IN PARTNERSHIP WITH:
CNRS

Université Claude Bernard (Lyon 1)

Ecole normale supérieure de Lyon

Activity Report 2014

Team ROMA

Optimisation des ressources : modèles, algorithmes et ordonnancement

IN COLLABORATION WITH: Laboratoire de l'Informatique du Parallélisme (LIP)

RESEARCH CENTER
Grenoble - Rhône-Alpes

THEME
Distributed and High Performance Computing

Table of contents

Team ROMA

Keywords: Scheduling, Parallel And Distributed Algorithms, Combinatorial Optimization, Exascale Systems, Fault Tolerance

Creation of the Team: 2012 February 01*, updated into Project-Team:* 2015 January 01.

1. Members

Research Scientists
Frédéric Vivien [Team leader, Inria, Senior Researcher, HdR]
Jean-Yves L'Excellent [Inria, Researcher, HdR]
Loris Marchal [CNRS, Researcher]
Bora Uçar [CNRS, Researcher]

Faculty Members
Anne Benoit [ENS Lyon, Associate Professor, and Institut Universitaire de France, HdR]
Yves Robert [ENS Lyon, Professor, and Institut Universitaire de France, HdR]

Engineers
Mohamed-Slim Bouguerra [Inria, until Oct. 2014]
Sheng Di [Inria, until May 2014, granted by Grand Equipement National de Calcul Intensif]
Guillaume Joslin [Univ. Lyon I, from July 2014]
Chiara Puglisi [Inria]

PhD Students
Guillaume Aupy [ENS Lyon, until Aug. 2014]
Aurélien Cavelan [Inria, from Oct. 2014]
Julien Herrmann [ENS Lyon]
Oguz Kaya [Inria, from Oct. 1 2014]
Wissam M. Sid-Lakhdar [ENS Lyon]
Dounia Zaidouni [Inria, until Aug. 2014]

Post-Doctoral Fellows
Enver Kayaaslan [Inria]
Hongyang Sun [Univ. Lyon I, from June 2014]
Guillaume Aupy [ENS Lyon, from Sep. 2014]

Administrative Assistants
Evelyne Blesle [Inria, until May 2014]
Virginie Bouyer [Inria, from Dec. 2014]
Laetitia Gauthe [Inria, from June until November 2014]

Others
Patrick Amestoy [INP Toulouse, external collaborator, HdR]
Alfredo Buttari [CNRS, external collaborator]
Franck Cappello [Argonne National Laboratory – USA, external collaborator, HdR]

2. Overall Objectives

2.1. Overall Objectives

The ROMA project aims at designing models, algorithms, and scheduling strategies to optimize the execution of scientific applications.

Scientists now have access to tremendous computing power. For instance, the four most powerful computing platforms in the TOP 500 list [43] each includes more than 500,000 cores and deliver a sustained performance of more than 10 Peta FLOPS. The volunteer computing platform BOINC [39] is another example with more than 440,000 enlisted computers and, on average, an aggregate performance of more than 9 Peta FLOPS. Furthermore, it had never been so easy for scientists to have access to parallel computing resources, either through the multitude of local clusters or through distant cloud computing platforms.

Because parallel computing resources are ubiquitous, and because the available computing power is so huge, one could believe that scientists no longer need to worry about finding computing resources, even less to optimize their usage. Nothing is farther from the truth. Institutions and government agencies keep building larger and more powerful computing platforms with a clear goal. These platforms must allow to solve problems in reasonable timescales, which were so far out of reach. They must also allow to solve problems more precisely where the existing solutions are not deemed to be sufficiently accurate. For those platforms to fulfill their purposes, their computing power must therefore be carefully exploited and not be wasted. This often requires an efficient management of all types of platform resources: computation, communication, memory, storage, energy, etc. This is often hard to achieve because of the characteristics of new and emerging platforms. Moreover, because of technological evolutions, new problems arise, and fully tried and tested solutions need to be thoroughly overhauled or simply discarded and replaced. Here are some of the difficulties that have, or will have, to be overcome:

- computing platforms are hierarchical: a processor includes several cores, a node includes several processors, and the nodes themselves are gathered into clusters. Algorithms must take this hierarchical structure into account, in order to fully harness the available computing power;
- the probability for a platform to suffer from a hardware fault automatically increases with the number of its components. Fault-tolerance techniques become unavoidable for large-scale platforms;
- the ever increasing gap between the computing power of nodes and the bandwidths of memories and networks, in conjunction with the organization of memories in deep hierarchies, requires to take more and more care of the way algorithms use memory;
- energy considerations are unavoidable nowadays. Design specifications for new computing platforms always include a maximal energy consumption. The energy bill of a supercomputer may represent a significant share of its cost over its lifespan. These issues must be taken into account at the algorithm-design level.

We are convinced that dramatic breakthroughs in algorithms and scheduling strategies are required for the scientific computing community to overcome all the challenges posed by new and emerging computing platforms. This is required for applications to be successfully deployed at very large scale, and hence for enabling the scientific computing community to push the frontiers of knowledge as far as possible. The ROMA project-team aims at providing fundamental algorithms, scheduling strategies, protocols, and software packages to fulfill the needs encountered by a wide class of scientific computing applications, including domains as diverse as geophysics, structural mechanics, chemistry, electromagnetism, numerical optimization, or computational fluid dynamics, to quote a few. To fulfill this goal, the ROMA project-team takes a special interest in dense and sparse linear algebra.

The work in the ROMA team is organized along three research themes.

1. **Algorithms for probabilistic environments.** In this theme, we consider problems where some of the platform characteristics, or some of the application characteristics, are described by probability distributions. This is in particular the case when considering the resilience of applications in failure-prone environments: the possibility of faults is modeled by probability distributions.
2. **Platform-aware scheduling strategies.** In this theme, we focus on the design of scheduling strategies that finely take into account some platform characteristics beyond the most classical ones, namely the computing speed of processors and accelerators, and the communication bandwidth of network links. In the scope of this theme, when designing scheduling strategies, we focus either on the energy consumption or on the memory behavior. All optimization problems under study are multi-criteria.

3. **High-performance computing and linear algebra.** We work on algorithms and tools for both sparse and dense linear algebra. In sparse linear algebra, we work on most aspects of direct multifrontal solvers for linear systems. In dense linear algebra, we focus on the adaptation of factorization kernels to emerging and future platforms. In addition, we also work on combinatorial scientific computing, that is, on the design of combinatorial algorithms and tools to solve combinatorial problems, such as those encountered, for instance, in the preprocessing phases of solvers of sparse linear systems.

3. Research Program

3.1. Algorithms for probabilistic environments

There are two main research directions under this research theme. In the first one, we consider the problem of the efficient execution of applications in a failure-prone environment. Here, probability distributions are used to describe the potential behavior of computing platforms, namely when hardware components are subject to faults. In the second research direction, probability distributions are used to describe the characteristics and behavior of applications.

3.1.1. Application resilience

An application is resilient if it can successfully produce a correct result in spite of potential faults in the underlying system. Application resilience can involve a broad range of techniques, including fault prediction, error detection, error containment, error correction, checkpointing, replication, migration, recovery, etc. Faults are quite frequent in the most powerful existing supercomputers. The Jaguar platform, which ranked third in the TOP 500 list in November 2011 [42], had an average of 2.33 faults per day during the period from August 2008 to February 2010 [66]. The mean-time between faults of a platform is inversely proportional to its number of components. Progresses will certainly be made in the coming years with respect to the reliability of individual components. However, designing and building high-reliability hardware components is far more expensive than using lower reliability top-of-the-shelf components. Furthermore, low-power components may not be available with high-reliability. Therefore, it is feared that the progresses in reliability will far from compensate the steady projected increase of the number of components in the largest supercomputers. Already, application failures have a huge computational cost. In 2008, the DARPA white paper on "System resilience at extreme scale" [41] stated that high-end systems wasted 20% of their computing capacity on application failure and recovery.

In such a context, any application using a significant fraction of a supercomputer and running for a significant amount of time will have to use some fault-tolerance solution. It would indeed be unacceptable for an application failure to destroy centuries of CPU-time (some of the simulations run on the Blue Waters platform consumed more than 2,700 years of core computing time [37] and lasted over 60 hours; the most time-consuming simulations of the US Department of Energy (DoE) run for weeks to months on the most powerful existing platforms [40]).

Our research on resilience follows two different directions. On the one hand we design new resilience solutions, either generic fault-tolerance solutions or algorithm-based solutions. On the other hand we model and theoretically analyze the performance of existing and future solutions, in order to tune their usage and help determine which solution to use in which context.

3.1.2. Scheduling strategies for applications with a probabilistic behavior

Static scheduling algorithms are algorithms where all decisions are taken before the start of the application execution. On the contrary, in non-static algorithms, decisions may depend on events that happen during the execution. Static scheduling algorithms are known to be superior to dynamic and system-oriented approaches in stable frameworks [47], [53], [54], [65], that is, when all characteristics of platforms and applications are perfectly known, known a priori, and do not evolve during the application execution. In practice, the prediction of application characteristics may be approximative or completely infeasible. For instance, the amount of computations and of communications required to solve a given problem in parallel may strongly depend on some input data that are hard to analyze (this is for instance the case when solving linear systems using full pivoting).

We plan to consider applications whose characteristics change dynamically and are subject to uncertainties. In order to benefit nonetheless from the power of static approaches, we plan to model application uncertainties and variations through probabilistic models, and to design for these applications scheduling strategies that are either static, or partially static and partially dynamic.

3.2. Platform-aware scheduling strategies

In this theme, we study and design scheduling strategies, focusing either on energy consumption or on memory behavior. In other words, when designing and evaluating these strategies, we do not limit our view to the most classical platform characteristics, that is, the computing speed of cores and accelerators, and the bandwidth of communication links.

In most existing studies, a single optimization objective is considered, and the target is some sort of absolute performance. For instance, most optimization problems aim at the minimization of the overall execution time of the application considered. Such an approach can lead to a very significant waste of resources, because it does not take into account any notion of efficiency nor of yield. For instance, it may not be meaningful to use twice as many resources just to decrease by 10% the execution time. In all our work, we plan to look only for algorithmic solutions that make a “clever” usage of resources. However, looking for the solution that optimizes a metric such as the efficiency, the energy consumption, or the memory-peak minimization, is doomed for the type of applications we consider. Indeed, in most cases, any optimal solution for such a metric is a sequential solution, and sequential solutions have prohibitive execution times. Therefore, it becomes mandatory to consider multi-criteria approaches where one looks for trade-offs between some user-oriented metrics that are typically related to notions of Quality of Service—execution time, response time, stretch, throughput, latency, reliability, etc.—and some system-oriented metrics that guarantee that resources are not wasted. In general, we will not look for the Pareto curve, that is, the set of all dominating solutions for the considered metrics. Instead, we will rather look for solutions that minimize some given objective while satisfying some bounds, or “budgets”, on all the other objectives.

3.2.1. Energy-aware algorithms

Energy-aware scheduling has proven an important issue in the past decade, both for economical and environmental reasons. Energy issues are obvious for battery-powered systems. They are now also important for traditional computer systems. Indeed, the design specifications of any new computing platform now always include an upper bound on energy consumption. Furthermore, the energy bill of a supercomputer may represent a significant share of its cost over its lifespan.

Technically, a processor running at speed s dissipates s^α watts per unit of time with $2 \leq \alpha \leq 3$ [45], [46], [51]; hence, it consumes $s^\alpha \times d$ joules when operated during d units of time. Therefore, energy consumption can be reduced by using speed scaling techniques. However it was shown in [67] that reducing the speed of a processor increases the rate of transient faults in the system. The probability of faults increases exponentially, and this probability cannot be neglected in large-scale computing [61]. In order to make up for the loss in *reliability* due to the energy efficiency, different models have been proposed for fault tolerance: (i) *re-execution* consists in re-executing a task that does not meet the reliability constraint [67]; (ii) *replication* consists in executing the same task on several processors simultaneously, in order to meet the reliability constraints [44]; and (iii) *checkpointing* consists in “saving” the work done at some certain instants, hence reducing the amount of work lost when a failure occurs [60].

Energy issues must be taken into account at all levels, including the algorithm-design level. We plan to both evaluate the energy consumption of existing algorithms and to design new algorithms that minimize energy consumption using tools such as resource selection, dynamic frequency and voltage scaling, or powering-down of hardware components.

3.2.2. Memory-aware algorithms

For many years, the bandwidth between memories and processors has increased more slowly than the computing power of processors, and the latency of memory accesses has been improved at an even slower pace. Therefore, in the time needed for a processor to perform a floating point operation, the amount of data transferred between the memory and the processor has been decreasing with each passing year. The risk is for an application to reach a point where the time needed to solve a problem is no longer dictated by the processor computing power but by the memory characteristics, comparable to the *memory wall* that limits CPU performance. In such a case, processors would be greatly under-utilized, and a large part of the computing power of the platform would be wasted. Moreover, with the advent of multicore processors, the amount of memory per core has started to stagnate, if not to decrease. This is especially harmful to memory intensive applications. The problems related to the sizes and the bandwidths of memories are further exacerbated on modern computing platforms because of their deep and highly heterogeneous hierarchies. Such a hierarchy can extend from core private caches to shared memory within a CPU, to disk storage and even tape-based storage systems, like in the Blue Waters supercomputer [38]. It may also be the case that heterogeneous cores are used (such as hybrid CPU and GPU computing), and that each of them has a limited memory.

Because of these trends, it is becoming more and more important to precisely take memory constraints into account when designing algorithms. One must not only take care of the amount of memory required to run an algorithm, but also of the way this memory is accessed. Indeed, in some cases, rather than to minimize the amount of memory required to solve the given problem, one will have to maximize data reuse and, especially, to minimize the amount of data transferred between the different levels of the memory hierarchy (minimization of the volume of memory inputs-outputs). This is, for instance, the case when a problem cannot be solved by just using the in-core memory and that any solution must be out-of-core, that is, must use disks as storage for temporary data.

It is worth noting that the cost of moving data has lead to the development of so called "communication-avoiding algorithms" [57]. Our approach is orthogonal to these efforts: in communication-avoiding algorithms, the application is modified, in particular some redundant work is done, in order to get rid of some communication operations, whereas in our approach, we do not modify the application, which is provided as a task graph, but we minimize the needed memory peak only by carefully scheduling tasks.

3.3. High-performance computing and linear algebra

Our work on high-performance computing and linear algebra is organized along three research directions. The first direction is devoted to direct solvers of sparse linear systems. The second direction is devoted to combinatorial scientific computing, that is, the design of combinatorial algorithms and tools that solve problems encountered in some of the other research themes, like the problems faced in the preprocessing phases of sparse direct solvers. The last direction deals with the adaptation of classical dense linear algebra kernels to the architecture of future computing platforms.

3.3.1. Direct solvers for sparse linear systems

The solution of sparse systems of linear equations (symmetric or unsymmetric, often with an irregular structure, from a few hundred thousand to a few hundred million equations) is at the heart of many scientific applications arising in domains such as geophysics, structural mechanics, chemistry, electromagnetism, numerical optimization, or computational fluid dynamics, to cite a few. The importance and diversity of applications are a main motivation to pursue research on sparse linear solvers. Because of this wide range of applications, any significant progress on solvers will have a significant impact in the world of simulation. Research on sparse direct solvers in general is very active for the following main reasons:

- many applications fields require large-scale simulations that are still too big or too complicated with respect to today's solution methods;
- the current evolution of architectures with massive, hierarchical, multicore parallelism imposes to overhaul all existing solutions, which represents a major challenge for algorithm and software development;

- the evolution of numerical needs and types of simulations increase the importance, frequency, and size of certain classes of matrices, which may benefit from a specialized processing (rather than resort to a generic one).

Our research in the field is strongly related to the software package MUMPS (see Section 5.1). MUMPS is both an experimental platform for academics in the field of sparse linear algebra, and a software package that is widely used in both academia and industry. The software package MUMPS enables us to (i) confront our research to the real world, (ii) develop contacts and collaborations, and (iii) receive continuous feedback from real-life applications, which is extremely critical to validate our research work. The feedback from a large user community also enables us to direct our long-term objectives towards meaningful directions.

In this context, we aim at designing parallel sparse direct methods that will scale to large modern platforms, and that are able to answer new challenges arising from applications, both efficiently—from a resource consumption point of view—and accurately—from a numerical point of view. For that, and even with increasing parallelism, we do not want to sacrifice in any manner numerical stability, based on threshold partial pivoting, one of the main originalities of our approach (our “trademark”) in the context of direct solvers for distributed-memory computers; although this makes the parallelization more complicated, applying the same pivoting strategy as in the serial case ensures numerical robustness of our approach, which we generally measure in terms of sparse backward error. In order to solve the hard problems resulting from the always-increasing demands in simulations, special attention must also necessarily be paid to memory usage (and not only execution time). This requires specific algorithmic choices and scheduling techniques. From a complementary point of view, it is also necessary to be aware of the functionality requirements from the applications and from the users, so that robust solutions can be proposed for a wide range of applications.

Among direct methods, we rely on the multifrontal method [55], [56], [59]. This method usually exhibits a good data locality and hence is efficient in cache-based systems. The task graph associated with the multifrontal method is in the form of a tree whose characteristics should be exploited in a parallel implementation.

Our work is organized along two main research directions. In the first one we aim at efficiently addressing new architectures that include massive, hierarchical parallelism. In the second one, we aim at reducing the running time complexity and the memory requirements of direct solvers, while controlling accuracy.

3.3.2. Combinatorial scientific computing

Combinatorial scientific computing (CSC) is a recently coined term (circa 2002) for interdisciplinary research at the intersection of discrete mathematics, computer science, and scientific computing. In particular, it refers to the development, application, and analysis of combinatorial algorithms to enable scientific computing applications. CSC’s deepest roots are in the realm of direct methods for solving sparse linear systems of equations where graph theoretical models have been central to the exploitation of sparsity, since the 1960s. The general approach is to identify performance issues in a scientific computing problem, such as memory use, parallel speed up, and/or the rate of convergence of a method, and to develop combinatorial algorithms and models to tackle those issues.

Our target scientific computing applications are (i) the preprocessing phases of direct methods (in particular MUMPS), iterative methods, and hybrid methods for solving linear systems of equations; and (ii) the mapping of tasks (mostly the sub-tasks of the mentioned solvers) onto modern computing platforms. We focus on the development and use of graph and hypergraph models, and related tools such as hypergraph partitioning algorithms, to solve problems of load balancing and task mapping. We also focus on bipartite graph matching and vertex ordering methods for reducing the memory overhead and computational requirements of solvers. Although we direct our attention on these models and algorithms through the lens of linear system solvers, our solutions are general enough to be applied to some other resource optimization problems.

3.3.3. Dense linear algebra on post-petascale multicore platforms

The quest for efficient, yet portable, implementations of dense linear algebra kernels (QR, LU, Cholesky) has never stopped, fueled in part by each new technological evolution. First, the LAPACK library [49] relied on BLAS level 3 kernels (Basic Linear Algebra Subroutines) that enable to fully harness the computing power of a single CPU. Then the SCALAPACK library [48] built upon LAPACK to provide a coarse-grain parallel version, where processors operate on large block-column panels. Inter-processor communications occur through highly tuned MPI send and receive primitives. The advent of multi-core processors has led to a major modification in these algorithms [50], [64], [58]. Each processor runs several threads in parallel to keep all cores within that processor busy. Tiled versions of the algorithms have thus been designed: dividing large block-column panels into several tiles allows for a decrease in the granularity down to a level where many smaller-size tasks are spawned. In the current panel, the diagonal tile is used to eliminate all the lower tiles in the panel. Because the factorization of the whole panel is now broken into the elimination of several tiles, the update operations can also be partitioned at the tile level, which generates many tasks to feed all cores.

The number of cores per processor will keep increasing in the following years. It is projected that high-end processors will include at least a few hundreds of cores. This evolution will require to design new versions of libraries. Indeed, existing libraries rely on a static distribution of the work: before the beginning of the execution of a kernel, the location and time of the execution of all of its component is decided. In theory, static solutions enable to precisely optimize executions, by taking parameters like data locality into account. At run time, these solutions proceed at the pace of the slowest of the cores, and they thus require a perfect load-balancing. With a few hundreds, if not a thousand, cores per processor, some tiny differences between the computing times on the different cores ("jitter") are unavoidable and irremediably condemn purely static solutions. Moreover, the increase in the number of cores per processor once again mandates to increase the number of tasks that can be executed in parallel.

We study solutions that are part-static part-dynamic, because such solutions have been shown to outperform purely dynamic ones [52]. On the one hand, the distribution of work among the different nodes will still be statically defined. On the other hand, the mapping and the scheduling of tasks inside a processor will be dynamically defined. The main difficulty when building such a solution will be to design lightweight dynamic schedulers that are able to guarantee both an excellent load-balancing and a very efficient use of data locality.

4. Application Domains

4.1. Application of sparse direct solvers

Sparse direct (multifrontal) solvers in distributed-memory environments have a wide range of applications as they are used at the heart of many numerical methods in simulation: whether a model uses finite elements or finite differences, or requires the optimization of a complex linear or nonlinear function, one often ends up solving a linear system of equations involving sparse matrices. There are therefore a number of application fields, among which some of the ones cited by the users of our sparse direct solver MUMPS (see Section 5.1) are: structural mechanics, biomechanics, medical image processing, tomography, geophysics, electromagnetism, fluid dynamics, econometric models, oil reservoir simulation, magneto-hydro-dynamics, chemistry, acoustics, glaciology, astrophysics, circuit simulation, and work on hybrid direct-iterative methods.

5. New Software and Platforms

5.1. MUMPS

Participants: Patrick Amestoy, Alfredo Buttari, Jean-Yves L'Excellent [correspondent], Chiara Puglisi, Wissam M. Sid-Lakhdar, Bora Uçar.

MUMPS (for *MUltifrontal Massively Parallel Solver*) see http://mumps-solver.org is a software package for the solution of large sparse systems of linear equations. It implements a direct method, the so called multifrontal method; it is a parallel code capable of exploiting distributed-memory computers as well as multithreaded libraries; its main originalities are its numerical robustness (including partial threshold pivoting in distributed-memory environment) and its wide range of features.

The latest public release is MUMPS 4.10.0 (May 2011); the new release is scheduled for February 2015 and will be under the Cecill-C licence, following an agreement between CERFACS, CNRS, ENS Lyon, INPT, Inria and University of Bordeaux.

6. New Results

6.1. Highlights of the Year

Yves Robert was awarded the 2014 IEEE Technical Committee on Scalable Computing (TCSC) Award for Excellence.

In October 2014, CERFACS, ENS Lyon, INPT, Inria and University of Bordeaux launched a consortium around the software package MUMPS (see http://mumps-consortium.org).

6.2. Cost-Optimal Execution of Boolean DNF Trees with Shared Streams

Several applications process queries expressed as trees of Boolean operators applied to predicates on sensor data streams, e.g., mobile apps and automotive apps. Sensor data must be retrieved from the sensors, which incurs a cost, e.g., an energy expense that depletes the battery of a mobile device, a bandwidth usage. The objective is to determine the order in which predicates should be evaluated so as to shortcut part of the query evaluation and minimize the expected cost. This problem has been studied assuming that each data stream occurs at a single predicate. In this work [17], [27] we study the case in which a data stream occurs in multiple predicates, either when each predicate references a single stream or when a predicate can reference multiple streams. In the single-stream case we give an optimal algorithm for a single-level tree and show that the problem is NP-complete for DNF trees. For DNF trees we show that there exists an optimal predicate evaluation order that is depth-first, which provides a basis for designing a range of heuristics. In the multi-stream case we show that the problem is NP-complete even for single-level trees. As in the single stream case, for DNF trees we show that there exists a depth-first leaf evaluation order that is optimal and we design efficient heuristics.

6.3. Efficient checkpoint/verification patterns for silent error detection

Errors have become a critical problem for high performance computing. Checkpointing protocols are often used for error recovery after fail-stop failures. However, silent errors cannot be ignored, and their peculiarity is that such errors are identified only when the corrupted data is activated. To cope with silent errors, we need a verification mechanism to check whether the application state is correct. Checkpoints should be supplemented with verifications to detect silent errors. When a verification is successful, only the last checkpoint needs to be kept in memory because it is known to be correct. In this work (RR UT-EECS-14-729), we analytically determine the best balance of verifications and checkpoints so as to optimize platform throughput. We introduce a balanced algorithm using a pattern with p checkpoints and q verifications, which regularly interleaves both checkpoints and verifications across same-size computational chunks. We show how to compute the waste of an arbitrary pattern, and we prove that the balanced algorithm is optimal when the platform MTBF (Mean Time Between Failures) is large in front of the other parameters (checkpointing, verification and recovery costs). We conduct several simulations to show the gain achieved by this balanced algorithm for well-chosen values of p and q, compared to the base algorithm that always perform a verification just before taking a checkpoint ($p = q = 1$), and we exhibit gains of up to 19%.

6.4. Assessing general-purpose algorithms to cope with fail-stop and silent errors

In this work (RR-Inria-8599), we combine the traditional checkpointing and rollback recovery strategies with verification mechanisms to address both fail-stop and silent errors. The objective is to minimize either makespan or energy consumption. While DVFS is a popular approach for reducing the energy consumption, using lower speeds/voltages can increase the number of errors, thereby complicating the problem. We consider an application workflow whose dependence graph is a chain of tasks, and we study three execution scenarios: (i) a single speed is used during the whole execution; (ii) a second, possibly higher speed is used for any potential re-execution; (iii) different pairs of speeds can be used throughout the execution. For each scenario, we determine the optimal checkpointing and verification locations (and the optimal speeds for the third scenario) to minimize either objective. The different execution scenarios are then assessed and compared through an extensive set of experiments.

6.5. Scheduling the I/O of HPC applications under congestion

A significant percentage of the computing capacity of large-scale platforms is wasted due to interferences incurred by multiple applications that access a shared parallel file system concurrently. One solution to handling I/O bursts in large-scale HPC systems is to absorb them at an intermediate storage layer consisting of burst buffers. However, our analysis of the Argonne's Mira system shows that burst buffers cannot prevent congestion at all times. As a consequence, I/O performance is dramatically degraded, showing in some cases a decrease in I/O throughput of 67%. In this work (RR-Inria-8519), we analyze the effects of interference on application I/O bandwidth, and propose several scheduling techniques to mitigate congestion. We show through extensive experiments that our global I/O scheduler is able to reduce the effects of congestion, even on systems where burst buffers are used, and can increase the overall system throughput up to 56%. We also show that it outperforms current Mira I/O schedulers.

6.6. Power-aware replica placement in tree networks with multiple servers per client

In this work (RR-Inria-8474), we revisit the well-studied problem of replica placement in tree networks. Rather than minimizing the number of servers needed to serve all client requests, we aim at minimizing the total power consumed by these servers. In addition, we use the most general (and powerful) server assignment policy, where the requests of a client can be served by multiple servers located in the (unique) path from this client to the root of the tree. We consider multi-modal servers that can operate at a set of discrete speeds, using the dynamic voltage and frequency scaling (DVFS) technique. The optimization problem is to determine an optimal location of the servers in the tree, as well as the speed at which each server is operated. A major result is the NP-completeness of this problem, to be contrasted with the minimization of the number of servers, which has polynomial complexity. Another important contribution is the formulation of a Mixed Integer Linear Program (MILP) for the problem, together with the design of several polynomial-time heuristics. We assess the efficiency of these heuristics by simulation. For mid-size instances (up to 30 nodes in the tree), we evaluate their absolute performance by comparison with the optimal solution (obtained via the MILP). The most efficient heuristics provide satisfactory results, within 20% of the optimal solution.

6.7. Parallel scheduling of task trees with limited memory

This work [28] investigates the execution of tree-shaped task graphs using multiple processors. Each edge of such a tree represents some large data. A task can only be executed if all input and output data fit into memory, and a data can only be removed from memory after the completion of the task that uses it as an input data. Such trees arise, for instance, in the multifrontal method of sparse matrix factorization. The peak memory needed for the processing of the entire tree depends on the execution order of the tasks. With one processor the objective of the tree traversal is to minimize the required memory. This problem was well studied and optimal polynomial algorithms were proposed.

Here, we extend the problem by considering multiple processors, which is of obvious interest in the application area of matrix factorization. With multiple processors comes the additional objective to minimize the time needed to traverse the tree, i.e., to minimize the makespan. Not surprisingly, this problem proves to be much harder than the sequential one. We study the computational complexity of this problem and provide inapproximability results even for unit weight trees. We design a series of practical heuristics achieving different trade-offs between the minimization of peak memory usage and makespan. Some of these heuristics are able to process a tree while keeping the memory usage under a given memory limit. The different heuristics are evaluated in an extensive experimental evaluation using realistic trees.

6.8. Scheduling Trees of Malleable Tasks for Sparse Linear Algebra

Scientific workloads are often described as directed acyclic task graphs. In this work [30], we focus on the multifrontal factorization of sparse matrices, whose task graph is structured as a tree of parallel tasks. Among the existing models for parallel tasks, the concept of *malleable* tasks is especially powerful as it allows each task to be processed on a time-varying number of processors. Following the model advocated by Prasanna and Musicus [62], [63] for matrix computations, we consider malleable tasks whose speedup is p^α, where p is the fractional share of processors on which a task executes, and α $(0 < \alpha \leq 1)$ is a parameter which does not depend on the task. We first motivate the relevance of this model for our application with actual experiments on multicore platforms. Then, we study the optimal allocation proposed by Prasanna and Musicus for makespan minimization using optimal control theory. We largely simplify their proofs by resorting only to pure scheduling arguments. Building on the insight gained thanks to these new proofs, we extend the study to distributed multicore platforms. There, a task cannot be distributed among several distributed nodes. In such a distributed setting (homogeneous or heterogeneous), we prove the NP-completeness of the corresponding scheduling problem, and propose some approximation algorithms. We finally assess the relevance of our approach by simulations on realistic trees. We show that the average performance gain of our allocations with respect to existing solutions (that are thus unaware of the actual speedup functions) is up to 16% for $\alpha = 0.9$ (the value observed in the real experiments).

6.9. Non-clairvoyant reduction algorithms for heterogeneous platforms

In this work [6], we have revisited the classical problem of the reduction collective operation in a heterogeneous environment. We have discussed and evaluated four algorithms that are non-clairvoyant, i.e., they do not know in advance the computation and communication costs. On the one hand, Binomial-stat and Fibonacci-stat are static algorithms that decide in advance which operations will be reduced, without adapting to the environment; they were originally defined for homogeneous settings. On the other hand, Tree-dyn and Non-Commut-Tree-dyn are fully dynamic algorithms, for commutative or non-commutative reductions. We have shown that these algorithms are approximation algorithms with constant or asymptotic ratios. We assessed the relative performance of all four non-clairvoyant algorithms with heterogeneous costs through a set of simulations. Our conclusions hold for a variety of distributions.

6.10. Memory-aware tree traversals with pre-assigned tasks

We have studied the complexity of traversing tree-shaped workflows whose tasks require large I/O files. We target a heterogeneous architecture with two resource types, each with a different memory, such as a multicore node equipped with a dedicated accelerator (FPGA or GPU). The tasks in the workflow are colored according to their type and can be processed if all there input and output files can be stored in the corresponding memory. The amount of used memory of each type at a given execution step strongly depends upon the ordering in which the tasks are executed, and upon when communications between both memories are scheduled. The objective is to determine an efficient traversal that minimizes the maximum amount of memory of each type needed to traverse the whole tree. In this study [11], we establish the complexity of this two-memory scheduling problem, and provide inapproximability results. In addition, we design several heuristics, based on both post-order and general traversals, and we evaluate them on a comprehensive set of tree graphs, including random trees as well as assembly trees arising in the context of sparse matrix factorizations.

6.11. Analysis of Dynamic Scheduling Strategies for Matrix Multiplication on Heterogeneous Platforms

The tremendous increase in the size and heterogeneity of supercomputers makes it very difficult to predict the performance of a scheduling algorithm. Therefore, dynamic solutions, where scheduling decisions are made at runtime have overpassed static allocation strategies. The simplicity and efficiency of dynamic schedulers such as Hadoop are a key of the success of the MapReduce framework. Dynamic schedulers such as StarPU, PaRSEC or StarSs are also developed for more constrained computations, e.g. task graphs coming from linear algebra. To make their decisions, these runtime systems make use of some static information, such as the distance of tasks to the critical path or the affinity between tasks and computing resources (CPU, GPU,. . .) and of dynamic information, such as where input data are actually located. In this study [16], we concentrate on two elementary linear algebra kernels, namely the outer product and the matrix multiplication. For each problem, we propose several dynamic strategies that can be used at runtime and we provide an analytic study of their theoretical performance. We prove that the theoretical analysis provides very good estimate of the amount of communications induced by a dynamic strategy and can be used in order to efficiently determine thresholds used in dynamic scheduler, thus enabling to choose among them for a given problem and architecture.

6.12. Determining the optimal redistribution

The classical redistribution problem aims at optimally scheduling communications when reshuffling from an initial data distribution to a target data distribution. This target data distribution is usually chosen to optimise some objective for the algorithmic kernel under study (good computational balance or low communication volume or cost), and therefore to provide high efficiency for that kernel. However, the choice of a distribution minimizing the target objective is not unique. This leads to generalizing the redistribution problem as follows: find a re-mapping of data items onto processors such that the data redistribution cost is minimal, and the operation remains as efficient. This work studies the complexity of this generalized problem. We compute optimal solutions and evaluate, through simulations, their gain over classical redistribution. We also show the NP-hardness of the problem to find the optimal data partition and processor permutation (defined by new subsets) that minimize the cost of redistribution followed by a simple computational kernel. Finally, experimental validation of the new redistribution algorithms are conducted on a multicore cluster, for both a 1D-stencil kernel and a more compute-intensive dense linear algebra routine.

6.13. On the hierarchically structured bin packing problem

We study the hierarchically structured bin packing problem [14]. In this problem, the items to be packed into bins are at the leaves of a tree. The objective of the packing is to minimize the total number of bins into which the descendants of an internal node are packed, summed over all internal nodes. We investigate an existing algorithm and make a correction to the analysis of its approximation ratio. Further results regarding the structure of an optimal solution and a strengthened inapproximability result are given.

6.14. Heuristics for the bipartite matching problem

We propose two heuristics for the bipartite matching problem that are amenable to shared-memory parallelization [18]. The first heuristic is very intriguing from parallelization perspective. It has no significant algorithmic synchronization overhead and no conflict resolution is needed across threads. We show that this heuristic has an approximation ratio of around 0.632. The second heuristic is designed to obtain a larger matching by employing the well-known Karp-Sipser heuristic on a judiciously chosen subgraph of the original graph. We show that the Karp-Sipser heuristic always finds a maximum cardinality matching in the chosen subgraph. Although the Karp-Sipser heuristic is hard to parallelize for general graphs, we exploit the structure of the selected subgraphs to propose a specialized implementation which demonstrates a very good scalability. Based on our experiments and theoretical evidence, we conjecture that this second heuristic obtains matchings with cardinality of at least 0.866 of the maximum cardinality. We discuss parallel implementations of the proposed heuristics on shared memory systems. Experimental results, for demonstrating speed-ups and verifying the theoretical results in practice, are provided.

6.15. Fill-in reduction in sparse matrix factorizations using hypergraphs

We discuss the use of hypergraph partitioning based methods in fill-reducing orderings of sparse matrices for Cholesky, LU and QR factorizations [33]. For the Cholesky factorization, we investigate a recent result on pattern-wise decomposition of sparse matrices, generalize the result, and develop algorithmic tools to obtain more effective ordering methods. The generalized results help us formulate the fill-reducing ordering problem for LU factorization as we do for the Cholesky case, without ever symmetrizing the given matrix A as $|A| + |A^T|$ or $|A^T||A|$. For the QR factorization, we adopt a recently proposed technique to use hypergraph models in a fairly standard manner. The method again does not form the possibly much denser matrix $|A^T||A|$. We also discuss alternatives for LU and QR factorization cases where the symmetrized matrix can be used. We provide comparisons with the most common alternatives in all three cases.

6.16. On partitioning two dimensional finite difference meshes for distributed memory parallel computers

We investigate the problem of partitioning finite difference meshes in two dimensions among the processors of a parallel computer [20]. The objective is to achieve a perfect load balance while minimizing the communication cost. There are well-known graph, hypergraph, and geometry-based partitioning algorithms for this problem. The known geometric algorithms have linear running time and obtain the best results for very special mesh sizes and processor numbers. We propose another geometric algorithm. The proposed algorithm is linear; is applicable to much more cases than some well-known alternatives; obtains better results than the graph partitioning algorithms; obtains better results than the hypergraph partitioning algorithms almost always. Our algorithm also obtains better results than a known asymptotically-optimal algorithm for some small number of processors. We also catalog related theoretical results.

6.17. A symmetry preserving algorithm for matrix scaling

We present an iterative algorithm which asymptotically scales the ∞-norm of each row and each column of a matrix to one [12]. This scaling algorithm preserves symmetry of the original matrix and shows fast linear convergence with an asymptotic rate of 1/2. We discuss extensions of the algorithm to the one-norm, and by inference to other norms. For the 1-norm case, we show again that convergence is linear, with the rate dependent on the spectrum of the scaled matrix. We demonstrate experimentally that the scaling algorithm improves the conditioning of the matrix and that it helps direct solvers by reducing the need for pivoting. In particular, for symmetric matrices the theoretical and experimental results highlight the potential of the proposed algorithm over existing alternatives.

6.18. Direct solvers for sparse linear systems

In the context of the MUMPS sparse direct solver (see Section 5.1), we worked in 2014 on: block-low-rank solvers and shared-memory parallelism [4], [13], hybrid (shared-distributed) parallelism and efficient collective communications in asynchronous environments [2], and scheduling strategies to decrease the memory-usage of multifrontal solvers. Quite significant performance gains have been obtained on up to 2000 cores of a Bullx DLC system (CALMIP mesocentre), some of the corresponding developments will be made available in the next release of our solver. We also worked on setting up a consortium of industrial users to fund engineers working on MUMPS (see Section 7.1). These activities were done in collaboration with INP Toulouse and with CERFACS, CNRS, ENS Lyon, Univ. Bordeaux, EDF, LSTC (Livermore, California) and EMGS (Norway).

7. Bilateral Contracts and Grants with Industry

7.1. Bilateral Contracts with Industry

Related to the evolutions and support of the MUMPS solver (see Section5.1), we worked on:

- setting up a consortium of industrial users to fund the project. Four membership contracts were signed this year by Altair, EDF, LSTC and Michelin.
- a contract with EMGS (Norway) related to low-rank compression for geophysics applications; the contract is managed by INP Toulouse.

8. Partnerships and Cooperations

8.1. National Initiatives

8.1.1. ANR

ANR White Project RESCUE (2010-2015), 4 years. The ANR White Project RESCUE was launched in November 2010, for a duration of 48 months (and was later extended for 6 additional months). It gathers three Inria partners (ROMA, Grand-Large and Hiepacs) and is led by ROMA. The main objective of the project is to develop new algorithmic techniques and software tools to solve the *exascale resilience problem*. Solving this problem implies a departure from current approaches, and calls for yet-to-be-discovered algorithms, protocols and software tools.

This proposed research follows three main research thrusts. The first thrust deals with novel *checkpoint protocols*. The second thrust entails the development of novel *execution models*, i.e., accurate stochastic models to predict (and, in turn, optimize) the expected performance (execution time or throughput) of large-scale parallel scientific applications. In the third thrust, we will develop novel *parallel algorithms* for scientific numerical kernels.

ANR Project SOLHAR (2013-2017), 4 years. The ANR Project SOLHAR was launched in November 2013, for a duration of 48 months. It gathers five academic partners (the HiePACS, Cepage, ROMA and Runtime Inria project-teams, and CNRS-IRIT) and two industrial partners (CEA/CESTA and EADS-IW). 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 proposed research is organized along three distinct research thrusts. The first objective deals with linear algebra kernels suitable for heterogeneous computing platforms. The second one focuses on runtime systems to provide efficient and robust implementation of dense linear algebra algorithms. The third one is concerned with scheduling this particular application on a heterogeneous and dynamic environment.

8.1.2. Inria Project Lab C2S@Exa - Computer and Computational Scienecs at Exascale

Participants: Olivier Aumage [RUNTIME project-team, Inria Bordeaux - Sud-Ouest], Jocelyne Erhel [SAGE project-team, Inria Rennes - Bretagne Atlantique], Philippe Helluy [TONUS project-team, Inria Nancy - Grand-Est], Laura Grigori [ALPINE project-team, Inria Saclay - Île-de-France], Jean-Yves L'excellent [ROMA project-team, Inria Grenoble - Rhône-Alpes], Thierry Gautier [MOAIS project-team, Inria Grenoble - Rhône-Alpes], Luc Giraud [HIEPACS project-team, Inria Bordeaux - Sud-Ouest], Michel Kern [POMDAPI project-team, Inria Paris - Rocquencourt], Stéphane Lanteri [Coordinator of the project], François Pellegrini [BACCHUS project-team, Inria Bordeaux - Sud-Ouest], Christian Perez [AVALON project-team, Inria Grenoble - Rhône-Alpes], Frédéric Vivien [ROMA project-team, Inria Grenoble - Rhône-Alpes].

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. At the current state of the art in technologies and methodologies, a multidisciplinary approach is required to overcome the challenges raised by the development of highly scalable numerical simulation software that can exploit computing platforms offering several hundreds of thousands of cores. Hence, the main objective of C2S@Exa is the establishment of a continuum of expertise in the computer science and numerical mathematics domains, by gathering researchers from Inria project-teams whose research and development activities are tightly linked to high performance computing issues in these domains. More precisely, 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.

8.2. European Initiatives

8.2.1. FP7 & H2020 Projects

8.2.1.1. SCORPIO

Type: FP7

Defi: Future and Emerging Technologies

Instrument: Specific Targeted Research Project

Objectif: Challenging current Thinking

Duration: June 2013 - May 2016

Coordinator: Nikolaos Bellas

Partners: CERTH, Greece; EPFL, Switzerland; RWTH Aachen University, Germany; The Queen's University of Belfast, UK; IMEC, Belgium

Inria contact: Frédéric Vivien

Abstract: A new computing paradigm that exploits uncertainty to design systems that are energy-efficient and scale gracefully under hardware errors by operating below the nominal operating point, in a controlled way, without inducing massive or fatal errors.

8.3. International Initiatives

8.3.1. Inria International Labs

In 2014, the University of Illinois at Urbana-Champaign, Inria, the French national computer science institute, Argonne National Laboratory, Barcelona Supercomputing Center, and Jülich Supercomputing Centre formed the Joint Laboratory on Extreme Scale Computing (JLESC), a follow-up of the Inria-Illinois Joint Laboratory for Petascale Computing. The Joint Laboratory is based at Illinois and includes researchers from Inria, and the National Center for Supercomputing Applications, ANL, BSC, and JSC. It focuses on software challenges found in extreme scale high-performance computers.

Research areas include:

- Scientific applications (big compute and big data) that are the drivers of the research in the other topics of the joint-laboratory.
- Modeling and optimizing numerical libraries, which are at the heart of many scientific applications.
- Novel programming models and runtime systems, which allow scientific applications to be updated or reimagined to take full advantage of extreme-scale supercomputers.
- Resilience and Fault-tolerance research, which reduces the negative impact when processors, disk drives, or memory fail in supercomputers that have tens or hundreds of thousands of those components.
- I/O and visualization, which are important part of parallel execution for numerical silulations and data analytics
- HPC Clouds, that may execute a portion of the HPC workload in the near future.

Several members of the ROMA team are involved in the JLESC joint lab through their research on resilience. Yves Robert is the scientific representant of Inria in JLESC.

8.3.2. Inria Associate Teams

The ALOHA associate-team is a joint project of the ROMA team and of the Information and Computer science Department of the University of Hawai'i (UH) at Mānoa, Honolulu, USA. Building on a vast array of theoretical techniques and expertise developed in the field of parallel and distributed computing, and more particularly application *scheduling*, we tackle database questions from a fresh perspective. To this end, this proposal includes:

- a group that specializes in database systems research and who has both industrial and academic experience, the group of Lipyeow Lim (UH);
- a group that specializes in practical aspects of scheduling problems and in simulation for emerging platforms and applications, and who has a long experience of multidisciplinary research, the group of Henri Casanova (UH);
- a group that specializes in the theoretical aspects of scheduling problems and resource management (the ROMA team).

The research work focuses on the following three thrusts:

1. Online, multi-criteria query optimization
2. Fault-Tolerance for distributed databases
3. Query scheduling for distributed databases

8.4. International Research Visitors

8.4.1. Visits to International Teams

8.4.1.1. Research stays abroad

Yves Robert has been appointed as a visiting scientist by the ICL laboratory (headed by Jack Dongarra) at the University of Tennessee Knoxville. He collaborates with several ICL researchers on high-performance linear algebra and resilience methods at scale.

9. Dissemination

9.1. Promoting Scientific Activities

9.1.1. Scientific council

Yves Robert is a member of the Scientifc Council of Maison de la Simulation, Saclay.

9.1.2. Scientific events organisation

9.1.2.1. General chair, scientific chair

Yves Robert is a member of the Steering Committee of IPDPS, of HCW of HeteroPar, and of Edu-EuroPar.

9.1.2.2. Member of the organizing committee

Yves Robert and Frédéric Vivien organized the *Scheduling for large-scale platforms* in Lyon, July 1-4, 2014. There were 55 participants.

Bora Uçar organized the CSC14 workshop, the *Sixth SIAM Workshop on Combinatorial Scientific Computing* in Lyon, July 21-23, 2014. There were 54 participants.

9.1.3. Scientific events selection

9.1.3.1. Responsability in conference program committees

Anne Benoit was vice-program chair, for the track *Algorithms*, of IPDPS 2014, and workshops co-chair of ICPP 2014.

Yves Robert was program vice-chair for the track *Algorithms* of SC'14.

Frédéric Vivien was program vice-chair, for the algorithms track, of HiPC 2014, and was co-responsible of the stream "Algorithmes distribués, multi-agents et calcul parallèle" for ROADEF 2014.

9.1.3.2. Member of conference program committees

Anne Benoit was a member of the program committees of the following conferences and workshops: HCW 2014, PDP 2014, Ena-HPC 2014 and CCGrid 2014.

Jean-Yves L'Excellent was a member of the program committees of ComPAS 2014, Vecpar 2014, CSC 2014.

Loris Marchal was a member of the program committee of IPDPS'2014 and HeteroPar'2014.

Yves Robert was a member of the program committee of the following conferences and workshops: ISCIS'14, EduPar'14, ICCS'14, and FTXS'14.

Bora Uçar was was a member of the program committee for MPP2014 (a workshop of SBAC-PAD2014), HiPC14, IPDPS 2014, PMAA14, and PCO 2014 (a workshop of IPDPS).

Frédéric Vivien was a member of the program committee of the following conferences and workshops: SC'14, posters of SC'14, IPDPS 2014, ComPAS'2014, PDP 2014, and EduHPC'14.

9.1.4. Roles in international organizations

Bora Uçar was elected as the Secretary of the SIAM activity group on Supercomputing. (1 January 2014–12 December 2015). He was also appointed as the member of the steering committee of Combinatorial Scientific Computing.

9.1.5. Journal

9.1.5.1. Member of the editorial board

Anne Benoit is an associate editor of the *Journal of Parallel and Distributed Computing (JPDC)* and of the *Journal of Sustainable Computing: Informatics and Systems (SUSCOM)*.

Yves Robert is an associate editor of *International Journal of High Performance Computing Applications (IJHPCA)*, *International Journal of Grid and Utility Computing (IJGUC)*, and *Journal of Computational Science (JOCS)*.

Frédéric Vivien is an associate editor of *Parallel Computing*.

9.2. Teaching - Supervision - Juries

9.2.1. Teaching

Licence: Anne Benoit, Systèmes et Réseaux, 48h, L3, École normale supérieure de Lyon, France.

Licence: Yves Robert, Algorithmes, 48h, L3, École normale supérieure de Lyon, France.

Master: Frédéric Vivien, Algorithmique et Programmation Parallèles, 36 h, M1, École normale supérieure de Lyon, France.

Master: Frédéric Vivien, Algorithms for High-Performance Computing Platforms, 36 h, M2, École normale supérieure de Lyon, France.

Master: Bora Uçar, Combinatorial Scientific Computing, 36 h, M2, École normale supérieure de Lyon, France.

9.2.2. Supervision

PhD: Guillaume Aupy, Resilient and energy-efficient scheduling algorithms at scale, École normale supérieure de Lyon, September 16, 2014, Anne Benoit and Yves Robert.

PhD: Dounia Zaidouni, Combining checkpointing and other resilience mechanisms for exascale systems, December 10, 2014, École normale supérieure de Lyon, Frédéric Vivien and Yves Robert.

PhD: Wissam M. Sid-Lakhdar, Scaling the solution of large sparse linear systems using multifrontal methods on hybrid shared-distributed memory architectures, December 1 2014, Jean-Yves L'Excellent.

PhD in progress: Aurélien Cavelan, Algorithms for detecting and correcting silent errors, September 1, 2014, Anne Benoît and Yves Robert.

PhD in progress: Oguz Kaya, Parallel and Distributed Sparse Tensor Computations, September 1, 2014, Yves Robert and Bora Uçar.

PhD in progress: Julien Herrmann, Numerical algorithms for large-scale platforms, September 1, 2012, Loris Marchal and Yves Robert.

9.2.3. Juries

Bora Uçar was a member of the PhD defense committee of Clément Vuchener as a "rapporteur", Université de Bordeaux, École doctoral de mathématiques et d'Informatique.

Yves Robert was a member of the HdR defense committee of Frédéric Suter, ENS Lyon.

10. Bibliography

Publications of the year

Doctoral Dissertations and Habilitation Theses

[1] G. AUPY. *Resilient and energy-efficient scheduling algorithms at scale*, École Normale Supérieure de Lyon, September 2014, https://hal.inria.fr/tel-01075111

[2] W. M. SID-LAKHDAR. *Scaling the solution of large sparse linear systems using multifrontal methods on hybrid shared-distributed memory architectures* , École Normale Supérieure de Lyon, December 2014, https://hal.inria.fr/tel-01111259

[3] D. ZAIDOUNI. *Combining checkpointing and other resilience mechanisms for exascale systems*, École normale supérieure de Lyon, December 2014, https://hal.inria.fr/tel-01110981

Articles in International Peer-Reviewed Journals

[4] P. R. AMESTOY, A. BUTTARI, G. JOSLIN, J.-Y. L'EXCELLENT, W. M. SID-LAKHDAR, C. WEISBECKER, M. FORZAN, C. POZZA, R. PERRIN, V. PELLISSIER. *Shared-Memory Parallelism and Low-Rank Approximation Techniques Applied to Direct Solvers in FEM Simulation*, in "IEEE Transactions on Magnetics", February 2014, vol. 50, n^{o} 2 [*DOI :* 10.1109/TMAG.2013.2284024], https://hal.inria.fr/hal-01060297

[5] G. AUPY, A. BENOIT, M. JOURNAULT, Y. ROBERT. *Power-aware replica placement in tree networks with multiple servers per client*, in "Sustainable Computing", September 2014, 18 p. [*DOI :* 10.1016/J.SUSCOM.2014.08.013], https://hal.inria.fr/hal-01059364

[6] A. BENOIT, L.-C. CANON, L. MARCHAL. *Non-clairvoyant reduction algorithms for heterogeneous platforms*, in "Concurrency and Computation Practice and Experience", 2014, 13 p. [*DOI :* 10.1002/CPE.3347], https://hal.inria.fr/hal-01090232

[7] G. BOSILCA, A. BOUTEILLER, J. DONGARRA, T. HÉRAULT, Y. ROBERT. *Composing resilience techniques: ABFT, periodic and incremental checkpointing*, in "The International Journal of Networking and Computing", March 2015, 18 p. , https://hal.inria.fr/hal-01091930

[8] M. BOUGERET, H. CASANOVA, Y. ROBERT, F. VIVIEN, D. ZAIDOUNI. *Using group replication for resilience on exascale systems*, in "International Journal of High Performance Computing Applications", May 2014, vol. 28, n^{o} 2, pp. 210-224 [*DOI :* 10.1177/1094342013505348], https://hal.inria.fr/hal-00881463

[9] H. CASANOVA, F. DUFOSSÉ, Y. ROBERT, F. VIVIEN. *Mapping Applications on Volatile Resources*, in "International Journal of High Performance Computing Applications", 2015, 19 p. , https://hal.inria.fr/hal-00923948

[10] J. DONGARRA, T. HÉRAULT, Y. ROBERT. *Performance and reliability trade-offs for the double checkpointing algorithm*, in "The International Journal of Networking and Computing", March 2014, vol. 4, n^{o} 1, 18 p. , https://hal.inria.fr/hal-01091928

[11] J. HERRMANN, L. MARCHAL, Y. ROBERT. *Memory-aware tree traversals with pre-assigned tasks*, in "Journal of Parallel and Distributed Computing", 2014, https://hal.inria.fr/hal-01026405

[12] A. KNIGHT, D. RUIZ, B. UÇAR. *A Symmetry Preserving Algorithm for Matrix Scaling*, in "SIAM Journal on Matrix Analysis and Applications", 2014, vol. 35, n^{o} 3, 25 p. [*DOI :* 10.1137/110825753], https://hal.inria.fr/inria-00569250

[13] J.-Y. L'EXCELLENT, W. M. SID-LAKHDAR. *A study of shared-memory parallelism in a multifrontal solver*, in "Parallel Computing", February 2014, vol. 40, n^{o} 3-4, pp. 34-46 [*DOI :* 10.1016/J.PARCO.2014.02.003], https://hal.inria.fr/hal-01060322

[14] T. LAMBERT, L. MARCHAL, B. UÇAR. *Comments on the hierarchically structured bin packing problem*, in "Information Processing Letters", 2015, vol. 115, n^{o} 2, pp. 306–309 [*DOI :* 10.1016/J.IPL.2014.10.001], https://hal.inria.fr/hal-01071414

International Conferences with Proceedings

[15] G. AUPY, A. BENOIT, M. JOURNAULT, Y. ROBERT. *Power-aware replica placement in tree networks with multiple servers per client*, in "EuroPar - 20th International European Conference on Parallel Processing", Porto, Portugal, August 2014, vol. 8632, 11 p. [*DOI :* 10.1007/978-3-319-09873-9_51], https://hal.inria.fr/hal-01059365

[16] O. BEAUMONT, L. MARCHAL. *Analysis of Dynamic Scheduling Strategies for Matrix Multiplication on Heterogeneous Platforms*, in "ACM Symposium on High-Performance Parallel and Distributed Computing", Vancouver, Canada, June 2014 [*DOI :* 10.1145/2600212.2600223], https://hal.inria.fr/hal-01090254

[17] H. CASANOVA, L. LIM, Y. ROBERT, F. VIVIEN, D. ZAIDOUNI. *Cost-Optimal Execution of Boolean Query Trees with Shared Streams*, in "28th IEEE International Parallel & Distributed Processing Symposium", Phoenix, United States, IEEE, May 2014, https://hal.inria.fr/hal-00923953

[18] F. DUFOSSÉ, K. KAYA, B. UÇAR. *Bipartite matching heuristics with quality guarantees on shared memory parallel computers*, in "IPDPS 2014", Phoenix, Arizona, United States, IEEE Computer Society, May 2014, 28 p. , https://hal.inria.fr/hal-00877211

[19] M. FAVERGE, J. HERRMANN, J. LANGOU, B. LOWERY, Y. ROBERT, J. DONGARRA. *Designing LU-QR hybrid solvers for performance and stability*, in "IEEE International Parallel & Distributed Processing Symposium", Phoenix, United States, May 2014, https://hal.inria.fr/hal-00930238

[20] A. GRANDJEAN, B. UÇAR. *On Partitioning Two Dimensional Finite Difference Meshes for Distributed Memory Parallel Computers*, in "PDP - 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing", Turin, Italy, IEEE, February 2014, pp. 9 - 16 [*DOI :* 10.1109/PDP.2014.10], https://hal.inria.fr/hal-01111292

[21] T. HÉRAULT, J. HERRMANN, L. MARCHAL, Y. ROBERT. *Determining the Optimal Redistribution for a Given Data Partition*, in "13th International Symposium on Parallel and Distributed Computing (ISPDC)", Marseille, France, 2014 [*DOI :* 10.1109/ISPDC.2014.16], https://hal.inria.fr/hal-01111537

Books or Proceedings Editing

[22] B. UÇAR (editor). *Book of Abstracts of the Sixth SIAM Workshop on Combinatorial Scientific Computing*, SIAM, August 2014, 82 p. , https://hal.inria.fr/hal-01054876

Research Reports

[23] G. AUPY, A. BENOIT. *Approximation algorithms for energy, reliability and makespan optimization problems*, July 2014, n^{o} RR-8107, 32 p. , https://hal.inria.fr/hal-00742754

[24] G. AUPY, A. BENOIT, H. CASANOVA, Y. ROBERT. *Scheduling computational workflows on failure-prone platforms*, ENS Lyon ; LIP ; Inria ; CNRS ; Université Lyon 1, October 2014, n^{o} RR-8609, https://hal.inria.fr/hal-01075100

[25] G. AUPY, A. BENOIT, M. JOURNAULT, Y. ROBERT. *Power-aware replica placement in tree networks with multiple servers per client*, February 2014, n^{o} RR-8474, https://hal.inria.fr/hal-00949252

[26] A. BENOIT, A. CAVELAN, Y. ROBERT, H. SUN. *Assessing general-purpose algorithms to cope with fail-stop and silent errors*, Inria, September 2014, n^{o} RR-8599, https://hal.inria.fr/hal-01066664

[27] H. CASANOVA, L. LIM, Y. ROBERT, F. VIVIEN, D. ZAIDOUNI. *Cost-Optimal Execution of Boolean DNF Trees with Shared Streams*, Inria, November 2014, n^{o} RR-8616, https://hal.inria.fr/hal-01079868

[28] L. EYRAUD-DUBOIS, L. MARCHAL, O. SINNEN, F. VIVIEN. *Parallel scheduling of task trees with limited memory*, 2014, n^{o} RR-8606, 37 p. , https://hal.inria.fr/hal-01070356

[29] A. GAINARU, G. AUPY, A. BENOIT, F. CAPPELLO, Y. ROBERT, M. SNIR. *Scheduling the I/O of HPC applications under congestion*, LIP, October 2014, n^{o} RR-8519, 25 p. , https://hal.inria.fr/hal-00983789

[30] A. GUERMOUCHE, L. MARCHAL, B. SIMON, F. VIVIEN. *Scheduling Trees of Malleable Tasks for Sparse Linear Algebra*, ENS Lyon, October 2014, n^{o} 8616, https://hal.inria.fr/hal-01077413

[31] J. HERRMANN, L. MARCHAL, Y. ROBERT. *Memory-aware list scheduling for hybrid platforms*, February 2014, n^{o} RR-8461, 30 p. , https://hal.inria.fr/hal-00944336

[32] T. HÉRAULT, J. HERRMANN, L. MARCHAL, Y. ROBERT. *Determining the optimal redistribution*, March 2014, n^{o} RR-8499, https://hal.inria.fr/hal-00960452

[33] O. KAYA, E. KAYAASLAN, B. UÇAR, I. S. DUFF. *Fill-in reduction in sparse matrix factorizations using hypergraphs*, January 2014, n^{o} RR-8448, https://hal.inria.fr/hal-00932882

[34] L. MARCHAL, F. VIVIEN, B. SIMON. *Scheduling malleable task trees*, September 2014, n^{o} RR-8587, https://hal.inria.fr/hal-01059704

Scientific Popularization

[35] B. UÇAR, A.-J. N. YZELMAN. *SIAM's CSC Workshop Series Marks 10th Year*, December 2014, A news article on the sixth SIAM Workshop on Combinatorial Scientific Computing,, https://hal.inria.fr/hal-01111302

Other Publications

[36] M. FAVERGE, J. HERRMANN, J. LANGOU, B. LOWERY, Y. ROBERT, J. DONGARRA. *Mixing LU and QR factorization algorithms to design high-performance dense linear algebra solvers*, December 2014, submitted to JPDC special issue for IPDPS14, https://hal.inria.fr/hal-01107457

References in notes

[37] *Blue Waters Newsletter*, dec 2012, http://cgi.ncsa.illinois.edu/BlueWaters/pdfs/bw-newsletter-1212.pdf

[38] *Blue Waters Resources*, 2013, https://bluewaters.ncsa.illinois.edu/data

[39] *The BOINC project*, 2013, http://boinc.berkeley.edu/

[40] *Final report of the Department of Energy Fault Management Workshop*, December 2012, http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/FaultManagement-wrkshpRpt-v4-final.pdf

[41] *System Resilience at Extreme Scale: white paper*, 2008, DARPA, http://institute.lanl.gov/resilience/docs/IBM%20Mootaz%20White%20Paper%20System%20Resilience.pdf

[42] *Top500 List - November*, 2011, http://www.top500.org/list/2011/11/

[43] *Top500 List - November*, 2012, http://www.top500.org/list/2012/11/

[44] I. ASSAYAD, A. GIRAULT, H. KALLA. *Tradeoff exploration between reliability power consumption and execution time*, in "Proceedings of SAFECOMP, the Conf. on Computer Safety, Reliability and Security", Washington, DC, USA, 2011

[45] H. AYDIN, Q. YANG. *Energy-aware partitioning for multiprocessor real-time systems*, in "IPDPS'03, the IEEE Int. Parallel and Distributed Processing Symposium", 2003, pp. 113–121

[46] N. BANSAL, T. KIMBREL, K. PRUHS. *Speed Scaling to Manage Energy and Temperature*, in "Journal of the ACM", 2007, vol. 54, n^{o} 1, pp. 1 – 39, http://doi.acm.org/10.1145/1206035.1206038

[47] A. BENOIT, L. MARCHAL, J.-F. PINEAU, Y. ROBERT, F. VIVIEN. *Scheduling concurrent bag-of-tasks applications on heterogeneous platforms*, in "IEEE Transactions on Computers", 2010, vol. 59, n^{o} 2, pp. 202-217

[48] L. S. BLACKFORD, J. CHOI, A. CLEARY, E. D'AZEVEDO, J. DEMMEL, I. DHILLON, J. DONGARRA, S. HAMMARLING, G. HENRY, A. PETITET, K. STANLEY, D. WALKER, R. C. WHALEY. *ScaLAPACK Users' Guide*, SIAM, 1997

[49] S. BLACKFORD, J. DONGARRA. *Installation Guide for LAPACK*, LAPACK Working Note, June 1999, n^{o} 41, originally released March 1992

[50] A. BUTTARI, J. LANGOU, J. KURZAK, J. DONGARRA. *Parallel tiled QR factorization for multicore architectures*, in "Concurrency: Practice and Experience", 2008, vol. 20, n^{o} 13, pp. 1573-1590

[51] J.-J. CHEN, T.-W. KUO. *Multiprocessor energy-efficient scheduling for real-time tasks*, in "ICPP'05, the Int. Conference on Parallel Processing", 2005, pp. 13–20

[52] S. DONFACK, L. GRIGORI, W. GROPP, L. V. KALE. *Hybrid Static/dynamic Scheduling for Already Optimized Dense Matrix Factorization*, in "Parallel Distributed Processing Symposium (IPDPS), 2012 IEEE 26th International", 2012, pp. 496-507, http://dx.doi.org/10.1109/IPDPS.2012.53

[53] J. DONGARRA, J.-F. PINEAU, Y. ROBERT, Z. SHI, F. VIVIEN. *Revisiting Matrix Product on Master-Worker Platforms*, in "International Journal of Foundations of Computer Science", 2008, vol. 19, n^{o} 6, pp. 1317-1336

[54] J. DONGARRA, J.-F. PINEAU, Y. ROBERT, F. VIVIEN. *Matrix Product on Heterogeneous Master-Worker Platforms*, in "13th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming", Salt Lake City, Utah, February 2008, pp. 53–62

[55] I. S. DUFF, J. K. REID. *The multifrontal solution of indefinite sparse symmetric linear systems*, in ""ACM Transactions on Mathematical Software"", 1983, vol. 9, pp. 302-325

[56] I. S. DUFF, J. K. REID. *The multifrontal solution of unsymmetric sets of linear systems*, in "SIAM Journal on Scientific and Statistical Computing", 1984, vol. 5, pp. 633-641

[57] L. GRIGORI, J. W. DEMMEL, H. XIANG. *Communication avoiding Gaussian elimination*, in "Proceedings of the 2008 ACM/IEEE conference on Supercomputing", Piscataway, NJ, USA, SC '08, IEEE Press, 2008, 29:1 p. , http://dl.acm.org/citation.cfm?id=1413370.1413400

[58] B. HADRI, H. LTAIEF, E. AGULLO, J. DONGARRA. *Tile QR Factorization with Parallel Panel Processing for Multicore Architectures*, in "IPDPS'10, the 24st IEEE Int. Parallel and Distributed Processing Symposium", 2010

[59] J. W. H. LIU. *The multifrontal method for sparse matrix solution: Theory and Practice*, in "SIAM Review", 1992, vol. 34, pp. 82–109

[60] R. MELHEM, D. MOSSÉ, E. ELNOZAHY. *The Interplay of Power Management and Fault Recovery in Real-Time Systems*, in "IEEE Transactions on Computers", 2004, vol. 53, n^{o} 2, pp. 217-231

[61] A. J. OLINER, R. K. SAHOO, J. E. MOREIRA, M. GUPTA, A. SIVASUBRAMANIAM. *Fault-aware job scheduling for bluegene/l systems*, in "IPDPS'04, the IEEE Int. Parallel and Distributed Processing Symposium", 2004, pp. 64–73

[62] G. N. S. PRASANNA, B. R. MUSICUS. *Generalized Multiprocessor Scheduling and Applications to Matrix Computations*, in "IEEE Trans. Parallel Distrib. Syst.", 1996, vol. 7, n^{o} 6, pp. 650-664

[63] G. N. S. PRASANNA, B. R. MUSICUS. *The Optimal Control Approach to Generalized Multiprocessor Scheduling*, in "Algorithmica", 1996, vol. 15, n^{o} 1, pp. 17-49

[64] G. QUINTANA-ORTÍ, E. QUINTANA-ORTÍ, R. A. VAN DE GEIJN, F. G. V. ZEE, E. CHAN. *Programming Matrix Algorithms-by-Blocks for Thread-Level Parallelism*, in "ACM Transactions on Mathematical Software", 2009, vol. 36, n^{o} 3

[65] Y. ROBERT, F. VIVIEN. *Algorithmic Issues in Grid Computing*, in "Algorithms and Theory of Computation Handbook", Chapman and Hall/CRC Press, 2009

[66] G. ZHENG, X. NI, L. V. KALE. *A scalable double in-memory checkpoint and restart scheme towards exascale*, in "Dependable Systems and Networks Workshops (DSN-W)", 2012, http://dx.doi.org/10.1109/DSNW.2012.6264677

[67] D. ZHU, R. MELHEM, D. MOSSÉ. *The effects of energy management on reliability in real-time embedded systems*, in "Proc. of IEEE/ACM Int. Conf. on Computer-Aided Design (ICCAD)", 2004, pp. 35–40