

- Contract with ATOS/Bull for the PhD CIFRE of Tassadit Célia Ait Kaci (2019-2021),
- Contract with Airbus for 1 year, renewable, on StarPU in Flusepa code (2019-), for the engineer contract of Alexis Juven,
- Contract with CEA for the PhD of Van Man Nguyen (2019-2021) and other short contracts.

10 Partnerships and cooperations

10.1 International initiatives

10.1.1 Inria International Labs

COHPC

Title: Correctness and Performance of HPC Applications

Duration: 2019 - 2021

Coordinator: Emmanuelle Saillard

Partner: Lawrence Berkeley National Laboratory (United States)

Inria contact: Emmanuelle Saillard

Summary: High Performance Computing (HPC) plays an important role in many fields like health, materials science, security or environment. The current supercomputer hardware trends lead to more complex HPC applications (heterogeneity in hardware and combinations of parallel programming models) that pose programmability challenges. As indicated by a recent US DOE report, progress to Exascale stresses the requirement for convenient and scalable debugging and optimization methods to help developers fully exploit the future machines; despite all recent advances these still remain manual complex tasks.

This collaboration aims to develop 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.

10.2 International research visitors

10.2.1 Visits of international scientists

Amina Guermouche, assistant professor at Telecom SudParis, visited the team 8 months till July 2020.

10.3 European initiatives

10.3.1 FP7 & H2020 Projects

EXA2PRO

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

Duration: May 2018 - July 2021

Coordinator: INSTITUTE OF COMMUNICATION AND COMPUTER SYSTEMS (Greece)

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 (Greece)
- LINKOPINGS UNIVERSITET (Sweden)
- MAXELER TECHNOLOGIES LIMITED (United Kingdom)
- 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 programming 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. EXA2PRO will be evaluated using 4 use cases from 4 different domains, which will be deployed in JUELICH supercomputing center. The use cases will leverage the EXA2PRO tool-chain and we expect:

- Increased applications performance based on EXA2PRO optimization tools (data and memory management).
- Efficient exploitation of heterogeneity by the applications that will allow the evaluation of more complex problems.
- Identification of trade-offs between design qualities (source code maintainability/reusability) and run-time constraints (performance/energy consumption).
- Evaluation of various fault-tolerance mechanisms for applications with different characteristics.

EXA2PRO outcome is expected to have major impact in a) the scientific and industrial community that focuses on application deployment in supercomputing centers: EXA2PRO environment will allow efficient application deployment with reduced effort. b) on application developers of exascale application: EXA2PRO will provide tools for improving source code maintainability/ reusability, which will allow application evaluation with reduced developers' effort. c) on the scientific community and the industry relevant to the EXA2PRO use cases. At least two of the EXA2PRO use cases will have significant impact to the CO2 capture and to the Supercapacitors industry.

PRACE-6IP

Title: PRACE 6th Implementation Phase Project

Duration: 05/2019 - 12/2021

Coordinator: FORSCHUNGSZENTRUM JULICH GMBH

Partners:

- AKADEMIA GORNICZO-HUTNICZA IM. STANISLAWA STASZICA W KRAKOWIE (Poland)
- ASSOCIACAO DO INSTITUTO SUPERIOR TECNICO PARA A INVESTIGACAO E DESENVOLVIMENTO (Portugal)
- ASSOCIATION NATIONAL CENTRE FOR SUPERCOMPUTING APPLICATIONS (Bulgaria)
- BARCELONA SUPERCOMPUTING CENTER - CENTRO NACIONAL DE SUPERCOMPUTACION (Spain)
- BAYERISCHE AKADEMIE DER WISSENSCHAFTEN (Germany)
- BILKENT UNIVERSITESI VAKIF (Turkey)
- CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE CNRS (France)
- CENTRUM SPOLOCNYCH CINNOSTI SLOVENSKEJ AKADEMIE VIED (Slovakia)
- CINECA CONSORZIO INTERUNIVERSITARIO (Italy)
- COMMISSARIAT A L ENERGIE ATOMIQUE ET AUX ENERGIES ALTERNATIVES (France)
- DANMARKS TEKNISKE UNIVERSITET (Denmark)
- EIDGENOESSISCHE TECHNISCHE HOCHSCHULE ZUERICH (Switzerland)
- FORSCHUNGSZENTRUM JULICH GMBH (Germany)
- FUNDACION PUBLICA GALLEGA CENTRO TECNOLOGICO DE SUPERCOMPUTACION DE GALICIA (Spain)
- GEANT VERENIGING (Netherlands)
- GRAND EQUIPEMENT NATIONAL DE CALCUL INTENSIF (France)
- Gauss Centre for Supercomputing (GCS) e.V. (Germany)
- ISTANBUL TEKNIK UNIVERSITESI (Turkey)
- KOBENHAVNS UNIVERSITET (Denmark)
- KORMANYZATI INFORMATIKAI FEJLESZTESI UGYNOKSEG (Hungary)
- KUNGLIGA TEKNISKA HOEGSKOLAN (Sweden)
- LINKOPINGS UNIVERSITET (Sweden)
- MACHBA - INTERUNIVERSITY COMPUTATION CENTER (Israel)
- MAX-PLANCK-GESELLSCHAFT ZUR FORDERUNG DER WISSENSCHAFTEN EV (Germany)
- NATIONAL INFRASTRUCTURES FOR RESEARCH AND TECHNOLOGY (Greece)
- NATIONAL UNIVERSITY OF IRELAND GALWAY (Ireland)
- NORGES TEKNISK-NATURVITENSKAPELIGE UNIVERSITET NTNU (Norway)
- PARTNERSHIP FOR ADVANCED COMPUTING IN EUROPE AISBL (Belgium)
- POLITECHNIKA GDANSKA (Poland)
- POLITECHNIKA WROCLAWSKA (Poland)
- SYDDANSK UNIVERSITET (Denmark)
- TECHNISCHE UNIVERSITAET WIEN (Austria)
- THE CYPRUS INSTITUTE (Cyprus)

- THE UNIVERSITY OF EDINBURGH (UK)
- UNINETT SIGMA2 AS (Norway)
- UNITED KINGDOM RESEARCH AND INNOVATION (UK)
- UNIVERSIDADE DE COIMBRA (Portugal)
- UNIVERSIDADE DE EVORA (Portugal)
- UNIVERSIDADE DO MINHO (Portugal)
- UNIVERSIDADE DO PORTO (Portugal)
- UNIVERSITAET INNSBRUCK (Austria)
- UNIVERSITAET STUTTGART (Germany)
- UNIVERSITE DU LUXEMBOURG (Luxembourg)
- UNIVERSITEIT ANTWERPEN (Belgium)
- UNIVERSITETET I OSLO (Norway)
- UNIVERZA V LJUBLJANI (Slovenia)
- UPPSALA UNIVERSITET (Sweden)
- VSB - Technical University of Ostrava (Czech Republic)

Inria contact: through GRAND EQUIPEMENT NATIONAL DE CALCUL INTENSIF

Summary: PRACE, the Partnership for Advanced Computing is the permanent pan-European High Performance Computing service providing world-class systems for world-class science. Systems at the highest performance level (Tier-0) are deployed by Germany, France, Italy, Spain and Switzerland, providing researchers with more than 17 billion core hours of compute time. HPC experts from 25 member states enabled users from academia and industry to ascertain leadership and remain competitive in the Global Race. Currently PRACE is finalizing the transition to PRACE 2, the successor of the initial five year period. 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 recognised 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. The proven project structure will be used to achieve each of the objectives in 7 dedicated work packages. The activities are designed to increase Europe's research and innovation potential especially through: seamless and efficient Tier-0 services and a pan-European HPC ecosystem including national capabilities; promoting take-up by industry and new communities and special offers to SMEs; assistance to PRACE 2 development; proposing strategies for deployment of leadership systems; collaborating with the ETP4HPC, CoEs and other European and international organisations on future architectures, training, application support and policies. This will be monitored through a set of KPIs.

10.4 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.

10.4.1 ANR

ANR SOLHARIS (https://www.irit.fr/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.

10.4.2 ADT - Inria Technological Development Actions

ADT Gordon

Participants Denis Barthou, Nathalie Furmento, Samuel Thibault, Pierre-André Wacrenier.

- Inria ADT Campaign 2018, 11/2018 - 11/2020 (24 months)
- Coordinator: Emmanuel Jeannot (Tadaam)
- Other partners: HiePACS, PLEIADE, Tadaam (Inria Bordeaux)

- Abstract: Teams HiePACS, Storm and Tadaam develop each a brick of an HPC software stack, namely solver, runtime, and communication library. The goal of the Gordon project is to consolidate the HPC stack, to improve interfaces between each brick, and to target a better scalability. The bioinformatics application involved in the project has been selected so as to stress the underlying systems.

ADT AFF3CT Matlab

Participants Denis Barthou, Olivier Aumage, Adrien Cassagne, Kun He.

- Inria ADT Campaign 2018, 03/2019 - 03/2020 (12 months)
- Coordinator: Denis Barthou
- Other partners: C. Jégo and C. Leroux (IMS lab, U.Bordeaux)
- Abstract: AFF3CT is a toolchain for designing, validation and experimentation of new Error Correcting codes. This toolchain is written in C++, and this constitutes a difficulty for many industrial users, who are mostly electronics engineers. The goal of this ADT is to widen the number of possible users by designing a Matlab and Python interface for AFF3CT, in collaboration with existing users, and proposing a parallel framework in OpenMP.

10.4.3 IPL - Inria Project Lab

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

Participants Samuel Thibault, Emmanuelle Saillard, Olivier Aumage, Idriss Daoudi.

- 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 http://hacspecis.gforge.inria.fr/ 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)

Participant Samuel Thibault.

- 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 https://project.inria.fr/hpcbigdata/ 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)

Participants Denis Barthou.

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

Since Nov 2020, the team is participating to the EQIP https://project.inria.fr/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.

10.5 Regional initiatives

HPC-scalable-ecosystem (HPC SCALABLE ECOSYSTEM)

Participants Denis Barthou, Marie-Christine Counilh, Nathalie Furmento, Samuel Thibault, Pierre-André Wacrenier.

- Regional Council 2018 - 2020
- Coordinator: Emmanuel Agullo (team HiePACS)

  Numerical simulation is today integrated in all cycles of scientific design and studies, whether academic or industrial, to predict or understand the behavior of complex phenomena often coupled or multi-physical. The quality of the prediction requires having precise and adapted models, but also to have computation algorithms efficiently implemented on computers with architectures in permanent evolution. Given the ever increasing size and sophistication of simulations implemented, the use of parallel computing on computers with up to several hundred thousand computing cores and consuming / generating massive volumes of data becomes unavoidable; this domain corresponds to what is now called High Performance Computing (HPC). On the other hand, the digitization of many processes and the proliferation of connected objects of all kinds generate ever-increasing volumes of data that contain multiple valuable information; these can only be highlighted through sophisticated treatments; we are talking about Big Data. The intrinsic complexity of these digital treatments requires a holistic approach with collaborations of multidisciplinary teams capable of mastering all the scientific skills required for each component of this chain of expertise.

  To have a real impact on scientific progress and advances, these skills must include the efficient management of the massive number of compute nodes using programming paradigms with a high level of expressiveness, exploiting high-performance communications layers, effective management for intensive I / O, efficient scheduling mechanisms on platforms with a large number of computing units and massive I / O volumes, innovative and powerful numerical methods for analyzing volumes of data produced and efficient algorithms that can be integrated into applications representing recognized scientific challenges with high societal and economic impacts. The project we propose aims to consider each of these links in a consistent, coherent and consolidated way.

  For this purpose, we propose to develop a unified Execution Support (SE) for large-scale numerical simulation and the processing of large volumes of data. We identified four Application Challenges (DA) identified by the Nouvelle-Aquitaine region that we propose to carry over this unified support. We will finally develop four Methodological Challenges (CM) to evaluate the impact of the project. This project will make a significant contribution to the

emerging synergy on the convergence between two yet relatively distinct domains, namely High Performance Computing (HPC) and the processing, management of large masses of data (Big Data); this project is therefore clearly part of the emerging field of High Performance Data Analytics (HPDA).

11 Dissemination

11.1 Promoting scientific activities

11.1.1 Scientific events: organisation

General chair, scientific chair

- Emmanuelle Saillard was associate chair of the C3PO workshop.

11.1.2 Scientific events: selection

Member of the conference program committees

- Emmanuelle Saillard: Correctness workshop, SC (Performance track)
- Samuel Thibault: Compas 20, HCW 20, SC 20, TBAA 20
- Olivier Aumage: CCGRID 2020, ICPADS 2020, SCAsia 2020

Reviewer STORM members have conducted reviewing activities for the following conferences and workshops: Correctness workshop, Supercomputing (SC), CCGRID 2020, ICPADS 2020, SCAsia 2020, ICPP 2020, ICPADS 2020.

Reviewer - reviewing activities STORM members have conducted reviewing activities for the following journals: TCSI, ACM TACO, ACM TOPC, Elsevier PARCO.

11.1.3 Invited talks

- Samuel Thibault: SIAM-PP (Seattle, Feb 2020), GTK2020 (Bruxelles, January 2020)
- Olivier Aumage: SIAM-PP(Seattle, Feb 2020), ONERA (visio, Sep 2020)

11.1.4 Scientific expertise

- Denis Barthou: reviewer for H2020 ERC Starting Grant projects

11.1.5 Research administration

- Olivier Aumage: Permanent Contact for Team STORM.
- Emmanuelle Saillard: Member of the Commission de délégation at Bordeaux-Sud-Ouest Inria Research Centre.

11.2 Teaching - Supervision - Juries

11.2.1 Teaching

- Engineering School: Emmanuelle Saillard, Introduction to Algorithms, 16HeCI, L3, ENSEIRB-MATMECA.
- Engineering School: Emmanuelle Saillard, Tree Structure, 16HeCI, L3, ENSEIRB-MATMECA.
- Engineering School: Emmanuelle Saillard, Languages of parallelism, 12HeC, M2, ENSEIRB-MATMECA.

- 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, 32HeTD, L1, University of Bordeaux.
- Licence: Samuel Thibault, Networking, 51HeTD, Licence Pro, University of Bordeaux.
- Licence: Samuel Thibault, Networking, 48HeTD, Licence 2, University of Bordeaux.
- Licence: Samuel Thibault, Compilation, 24HeTD, Licence 3, University of Bordeaux.
- Master: Samuel Thibault, Operating Systems, 24HeTD, L1, 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.

11.2.2 Supervision

- PhD in progress: Tassadit Célia Ait Kaci, March 2019, Emmanuelle Saillard, Marc Sergent, Denis Barthou.
- PhD in progress: Paul Beziau, October 2018, Raymond Namyst.
- PhD in progress: Idriss Daoudi, October 2018, Samuel Thibault, Thierry Gautier.
- PhD in progress: Romain Lion, October 2018, Samuel Thibault.
- PhD in progress: Gwenolé Lucas, November 2019, Raymond Namyst, Abdou Guermouche.
- PhD in progress: Van Man Nguyen, November 2019, Emmanuelle Saillard, Julien Jaeger, Denis Barthou, Patrick Carribault.

- PhD in progress: Maxime Gonthier, October 2020, Samuel Thibault, Loris Marchal.
- PhD in progress: Baptiste Coye, March 2020, Denis Barthou and Raymond Namyst.
- Internship: Mathieu Laurent, September 2020 - April 2021, Emmanuelle Saillard, Martin Quinson.
- Internship: Vincent Bridonneau, January 2020 - August 2020, Emmanuelle Saillard.
- Internship: Radjasouria Vinayagame, November 2019 - January 2020, Emmanuelle Saillard.
- Internship: Maxime Gonthier, April 2020 - July 2020.
- PhD completed: Adrien Cassagne, December 2020, Olivier Aumage, Denis Barthou, Christophe Jégo, Camille Leroux

11.2.3 Juries

- Emmanuelle Saillard was member of the following committees:
  - Associate Professor position at the University of Perpignan, Apr. and May 2020,
  - PhD of Jérémy Lagravière (reviewer): “The PGAS Programming Model and Mesh Based Computation: an HPC Challenge”, University of Oslo, June 2020,
  - PhD of Tao Chang (member): “Evaluation of programming models for manycore and / or heterogeneous architectures for Monte Carlo neutron transport codes”, Institut Polytechnique de Paris, December 2020.
- Olivier Aumage was member of the following committees:
  - Inria CRCN/ISFP selection committee for Inria BSO Research Center
- Marie-Christine Counilh was member of the following committees :
  - Associate Professor position at the University of Perpignan, Apr. and May 2020
  - ATER (Attaché Temporaire d’Enseignement et de Recherche) positions at the University of Bordeaux, May and Nov. 2020
- Denis Barthou was member of the following committees:
  - HDR of Olivier Aumage, U.Bordeaux, Dec. 2020 (member)
  - PhD of Riyane SID LAKHDAR, Dec. 2020, U. Grenoble Alpes (reviewer)

11.3 Popularization

11.3.1 Internal or external Inria responsibilities

- Samuel Thibault : Organization of a meet-a-researcher day for ENS Lyon, November 2020

11.3.2 Education

- Emmanuelle Saillard has created an unplugged activity: The Great Pyramid of China. This activity aims at introducing High performance computing to middle- and high-school students while doing maths.

11.3.3 Interventions

- Emmanuelle Saillard:
  - Unithé ou café, virtual, April 2020.
  - Welcoming ENS-Rennes undergraduate students, Inria, January 2020.
  - Welcoming ENS-Lyon undergraduate students, virtual, November 2020.
  - Circuit scientifique bordelais "hors les murs", October 2020, Nontron.
  - Chiche!, October 2020, Magendie high school, Bordeaux

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.

12.2 Publications of the year

International journals

[2] 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* (2020), pp. 1–29. DOI: 10.15803/ijnc.11.1_27. URL: https://hal.inria.fr/hal-03029847.

[3] J.-C. Papin, C. Denoual, L. Colombet and R. Namyst. 'SPAWN: An Iterative, Potentials-Based, Dynamic Scheduling and Partitioning Tool'. In: *International Journal of Parallel Programming* (15th Sept. 2020). DOI: 10.1007/s10766-020-00677-9. URL: https://hal.archives-ouvertes.fr/hal-03052422.

[4] R. Prat, T. Carrard, L. Soulard, O. Durand, R. Namyst and L. Colombet. 'AMR-based molecular dynamics for non-uniform, highly dynamic particle simulations'. In: *Computer Physics Communications* 253 (Aug. 2020), p. 107177. DOI: 10.1016/j.cpc.2020.107177. URL: https://hal.inria.fr/hal-03157035.

International peer-reviewed conferences

[5] G. Bathie, L. Marchal, Y. Robert and S. Thibault. 'Revisiting dynamic DAG scheduling under memory constraints for shared-memory platforms'. In: IPDPS - 2020 - IEEE International Parallel and Distributed Processing Symposium Workshops. New Orleans / Virtual, United States, 18th May 2020, pp. 1–10. DOI: 10.1109/IPDPSW50202.2020.00102. URL: https://hal.inria.fr/hal-03024626.

[6] I. Daoudi, P. Virouleau, T. Gautier, S. Thibault and O. Aumage. 'sOMP: Simulating OpenMP Task-Based Applications with NUMA Effects'. In: *OpenMP: Portable Multi-Level Parallelism on Modern Systems (IWOMP 2020)*. IWOMP 2020 - 16th International Workshop on OpenMP. Vol. 12295. LNCS. Austin / Virtual, United States, 1st Sept. 2020. DOI: 10.1007/978-3-030-58144-2_13. URL: https://hal.inria.fr/hal-02933803.

[7] A. Denis, E. Jeannot, P. Swartvagher and S. Thibault. 'Using Dynamic Broadcasts to improve Task-Based Runtime Performances'. In: Euro-Par - 26th International European Conference on Parallel and Distributed Computing. Euro-Par 2020. Warsaw, Poland: https://2020.euro-par.org/, 24th Aug. 2020. DOI: 10.1007/978-3-030-57675-2_28. URL: https://hal.inria.fr/hal-02872765.

[8] R. Lion and S. Thibault. 'From tasks graphs to asynchronous distributed checkpointing with local restart'. In: FTXS 2020 - IEEE/ACM 10th Workshop on Fault Tolerance for HPC at eXtreme Scale. Atlanta / Virtual, United States, 11th Nov. 2020. DOI: 10.1109/FTXS51974.2020.00009. URL: https://hal.archives-ouvertes.fr/hal-02970529.

[9] R. A. d. S. Lopes, S. Thibault and A. C. M. A. d. Melo. 'MASA-StarPU: Parallel Sequence Comparison with Multiple Scheduling Policies and Pruning'. In: SBAC-PAD 2020 - IEEE 32nd International Symposium on Computer Architecture and High Performance Computing. Porto, Portugal: https://sbac2020.dcc.fc.up.pt/, 8th Sept. 2020. DOI: 10.1109/SBAC-PAD49847.2020.00039. URL: https://hal.inria.fr/hal-02914793.

[10] V.-M. Nguyen, E. Saillard, J. Jaeger, D. Barthou and P. Carribault. 'Automatic Code Motion to Extend MPI Nonblocking Overlap Window'. In: C3PO'20 Workshop - First Workshop on Compiler-Assisted Correctness Checking and Performance Optimization for HPC. Vol. 12321. High Performance Computing: Part of the Lecture Notes in Computer Science book series (LNCS). Frankfurt / Virtual, Germany, 20th Oct. 2020, pp. 43–54. DOI: 10.1007/978-3-030-59851-8_4. URL: https://hal-cea.archives-ouvertes.fr/cea-03010533.

[11] V.-M. Nguyen, E. Saillard, J. Jaeger, D. Barthou and P. Carribault. 'PARCOACH Extension for Static MPI Nonblocking and Persistent Communication Validation'. In: *2020 IEEE/ACM 4th International Workshop on Software Correctness for HPC Applications (Correctness)*. Correctness 2020: Fourth International Workshop on Software Correctness for HPC Applications. Atlanta / Virtual, United States, 11th Nov. 2020. DOI: 10.1109/Correctness51934.2020.00009. URL: https://hal-cea.archives-ouvertes.fr/cea-03014171.

[12] M. Potse, E. Saillard, D. Barthou and Y. Coudière. 'Feasibility of Whole-Heart Electrophysiological Models With Near-Cellular Resolution'. In: CinC 2020 - Computing in Cardiology. Rimini / Virtual, Italy: https://www.cinc2020.org/, 16th Sept. 2020. URL: https://hal.inria.fr/hal-02943513.

[13] G. Tzanos, V. Soni, C. Prouveur, M. Haefele, S. Zouzoula, L. Papadopoulos, S. Thibault, N. Vandenbergen, D. Pleiter and D. Soudris. 'Applying StarPU runtime system to scientific applications: Experiences and lessons learned'. In: POMCO 2020 - 2nd International Workshop on Parallel Optimization using/for Multi- and Many-core High Performance Computing. Barcelona / Virtual, Spain, 10th Dec. 2020. URL: https://hal.inria.fr/hal-02985721.

Conferences without proceedings

[14] S. Thibault, L. Stanisic and A. Legrand. 'Faithful Performance Prediction of a Dynamic Task-based Runtime System, an Opportunity for Task Graph Scheduling'. In: SIAM PP 2020 - SIAM Conference on Parallel Processing for Scientific Computing. Seattle, United States: http://www.siam.org/meetings/pp20/, 12th Feb. 2020. URL: https://hal.inria.fr/hal-02943753.

Doctoral dissertations and habilitation theses

[15] O. Aumage. 'Instruments of Productivity for High Performance Computing'. Université de Bordeaux (UB), France, 14th Dec. 2020. URL: https://hal.inria.fr/tel-03105625.

[16] A. Cassagne. 'Optimization and parallelization methods for software-defined radio'. Université de Bordeaux, 8th Dec. 2020. URL: https://tel.archives-ouvertes.fr/tel-03118420.

Reports & preprints

[17] G. Bathie, L. Marchal, Y. Robert and S. Thibault. *Revisiting dynamic DAG scheduling under memory constraints for shared-memory platforms*. Inria, Feb. 2020. URL: https://hal.inria.fr/hal-02488399.

[18] M. Gonthier, L. Marchal and S. Thibault. *Locality-Aware Scheduling of Independant Tasks for Runtime Systems*. Inria, 2021. URL: https://hal.inria.fr/hal-03144290.

[19] A. Lasserre, R. Namyst and P.-A. Wacrenier. *EASYPAP: a Framework for Learning Parallel Programming*. 6th Feb. 2020. URL: https://hal.archives-ouvertes.fr/hal-02469919.

[20] P. Radojkovic, M. Marazakis, P. Carpenter, R. Jeyapaul, D. Gizopoulos, M. Schulz, A. Armejach, E. A. Ayguade, F. Bodin, R. Canal, F. Cappello, F. Chaix, G. Colin De Verdiere, S. Derradji, S. Di Carlo, C. Engelmann, I. Laguna, M. Moreto, O. Mutlu, L. Papadopoulos, O. Perks, M. Ploumidis, B. Salami, Y. Sazeides, D. Soudris, Y. Sourdis, P. Stenstrom, S. Thibault, W. Toms and O. Unsal. *Towards Resilient EU HPC Systems: A Blueprint*. European HPC resilience initiative, Apr. 2020. URL: https://hal.inria.fr/hal-02922257.

12.3 Other

Softwares

[21] [SW] S. Archipoff, C. Augonnet, O. Aumage, G. Beauchamp, B. Bramas, A. Buttari, A. Cassagne, J. Clet-Ortega, T. Cojean, N. Collin, V. Danjean, A. Denis, L. Eyraud-Dubois, N. Furmento, S. Henry, A. Hugo, M. Juhoor, A. Juven, M. Keryell-Even, Y. Khorsi, T. Lambert, E. Leria, B. Lizé, M. Makni, S. Nakov, R. Namyst, L. Nesi Lucas, P. Joris, D. Pasqualinotto, S. Pitoiset, Q.-D. Nguyen, C. Roelandt, C. Sakka, C. Salingue, L. Mello Schnorr, M. Sergent, A. Simonet, L. Stanisic, B. Subervie, F. Tessier, S. Thibault, B. Videau, L. Villeveygoux and P.-A. Wacrenier, *StarPU* version 1.3.3, 20th Jan. 2020. HAL: ⟨hal-02443512⟩, URL: https://hal.inria.fr/hal-02443512, SWHID: ⟨swh:1:dir:b6e19d99449a78805e7a55a341fbaba2bc431973;origin=https://hal.archives-ouvertes.fr/hal-02443512;visit=swh:1:snp:c21d3dfbd96e4fb502c534e59644dba14c542100;anchor=swh:1:rev:31be198773f103324593d26369f135fbde5b97f8;path=/⟩.