IN PARTNERSHIP WITH:
Institut Polytechnique de Bordeaux

Université de Bordeaux

Activity Report 2019

Project-Team STORM

STatic Optimizations, Runtime Methods

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

RESEARCH CENTER
Bordeaux - Sud-Ouest

THEME
Distributed and High Performance Computing

Table of contents

Project-Team STORM

Creation of the Team: 2015 January 01, updated into Project-Team: 2017 July 01

Keywords:

Computer Science and Digital Science:

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, 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 Fellows

Pierre Huchant [Inria, Post-Doctoral Fellow, from Apr 2019]
Philippe Virouleau [Inria, Post-Doctoral Fellow, until Apr 2019]

PhD Students

Tassadit Célia Ait Kaci [ATOS, PhD Student, from Mar 2019]
Hugo Brunie [Univ de Bordeaux, PhD Student, until Mar 2019]
Adrien Cassagne [Inria, PhD Student]
Baptiste Coye [Inria, PhD Student, from Nov 2019]
Idriss Daoudi [Inria, PhD Student]
Pierre Huchant [Univ de Bordeaux, PhD Student, until Mar 2019]
Romain Lion [Inria, PhD Student, from Oct 2019]
Arthur Loussert [CEA, PhD Student, until Oct 2019]
Gwenolé Lucas [Inria, PhD Student, from Nov 2019]
Van Man Nguyen [CEA, PhD Student, from Nov 2019]
Raphaël Prat [CEA, PhD Student, until Oct 2019]
Leo Villeveygoux [Univ de Bordeaux, PhD Student, until Jan 2019]

Technical staff

Nathalie Furmento [CNRS, Engineer]
Kun He [Inria, Engineer, from Apr 2019]
Alexis Juven [Inria, Engineer, from Dec 2019]
Mariem Makni [Inria, Engineer, from Apr 2019]
Corentin Salingue [Inria, Engineer, until Sep 2019]
Philippe Virouleau [Inria, Engineer, from May 2019 until Sep 2019]

Interns and Apprentices

Marc Duclusaud [Inria, from Nov 2019]
Nathan Edely [Inria, from Jul 2019 until Aug 2019]
Mael Keryell [Inria, until Apr 2019]
Quentin Kerzazi Vidal [Inria, Oct 2019]
Alice Lasserre [Inria, from May 2019 until Jul 2019]
Gwenolé Lucas [Inria, from Mar 2019 until Sep 2019]
Firmin Martin [Inria, from Jun 2019 until Jul 2019]
Louise Mercier [Inria, from Jul 2019 until Aug 2019]
Mehdi Naciri [Inria, from Mar 2019 until Jun 2019]
Gaspard Soulier Maitreau [Inria, Oct 2019]
Radjasouria Vinayagame [Inria, from Nov 2019]

Administrative Assistant

Sabrina Blondel-Duthil [Inria, Administrative Assistant]

Visiting Scientist

Baptiste Coye [no employer, from May 2019 until Sep 2019]

External Collaborators

Jean-Marie Couteyen [Airbus]
Amina Guermouche [Telecom Sud-Paris, from Dec 2019]

2. Overall Objectives

2.1. 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.

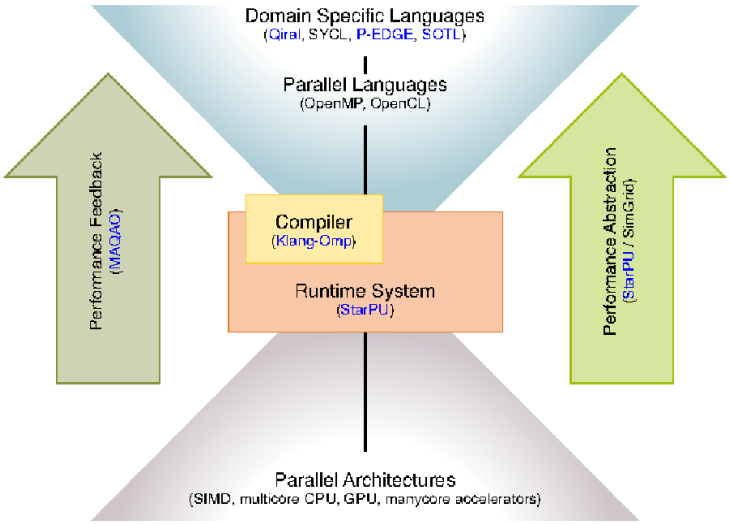

Figure 1. STORM Big Picture

- 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 2019 reflect this focus. Results presented in Sections 7.11, 7.15, 7.10 and 7.9 correspond to efforts for higher abstractions through DSL or libraries, and decouple algorithmics from parallel optimizations. Results in Section 7.8 correspond to efforts on parallelism expression and again abstraction, starting from standard parallel programming languages. Results described in Sections 7.1 and 7.16 provide feed-back information, through visualization and deadlock detection for parallel executions. The work described in Sections 7.3, 7.4, 7.5, 7.6,7.12, 7.7 and 7.13 focus in particular on StarPU and its development in order to better abstract architecture, resilience and optimizations. The work presented Section 7.2 aims to help developers with optimization.

Finally, Sections 7.14 and 7.17 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 9.2.2)
- Environment, in particular CO_2 capture (see Exa2PRO, 9.3.1)
- Health and heart disease analysis (see EXACARD, 9.2.1)
- Software infrastructures for Telecommunications (see AFF3CT, 7.9, 9.2.2)
- 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 9.2.1, Inria ADT SwLoc and Gordon 9.2.2, IPL HAC-SPECIS and BigData 9.2.3, PIA project ELCI 9.2, H2020 projects INTERTWinE and Exa2Pro 9.3.1 and PRACE project PRACE5IP 9.3.1).

5. Highlights of the Year

5.1. Highlights of the Year

- 1st year for the CoHPC team in collaboration with the Lawrence Berkeley National Lab. (9.4.1.1). A PhD student of the team obtained a joint postdoc with InriaSiliconValley at this lab to initiate the collaboration.

6. New Software and Platforms

6.1. 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 FUNCTIONAL DESCRIPTION: 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)

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

- Partners: Innovative Computing Laboratory (ICL) - King Abdullha University of Science and Technology - University of Colorado Denver
- Contact: Emmanuel Agullo
- URL: https://gitlab.inria.fr/solverstack/chameleon

6.2. hwloc

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.

- Participants: Brice Goglin and Valentin Hoyet
- Partners: Open MPI consortium - Intel - AMD - IBM
- Contact: Brice Goglin
- Publications: hwloc: a Generic Framework for Managing Hardware Affinities in HPC Applications - Managing the Topology of Heterogeneous Cluster Nodes with Hardware Locality (hwloc) - A Topology-Aware Performance Monitoring Tool for Shared Resource Management in Multicore Systems - Exposing the Locality of Heterogeneous Memory Architectures to HPC Applications - Towards the Structural Modeling of the Topology of next-generation heterogeneous cluster Nodes with hwloc - On the Overhead of Topology Discovery for Locality-aware Scheduling in HPC - Memory Footprint of Locality Information on Many-Core Platforms - M&MMs: Navigating Complex Memory Spaces with hwloc
- URL: http://www.open-mpi.org/projects/hwloc/

6.3. KaStORS

The KaStORS OpenMP Benchmark Suite

KEYWORDS: Benchmarking - HPC - Task-based algorithm - Task scheduling - OpenMP - Data parallelism

FUNCTIONAL DESCRIPTION: The KaStORS benchmarks suite has been designed to evaluate implementations of the OpenMP dependent task paradigm, introduced as part of the OpenMP 4.0 specification.

- Participants: François Broquedis, Nathalie Furmento, Olivier Aumage, Philippe Virouleau, Pierrick Brunet, Samuel Thibault and Thierry Gautier
- Contact: Thierry Gautier
- URL: http://kastors.gforge.inria.fr/#!index.md

6.4. KStar

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.

- Participants: Nathalie Furmento, Olivier Aumage, Philippe Virouleau and Samuel Thibault
- Contact: Olivier Aumage
- Publications: Bridging the gap between OpenMP and task-based runtime systems for the fast multipole method - Bridging the gap between OpenMP 4.0 and native runtime systems for the fast multipole method - Evaluation of OpenMP Dependent Tasks with the KASTORS Benchmark Suite
- URL: http://kstar.gforge.inria.fr/#!index.md

6.5. AFF3CT

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).

- Authors: Adrien Cassagne, Bertrand Le Gal, Camille Leroux, Denis Barthou and Olivier Aumage
- Partner: IMS
- Contact: Adrien Cassagne
- Publications: AFF3CT: A Fast Forward Error Correction Toolbox! - AFF3CT : Un environnement de simulation pour le codage de canal - Toward High-Performance Implementation of 5G SCMA Algorithms - An Efficient, Portable and Generic Library for Successive Cancellation Decoding of Polar Codes - Beyond Gbps Turbo Decoder on Multi-Core CPUs - Energy Consumption Analysis of Software Polar Decoders on Low Power Processors - Fast and Flexible Software Polar List Decoders - Fast Simulation and Prototyping with AFF3CT
- URL: https://aff3ct.github.io/

6.6. MORSE

KEYWORDS: High performance computing - Matrix calculation - Fast multipole method - Runtime system

FUNCTIONAL DESCRIPTION: MORSE (Matrices Over Runtime Systems @ Exascale) is a scientific project, its objectives are to solve matrix problems on complex architectures, using runtime systems. More specifically, the goal is to write codes that reach a high level of performance for all architectures. The algorithms are written independently of the architecture, and the runtime system dispatches the different computational parts to the different computing units. This methodology has been validated on three classes of problems: dense linear algebra, sparse and dense, and fast multipole methods. The corresponding codes have been incorporated into several softwares, MAGMA, Pastix and ScalFMM.

- Contact: Emmanuel Agullo
- URL: http://icl.cs.utk.edu/morse/

6.7. SwLoc

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.

- Contact: Corentin Salingue
- URL: http://swloc.gforge.inria.fr/

6.8. VITE

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.

- Participant: Mathieu Faverge
- Contact: Mathieu Faverge
- URL: http://vite.gforge.inria.fr/

6.9. PARCOACH

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.

- Participants: Emmanuelle Saillard, Denis Barthou and Pierre Huchant
- Partner: CEA
- Contact: Emmanuelle Saillard
- Publications: Combining Static and Dynamic Validation of MPI Collective Communication - PARCOACH: Combining static and dynamic validation of MPI collective communications - Static Validation of Barriers and Worksharing Constructs in OpenMP Applications - Correctness Analysis of MPI-3 Non-Blocking Communications in PARCOACH - Static/Dynamic Validation of MPI Collective Communications in Multi-threaded Context - MPI Thread-Level Checking for MPI+OpenMP Applications - PARCOACH Extension for Hybrid Applications with Interprocedural Analysis - PARCOACH Extension for a Full-Interprocedural Collectives Verification - Multi-Valued Expression Analysis for Collective Checking
- URL: https://esaillar.github.io/PARCOACH/

6.10. StarPU

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.

- Participants: Corentin Salingue, Andra Hugo, Benoît Lize, Cédric Augonnet, Cyril Roelandt, François 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 and Leo Villeveygoux
- Contact: Olivier Aumage
- Publications: Asynchronous Task-Based Execution of the Reverse Time Migration for the Oil and Gas Industry - A Compiler Algorithm to Guide Runtime Scheduling - Achieving high-performance with a sparse direct solver on Intel KNL - Modeling Irregular Kernels of Task-based codes: Illustration with the Fast Multipole Method - Scheduling of Dense Linear Algebra Kernels on Heterogeneous Resources - Critical resources management and scheduling under StarPU - Achieving High Performance on Supercomputers with a Sequential Task-based Programming Model - Programmation of heterogeneous architectures using moldable tasks - The StarPU Runtime System at Exascale ? : Scheduling and Programming over Upcoming Machines. - A Visual Performance Analysis

Framework for Task-based Parallel Applications running on Hybrid Clusters - Analyzing Dynamic Task-Based Applications on Hybrid Platforms: An Agile Scripting Approach - Detecção de Anomalias de Desempenho em Aplicações de Alto Desempenho baseadas em Tarefas em Clusters Híbridos - Resource aggregation for task-based Cholesky Factorization on top of heterogeneous machines - On Runtime Systems for Task-based Programming on Heterogeneous Platforms - Resource aggregation in task-based applications over accelerator-based multicore machines - Controlling the Memory Subscription of Distributed Applications with a Task-Based Runtime System - Exploiting Two-Level Parallelism by Aggregating Computing Resources in Task-Based Applications Over Accelerator-Based Machines - Exploiting Two-Level Parallelism by Aggregating Computing Resources in Task-Based Applications Over Accelerator-Based Machines - Achieving High Performance on Supercomputers with a Sequential Task-based Programming Model - Bridging the gap between OpenMP 4.0 and native runtime systems for the fast multipole method - Scalability of a task-based runtime system for dense linear algebra applications - Faithful Performance Prediction of a Dynamic Task-Based Runtime System for Heterogeneous Multi-Core Architectures - Towards seismic wave modeling on heterogeneous many-core architectures using task-based runtime system - Bridging the Gap between Performance and Bounds of Cholesky Factorization on Heterogeneous Platforms - Composing multiple StarPU applications over heterogeneous machines: A supervised approach - Evaluation of OpenMP Dependent Tasks with the KASTORS Benchmark Suite - A runtime approach to dynamic resource allocation for sparse direct solvers - Modeling and Simulation of a Dynamic Task-Based Runtime System for Heterogeneous Multi-Core Architectures - Toward OpenCL Automatic Multi-Device Support - Harnessing clusters of hybrid nodes with a sequential task-based programming model - Taking advantage of hybrid systems for sparse direct solvers via task-based runtimes - Modulariser les ordonnanceurs de tâches : une approche structurelle - Overview of Distributed Linear Algebra on Hybrid Nodes over the StarPU Runtime - StarPU-MPI: Task Programming over Clusters of Machines Enhanced with Accelerators - Modeling and Simulation of a Dynamic Task-Based Runtime System for Heterogeneous Multi-Core Architectures - Taking advantage of hybrid systems for sparse direct solvers via task-based runtimes - Adaptive Task Size Control on High Level Programming for GPU/CPU Work Sharing - Composing multiple StarPU applications over heterogeneous machines: a supervised approach - Implementation of FEM Application on GPU with StarPU - Le problème de la composition parallèle : une approche supervisée - Support exécutif scalable pour les architectures hybrides distribuées - SOCL: An OpenCL Implementation with Automatic Multi-Device Adaptation Support - C Language Extensions for Hybrid CPU/GPU Programming with StarPU - Programming Models and Runtime Systems for Heterogeneous Architectures - Programmation unifiée multi-accélérateur OpenCL - StarPU-MPI: Task Programming over Clusters of Machines Enhanced with Accelerators - Parallelization on Heterogeneous Multicore and Multi-GPU Systems of the Fast Multipole Method for the Helmholtz Equation Using a Runtime System - High-Level Support for Pipeline Parallelism on Many-Core Architectures - Programmability and Performance Portability Aspects of Heterogeneous Multi-/Manycore Systems - Programmation des architectures hétérogènes à l'aide de tâches divisibles - StarPU: a unified platform for task scheduling on heterogeneous multicore architectures - PEPPHER: Efficient and Productive Usage of Hybrid Computing Systems - The PEPPHER Approach to Programmability and Performance Portability for Heterogeneous many-core Architectures - Flexible runtime support for efficient skeleton programming on hybrid systems - LU Factorization for Accelerator-based Systems - QR Factorization on a Multicore Node Enhanced with Multiple GPU Accelerators - Programmation multi-accélérateurs unifiée en OpenCL - Détection optimale des coins et contours dans des bases d'images volumineuses sur architectures multicœurs hétérogènes - Association de modèles de programmation pour l'exploitation de clusters de GPUs dans le calcul intensif - Programming heterogeneous, accelerator-based multicore machines:current situation and main challenges - Scheduling Tasks over Multicore machines enhanced with acelerators: a Runtime System's Perspective - Composabilité de codes parallèles sur architectures hétérogènes - Data-Aware Task Scheduling on Multi-Accelerator based Platforms - Dynamically scheduled Cholesky factorization on multicore architectures with GPU accelerators. - StarPU: a Runtime System for Scheduling Tasks

over Accelerator-Based Multicore Machines - StarPU : un support exécutif unifié pour les architectures multicoeurs hétérogènes - Automatic Calibration of Performance Models on Heterogeneous Multicore Architectures - StarPU: A Unified Platform for Task Scheduling on Heterogeneous Multicore Architectures - Exploiting the Cell/BE architecture with the StarPU unified runtime system - Bridging the gap between OpenMP and task-based runtime systems for the fast multipole method - Composability of parallel codes on heterogeneous architectures - Are Static Schedules so Bad ? A Case Study on Cholesky Factorization - Scheduling of Linear Algebra Kernels on Multiple Heterogeneous Resources - Approximation Proofs of a Fast and Efficient List Scheduling Algorithm for Task-Based Runtime Systems on Multicores and GPUs - Resource aggregation for task-based Cholesky Factorization on top of modern architectures - Visual Performance Analysis of Memory Behavior in a Task-Based Runtime on Hybrid Platforms - Tolérance aux pannes dans l'exécution distribuée de graphes de tâches

- URL: http://starpu.gforge.inria.fr/

7. New Results

7.1. Multi-Valued Expression Analysis for Collective Checking

Determining if a parallel program behaves as expected on any execution is challenging due to non-deterministic executions. Static analysis helps to detect all execution paths that can be executed concurrently by identifying multi-valued expressions, i.e. expressions evaluated differently among processes. This can be used to find collective errors in parallel programs. The PARallel COntrol flow Anomaly CHecker (PARCOACH) framework has been extended with a multi-valued expressions detection to find such errors [9]. The new analysis corrects the previous one and analyzes parallel applications using MPI, OpenMP, UPC and CUDA.

7.2. Hiding the latency of MPI communication

As developers spend significant effort performing manual latency optimizations, the goal of Van Man Nguyen Ph.D Thesis is to automatically provide maximal communication overlap for MPI communication (collective, point-to-point and RMA Put/Get operations). A method that moves operations and their completion (e.g. Isend/Wait) as far apart as possible in the program while preserving memory consistency is under development.

7.3. 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.

7.4. Heterogeneous task scheduling

Taking advantage of heterogeneous systems requires to carefully choose which tasks should be accelerated. Simple heuristics allow to get fairly good performance, but do not have approximation ratio that would provide performance guarantees. The ROMA Inria team designed advanced heuristics which do have approximation ratios. We have implemented one within StarPU and indeed improved the performance over existing heuristics. The implementation required to revamp part of the StarPU toolkit dedicated to writing scheduling heuristics.

7.5. 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. The ROMA Inria team proposed a heuristic to introduce additional dependencies to the task graph enough so that any scheduling order will meet the memory constraint, while avoiding to extend the critical path length. On the other hand, banker algorithms allow to achieve this online, within the task scheduler, but do not have an overall view of the task graph, and may thus severely increase the critical path. We have thus started to design a collaboration between visionary heuristics which take a global but coarse view of the task graph, and online heuristics which have a local but precise view of the task graph.

7.6. 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.

7.7. Failure Tolerance for StarPU

Since supercomputers keep growing in terms of core numbers, the reliability decreases the same way. The project H2020 EXA2PRO 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. An implementation of a solution is currently being developed for evaluation, with an interface that can be easily plugged into StarPU.

7.8. Static and Dynamic Adaptation of Task parallelism

This work is the result of Pierre Huchant PhD thesis, and the objectives are to adapt statically and dynamically OpenCL tasks. The adaptation consists in splitting tasks automatically into multiple sub-tasks, taking into account the heterogeneity of the architecture (sub-tasks are specifically created for one processing unit), the load imbalance within a parallel OpenCL, between the different iterations in space, and if the task graph is repeatedly executed, between the iterations in time, and it takes into account the time of the communications generated by splitting the tasks [2]. The method is able to cope with sequential task graphs (tasks are parallel themselves but scheduled sequentially) and deals with tasks manipulating complex data structures as shown on an N-body particle simulation mini-app.

7.9. AFF3CT

The AFF3CT library, 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. Additional new algorithms have been designed and evaluated within the AFF3CT library [6], [7], and a new approach for generating and exploring automatically high performance error correction codes from matrix description is an on-going work. Besides, an on-going work is on the automatic parallelization of the tasks describing the simulation of a whole chain of signal transmission. In order to be able to make accessible the simulation results obtained with AFF3CT and to be able to replay easily the same simulations, a web interface allowing users to browse through the results, and the simulation setup for a large range of inputs has been designed. This graphical interface is currently in use at IMS by other researchers. Finally, a call for the creation of a consortium on AFF3CT is available on the web page of AFF3CT https://aff3ct.github.io/.

7.10. 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.

7.11. InKS framework

The InKS framework was developed by Ksander Ejjaaouani as part of his Ph.D Thesis co-supervised by the university of Strasbourg, the Maison de la Simulation and the STORM team. It enables separating algorithms of time loop based scientific simulations into platform independent algorithms and platform specific optimisation files, thus enforcing separation of concerns between the algorithmic design process on one side, and the optimization process of specifying platform-dependent aspects such as operation ordering and memory placement on the other side.

7.12. 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.

7.13. 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.

7.14. ADT Gordon

In collaboration with the HIEPACS and TADAAM Inria teams, we are strengthening 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.

7.15. 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, but this is still a work in progress.

7.16. Simulation of OpenMP task based programs

A simulator for OpenMP task based programs is being designed as part of Inria's IPL HAC-Specis project, and the Ph.D Thesis of Idriss Daoudi. The goal is to extend the SimGrid HPC simulation framework with the ability to simulate OpenMP applications.

7.17. 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.

8. Bilateral Contracts and Grants with Industry

8.1. 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 Arthur Loussert (2017-2019), for the PhD of Van Man Nguyen (2019-2021) and other short contracts.

9. Partnerships and Cooperations

9.1. Regional Initiatives

HPC/Big-Data Convergence

- Team participants : Olivier Aumage, Nathalie Furmento, Samuel Thibault.
- Other participants : David Auber, Olivier Beaumont, Lionel Eyraud-Dubois, Gérald Point
- Abstract: The goal of this project is to gather teams from the HPC and Big Data communities to work at the intersection between these domains. We will focus on how StarPU can be adapted to achieve good performances on Big Data platforms.

9.2. 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.

9.2.1. ANR

ANR SOLHAR (http://solhar.gforge.inria.fr/doku.php?id=start).

ANR MONU 2013 Program, 2013 - 2018 (36 months extended)

Identification: ANR-13-MONU-0007

Coordinator: Inria Bordeaux/LaBRI

Other partners: CNRS-IRIT, Inria-LIP Lyon, CEA/CESTA, EADS-IW

Abstract: This project aims at studying and designing algorithms and parallel programming models for implementing direct methods for the solution of sparse linear systems on emerging computers equipped with accelerators. The ultimate aim of this project is to achieve the implementation of a software package providing a solver based on direct methods for sparse linear systems of equations. Several attempts have been made to accomplish the porting of these methods on such architectures; the proposed approaches are mostly based on a simple offloading of some computational tasks (the coarsest grained ones) to the accelerators and rely on fine hand-tuning of the code and accurate performance modeling to achieve efficiency. This project proposes an innovative approach which relies on the efficiency and portability of runtime systems, such as the StarPU tool developed in the runtime team (Bordeaux). Although the SOLHAR project will focus on heterogeneous computers equipped with GPUs due to their wide availability and affordable cost, the research accomplished on algorithms, methods and programming models will be readily applicable to other accelerator devices such as ClearSpeed boards or Cell processors.

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.

9.2.2. ADT - Inria Technological Development Actions

ADT SwLoc (http://swloc.gforge.inria.fr/web/)

Participants: Raymond Namyst, Pierre-André Wacrenier, Andra Hugo, Brice Goglin, Corentin Salingue.

Inria ADT Campaign 2017, 10/2017 - 9/2019 (24 months)

Coordinator: Raymond Namyst

Abstract: The Inria action ADT SwLoc is aiming at developing a library allowing dynamic flexible partitioning of computing resources in order to execute parallel regions concurrently inside the same processes.

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, 12/2018 - 12/2019 (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.

9.2.3. 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.

9.3. European Initiatives

9.3.1. FP7 & H2020 Projects

Exa2PRO

- Title: Enhancing Programmability and boosting Performance Portability for Exascale Computing systems
- Program: H2020-FETHPC
- Duration: May 2018 - April 2021
- Coordinator: ICCS
- Inria contact: Samuel Thibault
- Partners:
  - Institute of Communications and Computer Systems (ICCS) (Greece)
  - Linköpiung University (LIU) (Sweden)
  - Centre for Research and Technology Hellas (CERTH) (Greece)

* Institut National de Recherche en Informatique et en Automatique (Inria) (France)
* Maxeler Technologies Limited (MAX) (UK)
* Forschungszentrum Jülich (JUELICH) (Germany)
* Centre National de la Recherche Scientifique (CNRS) (France)

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.

9.3.2. Collaborations in European Programs, Except FP7 & H2020

PRACE-5IP

- Title: PRACE 5th Implementation Phase
- Program: PRACE
- Duration: 2017 - 2019
- Coordinator: PRACE
- Inria contact for team STORM: Olivier Aumage
- Abstract: The objectives of PRACE-5IP 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 transition to PRACE2 including analysis of TransNational Access;
  * strengthening the internationally recognised PRACE brand;
  * continuing and extend advanced training which so far provided more than 18 800 person-training days;
  * preparing strategies and best practices towards Exascale computing;

* coordinating and enhancing the operation of the multi-tier HPC systems and services;
* 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 6 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;
* implementing a new flexible business model for PRACE 2;
* 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.

9.4. International Initiatives

9.4.1. Inria International Labs

Inria@SiliconValley
Associate Team involved in the International Lab:

9.4.1.1. COHPC

Title: Correctness and Performance of HPC Applications

International Partner (Institution - Laboratory - Researcher):

Lawrence Berkeley National Laboratory (United States) - Costin Iancu

Start year: 2019

See also: https://team.inria.fr/cohpc/

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.

9.5. International Research Visitors

9.5.1. Visits of International Scientists

- Scott Baden, LBNL (USA), from April 29 to May 3, 2019

10. Dissemination

10.1. Promoting Scientific Activities

10.1.1. Scientific Events Selection

10.1.1.1. Member of the Conference Program Committees

- Emmanuelle Saillard: Correctness workshop, Compas 2019, PPAM 2019, OMASE'19, 4PAD'19
- Olivier Aumage: SC Asia 2019, Euro-Par 2019, ICPADS 2019, PAW-ATM
- Samuel Thibault: HCW 19
- Raymond Namyst: SC 19, HiPC 19, CCGRID 19, ISC 19, ICCS 19, PPAM 19, RADR 19

10.1.1.2. Reviewer

STORM members have conducted reviewing activities for the following conferences and workshops: Correctness workshop, Compas, PPAM, OMASE, 4PAD, HCW 19

10.1.2. Journal

10.1.2.1. Reviewer - Reviewing Activities

STORM members have conducted reviewing activities for the following journals: JPDC, IEEE TII

10.1.3. Invited Talks

- Olivier Aumage: Cluster SysNum (Bordeaux, Jan. 24, 2019 and Mar. 8, 2019), 10-Year PlaFRIM (Bordeaux, May. 23, 2019), SPPEXA (Dresden, Germany, Oct. 21, 2019)
- Samuel Thibault: ENS Lyon seminar (Lyon, Jan 2019), ENS Lyon seminar (Lyon, May 2019), Workshop SIAM CSE (Spokane, Feb 2019), Workshop PADAL (Bordeaux, Sep 2019)
- Raymond Namyst: Aristote Seminar about "Toward ExaScale Computing" (Ecole Polytechnique, May 2019)

10.1.4. Scientific Expertise

- Olivier Aumage: Review of one project proposal for the French national research agency (ANR)
- Raymond Namyst: Scientific Advisor for CEA/DAM (French Department of Energy) HPC research groups

10.1.5. Research Administration

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

10.1.6. Standardization Activities

- Olivier Aumage represents Inria at the OpenMP ARB Standardization Committee.

10.2. Teaching - Supervision - Juries

10.2.1. 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, Harmonisation, 10HeTD, 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 new 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.
- 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.

10.2.2. 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: Adrien Cassagne, October 2017, Olivier Aumage, Denis Barthou, Christophe Jégo, Camille Leroux.
- PhD in progress: Idriss Daoudi, October 2018, Olivier Aumage, 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
- Internship: Firmin Martin, June 2019 - August 2019, Emmanuelle Saillard
- Internship: Radjasouria Vinayagame, November 2019 - January 2020, Emmanuelle Saillard
- Internship: Pierre-Antoine Rouby, May - July 2019 , Samuel Thibault
- Internship: Mehdi Naciri, January - August 2019, Adrien Cassagne, Denis Barthou

- Internship: Mael Keryell, January - April 2019, Denis Barthou
- Master's Thesis: Van Man Nguyen, January - September 2019 , Emmanuelle Saillard, Julien Jaeger, Denis Barthou, Patrick Carribault
- Master's Thesis: Gwenolé Lucas, March - Septembre 2019, Pierre-André Wacrenier, Abdou Guermouche, Mathieu Faverge
- PhD completed: Hugo Brunie, March 2019, Denis Barthou
- PhD completed: Pierre Huchant, March 2019, Denis Barthou
- PhD completed: Arthur Loussert, December 2019
- PhD completed: Raphaël Prat, October 2019

10.2.3. Juries

- Marie-Christine Counilh : Ph.D defense of Pierre Huchant at the University of Bordeaux, Mar. 2019 (member)
- Olivier Aumage : Ph.D defense of Ksander Ejjaaouani at the Maison de la simulation in Saclay, Oct. 25, 2019 (invited member)
- Denis Barthou was member of the following PhD/HdR committees:
  - PhD of Fabian Gruber (reviewer): "Debogage de performance pour code binaire: Analyse de sensitivité et profilage de dépendence", Grenoble Alpes University, December 2019
  - HdR of Guillaume Mercier (president): "Évolutions du passage de messages face aux défis de la gestion des topologies matérielles hiérarchiques", University of Bordeaux, December 2019
  - PhD of Maxime Schmitt (reviewer): "Génération automatique de codes adaptatifs", University of Strasbourg, September 2019
  - PhD of Gauthier Sornet (reviewer): "Parallélisme des calculs numériques appliqué aux géosciences", University of Orléans, September 2019
  - PhD of Youenn Lebras (president): "Code optimization based on source to source transformations using profile guided metrics", University of Paris Saclay, July 2019
  - PhD of Filip Arvid Jakobson (reviewer): "Static Analysis for BSPlib programs", University of Orléans, June 2019
  - PhD of Fotis Nicolaidis (president): "TROMOS: a SDK for virtual storage systems", University of Paris Saclay, May 2019
- Raymond Namyst was member of the following committees:
  - PhD of Benjamin Lorendeau (president): "Improving performance of a CFD code with unstructured meshes through multi level parallelism", University of Bordeaux, December 2019
  - PhD of Georgious Christodoulis (president): "Adapting a HPC runtime system to FPGAs", University of Grenoble, December 2019

10.3. Popularization

10.3.1. Internal or external Inria responsibilities

- Samuel Thibault : Organization of a meet-a-researcher day for ENS Lyon, December 2019

10.3.2. Interventions

- Emmanuelle Saillard, Raymond Namyst: Fête de la Science, Inria, October 2019.
- Emmanuelle Saillard: Printemps de la mixité, Inria, April 2019.
- Emmanuelle Saillard, Sabrina Duthil: Forum des métiers, Lycée des graves, April 2019.

- Emmanuelle Saillard: Moi Informaticienne - Moi Mathématicienne, University of Bordeaux, April 2019.
- Olivier Aumage, Nathalie Furmento, Emmanuelle Saillard, Adrien Cassagne: Welcoming ENS-Lyon undergraduate students, Inria, December 2019.
- Olivier Aumage, Marie-Christine Counilh, Emmanuelle Saillard, Kun He: Welcoming Schoolchildren, Inria, December 2019.

10.3.3. Creation of media or tools for science outreach

- Pierre-André Wacrenier and Raymond Namyst have developed a software platform named Easy-PAP to provide students with an easy-to-use programming environment to quickly learn parallel programming (using Pthreads, OpenMP, OpenCL and MPI). The platform has a nice graphical interface which makes it suitable for showing the benefits of High Performance Computing to a broad audience. More information on the web site: https://gforgeron.gitlab.io/easypap/

11. Bibliography

Publications of the year

Doctoral Dissertations and Habilitation Theses

[1] H. BRUNIE. *Data Allocation Optimisation for High Performance Computing Application on Heterogeneous Architecture*, Université de Bordeaux, January 2019, https://tel.archives-ouvertes.fr/tel-02101338

[2] P. HUCHANT. *Static Analysis and Dynamic Adaptation of Parallelism*, Université de Bordeaux, March 2019, https://hal.inria.fr/tel-02429785

[3] A. LOUSSERT. *Understanding and Guiding the Computing Resource Management in a Runtime Stacking Context*, Université de Bordeaux, October 2019, https://hal.inria.fr/tel-02438652

[4] R. PRAT. *Dynamic load balancing on exaflop supercomputer applied to molecular dynamics*, Université de Bordeaux, October 2019, https://tel.archives-ouvertes.fr/tel-02413331

Articles in International Peer-Reviewed Journals

[5] A. CASSAGNE, O. HARTMANN, M. LEONARDON, K. HE, C. LEROUX, R. TAJAN, O. AUMAGE, D. BARTHOU, T. TONNELLIER, V. PIGNOLY, B. LE GAL, C. JEGO. *AFF3CT: A Fast Forward Error Correction Toolbox!*, in "SoftwareX", July 2019, vol. 10, 7 p. , 100345 [*DOI :* 10.1016/J.SOFTX.2019.100345], https://hal.inria.fr/hal-02358306

[6] A. GHAFFARI, M. LEONARDON, A. CASSAGNE, C. LEROUX, Y. SAVARIA. *Toward High-Performance Implementation of 5G SCMA Algorithms*, in "IEEE Access", January 2019, vol. 7, pp. 10402-10414 [*DOI :* 10.1109/ACCESS.2019.2891597], https://hal.archives-ouvertes.fr/hal-01977885

[7] M. LEONARDON, A. CASSAGNE, C. LEROUX, C. JEGO, L.-P. HAMELIN, Y. SAVARIA. *Fast and Flexible Software Polar List Decoders*, in "Journal of Signal Processing Systems", January 2019 [*DOI :* 10.1007/s11265-018-1430-3], https://hal.inria.fr/hal-01987848

International Conferences with Proceedings

[8] A. ALONAZI, H. LTAIEF, D. KEYES, I. SAID, S. THIBAULT. *Asynchronous Task-Based Execution of the Reverse Time Migration for the Oil and Gas Industry*, in "CLUSTER 2019 - IEEE International Conference on Cluster Computing", Albuquerque, United States, IEEE, September 2019, pp. 1-11 [*DOI :* 10.1109/CLUSTER.2019.8891054], https://hal.inria.fr/hal-02403109

[9] P. HUCHANT, E. SAILLARD, D. BARTHOU, P. CARRIBAULT. *Multi-Valued Expression Analysis for Collective Checking*, in "EuroPar", Göttingen, Germany, August 2019, https://hal.archives-ouvertes.fr/hal-02390025

[10] L. LEANDRO NESI, S. THIBAULT, L. STANISIC, L. MELLO SCHNORR. *Visual Performance Analysis of Memory Behavior in a Task-Based Runtime on Hybrid Platforms*, in "CCGrid 2019 - 19th Annual IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing", Larnaca, Cyprus, IEEE, May 2019, pp. 142-151 [*DOI :* 10.1109/CCGRID.2019.00025], https://hal.inria.fr/hal-02275363

National Conferences with Proceedings

[11] R. LION. *Tolérance aux pannes dans l'exécution distribuée de graphes de tâches*, in "COMPAS'2019 - Conférence d'informatique en Parallélisme, Architecture et Système", Anglet, France, June 2019, https://hal.inria.fr/hal-02296118

Conferences without Proceedings

[12] T. C. AITKACI. *Analyse et Optimisations des Applications HPC à mémoire distribuée et globalement adressable*, in "COMPAS'2019 - Conférence d'informatique en Parallélisme, Architecture et Système", Anglet, France, June 2019, https://hal.inria.fr/hal-02429482

Research Reports

[13] C. ALIAS, S. THIBAULT, L. GONNORD. *A Compiler Algorithm to Guide Runtime Scheduling*, Inria Grenoble ; Inria Bordeaux - Sud-Ouest, December 2019, n^{o} RR-9315, https://hal.inria.fr/hal-02421327

Software

[14] 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, J. PABLO, D. PASQUALINOTTO, S. PITOISET, N. QUÔC-DINH, C. ROELANDT, C. SAKKA, C. SALINGUE, M. SCHNORR LUCAS, M. SERGENT, A. SIMONET, L. STANISIC, S. BÉRANGÈRE, F. TESSIER, S. THIBAULT, V. BRICE, L. VILLEVEYGOUX, P.-A. WACRENIER. *StarPU*, January 2020, Version : 1.3.3
[SWH-ID : swh:1:dir:b6e19d99449a78805e7a55a341fbaba2bc431973], Software, https://hal.inria.fr/hal-02443512