IN PARTNERSHIP WITH:
CNRS

Université Claude Bernard (Lyon 1)

Ecole normale supérieure de Lyon

Activity Report 2013

Team ROMA

Resource Optimization: Models, Algorithms, and scheduling

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.

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 and IUF, Associate Professor, HdR]
Yves Robert [ENS Lyon and IUF, Professor, HdR]

External Collaborators
Patrick Amestoy [INP Toulouse, HdR]
Alfredo Buttari [CNRS]

PhD Students
Guillaume Aupy [ENS Lyon]
Julien Herrmann [ENS Lyon]
Wissam M. Sid-Lakhdar [ENS Lyon]
Dounia Zaidouni [Inria, funded by ANR RESCUE project]

Post-Doctoral Fellows
Sheng Di [Inria, since Dec. 2013, funded by Genci project]
Enver Kayaaslan [Inria, since Oct. 2013]

Administrative Assistant
Evelyne Blesle [Inria]

2. Overall Objectives

2.1. Introduction

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 [68] each includes more than 500,000 cores and deliver a sustained performance of more than 10 Peta FLOPS. The volunteer computing platform BOINC [64] 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.

2.2. Highlights of the Year

Anne Benoit, Yves Robert and Frédéric Vivien published a textbook entitled "A Guide to Algorithm Design: Paradigms, Methods, and Complexity Analysis" [40].

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 [67], had an average of 2.33 faults per day during the period from August 2008 to February 2010 [91]. 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" [66] 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 [62] 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 [65]).

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 [72], [78], [79], [90], 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 $2 \leq \alpha \leq 3$ [70], [71], [76]; 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 [92] 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 [88]. 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 [92]; (ii) *replication* consists in executing the same task on several processors simultaneously, in order to meet the reliability constraints [69]; and (iii) *checkpointing* consists in "saving" the work done at some certain instants, hence reducing the amount of work lost when a failure occurs [87].

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 [63]. 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" [84]. 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 [80], [81], [86]. 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 [74] 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 [73] 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 [75], [89], [85]. 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 [77]. 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. Software and Platforms

5.1. MUMPS

Participants: Patrick Amestoy, Alfredo Buttari, Jean-Yves L'Excellent [correspondent], 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 and the wide range of functionalities available.

The latest public release is MUMPS 4.10.0 (May 2011).

The development of MUMPS was initiated by the European project PARASOL (Esprit 4, LTR project 20160, 1996-1999), whose results and developments were public domain. Since then, MUMPS has been supported by CERFACS, CNRS, ENS Lyon, INPT(ENSEEIHT)-IRIT, Inria, and University of Bordeaux. Following a contractual agreement signed by those institutes, the next release of MUMPS will be distributed under the Cecill-C license; a technical committee was also defined, currently composed of Patrick Amestoy, Abdou Guermouche, and Jean-Yves L'Excellent.

In the context of an ADT project (Action of Technological Development), Maurice Brémond (from Inria "SED" service in Grenoble) also worked part-time on the project, in particular on visualization tools helping researchers to analyze the behaviour of a parallel MUMPS execution.

More information on MUMPS is available on http://mumps-solver.org. See also Section 6.20 of this report.

6. New Results

6.1. Scheduling tree-shaped task graphs to minimize memory and makespan

In this work [37], we investigate the execution of tree-shaped task graphs using multiple processors. Each edge of such a tree represents a large IO file. A task can only be executed if all input and output files fit into memory, and a file can only be removed from memory after it has been consumed. Such trees arise, for instance, in the multifrontal method of sparse matrix factorization. The maximum amount of memory needed 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 the 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 an inapproximability result even for unit weight trees. Several heuristics are proposed, each with a different optimization focus, and they are analyzed in an extensive experimental evaluation using realistic trees.

6.2. Model and complexity results for tree traversals on hybrid platforms

In this work [35], we study the complexity of traversing tree-shaped workflows whose tasks require large I/O files. We target a heterogeneous architecture with two resources of different types, where each resource has its own memory, such as a multicore node equipped with a dedicated accelerator (FPGA or GPU). Tasks in the workflow are tagged with the type of resource needed for their processing. Besides, a task can be processed on a given resource only if all its input files and output files can be stored in the corresponding memory. At a given execution step, the amount of data stored in each memory 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 work, we establish the complexity of this two-memory scheduling problem, provide inapproximability results, and show how to determine the optimal depth-first traversal. Altogether, these results lay the foundations for memory-aware scheduling algorithms on heterogeneous platforms.

6.3. On the combination of silent error detection and checkpointing

In this work [19], we revisit traditional checkpointing and rollback recovery strategies, with a focus on silent data corruption errors. Contrarily to fail-stop failures, such latent errors cannot be detected immediately, and a mechanism to detect them must be provided. We consider two models: (i) errors are detected after some delays following a probability distribution (typically, an Exponential distribution); (ii) errors are detected through some verification mechanism. In both cases, we compute the optimal period in order to minimize the waste, i.e., the fraction of time where nodes do not perform useful computations. In practice, only a fixed number of checkpoints can be kept in memory, and the first model may lead to an irrecoverable failure. In this case, we compute the minimum period required for an acceptable risk. For the second model, there is no risk of irrecoverable failure, owing to the verification mechanism, but the corresponding overhead is included in the waste. Finally, both models are instantiated using realistic scenarios and application/architecture parameters.

6.4. Checkpointing algorithms and fault prediction

In this series of work [22], [49], we deal with the impact of fault prediction techniques on checkpointing strategies, when the fault-prediction system provides either prediction windows or exact predictions. We extend the classical first-order analysis of Young and Daly in the presence of a fault prediction system, characterized by its recall and its precision. In this framework, we provide optimal algorithms to decide whether and when to take predictions into account, and we derive the optimal value of the checkpointing period. These results allow us to analytically assess the key parameters that impact the performance of fault predictors at very large scale.

6.5. Mapping applications on volatile resources

In this series of work [12], [27], [28], we study the execution of iterative applications on volatile processors such as those found on desktop grids. We envision two models, one where all tasks are assumed to be independent, and another where all tasks are tightly coupled and keep exchanging information throughout the iteration. These two models cover the two extreme points of the parallelization spectrum. We develop master-worker scheduling schemes that attempt to achieve good trade-offs between worker speed and worker availability. Any iteration entails the execution of a fixed number of independent tasks or of tightly-coupled tasks. A key feature of our approach is that we consider a communication model where the bandwidth capacity of the master for sending application data to workers is limited. This limitation makes the scheduling problem more difficult both in a theoretical sense and in a practical sense. Furthermore, we consider that a processor can be in one of three states: available, down, or temporarily preempted by its owner. This preempted state also complicates the scheduling problem. In practical settings, e.g., desktop grids, master bandwidth is limited and processors are temporarily reclaimed. Consequently, addressing the aforementioned difficulties is necessary for successfully deploying master-worker applications on volatile platforms. Our first contribution is to determine the complexity of the scheduling problems in their offline versions, i.e., when processor availability behaviors are known in advance. Even with this knowledge, the problems are NP-hard. Our second contribution is an evaluation of the expectation of the time needed by a worker to complete a set of tasks. We obtain a close formula for independent tasks and an analytical approximation for tightly-coupled tasks. Those evaluations rely on a Markovian assumption for the temporal availability of processors, and are at the heart of some heuristics that aim at favoring “reliable” processors in a sensible manner. Our third contribution is a set of heuristics for both models, which we evaluate in simulation. Our results provide guidance to selecting the best strategy as a function of processor state availability versus average task duration.

6.6. Using group replication for resilience on exascale systems

High performance computing applications must be resilient to faults. The traditional fault-tolerance solution is checkpoint-recovery, by which application state is saved to and recovered from secondary storage throughout execution. It has been shown that, even when using an optimal checkpointing strategy, the checkpointing overhead precludes high parallel efficiency at large scale. Additional fault-tolerance mechanisms must thus be used. Such a mechanism is replication, i.e., multiple processors performing the same computation so that a processor failure does not necessarily imply an application failure. In spite of resource waste, replication can lead to higher parallel efficiency when compared to using only checkpoint-recovery at large scale. In this work [11], we propose to execute and checkpoint multiple application instances concurrently, an approach we term group replication. For Exponential failures we give an upper bound on the expected application execution time. This bound corresponds to a particular checkpointing period that we derive. For general failures, we propose a dynamic programming algorithm to determine non-periodic checkpoint dates as well as an empirical periodic checkpointing solution whose period is found via a numerical search. Using simulation we evaluate our proposed approaches, including comparison to the non-replication case, for both Exponential and Weibull failure distributions. Our broad finding is that group replication is useful in a range of realistic application and checkpointing overhead scenarios for future exascale platforms.

6.7. Unified model for assessing checkpointing protocols at extreme-scale

In this work [10], we present a unified model for several well-known checkpoint/restart protocols. The proposed model is generic enough to encompass both extremes of the checkpoint/restart space, from coordinated approaches to a variety of uncoordinated checkpoint strategies (with message logging). We identify a set of crucial parameters, instantiate them and compare the expected efficiency of the fault tolerant protocols, for a given application/platform pair. We then propose a detailed analysis of several scenarios, including some of the most powerful currently available HPC platforms, as well as anticipated Exascale designs. The results of this analytical comparison are corroborated by a comprehensive set of simulations. Altogether, they outline comparative behaviors of checkpoint strategies at very large scale, thereby providing insight that is hardly accessible to direct experimentation.

6.8. Revisiting the double checkpointing algorithm

In this work [33], we study fast checkpointing algorithms which require distributed access to stable storage. This work revisits the approach base upon double checkpointing, and compares the blocking algorithm of Zheng, Shi, and Kalé, with the non-blocking algorithm of Ni, Meneses, and Kalé in terms of both performance and risk. We also extend the model that they have proposed to assess the impact of the overhead associated to non-blocking communications. We then provide a new peer-to-peer checkpointing algorithm, called the triple checkpointing algorithm, that can work at constant memory, and achieves both higher efficiency and better risk handling than the double checkpointing algorithm. We provide performance and risk models for all the evaluated protocols, and compare them through comprehensive simulations.

6.9. Multi-criteria checkpointing strategies: Optimizing response-time versus resource utilization

Failures are increasingly threatening the efficiency of HPC systems, and current projections of Exascale platforms indicate that rollback recovery, the most convenient method for providing fault tolerance to general-purpose applications, reaches its own limits at such scales. One of the reasons explaining this unnerving situation comes from the focus that has been given to per-application completion time, rather than to platform efficiency. In this work [26], we discuss the case of uncoordinated rollback recovery where the idle time spent waiting recovering processors is used to progress a different, independent application from the system batch queue. We then propose an extended model of uncoordinated checkpointing that can discriminate between idle time and wasted computation. We instantiate this model in a simulator to demonstrate that, with this strategy, uncoordinated checkpointing per application completion time is unchanged, while it delivers near-perfect platform efficiency.

6.10. Optimal checkpointing period: Time vs. energy

In this work [18], we deal with parallel scientific applications using non-blocking and periodic coordinated checkpointing to enforce resilience. We provide a model and detailed formulas for total execution time and consumed energy. We characterize the optimal period for both objectives, and we assess the range of time/energy trade-offs to be made by instantiating the model with a set of realistic scenarios for Exascale systems. We give a particular emphasis to I/O transfers, because the relative cost of communication is expected to dramatically increase, both in terms of latency and consumed energy, for future Exascale platforms.

6.11. Energy-aware checkpointing of divisible tasks with soft or hard deadlines

In this work [20], we aim at minimizing the energy consumption when executing a divisible workload under a bound on the total execution time, while resilience is provided through checkpointing. We discuss several variants of this multi-criteria problem. Given the workload, we need to decide how many chunks to use, what are the sizes of these chunks, and at which speed each chunk is executed. Furthermore, since a failure may occur during the execution of a chunk, we also need to decide at which speed a chunk should be re-executed in the event of a failure. The goal is to minimize the expectation of the total energy consumption, while enforcing a deadline on the execution time, that should be met either in expectation (soft deadline), or in the worst case (hard deadline). For each problem instance, we propose either an exact solution, or a function that can be optimized numerically. The different models are then compared through an extensive set of experiments.

6.12. Assessing the performance of energy-aware mappings

In this work [8], we aim at mapping streaming applications that can be modeled by a series-parallel graph onto a 2-dimensional tiled chip multiprocessor (CMP) architecture. The objective of the mapping is to minimize the energy consumption, using dynamic voltage and frequency scaling (DVFS) techniques, while maintaining a given level of performance, reflected by the rate of processing the data streams. This mapping problem turns out to be NP-hard, and several heuristics are proposed. We assess their performance through comprehensive simulations using the StreamIt workflow suite and randomly generated series-parallel graphs, and various CMP grid sizes.

6.13. Computing the throughput of probabilistic and replicated streaming applications

In this work [7], we investigate how to compute the throughput of probabilistic and replicated streaming applications. We are given (i) a streaming application whose dependence graph is a linear chain; (ii) a one-to-many mapping of the application onto a fully heterogeneous target platform, where a processor is assigned at most one application stage, but where a stage can be replicated onto a set of processors; and (iii) a set of random variables modeling the computation and communication times in the mapping. We show how to compute the throughput of the application, i.e., the rate at which data sets can be processed, under two execution models, the Strict model where the actions of each processor are sequentialized, and the Overlap model where a processor can compute and communicate in parallel. The problem is easy when application stages are not replicated, i.e., assigned to a single processor: in that case the throughput is dictated by the critical hardware resource. However, when stages are replicated, i.e., assigned to several processors, the problem becomes surprisingly complicated: even in the deterministic case, the optimal throughput may be lower than the smallest internal resource throughput. The first contribution of this work is to provide a general method to compute the throughput when mapping parameters are constant or follow I.I.D. exponential laws. The second contribution is to provide bounds for the throughput when stage parameters (computation and communication times) form associated random sequences, and are N.B.U.E. (New Better than Used in Expectation) variables: the throughput is bounded from below by the exponential case and bounded from above by the deterministic case. An extensive set of simulation allows us to assess the quality of the model, and to observe the actual behavior of several distributions.

6.14. Reliability and performance optimization of pipelined real-time systems

In this work [6], we consider pipelined real-time systems that consist of a chain of tasks executing on a distributed platform. The processing of the tasks is pipelined: each processor executes only one interval of consecutive tasks. We are interested in minimizing both the input-output latency and the period of application mapping. For dependability reasons, we are also interested in maximizing the reliability of the system. We therefore assign several processors to each interval of tasks, so as to increase the reliability of the system. Both processors and communication links are unreliable and subject to transient failures. We assume that the arrival of the failures follows a constant parameter Poisson law, and that the failures are statistically independent events. We study several variants of this multiprocessor mapping problem, with several hypotheses on the target platform (homogeneous/heterogeneous speeds and/or failure rates). We provide NP-hardness complexity results, and optimal mapping algorithms for polynomial problem instances. Efficient heuristics are presented to solve the general case, and experimental results are provided.

6.15. Scheduling linear chain streaming applications on heterogeneous systems with failures

In this work [5], we study the problem of optimizing the throughput of streaming applications for heterogeneous platforms subject to failures. Applications are linear graphs of tasks (pipelines), with a type associated to each task. The challenge is to map each task onto one machine of a target platform, each machine having to be specialized to process only one task type, given that every machine is able to process all the types before being specialized in order to avoid costly setups. The objective is to maximize the throughput, i.e., the rate at which jobs can be processed when accounting for failures. Each instance can thus be performed by any machine specialized in its type and the workload of the system can be shared among a set of specialized machines. For identical machines, we prove that an optimal solution can be computed in polynomial time. However, the problem becomes NP-hard when two machines may compute the same task type at different speeds. Several polynomial time heuristics are designed for the most realistic specialized settings. Simulation results assess their efficiency, showing that the best heuristics obtain a good throughput, much better than the throughput obtained with a random mapping. Moreover, the throughput is close to the optimal solution in the particular cases where the optimal throughput can be computed.

6.16. A survey of pipelined workflow scheduling: Models and algorithms

In this survey [4], we consider a large class of applications that need to execute the same workflow on different data sets of identical size. Efficient execution of such applications necessitates intelligent distribution of the application components and tasks on a parallel machine, and the execution can be orchestrated by utilizing task-, data-, pipelined-, and/or replicated-parallelism. The scheduling problem that encompasses all of these techniques is called pipelined workflow scheduling, and it has been widely studied in the last decade. Multiple models and algorithms have flourished to tackle various programming paradigms, constraints, machine behaviors or optimization goals. This work surveys the field by summing up and structuring known results and approaches.

6.17. Reclaiming the energy of a schedule: Models and algorithms

In this work [1], we consider a task graph to be executed on a set of processors. We assume that the mapping is given, say by an ordered list of tasks to execute on each processor, and we aim at optimizing the energy consumption while enforcing a prescribed bound on the execution time. Although it is not possible to change the allocation of a task, it is possible to change its execution speed. Rather than using a local approach such as backfilling, we consider the problem as a whole and study the impact of several speed variation models on its complexity. For continuous speeds, we give a closed-form formula for trees and series-parallel graphs, and we cast the problem into a geometric programming problem for general directed acyclic graphs. We show that the classical dynamic voltage and frequency scaling (DVFS) model with discrete modes leads to an NP-complete problem, even if the modes are regularly distributed (an important particular case in practice, which we analyze as the incremental model). On the contrary, the Vdd-hopping model that allows to switch between different supply voltages (VDD) while executing a task leads to a polynomial solution. Finally, we provide an approximation algorithm for the incremental model, which we extend for the general DVFS model.

6.18. Non-clairvoyant reduction algorithms for heterogeneous platforms

In this work [24], we revisit the classical problem of the reduction collective operation in a heterogeneous environment. We discuss and evaluate 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. With identical computation costs, we show that these algorithms are approximation algorithms with constant or asymptotic ratios. When costs are exponentially distributed, we perform an analysis of Tree-dyn based on Markov chains. Finally, we assess the relative performance of all four non-clairvoyant algorithms with heterogeneous costs through a set of simulations.

6.19. Non-linear divisible loads: There is no free lunch

Divisible Load Theory (DLT) has received a lot of attention in the past decade. A divisible load is a perfect parallel task, that can be split arbitrarily and executed in parallel on a set of possibly heterogeneous resources. The success of DLT is strongly related to the existence of many optimal resource allocation and scheduling algorithms, what strongly differs from general scheduling theory. Moreover, recently, close relationships have been underlined between DLT, that provides a fruitful theoretical framework for scheduling jobs on heterogeneous platforms, and MapReduce, that provides a simple and efficient programming framework to deploy applications on large scale distributed platforms.

The success of both have suggested to extend their framework to non-linear complexity tasks. In this work [23], we show that both DLT and MapReduce are better suited to workloads with linear complexity. In particular, we prove that divisible load theory cannot directly be applied to quadratic workloads, such as it has been proposed recently. We precisely state the limits for classical DLT studies and we review and propose solutions based on a careful preparation of the dataset and clever data partitioning algorithms. In particular, through simulations, we show the possible impact of this approach on the volume of communications generated by MapReduce, in the context of Matrix Multiplication and Outer Product algorithms.

6.20. Direct solvers for sparse linear systems

This work is closely related to the MUMPS solver (see Section 5.1) and was performed in close collaboration with INPT (Toulouse). First, we have pursued the study of low-rank representations to speed-up sparse direct solvers using the so called BLR (Block Low Rank) format [44]. This work was done in collaboration with LSTC (Livermore Software Technology Corp., USA) and in the context of a contract with EDF which funded the PhD thesis of Clément Weisbecker at INPT. We also worked on shared-memory parallelism [61] in the context of the PhD thesis of Wissam M. Sid-Lakhdar. Concerning low-rank approximations, they were experimented on geophysics applications [38] (Helmholtz equations) in the context of a collaboration with members of the ISTerre and Geoazur laboratories. The impact of both low-rank compression and shared-memory parallelism was also studied on electromagnetism problems [17], in collaboration with University of Padova (Italy) and CEDRAT.

We have started the design and implementation of a distributed-memory low-rank multifrontal solver. When computations are faster (thanks to low-rank compression or multithreading within each node), we observed that communications become critical; we are therefore currently studying the limits of the communication schemes from the MUMPS approach and their possible improvements.

On numerical and industrial aspects, we worked on rank detection and null space basis computations (in collaboration with CERFACS and Total/Hutchinson) as well as on improved parallel pivoting strategies for symmetric indefinite systems, in collaboration with ESI-Group (see Section 7.1).

6.21. Push-relabel based algorithms for the maximum transversal problem

In this work [14], we investigate the push-relabel algorithm for solving the problem of finding a maximum cardinality matching in a bipartite graph in the context of the maximum transversal problem. We describe in detail an optimized yet easy-to-implement version of the algorithm and fine-tune its parameters. We also introduce new performance-enhancing techniques. On a wide range of real-world instances, we compare the push-relabel algorithm with state-of-the-art algorithms based on augmenting paths and pseudoflows. We conclude that a carefully tuned push-relabel algorithm is competitive with all known augmenting path-based algorithms, and superior to the pseudoflow-based ones.

6.22. Constructing elimination trees for sparse unsymmetric matrices

The elimination tree model for sparse unsymmetric matrices and an algorithm for constructing it have been recently proposed [82], [83]. The construction algorithm has a worst-case time complexity of $\Theta(mn)$ for an $n \times n$ unsymmetric matrix having m off-diagonal nonzeros. In this work [15], we propose another algorithm that has a worst-case time complexity of $\mathcal{O}(m \log n)$. We compare the two algorithms experimentally and show that both algorithms are efficient in general. The algorithm of Eisenstat and Liu is faster in many practical cases, yet there are instances in which there is a significant difference between the running time of the two algorithms in favor of the proposed one.

6.23. Semi-matching algorithms for scheduling parallel tasks under resource constraints

In this work [25], we study the problem of minimum makespan scheduling when tasks are restricted to subsets of the processors (resource constraints), and require either one or multiple distinct processors to be executed (parallel tasks). This problem is related to the minimum makespan scheduling problem on unrelated machines, as well as to the concurrent job shop problem, and it amounts to finding a semi-matching in bipartite graphs or hypergraphs. The problem is known to be NP-complete for bipartite graphs with general vertex (task) weights, and solvable in polynomial time for unweighted graphs with unit weights (i.e., unit-weight tasks). We prove that the problem is NP-complete for hypergraphs even in the unweighted case. We design several greedy algorithms of low complexity to solve two versions of the problem, and assess their performance through a set of exhaustive simulations. Even though there is no approximation guarantee for these low-complexity algorithms, they return solutions close to the optimal (or a known lower bound) in average.

6.24. Maximum cardinality bipartite matching algorithms on GPUs

In two studies [30], [31], we propose, develop, and evaluate maximum cardinality matching algorithms from two different families (called push-relabel and augmenting-path based) on GPUs. The problem of finding a maximum cardinality matching in bipartite graphs has applications in computer science, scientific computing, bioinformatics, and other areas. To the best of our knowledge, the proposed algorithms are the first investigation of the push-relabel and augmenting-path based on GPUs/ We compare the proposed algorithms with serial and multicore implementations from the literature on a large set of real-life problems where in majority of the cases one of our GPU-accelerated algorithms is demonstrated to be faster than both the sequential and multicore implementations.

6.25. Analysis of partitioning models and metrics in parallel sparse matrix-vector multiplication

Graph/hypergraph partitioning models and methods have been successfully used to minimize the communication among processors in several parallel computing applications. Parallel sparse matrix-vector multiplication (SpMxV) is one of the representative applications that renders these models and methods indispensable in many scientific computing contexts. In this work [36], [55], we investigate the interplay of the partitioning metrics and execution times of SpMxV implementations in three libraries: Trilinos, PETSc, and an in-house one. We carry out experiments with up to 512 processors and investigate the results with regression analysis. Our experiments show that the partitioning metrics influence the performance greatly in a distributed memory setting. The regression analyses demonstrate which metric is the most influential for the execution time of the libraries.

6.26. On partitioning and reordering problems in a hierarchically parallel hybrid linear solver

PDSLin is a general-purpose algebraic parallel hybrid (direct/iterative) linear solver based on the Schur complement method. The most challenging step of the solver is the computation of a preconditioner based on the global Schur complement. Efficient parallel computation of the preconditioner gives rise to partitioning problems with sophisticated constraints and objectives. In this work [39], we identify two such problems and propose hypergraph partitioning methods to address them. The first problem is to balance the workloads associated with different subdomains to compute the preconditioner. We first formulate an objective function and a set of constraints to model the preconditioner computation time. Then, to address these complex constraints, we propose a recursive hypergraph bisection method. The second problem is to improve the data locality during the parallel solution of a sparse triangular system with multiple sparse right-hand sides. We carefully analyze the objective function and show that it can be well approximated by a standard hypergraph partitioning method. Moreover, an ordering compatible with a post ordering of the subdomain elimination tree is shown to be very effective in preserving locality. To evaluate the two proposed methods in practice, we present experimental results using linear systems arising from some applications of our interest. First, we show that in comparison to a commonly-used nested graph dissection method, the proposed recursive hypergraph partitioning method reduces the preconditioner construction time, especially when the number of subdomains is moderate. This is the desired result since PDSLin is based on a two-level parallelization to keep the number of subdomains small by assigning multiple processors to each subdomain. We also show that our second proposed hypergraph method improves the data locality during the sparse triangular solution and reduces the solution time. Moreover, we show that partitioning time can be greatly reduced while maintaining its quality by removing quasi-dense rows from the solution vectors.

6.27. UMPA: A Multi-objective, multi-level partitioner for communication minimization

In this work [42], we propose a directed hypergraph model and a refinement heuristic to distribute communicating tasks among the processing units in a distributed memory setting. The aim is to achieve load balance and minimize the maximum data sent by a processing unit. We also take two other communication metrics into account with a tie-breaking scheme. With this approach, task distributions causing an excessive use of network or a bottleneck processor which participates to almost all of the communication are avoided. We show on a large number of problem instances that our model improves the maximum data sent by a processor up to 34% for parallel environments with 4, 16, 64, and 256 processing units compared to the state of the art which only minimizes the total communication volume.

6.28. A Partitioning-based divisive clustering technique for maximizing the modularity

In this work [43], we present a new graph clustering algorithm aimed at obtaining clusterings of high modularity. The algorithm pursues a divisive clustering approach and uses established graph partitioning algorithms and techniques to compute recursive bipartitions of the input as well as to refine clusters. Experimental evaluation shows that the modularity scores obtained compare favorably to many previous approaches. In the majority of test cases, the algorithm outperformed the best known alternatives. In particular, among 13 problem instances common in the literature, the proposed algorithm improves the best known modularity in 9 cases.

6.29. Randomized matching heuristics with quality guarantees on shared memory parallel computers

In this work [56], we propose two heuristics for the bipartite matching problem that are amenable to shared-memory parallelization. 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.

6.30. On the minimum edge cover and vertex partition by quasi-cliques problems

A γ-quasi-clique in a simple undirected graph is a set of vertices which induces a subgraph with the edge density of at least γ for $0 < \gamma < 1$. A cover of a graph by γ-quasi-cliques is a set of γ-quasi-cliques where each edge of the graph is contained in at least one quasi-clique. The minimum cover by γ-quasi-cliques problem asks for a γ-quasi-clique cover with the minimum number of quasi-cliques. A partition of a graph by γ-quasi-cliques is a set of γ-quasi-cliques where each vertex of the graph belongs to exactly one quasi-clique. The minimum partition by γ-quasi-cliques problem asks for a vertex partition by γ-quasi-cliques with the minimum number of quasi-cliques. In this work [60], we show that the decision versions of the minimum cover and partition by γ-quasi-cliques problems are NP-complete for any fixed γ satisfying $0 < \gamma < 1$.

7. Bilateral Contracts and Grants with Industry

7.1. Bilateral Contracts with Industry

Related to evolutions of the MUMPS solver (see Section 5.1), and in order to continue funding two engineers while working on the design of a consortium of industrial users, we worked on the following contracts with industry, that were managed by CERFACS and INPT, respectively:

- Total/Hutchinson. In this contract, we worked more specifically on numerical aspects related to rank detection and null-space computations. This feature will be available in a future version of the solver.
- ESI-Group. We worked on modified pivoting strategies for hard symmetric indefinite problems. The proposed solutions could be validated by the industrial partner. This feature will be available in the next release of our package.

8. Partnerships and Cooperations

8.1. National Initiatives

8.1.1. ANR

ANR White Project RESCUE (2010-2014), 4 years. The ANR White Project RESCUE was launched in November 2010, for a duration of 48 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 Projects

8.2.1.1. SCORPIO

Type: COOPERATION

Instrument: Specific Targeted Research Project

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 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 of International Scientists

Ana Gainaru (from UIUC and Argonne National Laboratory) has visited our team for three weeks in October and November 2013. She initiated a collaboration with Guillaume Aupy, Anne Benoit, Franck Cappello and Yves Robert on scheduling I/O activity to avoid congestion and increase performance when executing several scientific applications on large-scale platforms.

8.4.2. Visits to International Teams

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. Scientific Animation

Anne Benoit is an associate editor of the *Journal of Parallel and Distributed Computing (JPDC)*. She was the workshops co-chair of ICPP 2013. She was a member of the program committees of the following conferences and workshops: HiPC 2013, ICPE 2013, CCGrid 2013, IPDPS 2013, CLOSER 2013, HCW 2013, IGCC 2013.

Jean-Yves L'Excellent was a member of the program committees of Renpar'13 and ICPP'2013, where he was also local arrangements co-chair. He co-organized the third MUMPS Users days, EDF, Clamart, May 29-30, 2013.

Loris Marchal is or was a member of the program committees of IPDPS'2013, ICPP'2013, and IPDPS'2014.

Yves Robert is an associate editor of *IJHPCA*, *IJGUC* and *JOCS*. He was Program Chair of ICPP 2013 (Int. Conference on Parallel Processing) and of HiPC 2013 (Int. Conference on High Performance Computing). He is or was a member of the program committees of the following conferences and workshops: EduPar 2013, FTXS 2013, ICCS 2013, IGCC 2013, ISC tutorials 2013 ISCIS 2013 and SC 2013.

Bora Uçar was the chair of the applications track of ICPP 2013. He was a member of the program committee for IPDPS 2013, PCO 2013 (a workshop of IPDPS), and PPAM.

Frédéric Vivien is an associate editor of *Parallel Computing*. Frédéric Vivien was program vice-chair, for the algorithms track, of IPDPS 2013, is program vice-chair, for the algorithms track, of HiPC 2014, and is co-responsible of the stream “Algorithmes distribués, multi-agents et calcul parallèle” for ROADEF 2014.

He is or was a member of the program committee of the following conferences and workshops: SC'14, IPDPS 2014, ComPAS'2014, PDP 2014, SC'13, EduPDHPC, ICPP 2013, EduPar-13, PDP 2013, ROADEF 2013, RenPar'21 - ComPAS'2013.

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 in progress: Guillaume Aupy, Multi-criteria scheduling on volatile platforms, September 1, 2011, Anne Benoit and Yves Robert.

PhD in progress: Dounia Zaidouni, Performance and execution models for exascale applications in failure-prone environments, October 1, 2011, Frédéric Vivien and Yves Robert.

PhD in progress: Wissam M. Sid-Lakhdar, Exploitation of multicore architectures in the resolution of sparse linear systems by multifrontal methods, October 1, 2011, Jean-Yves L'Excellent.

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

9.2.3. Juries

PhD: Anne Benoit was a “rapporteur” and member of the jury for the PhD defense of Przemysław Uznański, Bordeaux, France, October 11, 2013.

PhD: Jean-Yves L'Excellent was a “rapporteur” and member of the jury for the PhD defense of Sethy Montan, University Pierre et Marie Curie, France, October 17, 2013.

PhD: Yves Robert was a “rapporteur” and member of the jury for the PhD defense of Amal Khabou, University Orsay Paris XI, Saclay, France, on February 11, 2013.

PhD: Yves Robert was a member of the jury for the PhD defense of Dimitris Letsios, University of Evry Val d'Essonne, Paris, France, on October 22, 2013.

Habilitation: Yves Robert chaired the jury for the *Habilitation à Diriger des Recherches* of Laurent Lefèvre, ENS Lyon, France, November 22, 2013.

PhD: Frédéric Vivien was a member of the jury for the PhD defense of Javier Celaya, University of Zaragoza, Zaragoza, Spain, September 6, 2013.

PhD: Frédéric Vivien was an "expert" for the PhD defense of Marco Meoni, EPFL, Lausanne, Switzerland, December 11, 2013.

PhD: Bora Uçar was an "evaluator" for the PhD defense of Bastian Onne Fagginger Auer, Department of Mathematics, Utrecht University, the Netherlands, August 26 2013.

10. Bibliography

Publications of the year

Articles in International Peer-Reviewed Journals

[1] G. Aupy, A. Benoit, F. Dufossé, Y. Robert. *Reclaiming the energy of a schedule: models and algorithms*, in "Concurrency and Computation: Practice and Experience", 2013, vol. 25, pp. 1505-1523 [*DOI :* 10.1002/cpe.2889], http://hal.inria.fr/hal-00763388

[2] G. Aupy, Y. Robert, F. Vivien, D. Zaidouni. *Checkpointing algorithms and fault prediction*, in "Journal of Parallel and Distributed Computing", November 2013 [*DOI :* 10.1016/j.jpdc.2013.10.010], http://hal.inria.fr/hal-00908446

[3] M. Baboulin, J. Dongarra, J. Herrmann, S. Tomov. *Accelerating linear system solutions using randomization technique*, in "ACM Transactions on Mathematical Software", February 2013, vol. 39, n^o 2 [*DOI :* 10.1145/2427023.2427025], http://hal.inria.fr/hal-00908496

[4] A. Benoit, V. U. Catalyurek, Y. Robert, E. Saule. *A Survey of Pipelined Workflow Scheduling: Models and Algorithms*, in "ACM Computing Surveys", 2013, vol. 45, n^o 4 [*DOI :* 10.1145/2501654.2501664], http://hal.inria.fr/hal-00926178

[5] A. Benoit, A. Dobrila, J.-M. Nicod, L. Philippe. *Scheduling linear chain streaming applications on heterogeneous systems with failures*, in "Future Generation Computer Systems", 2013, vol. 29, n^o 5, pp. 1140-1151 [*DOI :* 10.1016/j.future.2012.12.015], http://hal.inria.fr/hal-00926146

[6] A. Benoit, F. Dufossé, A. Girault, Y. Robert. *Reliability and performance optimization of pipelined real-time systems*, in "Journal of Parallel and Distributed Computing", 2013, vol. 73, n^o 6, pp. 851-865 [*DOI :* 10.1016/j.jpdc.2013.02.009], http://hal.inria.fr/hal-00926123

[7] A. Benoit, M. Gallet, B. Gaujal, Y. Robert. *Computing the throughput of probabilistic and replicated streaming applications*, in "Algorithmica", March 2013, http://hal.inria.fr/hal-00800083

[8] A. Benoit, R. Melhem, P. Renaud-Goud, Y. Robert. *Assessing the performance of energy-aware mappings*, in "Parallel Processing Letters", 2013, vol. 23, n^o 2 [*DOI :* 10.1142/S0129626413400033], http://hal.inria.fr/hal-00926105

[9] A. Benoit, Y. Robert, A. Rosenberg, F. Vivien. *Static strategies for worksharing with unrecoverable interruption*, in "Theory of Computing Systems", 2013, vol. 53, n^o 3, pp. 386-423 [*DOI :* 10.1007/s00224-012-9426-z], http://hal.inria.fr/hal-00763321

[10] G. Bosilca, A. Bouteiller, É. Brunet, F. Cappello, J. Dongarra, A. Guermouche, T. Hérault, Y. Robert, F. Vivien, D. Zaidouni. *Unified Model for Assessing Checkpointing Protocols at Extreme-Scale*, in "Journal of Concurrency and Computation: Practice and Experience", November 2013 [*DOI :* 10.1002/cpe.3173], http://hal.inria.fr/hal-00908447

[11] 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", October 2013 [*DOI :* 10.1177/1094342013505348], http://hal.inria.fr/hal-00881463

[12] H. Casanova, F. Dufossé, Y. Robert, F. Vivien. *Mapping Applications on Volatile Resources*, in "International Journal of High Performance Computing Applications", 2013, http://hal.inria.fr/hal-00923948

[13] J. Dongarra, M. Faverge, T. Hérault, M. Jacquelin, J. Langou, Y. Robert. *Hierarchical QR factorization algorithms for multi-core clusters*, in "Parallel Computing", 2013, vol. 39, n^{o} 4-5, pp. 212-232 [*DOI :* 10.1016/j.parco.2013.01.003], http://hal.inria.fr/hal-00809770

[14] K. Kaya, J. Langguth, F. Manne, B. Uçar. *Push-relabel based algorithms for the maximum transversal problem*, in "Computers & Operations Research", 2013, vol. 40, n^{o} 5, pp. 1266-1275 [*DOI :* 10.1016/j.cor.2012.12.009], http://hal.inria.fr/hal-00763920

[15] K. Kaya, B. Uçar. *Constructing elimination trees for sparse unsymmetric matrices*, in "SIAM Journal on Matrix Analysis and Applications", April 2013, vol. 34, n^{o} 2, pp. 345-354 [*DOI :* 10.1137/110825443], http://hal.inria.fr/inria-00567970

[16] S. Prasad, A. Gupta, K. Kant, A. Lumsdaine, D. Padua, Y. Robert, A. Rosenberg, A. Sussman, C. Weems. *Literacy for all in parallel and distributed computing: guidelines for an undergraduate core curriculum*, in "CSI Journal of Computing", 2013, To appear, http://hal.inria.fr/hal-00764026

International Conferences with Proceedings

[17] P. Amestoy, O. Boiteau, A. Buttari, G. Joslin, J.-Y. L'Excellent, W. M. Sid-Lakhdar, C. Weisbecker, M. Forzan, C. Pozza, V. Pellissier, R. Perrin. *Shared memory parallelism and low-rank approximation techniques applied to direct solvers in FEM simulation (regular paper)*, in "IEEE International Conference on the Computation of Electromagnetic Fields (COMPUMAG)", Budapest, Hungary, 2013, http://hal.inria.fr/hal-00924660

[18] G. Aupy, A. Benoit, T. Hérault, Y. Robert, J. Dongarra. *Optimal Checkpointing Period: Time vs. Energy*, in "Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems", Denver, United States, November 2013, http://hal.inria.fr/hal-00926199

[19] G. Aupy, A. Benoit, T. Hérault, Y. Robert, F. Vivien, D. Zaidouni. *On the Combination of Silent Error Detection and Checkpointing*, in "PRDC - The 19th IEEE Pacific Rim International Symposium on Dependable Computing - 2013", Vancouver, Canada, IEEE, December 2013, http://hal.inria.fr/hal-00847620

[20] G. Aupy, A. Benoit, R. Melhem, P. Renaud-Goud, Y. Robert. *Energy-aware checkpointing of divisible tasks with soft or hard deadlines*, in "IGCC - 4th International Green Computing Conference - 2013", Arlington, United States, February 2013, http://hal.inria.fr/hal-00857244

[21] G. AUPY, M. FAVERGE, Y. ROBERT, J. KURZAK, P. LUSZCZEK, J. DONGARRA. *Implementing a systolic algorithm for QR factorization on multicore clusters with PaRSEC*, in "PROPER 2013 - 6th Workshop on Productivity and Performance", Aachen, Germany, August 2013, http://hal.inria.fr/hal-00844492

[22] G. AUPY, Y. ROBERT, F. VIVIEN, D. ZAIDOUNI. *Checkpointing strategies with prediction windows*, in "PRDC - The 19th IEEE Pacific Rim International Symposium on Dependable Computing - 2013", Vancouver, Canada, IEEE, December 2013, http://hal.inria.fr/hal-00847622

[23] O. BEAUMONT, H. LARCHEVÊQUE, L. MARCHAL. *Non Linear Divisible Loads: There is No Free Lunch*, in "IPDPS 2013, 27th IEEE International Parallel & Distributed Processing Symposium", Boston, United States, IEEE, 2013, http://hal.inria.fr/hal-00771640

[24] A. BENOIT, L.-C. CANON, L. MARCHAL. *Non-clairvoyant reduction algorithms for heterogeneous platforms*, in "HeteroPar'2013, in conjunction with Euro-Par 2013", Aachen, Germany, 2013, http://hal.inria.fr/hal-00926093

[25] A. BENOIT, J. LANGGUTH, B. UÇAR. *Semi-matching algorithms for scheduling parallel tasks under resource constraints*, in "IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum", Cambridge, MA, United States, IEEE Computer Society, 2013, pp. 1744-1753 [*DOI :* 10.1109/IPDPSW.2013.30], http://hal.inria.fr/hal-00738393

[26] A. BOUTEILLER, F. CAPPELLO, J. DONGARRA, A. GUERMOUCHE, T. HÉRAULT, Y. ROBERT. *Multi-criteria checkpointing strategies: response-time versus resource utilization*, in "Euro-Par 2013", Aachen, Germany, S. VERLAG (editor), LNCS, 2013, vol. 8097, pp. 420-431 [*DOI :* 10.1007/978-3-642-40047-6_43], http://hal.inria.fr/hal-00926606

[27] H. CASANOVA, F. DUFOSSÉ, Y. ROBERT, F. VIVIEN. *Mapping tightly-coupled applications on volatile resources*, in "PDP'2013, the 21st Euromicro Int. Conf. on Parallel, Distributed, and Network-Based Processing", Belfast, United Kingdom, IEEE Computer Society Press, 2013, http://hal.inria.fr/hal-00763376

[28] H. CASANOVA, F. DUFOSSÉ, Y. ROBERT, F. VIVIEN. *Scheduling Tightly-Coupled Applications on Heterogeneous Desktop Grids*, in "HCW 2013 - 22nd International Heterogeneity in Computing Workshop", Boston, United States, May 2013, http://hal.inria.fr/hal-00788606

[29] 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, http://hal.inria.fr/hal-00923953

[30] M. DEVECI, K. KAYA, B. UÇAR, V. U. CATALYUREK. *A Push-Relabel-Based Maximum Cardinality Bipartite Matching Algorithm on GPUs*, in "42nd International Conference on Parallel Processing", Lyon, France, IEEE Computer Society, 2013, pp. 21 - 29 [*DOI :* 10.1109/ICPP.2013.11], http://hal.inria.fr/hal-00923464

[31] M. DEVECI, K. KAYA, B. UÇAR, V. U. CATALYUREK. *GPU accelerated maximum cardinality matching algorithms for bipartite graphs*, in "Euro-Par 2013", Aachen, Germany, F. WOLF, B. MOHR, D. AN MEY (editors), Springer, August 2013, pp. 850-861 [*DOI :* 10.1007/978-3-642-40047-6_84], http://hal.inria.fr/hal-00923449

[32] S. DI, Y. ROBERT, F. VIVIEN, D. KONDO, C.-L. WANG, F. CAPPELLO. *Optimization of Cloud Task Processing with Checkpoint-Restart Mechanism*, in "SC13 - Supercomputing - 2013", Denver, United States, ACM, November 2013 [*DOI :* 10.1145/2503210.2503217], http://hal.inria.fr/hal-00847635

[33] J. DONGARRA, T. HÉRAULT, Y. ROBERT. *Revisiting the double checkpointing algorithm*, in "APDCM 2013", Boston, United States, IEEE, 2013, http://hal.inria.fr/hal-00925168

[34] 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, December 2013, http://hal.inria.fr/hal-00930238

[35] J. HERRMANN, L. MARCHAL, Y. ROBERT. *Model and complexity results for tree traversals on hybrid platforms*, in "HeteroPar - International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Platforms", Aachen, Germany, August 2013, http://hal.inria.fr/hal-00926502

[36] K. KAYA, B. UÇAR, V. U. CATALYUREK. *Analysis of Partitioning Models and Metrics in Parallel Sparse Matrix-Vector Multiplication*, in "10th PPAM - Parallel Processing and Applied Mathematics", Varsovie, Poland, Springer, 2014, to appear, http://hal.inria.fr/hal-00923454

[37] L. MARCHAL, O. SINNEN, F. VIVIEN. *Scheduling tree-shaped task graphs to minimize memory and makespan*, in "IPDPS 2013 - 27th IEEE International Parallel & Distributed Processing Symposium", Boston, United States, May 2013, http://hal.inria.fr/hal-00788612

[38] C. WEISBECKER, P. R. AMESTOY, O. BOITEAU, R. BROSSIER, A. BUTTARI, J.-Y. L'EXCELLENT, S. OPERTO, J. VIRIEUX. *3D frequency-domain seismic modeling with a Block Low-Rank algebraic multifrontal direct solver*, in "SEG Technical Program Expanded Abstracts, SEG annual meeting", Houston, Texas, United States, 2013 [*DOI :* 10.1190/SEGAM2013-0603.1], http://hal.inria.fr/hal-00924638

[39] I. YAMAZAKI, X. S. LI, F.-H. ROUET, B. UÇAR. *On Partitioning and Reordering Problems in a Hierarchically Parallel Hybrid Linear Solver*, in "2013 IEEE 27th International Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW)", Cambridge, MA, United States, IEEE Computer Society, May 2013, http://hal.inria.fr/hal-00923447

Scientific Books (or Scientific Book chapters)

[40] A. BENOIT, Y. ROBERT, F. VIVIEN. , *A Guide to Algorithm Design: Paradigms, Methods, and Complexity Analysis*, Applied Algorithms and Data Structures series, Chapman & Hall/CRC, August 2013, 380 p. , http://hal.inria.fr/hal-00908448

[41] A. BENOIT, L. MARCHAL, Y. ROBERT, B. UÇAR, F. VIVIEN. *Scheduling for Large-Scale Systems*, in "The Computing Handbook Set, vol. 1", T. GONZALEZ, J. L. DÍAZ HERRERA (editors), Chapman and Hall/CRC Press, 2013, To appear, http://hal.inria.fr/hal-00763372

[42] V. U. CATALYUREK, M. DEVECI, K. KAYA, B. UÇAR. *UMPA: A Multi-objective, multi-level partitioner for communication minimization*, in "Graph Partitioning and Graph Clustering 2012", D. A. BADER, H. MEYERHENKE, P. SANDERS, D. WAGNER (editors), Contemporary Mathematics, AMS, 2013, vol. 588, pp. 53-66 [*DOI :* 10.1090/CONM/588/11704], http://hal.inria.fr/hal-00763563

[43] V. U. CATALYUREK, K. KAYA, J. LANGGUTH, B. UÇAR. *A Partitioning-based divisive clustering technique for maximizing the modularity*, in "Graph Partitioning and Graph Clustering 2012", D. A. BADER, H. MEYERHENKE, P. SANDERS, D. WAGNER (editors), Contemporary Mathematics, AMS, 2013, vol. 588, pp. 171-186 [*DOI :* 10.1090/CONM/588/11712], http://hal.inria.fr/hal-00763559

Research Reports

[44] P. R. AMESTOY, C. ASHCRAFT, O. BOITEAU, A. BUTTARI, J.-Y. L'EXCELLENT, C. WEISBECKER. , *Improving multifrontal methods by means of block low-rank representations*, Inria, January 2013, n^o RR-8199, Submitted for publication to SIAM, http://hal.inria.fr/hal-00776859

[45] G. AUPY, A. BENOIT, T. HÉRAULT, Y. ROBERT, J. DONGARRA. , *Optimal Checkpointing Period: Time vs. Energy*, Inria, October 2013, n^o RR-8387, 19 p. , http://hal.inria.fr/hal-00878938

[46] G. AUPY, A. BENOIT, T. HÉRAULT, Y. ROBERT, F. VIVIEN, D. ZAIDOUNI. , *On the Combination of Silent Error Detection and Checkpointing*, Inria, June 2013, n^o RR-8319, http://hal.inria.fr/hal-00836871

[47] G. AUPY, A. BENOIT, R. MELHEM, P. RENAUD-GOUD, Y. ROBERT. , *Energy-aware checkpointing of divisible tasks with soft or hard deadlines*, Inria, February 2013, n^o RR-8238, 33 p. , http://hal.inria.fr/hal-00788641

[48] G. AUPY, M. FAVERGE, Y. ROBERT, J. KURZAK, P. LUSZCZEK, J. DONGARRA. , *Implementing a Systolic Algorithm for QR Factorization on Multicore Clusters with PaRSEC*, Inria, November 2013, n^o RR-8390, 16 p. , Published in ProPer'13, http://hal.inria.fr/hal-00879248

[49] G. AUPY, Y. ROBERT, F. VIVIEN, D. ZAIDOUNI. , *Checkpointing algorithms and fault prediction*, Inria, February 2013, n^o RR-8237, Accepted to be published in JPDC, http://hal.inria.fr/hal-00788313

[50] G. AUPY, Y. ROBERT, F. VIVIEN, D. ZAIDOUNI. , *Checkpointing strategies with prediction windows*, Inria, February 2013, n^o RR-8239, 44 p. , http://hal.inria.fr/hal-00789109

[51] G. AUPY, Y. ROBERT, F. VIVIEN, D. ZAIDOUNI. , *Comments on "Improving the computing efficiency of HPC systems using a combination of proactive and preventive checkpoint"*, Inria, June 2013, n^o RR-8318, http://hal.inria.fr/hal-00836629

[52] G. AUPY, M. SHANTHARAM, A. BENOIT, Y. ROBERT, P. RAGHAVAN. , *Co-Scheduling Algorithms for High-Throughput Workload Execution*, Inria, April 2013, n^o RR-8293, 21 p. , http://hal.inria.fr/hal-00819036

[53] A. BENOIT, L.-C. CANON, L. MARCHAL. , *Non-clairvoyant reduction algorithms for heterogeneous platforms*, Inria, June 2013, n^o RR-8315, http://hal.inria.fr/hal-00832102

[54] H. CASANOVA, L. LIM, Y. ROBERT, F. VIVIEN, D. ZAIDOUNI. , *Cost-Optimal Execution of Trees of Boolean Operators with Shared Streams*, Inria, October 2013, n^o RR-8373, 39 p. , http://hal.inria.fr/hal-00869340

[55] V. U. CATALYUREK, K. KAYA, B. UÇAR. , *On analysis of partitioning models and metrics in parallel sparse matrix-vector multiplication*, Inria, May 2013, n^o RR-8301, 25 p. , http://hal.inria.fr/hal-00821523

[56] F. DUFOSSÉ, K. KAYA, B. UÇAR. , *Randomized matching heuristics with quality guarantees on shared memory parallel computers*, Inria, October 2013, n° RR-8386 ; Rapport LAAS n°13578, 28 p. , http://hal.inria.fr/hal-00877211

[57] J. HERRMANN, L. MARCHAL, Y. ROBERT. , *Tree traversals with task-memory affinities*, Inria, February 2013, n° RR-8226, 31 p. , http://hal.inria.fr/hal-00787753

[58] J. HERRMANN, L. MARCHAL, Y. ROBERT. , *Memory-aware list scheduling for hybrid platforms*, Inria, February 2014, n° RR-8461, 30 p. , http://hal.inria.fr/hal-00944336

[59] O. KAYA, E. KAYAASLAN, B. UÇAR, I. S. DUFF. , *Fill-in reduction in sparse matrix factorizations using hypergraphs*, Inria, January 2014, n° RR-8448, http://hal.inria.fr/hal-00932882

[60] O. KAYA, E. KAYAASLAN, B. UÇAR. , *On the minimum edge cover and vertex partition by quasi-cliques problems*, Inria, February 2013, n° RR-8255, http://hal.inria.fr/hal-00795429

[61] J.-Y. L'EXCELLENT, M. W. SID-LAKHDAR. , *Introduction of shared-memory parallelism in a distributed-memory multifrontal solver*, Inria, February 2013, n° RR-8227, 35 p. , http://hal.inria.fr/hal-00786055

References in notes

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

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

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

[65] , *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

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

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

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

[69] 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

[70] 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

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

[72] 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

[73] 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

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

[75] 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

[76] 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

[77] 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

[78] 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

[79] 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

[80] 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

[81] 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

[82] S. C. Eisenstat, J. W. H. Liu. *The theory of elimination trees for sparse unsymmetric matrices*, in "SIAM Journal on Matrix Analysis and Applications", 2005, vol. 26, n^{o} 3, pp. 686–705

[83] S. C. Eisenstat, J. W. H. Liu. *Algorithmic aspects of elimination trees for sparse unsymmetric matrices*, in "SIAM Journal on Matrix Analysis and Applications", 2008, vol. 29, n^{o} 4, pp. 1363–1381

[84] 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

[85] 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

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

[87] 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

[88] 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

[89] 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

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

[91] 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

[92] 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