Activity Report 2015

Team STORM

STatic Optimizations, Runtime Methods

Inria teams are typically groups of researchers working on the definition of a common project, and objectives, with the goal to arrive at the creation of a project-team. Such project-teams may include other partners (universities or research institutions).

RESEARCH CENTER
Bordeaux - Sud-Ouest

THEME
Distributed and High Performance Computing

Table of contents

Team STORM

Creation of the Team: 2015 January 01

Keywords:

Computer Science and Digital Science:

2.1.10. - Domain-specific languages
2.1.6. - Concurrent programming
2.2.1. - Static analysis
2.2.3. - Run-time systems
2.2.4. - Parallel architectures
2.2.5. - GPGPU, FPGA, etc.
6.2.6. - Optimization
6.2.7. - High performance computing

Other Research Topics and Application Domains:

3.3.1. - Earth and subsoil
4.1.1. - Oil, gas
5.2.3. - Aviation

1. Members

Research Scientist

Olivier Aumage [Inria, Researcher]

Faculty Members

Denis Barthou [Team leader, INP Bordeaux, Professor, HdR]
Marie Christine Counilh [Univ. Bordeaux, Associate Professor]
Raymond Namyst [Univ. Bordeaux, Professor, HdR]
Samuel Thibault [Univ. Bordeaux, Associate Professor]
Pierre Andre Wacrenier [Univ. Bordeaux, Associate Professor]

Engineers

Adrien Cassagne [Univ. Bordeaux, from Mar 2015]
Jerome Clet-Ortega [Inria]
Nathalie Furmento [CNRS]
Alexandre Honorat [Inria, from Feb 2015, until Aug 2015]
Andra Hugo [Inria, until Aug. 2015]
Samuel Pitoiset [Inria]
James Tombi A Mba [Inria, until Dec 2015]

PhD Students

Hugo Brunie [CEA]
Antoine Capra [CEA, until Oct 2015]
Emmanuel Cieren [CEA, until Oct 2015]
Terry Cojean [Inria]
Christopher Haine [Inria]
Pierre Huchant [Univ. Bordeaux, from Feb 2015]
Suraj Kumar [Inria]
Pei Li [Telecom Sud-Paris, until Dec 2015]

Jean-Charles Papin [ENS Cachan, until Apr 2015]
Corentin Rossignon [TOTAL SA, until Mar 2015, granted by CIFRE]
Emmanuelle Saillard [CEA, until Sep 2015, granted by CIFRE]
Marc Sergent [Inria]
Gregory Vaumourin [CEA]

Post-Doctoral Fellow
Luka Stanisic [Inria]

Administrative Assistants
Sylvie Embolla [Inria]
Flavie Tregan [Inria, until Jun 2015]

Others
Thomas Cledel [Univ. Bordeaux, internship, from Jun 2015 until Aug 2015]
Mathieu Lirzin [Univ. Bordeaux, internship, from May 2015 until Jul 2015]
Anthony Simonet [Univ. Bordeaux, internship, from May 2015 until Jul 2015]

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, fit 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.

- 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.

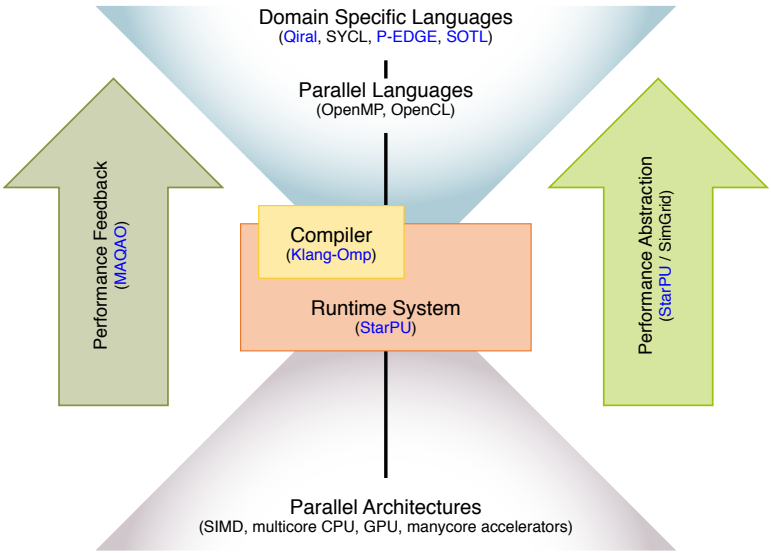

Figure 1. STORM Big Picture

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 at horizon 2020 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. Parallel Computing and Architectures

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 developpers, abstracting away architectural details providing some kind of performance portability. 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. 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.

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. We recall in the following section the expertise of the team.

4. Application Domains

4.1. Supporting Numeric Libraries and Scientific Simulation Applications

The application of our work covers linear algebra, solvers, fast-multipole methods, in collaboration with other Inria teams and with industry. This allows the scientific development of new techniques adapted to these applications, and opens its use in a large variety of physic simulations in high performance computing.

In terms of direct application, the software developped in the team have allowed applications in various physics fields, ranging from sismic, mechanic of fluids, molecular dynamics, high energy physics or material simulations. Similarly, the domains of image processing and signal processing can take advantage of the expertise and software of the team.

5. Highlights of the Year

5.1. Highlights of the Year

STORM received an H2020 FETHPC Grant for taking part in the INTERTWinE European project to be run from Oct. 2015 to Sep. 2018, to promote interoperability between multiple runtime systems and application support layers.

6. New Software and Platforms

6.1. Chameleon

KEYWORDS: HPC - Dense linear algebra - Task-based algorithm - Runtime system - 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 (e.i. 4 min).

- Participants: Marc Sergent, Suraj Kumar, Samuel Thibault, Andra Hugo, Terry Cojean, Nathalie Furmento
- Partners: Innovative Computing Laboratory (ICL) - King Abdullha University of Science and Technology - University of Colorado Denver
- Contact: Emmanuel Agullo
- URL: https://project.inria.fr/chameleon/

6.2. KLANG-OMP

The Klang-Omp OpenMP Compiler
KEYWORDS: Compilers - Task scheduling - OpenMP - Source-to-source compiler - Data parallelism
FUNCTIONAL DESCRIPTION

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

- Participants: Olivier Aumage, Nathalie Furmento, Samuel Pitoiset and Samuel Thibault
- Contact: Olivier Aumage
- URL: http://kstar.gforge.inria.fr/#!index.md

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

6.4. MAQAO

SCIENTIFIC DESCRIPTION

MAQAO relies on binary codes for Intel x86 and ARM architectures. For x86 architecture, it can insert probes for instrumentation directly inside the binary. There is no need to recompile. The static/dynamic approach of MAQAO analysis is the main originality of the tool, combining performance model with values collected through instrumentation.

MAQAO has a static performance model for x86 and ARM architectures. This model analyzes performance of the codes on the architectures and provides some feed-back hints on how to improve these codes, in particular for vector instructions.

The dynamic collection of data in MAQAO enables the analysis of thread interactions, such as false sharing, amount of data reuse, runtime scheduling policy, ...

FUNCTIONAL DESCRIPTION

MAQAO is a performance tuning tool for OpenMP parallel applications. It relies on the static analysis of binary codes and the collection of dynamic information (such as memory traces). It provides hints to the user about performance bottlenecks and possible workarounds.

- Participants: Denis Barthou, Olivier Aumage, Christopher Haine and James Tombi A Mba
- Contact: Denis Barthou
- URL: https://gforge.inria.fr/projects/maqao

6.5. P-EDGE

An Efficient and Portable Library for Error Correction Code Design and Optimization

KEYWORDS: Code generation - Error Correction Code

FUNCTIONAL DESCRIPTION

The P-Edge library joins genericity techniques together with code generation capabilities to enable implementing efficient and portable error correction codes. The genericity offered allows to easily experiment with a large panel of algorithmic variants.

- Authors: Adrien Cassagne, Olivier Aumage, Bertrand Le Gal, Camille Leroux and Denis Barthou
- Partner: IMS
- Contact: Adrien Cassagne

6.6. StarPU

The StarPU Runtime System

KEYWORDS: HPC - Scheduling - GPU - Multicore - Performance

SCIENTIFIC DESCRIPTION

Traditional processors have reached architectural limits which heterogeneous multicore designs and hardware specialization (e.g. 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 (e.g. 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 (e.g. 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: Samuel Thibault, Nathalie Furmento, Jérôme Clet-Ortega, Pierre-André Wacrenier, Terry Cojean, Andra Hugo, Raymond Namyst, Olivier Aumage, Marc Sergent and Samuel Pitoiset.
- Contact: Raymond Namyst
- URL: http://starpu.gforge.inria.fr/

6.7. hwloc

Hardware Locality

KEYWORDS: HPC - Topology - Open MPI - Affinities - GPU

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 the application so as to help them adapt their behavior to the hardware characteristics.

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 .

- Participants: Brice Goglin and Samuel Thibault
- Contact: Brice Goglin
- URL: http://www.open-mpi.org/projects/hwloc/

7. New Results

7.1. Memory Management and Distributed Computing with StarPU

Task-based programming models manage to abstract away much of the architecture complexity while efficiently meeting the performance challenge, even at large scale. While computation scheduling has been well studied, the dynamic management of memory resource subscription inside such run-times has however been little explored, despite the fact that the lookahead, anticipative capabilities offered by the decoupled task submission/task execution steps may sometime come with a high memory cost, especially in distributed context where buffers for receiving incoming contributions have to be accounted for. We therefore studied the cooperation between a task-based application code and a run-time system engine to control the memory subscription levels throughout the execution. We showed that the task paradigm allows to control the memory footprint of the application by throttling the task submission flow rate, striking a compromise between the performance benefits of anticipative task submission and the resulting memory consumption.

7.2. Simulation Execution with StarPU and SimGrid

The combination of StarPU and SimGrid allows to fast, accurate, and reproducible simulation the execution of task-based HPC applications.

This has proved to be very useful for theoretical research on scheduling heuristics [10]. It notably allowed to modify the simulated platform in order to easily observe and understand which parts of the platform (bandwidth, computation power) cause a bottleneck. It also allowed to remove some parts of the problem, such as the cost of data transfers, to simplify the problem and be able to deeply study scheduling solutions and compare them with optimum solutions in a simple environment before tackling the complete platform.

We have also extended the modelization of computation nodes, to take into account the PCI hierarchy of the system. This allows to get a more accurate simulation of systems which have dedicated channels between GPUs.

Last but not least, we have started to extend the StarPU+Simgrid combination to StarPU+MPI+Simgrid, to simulate the execution of HPC applications on *clusters* of heterogeneous systems. The preliminary results seem to show good accuracy. This will allow to easily study how applications scale, and study for instance how network performance have impact on it.

7.3. Scheduling heuristics for dense linear algebra

In the context of Suraj KUMAR's PhD thesis, we are studying the scheduling of the Cholesky factorization on heterogeneous systems.

We have started to introduce communication costs into the constraint programming. Since this increases resolution time a lot, we had to optimize the expression of the data transfers to simplify the resolution. We modified the StarPU runtime system to be able to inject not only a static schedule of tasks, but also a static schedule of data transfers. This allows to inject the schedule optimized by constraint programming into real executions.

We have also shown how static schedules and dynamic scheduling strategies compare on heterogeneous platforms, and notably in the context of varying task execution time can typically be a problem for static scheduling. Static schedules have actually proven to be robust against variation in execution time. We have also studied injecting static information into dynamic schedulers, which improves the resulting performance with little offline analysis.

7.4. Out-of-core support for task graphs

In the context of the Hi-BOX project, Airbus factorizes very large compressed matrices, which can not fit in the main memory, and most of the data thus have to be temporarily transferred to the disk, and loaded on-demand. We have thus extended the StarPU out-of-core support to the case of compressed matrices, and improved the eviction heuristics, so as to transfer data to the disk in advance.

7.5. Parallel Tasks within StarPU

One of the biggest challenge raised by the design of high performance task-based applications on top of heterogeneous accelerator-based machines lies in choosing the adequate granularity of tasks. Indeed, GPUs generally exhibit better performance when executing kernels featuring numerous threads whereas regular CPU cores typically reach their peak performance with fine grain tasks working on a reduced memory footprint. As a consequence, task-based applications running on such heterogeneous platforms have to adopt an intermediate granularity, chosen as a trade-off between coarse-grain and fine-grain tasks. We have tackle this granularity problem via resource aggregation : our approach consists in reducing the performance gap between accelerators and single cores by scheduling parallel tasks over cluster of CPUs. For this purpose, we have extended the concept of scheduling context, which we introduced in a previous work, so that the main runtime system only sees virtual computing resources on which it can schedule parallel tasks (e.g. BLAS kernels). The execution of tasks inside such clusters can virtually rely on any thread-based runtime system, and does not interfere with the outer scheduler. As a proof of concept we allow the interoperability of StarPU and OpenMP to co-manage task parallelism. We showed that our approach is able to outperform the magma, dplasma and chameleon state-of-the-art dense linear algebra libraries when dealing with matrices of small and medium size.

7.6. Running Compliant OpenMP Applications on top of StarPU with the Klang-Omp Compiler

Several robust runtime systems proposed recently have shown the benefits of task-based parallelism models. However, the common weakness of these supports is to tie applications to specific APIs. The OpenMP specification aims at providing a common parallel programming means for shared-memory platforms. It appears a good candidate to address this issue. We assessed the effectiveness and limits of this approach on the ScalFMM library developed by Inria HiePACS team, implementing fast multipole methods (FMM) algorithms. We showed that OpenMP dependent tasks allow for significant performance improvements over OpenMP loops and independent tasks for this application. We also demonstrated that suitable, targeted language extensions can further improve performances by a important margin in some cases.

7.7. Task-based Seismology Simulation on top of StarPU

Understanding three-dimensional seismic wave propagation in complex media is still one of the main challenges of quantitative seismology. Because of its simplicity and numerical efficiency, the finite-differences method is one of the standard techniques implemented to consider the elastodynamics equation. Additionally, this class of modelling heavily relies on parallel architectures in order to tackle large scale geometries including a detailed description of the physics. Last decade, significant efforts have been devoted towards efficient implementation of the finite-differences methods on emerging architectures. These contributions have demonstrated their efficiency leading to robust industrial applications. The growing representation of

heterogeneous architectures combining general purpose multicore platforms and accelerators leads to re-design current parallel application. We thus considered the StarPU task-based runtime system in order to harness the power of heterogeneous CPU+GPU computing nodes. Preliminary results demonstrate significant speedups in comparison with the best implementation suitable for homogeneous cores.

7.8. Interfacing the MPC Parallel Framework with StarPU

CEA has developped a framework named MPC that transforms MPI+OpenMP applications into a lightweight thread-based program which can flexibly and efficiently exploit multicore architectures. StarPU, on its side, is mainly dedicated to schedule coarse grain tasks over accelerators, and is less suited to fine grain task scheduling. We have thus initiated a software interoperability effort between StarPU and MPC. The first step was to implement a new StarPU task scheduling strategy based on a NUMA-aware adaptative task granularity according to the target device (GPU or CPU). We observed significative performance gains for a Cholesky application in comparison to an eager strategy, thanks to the NUMA-aware aspect. However more work is still needed with respect to task decomposition as it implies data partitioning during the execution. We are also working on a variable granularity task programming interface in order to simplify the developer's coding effort. Finally, we develop a mechanism in StarPU to isolate some parts of the computing platform for another runtime. We used nested *scheduling contexts* to redirect some tasks to a scheduling component that StarPU may or may not control. The idea is to associate a scheduling subcontext to a runtime, for instance MPC, that would access to a dedicated set of computing resources for executing parallel kernels.

7.9. A Domain Specific Framework for Executing Stencil Kernels on Accelerated Platforms with SYCL

Stencil kernels arise in many scientific codes as the result from discretizing natural, continuous phenomenons. Many research works have designed stencil frameworks to help programmer optimize stencil kernels for performance, and to target CPUs or accelerators. However, existing stencil kernels, either library-based or language-based necessitate to write distinct source codes for accelerated kernels and for the core application, or to resort to specific keywords, pragmas or language extensions. SYCL is a C++ based approach designed by the Khronos Group to program the core application as well as the application kernels with a single unified, C++ compliant source code. A SYCL application can then be linked with a CPU-only runtime library or processed by a SYCL-enabled compiler to automatically build an OpenCL accelerated application. We designed a stencil-dedicated domain specific embedded language (DSEL) which leverage SYCL together with expression template techniques to implement statically optimized stencil applications able to run on platforms equipped with OpenCL devices, while preserving the single source benefits from SYCL. Our stencil DSL has been tested using the SYCL compiler ComputeCpp from the CodePlay company on an accelerated platform, as well as with the TriSYCL library designed as a compilerless approach for CPU-only prototyping.

7.10. Combining Code Generation and Template Specialization Techniques in the P-EDGE Generic Polar Error Correction Code Framework

Error Correction Code decoding algorithms for consumer products such as *Internet of Things* (IoT) devices are usually implemented as dedicated hardware circuits. As processors are becoming increasingly powerful and energy efficient, there is now a strong desire to perform this processing in software to reduce production costs and time to market. The recently introduced family of Successive Cancellation decoders for Polar codes has been shown in several research works to efficiently leverage the ubiquitous SIMD units in modern CPUs, while offering strong potentials for a wide range of optimizations. Together with the IMS Laboratory, we designed the P-EDGE framework which combines a specialized skeleton generator and a building blocks library routines to provide a generic, extensible Polar code exploration workbench. It enables ECC code designers to easily experiments with combinations of existing and new optimizations, while delivering performance close to state-of-art decoders.

7.11. Binary Kernel Analysis, Hinting and Transformation for SIMDization

SIMD processor units have become ubiquitous. Using SIMD instructions is the key for performance for many applications. Modern compilers have made immense progress in generating efficient SIMD code. However, they still may fail or SIMDize poorly, due to conservativeness, source complexity or missing capabilities. When SIMDization fails, programmers are left with little clues about the root causes and actions to be taken. Our proposed guided SIMDization framework builds on the assembly-code quality assessment toolkit MAQAO to analyzes binaries for possible SIMDization hindrances. It proposes improvement strategies and readily quantifies their impact, using *in vivo* evaluations of suggested transformation. Thanks to our framework, the programmer gets clear directions and quantified expectations on how to improve his/her code SIMDizability. We show results of our technique on TSVC benchmark.

7.12. Dynamic Granularity Adaptation of OpenCL Kernels on Heterogeneous Multi-device Systems

On-going work as part of the PhD of P. Huchant aims to transparently execute an OpenCL kernel, and further a complete task graph, on an heterogeneous multi-device system. We propose methods to split an OpenCL kernel at compile time and adapt its granularity dynamically to ensure load balance. If the kernel is executed multiple times, we propose to determine its granularity by using a linear program whose constraints are built from performance measurements collected during the first invocations of the kernel with predefined granularities. Splitting the execution of one kernel into different executions does not require additional information from the user, therefore increasing the level of portability of OpenCL codes. First experiments show the interest of our approach.

8. Bilateral Contracts and Grants with Industry

8.1. Bilateral Contracts with Industry

CodePlay A Contract has been established between CodePlay and Team Storm to experiment with the ComputeCpp compiler for the SYCL language and OpenCL based framework on hybrid, accelerated architectures.

8.2. Bilateral Grants with Industry

TOTAL SA Total is granting the CIFRE PhD thesis of Corentin Rossignon on Sparse GMRES on heterogeneous platforms in oil extraction simulation from april 2012 to march 2015.

CEA CEA is granting the CIFRE PhD thesis of Emmanuelle Saillard (2012-2015) on Static/Dynamic Analysis for the validation and optimization of parallel applications, Grégory Vaumourin (2013-2016) on Hybrid Memory Hierarchy and Dynamic data optimization for embedded parallel architectures, Emmanuel Cieren (2012-2015) on Molecular Dynamics on Exascale Supercomputers, and Jean-Charles Papin (2013-2016) on Potential-based Dynamic Scheduling techniques and Partitioning tools for domain decomposition simulations.

CEA - REGION AQUITAINE CEA together with the Aquitaine Region Council is funding the PhD thesis of Marc Sergent (2013-2016) on Scalability for Task-based Runtimes.

RAPID HiBOX This contract between IMACS an EADS France aims to develop a state of the art library for fast iterative, direct and hybrid methods, efficient on new heterogeneous parallel and hybrid architectures, that can be used on Boundary Element Methods. Applications targeted are acoustics, elastodynamics and electromagnetism. The contrat grants 2 year engineer for the parallelization of the library based on StarPU.

9. Partnerships and Cooperations

9.1. Regional Initiatives

REGION AQUITAINE - CEA The Aquitaine Region Council together with CEA is funding PhD thesis of Marc Sergent (2013-2016) on Scalability for Task-based Runtimes

Labex CPU The Labex CPU local cluster from the University of Bordeaux is funding the engineer position of Adrien Cassagne (2015-2016) to explore the optimization Error Correction Code (ECC) algorithms and simulation chains from IMS Laboratory using STORM software and expertize, for designing the upcoming 5G mobile phone communication technology.

9.2. National Initiatives

9.2.1. PIA

ELCI The ELCI 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.

9.2.2. ANR

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

ANR MONU 2013 Program, 2013 - 2016 (36 months)

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 Songs Simulation of next generation systems (http://infra-songs.gforge.inria.fr/).

ANR INFRA 2011, 01/2012 - 12/2015 (48 months)

Identification: ANR-11INFR01306

Coordinator: Martin Quinson (Inria Nancy)

Other partners: Inria Nancy, Inria Rhône-Alpes, IN2P3, LSIIT, Inria Rennes, I3S.

Abstract: The goal of the SONGS project is to extend the applicability of the SimGrid simulation framework from Grids and Peer-to-Peer systems to Clouds and High Performance Computation systems. Each type of large-scale computing system will be addressed through a set of use cases and lead by researchers recognized as experts in this area.

9.2.3. ADT - Inria Technological Development Actions

ADT K'Star (http://kstar.gforge.inria.fr/#!index.md)

Participants: Olivier Aumage, Nathalie Furmento, Samuel Pitoiset, Samuel Thibault.

Inria ADT Campaign 2013, 10/2013 - 9/2015 (24 months)

Coordinator: Thierry Gautier (team AVALON, Inria Grenoble - Rhône-Alpes) and Olivier Aumage (team RUNTIME, Inria Bordeaux - Sud-Ouest)

Abstract: The Inria action ADT K'Star is a joint effort from Inria teams AVALON and RUNTIME to design the Klang-Omp source-to-source OpenMP compiler to translate OpenMP directives into calls to the API of AVALON and RUNTIME respective runtime systems (XKaapi for AVALON, StarPU for RUNTIME).

9.2.4. IPL - Inria Project Lab

C2S@Exa - Computer and Computational Sciences at Exascale **Participant:** Olivier Aumage.

Inria IPL 2013 - 2017 (48 months)

Coordinator: Stéphane Lantéri (team Nachos, Inria Sophia)

Since January 2013, the team is participating to the C2S@Exa http://www-sop.inria.fr/c2s_at_exa Inria Project Lab (IPL). This national initiative aims at the development of numerical modeling methodologies that fully exploit the processing capabilities of modern massively parallel architectures in the context of a number of selected applications related to important scientific and technological challenges for the quality and the security of life in our society. This collaborative effort involves computer scientists that are experts of programming models, environments and tools for harnessing massively parallel systems, algorithmists that propose algorithms and contribute to generic libraries and core solvers in order to take benefit from all the parallelism levels with the main goal of optimal scaling on very large numbers of computing entities and, numerical mathematicians that are studying numerical schemes and scalable solvers for systems of partial differential equations in view of the simulation of very large-scale problems.

9.3. European Initiatives

9.3.1. FP7 & H2020 Projects

9.3.1.1. INTERTWINE

Title: Programming Model INTERoperability ToWards Exascale

Programm: H2020

Duration: October 2015 - October 2018

Coordinator: EPCC

Partners:

Barcelona Supercomputing Center - Centro Nacional de Supercomputacion (Spain)

Deutsches Zentrum für Luft - und Raumfahrt Ev (Germany)

Fraunhofer Gesellschaft Zur Forderung Der Angewandten Forschung Ev (Germany)

Institut National de Recherche en Informatique et en Automatique (France)

Kungliga Tekniska Hoegskolan (Sweden)

T-Systems Solutions for Research (Germany)

The University of Edinburgh (United Kingdom)

Universitat Jaume I de Castellon (Spain)

The University of Manchester (United Kingdom)

Inria contact: Olivier Aumage

This project addresses the problem of programming model design and implementation for the Exascale. The first Exascale computers will be very highly parallel systems, consisting of a hierarchy of architectural levels. To program such systems effectively and portably, programming APIs with efficient and robust implementations must be ready in the appropriate timescale. A single, "silver bullet" API which addresses all the architectural levels does not exist and seems very unlikely to emerge soon enough. We must therefore expect that using combinations of different APIs at different system levels will be the only practical solution in the short to medium term. Although there remains room for improvement in individual programming models and their implementations, the main challenges lie in interoperability between APIs. It is this interoperability, both at the specification level and at the implementation level, which this project seeks to address and to further the state of the art. INTERTWinE brings together the principal European organisations driving the evolution of programming models and their implementations. The project will focus on seven key programming APIs: MPI, GASPI, OpenMP, OmpSs, StarPU, QUARK and PaRSEC, each of which has a project partner with extensive experience in API design and implementation. Interoperability requirements, and evaluation of implementations will be driven by a set of kernels and applications, each of which has a project partner with a major role in their development. The project will implement a co- design cycle, by feeding back advances in API design and implementation into the applications and kernels, thereby driving new requirements and hence further advances.

9.3.1.2. Mont-Blanc 2

Title: Programming Model INTERoperability ToWards Exascale

Programm: H2020

Duration: Sep. 2013 - Sep. 2016

Coordinator: BSC

Partners: Atos/Bull, ARM, Jülich, LRZ, Univ. Stuttgart, CINECA, CNRS, CEA, Univ. Bristol, Allinea Software, Univ. Cantabria

Inria contact: Olivier Aumage

The Mont-Blanc project aims to develop a European Exascale approach leveraging on commodity power-efficient embedded technologies. The project has developed a HPC system software stack on ARM, and will deploy the first integrated ARM-based HPC prototype by 2014, and is also working on a set of 11 scientific applications to be ported and tuned to the prototype system.

9.3.2. Collaborations with Major European Organizations

PRACE (Europe): Two-days training session on runtime systems, as part of the Prace Advanced Training Center Program (together with *La Maison de la Simulation*).

9.4. International Initiatives

9.4.1. Inria Associate Teams not involved in an Inria International Labs

MORSE Matrices Over Runtime Systems at Exascale

- Inria Associate-Teams program: 2011-2016
- Coordinator: Emmanuel Agullo (Hiepacs)
- Parners: Inria (Runtime & Hiepacs), University of Tennessee Knoxville, University of Colorado Denver and KAUST.
- Abstract: The Matrices Over Runtime Systems at Exascale (MORSE) associate team has vocation to design dense and sparse linear algebra methods that achieve the fastest possible time to an accurate solution on large-scale multicore systems with GPU accelerators, using all the processing power that future high end systems can make available. To develop software that will perform well on petascale and exascale systems with thousands of nodes and millions of cores, several daunting challenges have to be overcome both by the numerical linear algebra and the runtime system communities. With Inria Hiepacs, University of Tennessee, Knoxville and University of Colorado, Denver.

10. Dissemination

10.1. Promoting Scientific Activities

10.1.1. Scientific events organisation

10.1.1.1. Member of the organizing committees

- Raymond NAMYST was publicity co-chair of ACM International Conference on Computing Frontiers 2015 (CF'15).

10.1.2. Scientific events selection

10.1.2.1. Member of the conference program committees

- Samuel THIBAULT was a program committee member for EuroPar'15.
- Olivier AUMAGE was a program committee member for HUCAA'15.
- Raymond NAMYST was member of the program committee for IEEE Cluster 2015, ROSS 2015, PPAM 2015, RESPA 2015, SAC/MUSEPAT 2016 and EuroPar 2016.

10.1.2.2. Reviewer

The members of the team reviewed numerous papers for various internationcal conferences such as IPDPS, Super-Computing, Euro-Par, ICPP.

10.1.3. Journal

10.1.3.1. Reviewer - Reviewing activities

The members of the team review papers from many high-level journals such as TPDS, CCPE, TACO, JPDC.

10.1.4. Invited talks

- Samuel THIBAULT was invited to participate to the HCW panel at IPDPS'15, and to the GPU thematic school in Grenoble in December 2015.
- Olivier AUMAGE was invited to participate to the workshop on Graph-based Languages and Task Programming workshop organized by the company Total in Pau in March 2015, and to the Parallel Runtimes and Architectures panel at the HiPEAC Computer System Week in Oslo in May 2015.

10.1.5. Scientific expertise

Raymond NAMYST has been Scientific Advisor at CEA/DAM (French Department of Energy) since 2007.

Denis Barthou was scientific expert for ANR CIFRE PhD files and for ANR HPC/simulation proposal.

10.1.6. Research administration

Raymond NAMYST is member of the committee in charge of allocating research delegations at Inria Bordeaux Sud-Ouest.

10.2. Teaching - Supervision - Juries

10.2.1. Teaching administration

- Raymond NAMYST is co-chair of Teaching Department in Computer Science of University of Bordeaux
- Denis BARTHOU is in charge of the RSR option (M2 level) at Bordeaux INP.
- Samuel THIBAULT is responsible for the computer science topic of the first university semester.

10.2.2. Teaching

Licence : Marie-Christine Counilh, Introduction to Computer Science, 42HeTD, L1, University of Bordeaux

Licence : Marie-Christine Counilh, Object Oriented Programming, 51HeTD, L2, University of Bordeaux

Licence : Raymond Namyst, Computer Architecture, 38HeTD, L1, University of Bordeaux

Licence : Raymond Namyst, System Programming, 59HeTD, L3, University of Bordeaux

Licence : Samuel Thibault, Introduction to Computer Science, 42HeTD, L1, University of Bordeaux

Licence : Samuel Thibault, Networking and programming project, 24HeTD, L3, University of Bordeaux

Licence : Pierre-André Wacrenier, Introduction to Computer Science, 42HeTD, L1, University of Bordeaux

Licence : Pierre-André Wacrenier, System Programming, 35HeTD, L3, University of Bordeaux

Master : Raymond Namyst, Operating Systems, 60HeTD, M1, University of Bordeaux

Master : Raymond Namyst, Parallel Programming, 33HeTD, M1, University of Bordeaux

Master : Samuel Thibault, Operating Systems, 24HeTD, M1, University of Bordeaux

Master : Pierre-André Wacrenier, Parallel Programming, 51HeTD, M1, University of Bordeaux

Engineer School : Olivier Aumage, High Performance Communication Libraries, 20HeTD, Bac+5, ENSEIRB/IPB

Engineer School : Olivier Aumage, Languages and Supports for Parallelism, 14HeTD, Bac+5, ENSEIRB/IPB

Engineer School : Nathalie Furmento, Operating Systems, 20HeTD, Bac+5, ENSEIRB/IPB

Engineer School: Denis Barthou, Compilation, 28HeTD, M1, ENSEIRB/IPB

Engineer School: Denis Barthou, Computer Architectures, 54HeTD, L3, ENSEIRB/IPB

Engineer School/M2: Denis Barthou, Architecture of parallel computers, 24HeTD, M2, ENSEIRB/IPB

Engineer School: Denis Barthou, Games and interaction, 20HeTD, M2, ENSEIRB/IPB

Engineer School: Denis Barthou, Programming and Software Engineering, 55HeTD, L3/M1, ENSEIRB/IPB

10.2.3. Supervision

PhD: Corentin Rossignon, Design of an object-oriented runtime system for oil reserve simulations on heterogeneous architectures, 2015/07, Olivier Aumage and Pascal Hénon (TOTAL) and Raymond Namyst and Samuel Thibault

PhD: Paul-Antoine Arras, Ordonnancement d'applications à flux de données pour les MPSoC embarqués hybrides comprenant des unités de calcul programmables et des accélérateurs matériels, 2015/02, Emmanuel Jeannot, Samuel Thibault, Didier Fuin, Arthur Stoutchinin

PhD: Emmanuelle Saillard, Static/dynamic/iterative analyses for validation and improvement of multi-models HPC applications, U. Bordeaux, 2015/09, Patrick Carribault (CEA) and Denis Barthou

PhD: Emmanuel Cieren, Molecular Dynamics on Exascale Supercomputers, 2015/10, Laurent Colombet (CEA), Raymond Namyst

PhD: Antoine Capra, Virtualization in the context of High Parallel Computing, 2015/12, Marc Pérache (CEA), François Diakhaté (CEA), Raymond Namyst

PhD: Pei Li, Unified system of code transformation and execution for heterogeneous multi-core architectures, 2015/12, Elisabeth Brunet (Telecom Sud-Paris), Raymond Namyst

PhD in progress: Christopher Haine, Estimating efficiency and automatic restructuration of data layout, 2014/01, Olivier Aumage, Denis Barthou

PhD in progress: Jérôme Richard, Conception of a software component model with task scheduling for many-core based parallel architecture, application to the Gysela5D code, 2014/11, Christian Perez (LIP/ENSL), Julien Bigot (Maison de la Simulation), Olivier Aumage, Guillaume LATU (IRFM).

PhD in progress: Marc Sergent , Passage à l'échelle de moteur d'exécution à base de graphes de tâches, 2013/09, Olivier Aumage , David Goudin (CEA/CESTA), Samuel Thibault , Raymond Namyst

PhD in progress: Suraj Kumar, Task-based programming paradigms and scheduling, 2013/12, Emmanuel Agullo, Olivier Beaumont, Samuel Thibault

PhD in progress: Hugo Brunie. Characterizing and Using Hierarchical Heterogeneous Memories. 2015/09, Julien Jaeger (CEA), Patrick Carribault (CEA), Denis Barthou

PhD in progress: P. Huchant. Static/Dynamic Parallelism Adaptation. 2015/09, Denis Barthou, Raymond Namyst

PhD in progress: G. Vaumourin. Hybrid Memory Hierarchy and Dynamic Data Handling in Embedded Parallel Architectures. Alexandre Guerre (CEA), Thomas Dombek (CEA), Denis Barthou

PhD in progress: J.C. Papin, Potentials-based dynamic scheduling and partitioning tools for domain decomposition based simulations, Laurent Colombet (CEA), Raymond Namyst

10.2.4. Juries

Samuel THIBAULT was member of PhD defense jury of the following candidates:

- Nicolas Bertrand (INP Toulouse, Examinator)
- Sébastien Frémal (Mons Université (Belgium), Examinator)

Denis BARTHOU was member of PhD defense jury of the following candidates:

- Imen Fassi (U. Strasbourg, Reviewer)
- Leandro Fontoura Cupertino (U. Toulouse, Reviewer)
- Vincent Palomares (U. Versailles St Quentin, Reviewer)

Raymond NAMYST was member of the PhD defense jury of the following candidates:

- Aurèle Mahéo (Université Versailles Saint-Quentin, Reviewer)
- Luka Stanisic (Université de Grenoble, Reviewer)
- Safae Dahmani (CEA, Reviewer)
- Andi Drebes (Université Pierre et Marie Curie, President)
- Salli Moustafa (EDF, President)
- Vincent Lanore (ENS Lyon, President)
- Damien Dosimont (Université de Grenoble, Examinator)
- Florent Lopez (Université de Toulouse, Examinator)

He was also member of the HDR defense jury of the following candidates:

- Georges Da Costa (Université Paul Sabatier de Toulouse, President)

Nathalie FURMENTO was member of a Recruiter Committee for a permanent engineer position at the Université de la Rochelle.

10.3. Popularization

- Olivier Aumage gave a tutorial on task-based runtime systems at the HiPEAC conference in Amsterdam in January 2015.
- Olivier Aumage participated to the Inria Booth at the SuperComputing conference in Austin in November 2015.
- Samuel Thibault gave a talk at the ELCI Workshop in Paris.
- Samuel Thibault published a paper « L'accessibilité partout ! Retour d'expérience sous Debian » in the professional « Programmez ! » journal.

11. Bibliography

Publications of the year

Doctoral Dissertations and Habilitation Theses

[1] A. CAPRA. *Virtualisation en contexte HPC*, Université de Bordeaux, 2015

[2] E. CIEREN. *Molecular Dynamics for Exascale Supercomputers*, Université de Bordeaux, 2015

[3] A. HUGO. *Composability of parallel codes on heterogeneous architectures*, Université de Bordeaux, 2015

[4] P. LI. *Système unifié de transformation de code et d'exécution pour un passage aux architectures multi-coeurs hétérogènes*, Université de Bordeaux, 2015

[5] C. ROSSIGNON. *Un modèle de programmation à grain fin pour la parallélisation de solveurs linéaires creux*, Université de Bordeaux, 2015

[6] E. SAILLARD. *Static/Dynamic Analyses for Validation and Improvements of Multi-Model HPC Applications.*, Université de Bordeaux, 2015, Thèse de doctorat dirigée par Barthou, Denis Informatique Bordeaux 2015, http://www.theses.fr/2015BORD0176/document

Articles in International Peer-Reviewed Journals

[7] P.-A. ARRAS, D. FUIN, E. JEANNOT, A. STOUTCHININ, S. THIBAULT. *List Scheduling in Embedded Systems Under Memory Constraints*, in "International Journal of Parallel Programming", December 2015, vol. 43, n° 6, pp. 1103-1128 [*DOI :* 10.1007/s10766-014-0338-1], https://hal.inria.fr/hal-01087067

[8] L. STANISIC, S. THIBAULT, A. LEGRAND, B. VIDEAU, J.-F. MÉHAUT. *Faithful Performance Prediction of a Dynamic Task-Based Runtime System for Heterogeneous Multi-Core Architectures*, in "Concurrency and Computation: Practice and Experience", May 2015, 16 p. [*DOI :* 10.1002/CPE], https://hal.inria.fr/hal-01147997

Invited Conferences

[9] J.-C. PAPIN, C. DENOUAL, L. COLOMBET, R. NAMYST. *SPAWN: An Iterative, Potentials-Based, Dynamic Scheduling and Partitioning Tool*, in "SuperComputing'15 - RESPA Workshop", Austin, United States, November 2015, https://hal.inria.fr/hal-01223897

International Conferences with Proceedings

[10] E. AGULLO, O. BEAUMONT, L. EYRAUD-DUBOIS, J. HERRMANN, S. KUMAR, L. MARCHAL, S. THIBAULT. *Bridging the Gap between Performance and Bounds of Cholesky Factorization on Heterogeneous Platforms*, in "Heterogeneity in Computing Workshop 2015", Hyderabad, India, May 2015, https://hal.inria.fr/hal-01120507

[11] E. AGULLO, O. BEAUMONT, L. EYRAUD-DUBOIS, S. KUMAR. *Are Static Schedules so Bad ? A Case Study on Cholesky Factorization*, in "IPDPS'16", Chicago, IL, United States, Proceedings of the 30th IEEE International Parallel & Distributed Processing Symposium, IPDPS'16, IEEE, May 2016, https://hal.inria.fr/hal-01223573

[12] P.-A. ARRAS, D. FUIN, E. JEANNOT, S. THIBAULT. *DKPN: A Composite Dataflow/Kahn Process Networks Execution Model*, in "24th Euromicro International Conference on Parallel, Distributed and Network-based processing", Heraklion Crete, Greece, February 2016, https://hal.inria.fr/hal-01234333

[13] A. CASSAGNE, B. LE GAL, C. LEROUX, O. AUMAGE, D. BARTHOU. *An Efficient, Portable and Generic Library for Successive Cancellation Decoding of Polar Codes*, in "The 28th International Workshop on Languages and Compilers for Parallel Computing (LCPC 2015)", Raleigh, United States, September 2015, https://hal.inria.fr/hal-01203105

[14] J. JAEGER, E. SAILLARD, P. CARRIBAULT, D. BARTHOU. *Correctness Analysis of MPI-3 Non-Blocking Communications in PARCOACH*, in "European MPI Users' Group Meeting", Bordeaux, France, EuroMPI '15 The 22nd European MPI Users' Group Meeting, September 2015 [*DOI :* 10.1145/1235], https://hal.inria.fr/hal-01252321

[15] V. MARTÍNEZ, D. MICHÉA, F. DUPROS, O. AUMAGE, S. THIBAULT, H. AOCHI, P. O. A. NAVAUX. *Towards seismic wave modeling on heterogeneous many-core architectures using task-based runtime system*, in "27th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)", Florianopolis, Brazil, IEEE 27th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), IEEE, October 2015, https://hal.inria.fr/hal-01182746

Other Publications

[16] T. COJEAN, A. GUERMOUCHE, A. HUGO, R. NAMYST, P.-A. WACRENIER. *Exploiting two-level parallelism by aggregating computing resources in task-based applications over accelerator-based machines*, July 2015, working paper or preprint, https://hal.inria.fr/hal-01181135