RESEARCH CENTRE

Grenoble - Rhône-Alpes

IN PARTNERSHIP WITH:

CNRS, Université Claude Bernard (Lyon 1), Ecole normale supérieure de Lyon

2021
ACTIVITY REPORT

Project-Team
ROMA

Optimisation des ressources : modèles, algorithmes et ordonnancement

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

DOMAIN

Networks, Systems and Services, Distributed Computing

THEME

Distributed and High Performance Computing

Contents

Project-Team ROMA **1**

1 Team members, visitors, external collaborators **2**

2 Overall objectives **3**

3 Research program **3**
- 3.1 Resilience for very large scale platforms . . . 3
- 3.2 Multi-criteria scheduling strategies . . . 4
- 3.3 Sparse direct solvers and sparsity in computing . . . 4

4 Application domains **5**

5 Highlights of the year **5**
- 5.1 Awards . . . 5

6 New software and platforms **6**
- 6.1 New software . . . 6
  - 6.1.1 MatchMaker . . . 6
  - 6.1.2 PaStiX . . . 6

7 New results **7**
- 7.1 Resilience for very large scale platforms . . . 7
  - 7.1.1 Resilient scheduling of moldable jobs to cope with silent errors. . . . 7
  - 7.1.2 Optimal Checkpointing Strategies for Iterative Applications. . . . 7
- 7.2 Multi-criteria scheduling strategies . . . 8
  - 7.2.1 Max-stretch minimization on an edge-cloud platform. . . . 8
  - 7.2.2 Update on the Asymptotic Optimality of LPT. . . . 8
  - 7.2.3 Shelf schedules for independent moldable tasks to minimize the energy consumption. 8
  - 7.2.4 Locality-Aware Scheduling of Independent Tasks for Runtime Systems. . . . 9
  - 7.2.5 Taming tail latency in key-value stores: a scheduling perspective. . . . 9
  - 7.2.6 Evaluating Task Dropping Strategies for Overloaded Real-Time Systems. . . . 10
- 7.3 Sparse direct solvers and sparsity in computing . . . 10
  - 7.3.1 Trading Performance for Memory in Sparse Direct Solvers using Low-rank Compression. . . . 10
  - 7.3.2 Deciding Non-Compressible Blocks in Sparse Direct Solvers using Incomplete Factorization. . . . 11
  - 7.3.3 Algorithms and data structures for hyperedge queries. . . . 11
  - 7.3.4 Shared-memory implementation of the Karp-Sipser kernelization process. . . . 11
  - 7.3.5 Streaming hypergraph partitioning algorithms on limited memory environments. . . . 12
  - 7.3.6 Fully-dynamic weighted matching approximation in practice. . . . 12
  - 7.3.7 Strongly connected components in directed hypergraphs. . . . 12

8 Partnerships and cooperations **13**
- 8.1 International initiatives . . . 13
  - 8.1.1 Associate Teams in the framework of an Inria International Lab or in the framework of an Inria International Program . . . 13
  - 8.1.2 Inria associate team not involved in an IIL or an international program . . . 13
  - 8.1.3 Participation in other International Programs . . . 14
- 8.2 International research visitors . . . 14
  - 8.2.1 Visits to international teams . . . 14
- 8.3 National initiatives . . . 14
  - 8.3.1 ANR Project SOLHARIS (2019-2023), 4 years. . . . 14

9 Dissemination **14**
9.1 Promoting scientific activities . . . 14
9.1.1 Scientific events: organisation . . . 14
9.1.2 Scientific events: selection . . . 15
9.1.3 Journal . . . 15
9.1.4 Invited talks . . . 16
9.1.5 Leadership within the scientific community . . . 16
9.1.6 Scientific expertise . . . 16
9.1.7 Research administration . . . 16
9.2 Teaching - Supervision - Juries . . . 17
9.2.1 Teaching . . . 17
9.2.2 Supervision . . . 17
9.2.3 Juries . . . 18
9.3 Popularization . . . 18
9.3.1 Articles and contents . . . 18

10 Scientific production **18**
10.1 Major publications . . . 18
10.2 Publications of the year . . . 18
10.3 Cited publications . . . 21

Project-Team ROMA

Creation of the Project-Team: 2015 January 01

Keywords

Computer sciences and digital sciences

A1.1.1. – Multicore, Manycore

A1.1.2. – Hardware accelerators (GPGPU, FPGA, etc.)

A1.1.3. – Memory models

A1.1.4. – High performance computing

A1.1.5. – Exascale

A1.1.9. – Fault tolerant systems

A1.6. – Green Computing

A6.1. – Methods in mathematical modeling

A6.2.3. – Probabilistic methods

A6.2.5. – Numerical Linear Algebra

A6.2.6. – Optimization

A6.2.7. – High performance computing

A6.3. – Computation-data interaction

A7.1. – Algorithms

A8.1. – Discrete mathematics, combinatorics

A8.2. – Optimization

A8.7. – Graph theory

A8.9. – Performance evaluation

Other research topics and application domains

B3.2. – Climate and meteorology

B3.3. – Geosciences

B4. – Energy

B4.5.1. – Green computing

B5.2.3. – Aviation

B5.5. – Materials

1 Team members, visitors, external collaborators

Research Scientists

- Loris Marchal [Team leader, CNRS, Researcher, HDR]
- Bora Uçar [CNRS, Researcher, HDR]
- Frédéric Vivien [Inria, Senior Researcher, HDR]

Faculty Members

- Anne Benoit [École Normale Supérieure de Lyon, Associate Professor, HDR]
- Grégoire Pichon [Univ Claude Bernard, Associate Professor]
- Yves Robert [École Normale Supérieure de Lyon, Professor, HDR]

Post-Doctoral Fellow

- Somesh Singh [CNRS, from Sep 2021, Inria after Nov. 2021]

PhD Students

- Yishu Du [Université Tongji - Chine]
- Anthony Dugois [Inria]
- Redouane Elghazi [Univ de Franche-Comté]
- Yiqin Gao [Univ de Lyon, until Sep 2021]
- Maxime Gonthier [Inria]
- Lucas Perotin [École Normale Supérieure de Lyon]
- Zhiwei Wu [East China Normal University de Shanghai]

Interns and Apprentices

- Elodie Bernard [Inria, from May 2021 until Jul 2021]
- Jules Bertrand [École Normale Supérieure de Lyon, from Feb 2021 until Jul 2021]

Administrative Assistant

- Evelyne Blesle [Inria]

External Collaborators

- Theo Mary [CNRS]
- Hongyang Sun [University of Kansas (USA), from Oct 2021]

2 Overall objectives

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

Scientists now have access to tremendous computing power. For instance, the top supercomputers contain more than 100,000 cores, and volunteer computing grids gather millions of processors. 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.

3 Research program

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

3.1 Resilience for very large scale platforms

For HPC applications, scale is a major opportunity. The largest supercomputers contain tens of thousands of nodes and future platforms will certainly have to enroll even more computing resources to enter the

Exascale era. Unfortunately, scale is also a major threat. Indeed, even if each node provides an individual MTBF (Mean Time Between Failures) of, say, one century, a machine with 100,000 nodes will encounter a failure every 9 hours in average, which is shorter than the execution time of many HPC applications.

To further darken the picture, several types of errors need to be considered when computing at scale. In addition to classical fail-stop errors (such as hardware failures), silent errors (a.k.a silent data corruptions) must be taken into account. The cause for silent errors may be for instance soft errors in L1 cache, or bit flips due to cosmic radiations. The problem is that the detection of a silent error is not immediate, and that they only manifest later, once the corrupted data has propagated and impacted the result.

Our work investigates new models and algorithms for resilience at extreme-scale. Its main objective is to cope with both fail-stop and silent errors, and to design new approaches that dramatically improve the efficiency of state-of-the-art methods. Application resilience currently involves a broad range of techniques, including fault prediction, error detection, error containment, error correction, checkpointing, replication, migration, recovery, etc. Extending these techniques, and developing new ones, to achieve efficient execution at extreme-scale is a difficult challenge, but it is the key to a successful deployment and usage of future computing platforms.

3.2 Multi-criteria 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. Our work mainly considers the following two platform characteristics:

Energy consumption. Power management in HPC is necessary due to both monetary and environmental constraints. Using dynamic voltage and frequency scaling (DVFS) is a widely used technique to decrease energy consumption, but it can severely degrade performance and increase execution time. Part of our work in this direction studies the trade-off between energy consumption and performance (throughput or execution time). Furthermore, our work also focuses on the optimization of the power consumption of fault-tolerant mechanisms. The problem of the energy consumption of these mechanisms is especially important because resilience generally requires redundant computations and/or redundant communications, either in time (re-execution) or in space (replication), and because redundancy consumes extra energy.

Memory usage and data movement. In many scientific computations, memory is a bottleneck and should be carefully considered. Besides, data movements, between main memory and secondary storages (I/Os) or between different computing nodes (communications), are taking an increasing part of the cost of computing, both in term of performance and energy consumption. In this context, our work focuses on scheduling scientific applications described as task graphs both on memory constrained platforms, and on distributed platforms with the objective of minimizing communications. The task-based representation of a computing application is very common in the scheduling literature but meets an increasing interest in the HPC field thanks to the use of runtime schedulers. Our work on memory-aware scheduling is naturally multi-criteria, as it is concerned with both memory consumption, performance and data-movements.

3.3 Sparse direct solvers and sparsity in computing

In this theme, we work on various aspects of sparse direct solvers for linear systems. Target applications lead to sparse systems made of millions of unknowns. In the scope of the PaStiX solver, co-developed with the Inria HiePACS team, there are two main objectives: reducing as much as possible memory requirements and exploiting modern parallel architectures through the use of runtime systems.

A first research challenge is to exploit the parallelism of modern computers, made of heterogeneous (CPUs+GPUs) nodes. The approach consists of using dynamic runtime systems (in the context of the PaStiX solver, Parsec or StarPU) to schedule tasks.

Another important direction of research is the exploitation of low-rank representations. Low-rank approximations are commonly used to compress the representation of data structures. The loss of information induced is often negligible and can be controlled. In the context of sparse direct solvers, we exploit the notion of low-rank properties in order to reduce the demand in terms of floating-point operations and memory usage. To enhance sparse direct solvers using low-rank compression, two orthogonal approaches are followed: (i) integrate new strategies for a better scalability and (ii) use preprocessing steps to better identify how to cluster unknowns, when to perform compression and which blocks not to compress.

CSC is a term (coined 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. Most of the time, the research output includes experiments with real life data to validate the developed combinatorial algorithms and fine tune them.

In this context, our work targets (i) the preprocessing phases of direct methods, iterative methods, and hybrid methods for solving linear systems of equations; (ii) high performance tensor computations. The core topics covering our contributions include partitioning and clustering in graphs and hypergraphs, matching in graphs, data structures and algorithms for sparse matrices and tensors (different from partitioning), and task mapping and scheduling.

4 Application domains

Sparse linear system solvers have a wide range of applications as they are used at the heart of many numerical methods in computational science: 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 system of linear equations involving sparse matrices. There are therefore a number of application fields: structural mechanics, seismic modeling, 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.

Tensors, or multidimensional arrays, are becoming very important because of their use in many data analysis applications. The additional dimensions over matrices (or two dimensional arrays) enable gleaning information that is otherwise unreachable. Tensors, like matrices, come in two flavors: dense tensors and sparse tensors. Dense tensors arise usually in physical and simulation applications: signal processing for electroencephalography (also named EEG, electrophysiological monitoring method to record electrical activity of the brain); hyperspectral image analysis; compression of large grid-structured data coming from a high-fidelity computational simulation; quantum chemistry etc. Dense tensors also arise in a variety of statistical and data science applications. Some of the cited applications have structured sparsity in the tensors. We see sparse tensors, with no apparent/special structure, in data analysis and network science applications. Well known applications dealing with sparse tensors are: recommender systems; computer network traffic analysis for intrusion and anomaly detection; clustering in graphs and hypergraphs modeling various relations; knowledge graphs/bases such as those in learning natural languages.

5 Highlights of the year

5.1 Awards

Bora Uçar received IEEE TPDS Award for Editorial Excellence 2021.

6 New software and platforms

6.1 New software

6.1.1 MatchMaker

Name: Maximum matchings in bipartite graphs

Keywords: Graph algorithmics, Matching

Scientific Description: The implementations of ten exact algorithms and four heuristics for solving the problem of finding a maximum cardinality matching in bipartite graphs are provided.

Functional Description: This software provides algorithms to solve the maximum cardinality matching problem in bipartite graphs.

URL: https://gitlab.inria.fr/bora-ucar/matchmaker

Publications: hal-00786548, hal-00763920

Contact: Bora Uçar

Participants: Kamer Kaya, Johannes Langguth

6.1.2 PaStiX

Name: Parallel Sparse matriX package

Keywords: Linear algebra, High-performance calculation, Sparse Matrices, Linear Systems Solver, Low-Rank compression

Scientific Description: PaStiX is based on an efficient static scheduling and memory manager, in order to solve 3D problems with more than 50 million of unknowns. The mapping and scheduling algorithm handles a combination of 1D and 2D block distributions. A dynamic scheduling can also be applied to take care of NUMA architectures while taking into account very precisely the computational costs of the BLAS 3 primitives, the communication costs and the cost of local aggregations.

Functional Description: PaStiX is a scientific library that provides a high performance parallel solver for very large sparse linear systems based on block direct and block ILU(k) methods. It can handle low-rank compression techniques to reduce the computation and the memory complexity. Numerical algorithms are implemented in single or double precision (real or complex) for LLt, LDLt and LU factorization with static pivoting (for non symmetric matrices having a symmetric pattern). The PaStiX library uses the graph partitioning and sparse matrix block ordering packages Scotch or Metis.

The PaStiX solver is suitable for any heterogeneous parallel/distributed architecture when its performance is predictable, such as clusters of multicore nodes with GPU accelerators or KNL processors. In particular, we provide a high-performance version with a low memory overhead for multicore node architectures, which fully exploits the advantage of shared memory by using a hybrid MPI-thread implementation.

The solver also provides some low-rank compression methods to reduce the memory footprint and/or the time-to-solution.

URL: https://gitlab.inria.fr/solverstack/pastix

Contact: Pierre Ramet

Participants: Tony Delarue, Grégoire Pichon, Mathieu Faverge, Esragul Korkmaz, Pierre Ramet

Partners: INP Bordeaux, Université de Bordeaux

7 New results

7.1 Resilience for very large scale platforms

The ROMA team has been working on resilience problems for several years. In 2021, we have focused on several problems.

7.1.1 Resilient scheduling of moldable jobs to cope with silent errors.

Participants: Anne Benoit, Valentin Le Fèvre, Lucas Perotin, Yves Robert, Padma Raghavan *(Vanderbilt University)*, Hongyang Sun *(Vanderbilt University)*.

We have focused on the resilient scheduling of moldable parallel jobs on high-performance computing (HPC) platforms. Moldable jobs allow for choosing a processor allocation before execution, and their execution time obeys various speedup models. The objective is to minimize the overall completion time of the jobs, or the makespan, when jobs can fail due to silent errors and hence may need to be re-executed after each failure until successful completion. Our work generalizes the classical scheduling framework for failure-free jobs. To cope with silent errors, we introduce two resilient scheduling algorithms, LPA-List and Batch-List, both of which use the List strategy to schedule the jobs. Without knowing a priori how many times each job will fail, LPA-List relies on a local strategy to allocate processors to the jobs, while Batch-List schedules the jobs in batches and allows only a restricted number of failures per job in each batch. We prove new approximation ratios for the two algorithms under several prominent speedup models (e.g., roofline, communication, Amdahl, power, monotonic, and a mixed model). An extensive set of simulations is conducted to evaluate different variants of the two algorithms, and the results show that they consistently outperform some baseline heuristics. Overall, our best algorithm is within a factor of 1.6 of a lower bound on average over the entire set of experiments, and within a factor of 4.2 in the worst case.

Preliminary results with a subset of speedup models were published in Cluster 2020 [37], and this year, an extended version appeared in IEEE Transactions on Computers [8].

7.1.2 Optimal Checkpointing Strategies for Iterative Applications.

Participants: Yishu Du, Loris Marchal, Yves Robert, Guillaume Pallez *(Inria Bordeaux)*.

We have studied how to provide an optimal checkpointing strategy to protect iterative applications from fail-stop errors. We consider a general framework, where the application repeats the same execution pattern by executing consecutive iterations, and where each iteration is composed of several tasks. These tasks have different execution lengths and different checkpoint costs. Assume that there are n tasks and that task a_i, where $0 \leq i < n$, has execution time t_i and checkpoint cost c_i. A naive strategy would checkpoint after each task. Another naive strategy would checkpoint at the end of each iteration. A strategy inspired by the Young/Daly formula would work for $\sqrt{2\mu c_{ave}}$ seconds, where μ is the application MTBF and c_{ave} the average checkpoint time, and checkpoint at the end of the current task (and repeat). Another strategy, also inspired by the Young/Daly formula, would select the task $a_{\min}$ with smallest checkpoint cost $c_{\min}$ and would checkpoint after every p^{th} instance of that task, leading to a checkpointing period pT, where $T = \sum_{i=0}^{n-1} a_i$ is the time per iteration. One would choose the period so that $pT \approx \sqrt{2\mu c_{\min}}$ to obey the Young/Daly formula. All these naive and Young/Daly strategies are suboptimal. Our main contribution in this work has been to show that the optimal checkpoint strategy is globally periodic, and to design a dynamic programming algorithm that computes the optimal checkpointing pattern. This pattern may well checkpoint many different tasks, and this across many different iterations. We have shown through simulations, both from synthetic and real-life application scenarios, that the optimal strategy outperforms the naive and Young/Daly strategies.

This work has been published in IEEE Transaction of Parallel and Distributed Computers [11].

7.2 Multi-criteria scheduling strategies

We report here the work undertaken by the ROMA team in multi-criteria strategies, which focuses on taking into account energy and memory constraints, but also budget constraints or specific constraints for scheduling online requests.

7.2.1 Max-stretch minimization on an edge-cloud platform.

Participants: Anne Benoit, Redouane Elghazi, Yves Robert.

We have considered the problem of scheduling independent jobs that are generated by processing units at the edge of the network. These jobs can either be executed locally, or sent to a centralized cloud platform that can execute them at greater speed. Such edge-generated jobs may come from various applications, such as e-health, disaster recovery, autonomous vehicles or flying drones. The problem is to decide where and when to schedule each job, with the objective to minimize the maximum stretch incurred by any job. The stretch of a job is the ratio of the time spent by that job in the system, divided by the minimum time it could have taken if the job was alone in the system. We formalize the problem and explain the differences with other models that can be found in the literature. We prove that minimizing the max-stretch is NP-complete, even in the simpler instance with no release dates (all jobs are known in advance). This result comes from the proof that minimizing the max-stretch with homogeneous processors and without release dates is NP-complete, a complexity problem that was left open before this work. We design several algorithms to propose efficient solutions to the general problem, and we conduct simulations based on real platform parameters to evaluate the performance of these algorithms.

This work appeared in the proceedings of IPDPS 2021 [17].

7.2.2 Update on the Asymptotic Optimality of LPT.

Participants: Anne Benoit, Redouane Elghazi, Louis-Claude Canon *(Univ. Besançon)*, Pierre-Cyrille Héam *(Univ. Besançon)*.

When independent tasks are to be scheduled onto identical processors, the typical goal is to minimize the makespan. A simple and efficient heuristic consists in scheduling first the task with the longest processing time (LPT heuristic), and to plan its execution as soon as possible. While the performance of LPT has already been largely studied, in particular its asymptotic performance, we revisit results and propose a novel analysis for the case of tasks generated through uniform integer compositions. Also, we perform extensive simulations to empirically assess the asymptotic performance of LPT. Results demonstrate that the absolute error rapidly tends to zero for several distributions of task costs, including ones studied by theoretical models, and realistic distributions coming from benchmarks.

This work appeared in the proceedings of EuroPar 2021 [16].

7.2.3 Shelf schedules for independent moldable tasks to minimize the energy consumption.

Participants: Anne Benoit, Redouane Elghazi, Louis-Claude Canon *(Univ. Besançon)*, Pierre-Cyrille Héam *(Univ. Besançon)*.

Scheduling independent tasks on a parallel platform is a widely-studied problem, in particular when the goal is to minimize the total execution time, or makespan ($P||C_{max}$ problem in Graham's notations). Also, many applications do not consist of sequential tasks, but rather of parallel moldable tasks that can decide their degree of parallelism at execution (i.e., on how many processors they are executed). Furthermore, since the energy consumption of data centers is a growing concern, both from an environmental and economical point of view, minimizing the energy consumption of a schedule is a main challenge to be addressed. One should decide, for each task, on how many processors it is executed, and at which speed the processors are operated, with the goal to minimize the total energy consumption. We further focus on co-schedules, where tasks are partitioned into shelves, and we prove that the problem of minimizing the energy consumption remains NP-complete when static energy is consumed during the whole duration of the application. We are however able to provide an optimal algorithm for the schedule within one shelf, i.e., for a set of tasks that start at the same time. Several approximation results are derived, and simulations are performed to show the performance of the proposed algorithms.

This work appeared in the proceedings of SBAC-PAD 2021 [15].

7.2.4 Locality-Aware Scheduling of Independent Tasks for Runtime Systems.

Participants: Maxime Gonthier, Loris Marchal, Samuel Thibault *(Inria Bordeaux)*.

A now-classical way of meeting the increasing demand for computing speed by HPC applications is the use of GPUs and/or other accelerators. Such accelerators have their own memory, which is usually quite limited, and are connected to the main memory through a bus with bounded bandwidth. Thus, particular care should be devoted to data locality in order to avoid unnecessary data movements. Task-based runtime schedulers have emerged as a convenient and efficient way to use such heterogeneous platforms. When processing an application, the scheduler has the knowledge of all tasks available for processing on a GPU, as well as their input data dependencies. Hence, it is able to order tasks and prefetch their input data in the GPU memory (after possibly evicting some previously-loaded data), while aiming at minimizing data movements, so as to reduce the total processing time. In this work, we focus on how to schedule tasks that share some of their input data (but are otherwise independent) on a GPU. We provide a formal model of the problem, exhibit an optimal eviction strategy, and show that ordering tasks to minimize data movement is NP-complete. We review and adapt existing ordering strategies to this problem, and propose a new one based on task aggregation. These strategies have been implemented in the StarPU runtime system. We present their performance on tasks from tiled 2D and 3D matrix products. Our experiments demonstrate that using our new strategy together with the optimal eviction policy reduces the amount of data movement as well as the total processing time.

A preliminary version of this work has been presented at the COLOC workshop of EuroPar 2021 [20]. An extended version is available as a research report [35].

7.2.5 Taming tail latency in key-value stores: a scheduling perspective.

Participants: Anthony Dugois, Loris Marchal, Anne Benoit, Louis-Claude Canon *(Univ. Besançon)*, Sonia Ben Mokhtar *(Univ. Lyon 1)*, Étienne Rivière *(Univ. Louvain, Belgique)*.

Distributed key-value stores employ replication for high availability. Yet, they do not always efficiently take advantage of the availability of multiple replicas for each value, and read operations often exhibit high tail latencies. Various replica selection strategies have been proposed to address this problem, together with local request scheduling policies. It is difficult, however, to determine what is the absolute performance gain each of these strategies can achieve. We present a formal framework allowing the systematic study of request scheduling strategies in key-value stores. We contribute a definition of the optimization problem related to reducing tail latency in a replicated key-value store as a minimization problem with respect to the maximum weighted flow criterion. By using scheduling theory, we show the difficulty of this problem, and therefore the need to develop performance guarantees. We also study the behavior of heuristic methods using simulations, which highlight which properties are useful for limiting tail latency: for instance, the EFT strategy—which uses the earliest available time of servers—exhibits a tail latency that is less than half that of state-of-the-art strategies, often matching the lower bound. Our study also emphasizes the importance of metrics such as the stretch to properly evaluate replica selection and local execution policies.

This work has been accepted at IPDPS 2022 [24]. An extended version is available as a research report [28].

7.2.6 Evaluating Task Dropping Strategies for Overloaded Real-Time Systems.

Participants: Yiqin Gao, Yves Robert, Frédéric Vivien, Guillaume Pallez *(Inria Bordeaux)*.

This work proposes evaluation criteria and scheduling strategies for the analysis of overloaded real-time systems. A single task periodic task must be processed by a server, under an arbitrary deadline (i.e., the relative deadline is larger than the period). The system is overloaded, which means that the period is smaller than the average execution time of the task instances. A task instance that does not meet its deadline is automatically killed. The problem is then to minimize the deadline-miss ratio. This work builds upon techniques from queueing theory and proposes a new approach for real-time systems.

A preliminary version of this work was accepted as a work-in-progress at the RTSS 2021 conference [19].

7.3 Sparse direct solvers and sparsity in computing

We continued our work on the optimization of sparse solvers by concentrating on data locality when mapping tasks to processors, and by studying the tradeoff between memory and performance when using low-rank compression. We worked on combinatorial problems arising in sparse matrix and tensors computations. The computations involved direct methods for solving sparse linear systems and tensor factorizations. The combinatorial problems were based on matchings on bipartite graphs, partitionings, and hyperedge queries.

7.3.1 Trading Performance for Memory in Sparse Direct Solvers using Low-rank Compression.

Participants: Grégoire Pichon, Loris Marchal, Thibault Marette *(ENS Lyon)*, Frédéric Vivien.

Sparse direct solvers using Block Low-Rank compression have been proven efficient to solve problems arising in many real-life applications. Improving those solvers is crucial for being able to 1) solve larger problems and 2) speed up computations. A main characteristic of a sparse direct solver using low-rank compression is at what point in the algorithm the compression is performed. There are two distinct approaches: (1) all blocks are compressed before starting the factorization, which reduces the memory as much as possible, or (2) each block is compressed as late as possible, which usually leads to better speedup. Approach 1 reaches a very small memory footprint generally at the expense of a greater execution time. Approach 2 achieves a smaller execution time but requires more memory. The objective of the proposed approach is to design a composite approach, to speedup computations while staying under a given memory limit. This should allow to solve large problems that cannot be solved with Approach 2 while reducing the execution time compared to Approach 1. We propose a memory-aware strategy where each block can be compressed either at the beginning or as late as possible. We first consider the problem of choosing when to compress each block, under the assumption that all information on blocks is perfectly known, i.e., memory requirement and execution time of a block when compressed or not. We show that this problem is a variant of the NP-complete Knapsack problem, and adapt an existing approximation algorithm for our problem. Unfortunately, the required information on blocks depends on numerical properties and in practice cannot be known in advance. We thus introduce models to estimate those values. Experiments on the PaStiX solver demonstrate that our new approach can achieve an excellent trade-off between memory consumption and computational cost. For instance on matrix Geo1438, Approach 2 uses three times as much memory as Approach 1 while being three times faster. Our new approach leads to an execution time only 30% larger than Approach 2 when given a memory 30% larger than the one needed by Approach 1.

This work is in press and will appear in FGCS in 2022 [13].

7.3.2 Deciding Non-Compressible Blocks in Sparse Direct Solvers using Incomplete Factorization.

Participants: Grégoire Pichon, Esragul Korkmaz *(Inria Bordeaux)*, Mathieu Faverge *(Inria Bordeaux)*, Pierre Ramet *(Inria Bordeaux)*.

Low-rank compression techniques are very promising for reducing memory footprint and execution time on a large spectrum of linear solvers. Sparse direct supernodal approaches are one of these techniques. However, despite providing a very good scalability and reducing the memory footprint, they suffer from an important flops overhead in their unstructured low-rank updates. As a consequence, the execution time is not improved as expected. In this work, we study a solution to improve low-rank compression techniques in sparse supernodal solvers. The proposed method tackles the overprice of the low-rank updates by identifying the blocks that have poor compression rates. We show that the fill-in levels of the graph based block incomplete LU factorization can be used in a new context to identify most of these non-compressible blocks at low cost. This identification enables to postpone the low-rank compression step to trade small extra memory consumption for a better time to solution. The solution is validated within the PaStiX library with a large set of application matrices. It demonstrates sequential and multithreaded speedup up to 8.5x, for small memory overhead of less than 1.49x with respect to the original version.

This work appeared in the proceedings of the HIPC 2021 conference [22]

7.3.3 Algorithms and data structures for hyperedge queries.

Participants: Bora Uçar, Jules Bertrand *(ENS Lyon)*, Fanny Dufossé *(Inria Grenoble)*.

In this work [33], we consider the problem of querying the existence of hyperedges in hypergraphs. More formally, we are given a hypergraph, and we need to answer queries of the form “does the following set of vertices form a hyperedge in the given hypergraph?”. Our aim is to set up data structures based on hashing to answer these queries as fast as possible. We propose an adaptation of a well-known perfect hashing approach for the problem at hand. We analyze the space and run time complexity of the proposed approach, and experimentally compare it with the state of the art hashing-based solutions. Experiments demonstrate that the proposed approach has shorter query response time than the other considered alternatives, while having the shortest or the second shortest construction time. During the internship of Jules Bernard (in collaboration with Fanny Dufossé of DataMove), we looked at the same problem in a dynamical setting, where hyperedges come and go. We have also continued to work on the associated software (gitlab link) and its parallelization.

7.3.4 Shared-memory implementation of the Karp-Sipser kernelization process.

Participants: Ioannis Panagiotas, Bora Uçar, Johannes Langguth *(Univ. Bergen, Norway)*.

In this work [23], we investigate the parallelization of the Karp-Sipser kernelization technique, which constitutes the central part of the well-known Karp-Sipser heuristic for the maximum cardinality matching problem. The technique reduces a given problem instance to a smaller but equivalent one, by repeated applications of two operations: vertex removal, and merging two vertices. The operation of merging two vertices poses the principal challenge in parallelizing the technique. We describe an algorithm that minimizes the need for synchronization and present an efficient shared-memory parallel implementation of the kernelization technique for bipartite graphs. Using extensive experiments on a variety of multicore CPUs, we show that our implementation scales well up to 32 cores on one socket.

7.3.5 Streaming hypergraph partitioning algorithms on limited memory environments.

Participants: Bora Uçar, Fatih Taşyaran *(Sabanci University, Turkey)*, Berkay Demireller *(Sabanci University, Turkey)*, Kamer Kaya *(Sabanci University, Turkey)*.

In this work [25], we assume a streaming model where the data items and their relations are modeled as a hypergraph, which is generated at the edge (Internet of Things). This hypergraph is then partitioned, and the parts are sent to remote nodes via an algorithm running on a memory-restricted device, such as a single board computer. Such a partitioning is usually performed by taking a connectivity metric into account to minimize the communication cost of later analyses that will be performed in a distributed fashion. Although there are many offline tools that can partition static hypergraphs effectively, algorithms for the streaming settings are rare. We analyze a well-known algorithm from the literature and significantly improve its run time by altering its inner data structure. On a medium-scale hypergraph, the new algorithm reduces the run time from 17800 seconds to 10 seconds. We then propose sketch-and hash-based algorithms, as well as ones that can leverage extra memory to store a small portion of the data to enable the refinement of partitioning when possible. We experimentally analyze the performance of these algorithms and report their run times, connectivity metric scores, and memory uses on a high-end server and four different single-board computer architectures.

7.3.6 Fully-dynamic weighted matching approximation in practice.

Participants: Bora Uçar, Eugenio Angriman *(Humboldt-Universität zu Berlin, Germany)*, Henning Meyerhenke *(Humboldt-Universität zu Berlin, Germany)*, Christian Schulz *(Universität Heidelberg, Germany)*.

In this work [14], we engineer the first non-trivial implementations for approximating the dynamic weighted matching problem. Our first algorithm is based on random walks/paths combined with dynamic programming. The second algorithm implements a recent algorithm for which there was not any implementation before our work. has been introduced by Stubbs and Williams without an implementation. We exploit previous work on dynamic unweighted matching algorithms as a black box in order to obtain a fully-dynamic weighted matching algorithm for implementing the second algorithm. We empirically study the algorithms on an extensive set of dynamic instances and compare them with optimal weighted matchings. Our experiments show that the random walk algorithm typically fares much better than Stubbs/Williams (regarding the time/quality tradeoff), and its results are often not far from the optimum.

7.3.7 Strongly connected components in directed hypergraphs.

Participants: Anne Benoit, Bora Uçar, Élodie Bernard *(ENS Lyon)*.

There are recent work developing numerical analysis of nonnegative sparse tensors. Much of the theoretical results are applicable only for irreducible tensors; the irreducibility of tensors has a definition similar to that of the irreducibility of matrices. However, there is no efficient tool support to test irreducibility and to find the irreducible blocks of large sparse tensors arising in real-life applications. We have initiated a study to address this lack of tool support by developing algorithms for detecting irreducible tensors based on hypergraphs and for detecting the maximally irreducible blocks when a tensor is reducible. Bora Uçar and Anne Benoît have worked with Elodie Bernard, an L3 intern, on this subject.

8 Partnerships and cooperations

8.1 International initiatives

8.1.1 Associate Teams in the framework of an Inria International Lab or in the framework of an Inria International Program

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

Research areas include:

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

Several members of the ROMA team are involved in the JLESC joint lab through their research on scheduling and resilience. Yves Robert is the INRIA executive director of JLESC.

8.1.2 Inria associate team not involved in an IIL or an international program

PEACHTREE

Participants: Bora Uçar, Anne Benoit, Loris Marchal.

Title: Shared memory sparse tensors computations: Combinatorial tools, scheduling, and numerical algorithms

Duration: 2020 – 2022

Coordinator: Umit V. Çatalyürek (umit@gatech.edu)

Partners:

- GeorgiaTech

Inria contact: Bora Uçar

Summary: Tensors, or multidimensional arrays, have many uses in data analysis applications. The additional dimensions over matrices (or two dimensional arrays) enable gleaning information that is otherwise unreachable. A remarkable example comes from the Netflix Challenge. The aim of the challenge was to improve the company's algorithm for predicting user ratings on movies using a dataset containing a set of ratings of users on movies. The winning algorithm, when the challenge was concluded, had to use the time dimension on top of user x movie rating, during the analysis. Tensors from many applications, such as the mentioned one, are sparse, which means that not all entries of the tensor are relevant or known. The PeachTree project investigates the building blocks of numerical parallel tensor computation algorithms on high end systems, and designs a set of scheduling and combinatorial tools for achieving efficiency. More information at PeachTree web page.

8.1.3 Participation in other International Programs

- PHC Aurora Project with J. Langguth of Simula Labs, Norway. More information at project web page.

8.2 International research visitors

8.2.1 Visits to international teams

Research stays abroad

Yves Robert

Visited institution: ICL laboratory, University of Tennessee in Knoxville

Country: USA

Dates: several visits

Context of the visit: Yves Robert has been appointed as a visiting scientist by the ICL laboratory (headed by Jack Dongarra) since 2011.

Mobility program/type of mobility: Visiting scientist.

8.3 National initiatives

8.3.1 ANR Project SOLHARIS (2019-2023), 4 years.

Participants: Maxime Gonthier, Gréfoire Pichon, Loris Marchal, Bora Uçar.

The ANR Project SOLHAR was launched in November 2019, for a duration of 48 months. It gathers five academic partners (the HiePACS, ROMA, RealOpt, STORM and TADAAM INRIA project-teams, and CNRS-IRIT) and two industrial partners (CEA/CESTA and Airbus CRT). This project aims at producing scalable methods for direct methods for the solution of sparse linear systems on large scale and heterogeneous computing platforms, based on task-based runtime systems.

The proposed research is organized along three distinct research thrusts. The first objective deals with the development of scalable linear algebra solvers on task-based runtimes. The second one focuses on the deployement of runtime systems on large-scale heterogeneous platforms. The last one is concerned with scheduling these particular applications on a heterogeneous and large-scale environment.

9 Dissemination

9.1 Promoting scientific activities

9.1.1 Scientific events: organisation

General chair, scientific chair

- Anne Benoit is the general co-chair of IEEE IPDPS'22 (36th IEEE International Parallel & Distributed Processing Symposium).

9.1.2 Scientific events: selection

Chair of conference program committees

- Anne Benoit was the Poster chair of ICPP'21.
- Yves Robert and Bora Uçar are the program co-chairs of IEEE IPDPS'22.
- Bora Uçar is the program vice-chair of SEA 2022 (20th Symposium on Experimental Algorithms).
- Yves Robert is the ACM Posters vice-chair of SC'22.
- Anne Benoit is the ACM SRC Graduate Posters chair of SC'22.
- Frédéric Vivien is Research Posters vice-chair for SC'22.

Member of the conference program committees

- Anne Benoit was a member of the program committees of IPDPS'21, SC'21, Compas'21, SuperCheck'21. She is a member of the program committee of SC'22 and PPAM'22.
- Loris Marchal was/is a member of the program committees of APDCM'2021, EuroPar'2021, IPDPS'2022, APDCM'2022 and ICPP'2022.
- Grégoire Pichon was a member of the program committee of SBAC-PAD'21
- Bora Uçar was a member of the Proceedings Paper Committee of the 20th SIAM Conference on Parallel Processing for Scientific Computing, (to be held in February 2022).
- Frédéric Vivien was a "Special Committee Member" of the program committee os IPDPS'22, and a member of the program committees of IPDPS'21 and PDP 2021.
- Yves Robert was a member of the program committees of FTXS'21, SCALA'21, PMBS'21, SuperCheck'21 (co-located with SC'21) and Resilience (co-located with Euro-Par'21).

Reviewer

- Bora Uçar reviewed papers for SIAM Conference on Applied and Computational Discrete Algorithms.

9.1.3 Journal

Member of the editorial boards

- Anne Benoit is Associate Editor (in Chief) of the journal of Parallel Computing: Systems and Applications (ParCo).
- Bora Uçar is a member of the editorial board of IEEE Transactions on Parallel and Distributed Systems (IEEE TPDS), SIAM Journal on Scientific Computing (SISC), SIAM Journal on Matrix Analysis and Applications (SIMAX), and Parallel Computing. He is also acting as a guest editor for a special issue of Journal of Parallel and Distributed Computing.
- Frédéric Vivien is a member of the editorial board of Journal of Parallel and Distributed Computing and of the ACM Transactions on Parallel Computing.
- Yves Robert is a member of the editorial board of ACM Transactions on Parallel Computing (TOPC), the International Journal of High Performance Computing (IJHPCA) and the Journal of Computational Science (JOCS).

Reviewer - reviewing activities

- Anne Benoit reviewed papers for JPDC, TOPC and COMNET.
- Loris Marchal reviewed papers for CCPE and ParCo.
- Grégoire Pichon reviewed papers for TPDS, SIMAX.
- Yves Robert reviewed papers for ACM TOPC, IEEE TC (Trans. Computers), IEEE TPDS (Trans. Parallel Distributed Systems) and IEEE TCC (Trans. Cloud Computing).
- Bora Uçar reviewed papers for Journal of Computational and Applied Mathematics, Mathematical Geosciences, Concurrency and Computation: Practice and Experience, and Journal of Parallel and Distributed Computing.

9.1.4 Invited talks

- Anne Benoit gave a keynote talk at the 18th International Conference on Network and Parallel Computing (IFIP NPC), Paris, November 2021, on Resilient scheduling for high-performance computing.
- Bora Uçar gave an invited talk to the Numerical Analysis group at the University of Strathclyde, 30 March 2021 (online during the Covid-19 pandemic); he also gave a guest lecture (4 times, 75 minutes each) on matching heuristics for Alex Pothen's graduate level course at Purdue.

9.1.5 Leadership within the scientific community

- Anne Benoit is elected as chair of IEEE TCPP, the Technical Committee on Parallel Processing (2020-2022). She serves in the steering committees of IPDPS, HCW, and HeteroPar.
- Bora Uçar serves as the secretary of SIAM Activity Group on Applied and Computational Discrete Algorithms (for the period Jan 21 – Dec 22).
- Bora Uçar serves in the steering committee of HiPC (2019–2021)
- Yves Robert serves in the steering committee of IPDPS and HCW.

9.1.6 Scientific expertise

- Frédéric Vivien is an elected member of the scientific council of the École normale supérieure de Lyon.
- Frédéric Vivien is a member of the scientific council of the IRMIA labex.

9.1.7 Research administration

- Loris Marchal is a member of the committee of the "Complexity of Algorithm" working group of the Gdr IM (CNRS research group on theoretical computer science)
- Bora Uçar is an elected member of the Council of LIP (Conseil du LIP), and also an elected member of Council of the Fédération d'Informatique de Lyon (Conseil de la FIL). He is also co-chair of the thesis committee of LIP (Co-responsable de la commission des thèses du LIP).
- Frédéric Vivien is the vice-head of the Fédération Informatique de Lyon.

9.2 Teaching - Supervision - Juries

9.2.1 Teaching

- Licence: Anne Benoit, Responsible of the L3 students at ENS Lyon, France
- Licence: Anne Benoit, Algorithmique avancée, 48h, L3, ENS Lyon, France
- Master: Anne Benoit, Parallel and Distributed Algorithms and Programs, 42h, M1, ENS Lyon, France
- Master: Grégoire Pichon, Resource optimization for linear system solvers, 10h, M2, ENS Lyon, France
- Master: Grégoire Pichon, Compilation / traduction des programmes, 24h, M1, Univ. Lyon 1, France
- Master: Grégoire Pichon, Programmation système et temps réel, 27.75h, M1, Univ. Lyon 1, France
- Master: Grégoire Pichon, Réseaux, 12h, M1, Univ. Lyon 1, France
- Licence: Grégoire Pichon, Programmation concurrente, 28.5h, L3, Univ. Lyon 1, France
- Licence: Grégoire Pichon, Réseaux, 34h, L3, Univ. Lyon 1, France
- Licence: Grégoire Pichon, Système d'exploitation, 24h, L2, Univ. Lyon 1, France
- Licence: Grégoire Pichon, Introduction aux réseaux et au web, 21h, L1, Univ. Lyon 1, France
- Licence: Grégoire Pichon, Référent pédagogique, 30h, L1/L2/L3, Univ. Lyon 1, France
- Master: Grégoire Pichon, Bora Uçar, and Frédéric Vivien, Resource optimization for linear system solvers, 10h each, M2, ENS Lyon, France.
- Master: Yves Robert, Responsible of the M2 students at ENS Lyon, France (2018-2021)
- Licence: Yves Robert, Probabilités et algorithmes randomisés, 48h cours +32h TD, L3, ENS Lyon, France
- Agrégation Informatique: Yves Robert, Algorithmique, NP-complétude et algorithmes d'approximation, probabilités, 74h, ENS Lyon, France

9.2.2 Supervision

- PhD defended: Yiqin Gao, "Replication Algorithms for Real-time Tasks with Precedence Constraints", defended on September 29, 2021, funding: ENS Lyon, advisors: Yves Robert and Frédéric Vivien.
- PhD in progress: Lucas Perotin, "Fault-tolerant scheduling of parallel jobs", started in October 2020, funding: ENS Lyon, advisors: Anne Benoit and Yves Robert.
- PhD in progress: Redouane Elghazi, "Stochastic Scheduling for HPC Systems", started in September 2020, funding: Région Franche-Comté, advisors: Anne Benoit, Louis-Claude Canon and Pierre-Cyrille Héam.
- PhD in progress: Zhiwei Wu, "Energy-aware strategies for periodic scientific workflows under reliability constraints on heterogeneous platforms", started in October 2020, funding: China Scholarship Council, advisors: Frédéric Vivien, Yves Robert, Li Han (ECNU) and Jing Liu (ECNU).
- PhD in progress: Yishu Du, "Resilient algorithms and scheduling techniques for numerical algorithms", started in December 2019, funding: China Scholarship Council, advisors: Loris Marchal and Yves Robert.
- PhD in progress: Anthony Dugois "Scheduling for key value stores", started in October 2020, funding: Inria, advisors: Loris Marchal and Louis-Claude Canon (Univ. Besançon).

- PhD in progress: Maxime Gonthier “Memory-Aware scheduling for task-based runtime systems”, started in October 2020, funding: Inria, advisors: Loris Marchal and Samuel Thibault (Univ. Bordeaux).

9.2.3 Juries

- Anne Benoit was a reviewer and a member of the jury for the thesis of Alena Shilova (December 2021, Université de Bordeaux).
- Loris Marchal is a responsible of the competitive selection of ENS Lyon students for Computer Science, and is a member of the jury of this competitive exam.
- Yves Robert was the chair of the 2021 ACM/IEEE-CS George Michael HPC Fellowship committee. He was a member of the 2021 IEEE Fellow Committee, and of the 2021 IEEE Charles Babbage Award Committee.

9.3 Popularization

9.3.1 Articles and contents

- Yves Robert, together with George Bosilca, Aurélien Bouteiller and Thomas Herault, gave a full-day tutorial at SC’21 on *Fault-tolerant techniques for HPC and Big Data: theory and practice*.

10 Scientific production

10.1 Major publications

[1] A. Benoit, T. Hérault, V. Le Fèvre and Y. Robert. ‘Replication Is More Efficient Than You Think’. In: *SC 2019 - International Conference for High Performance Computing, Networking, Storage, and Analysis (SC’19)*. Denver, United States, Nov. 2019. URL: https://hal.inria.fr/hal-02273142.

[2] M. Bougeret, H. Casanova, M. Rabie, Y. Robert and F. Vivien. ‘Checkpointing strategies for parallel jobs.’ In: *SuperComputing (SC) - International Conference for High Performance Computing, Networking, Storage and Analysis, 2011*. United States, 2011, pp. 1–11. URL: https://hal.archives-ouvertes.fr/hal-00738504.

[3] J. Dongarra, T. Hérault and Y. Robert. ‘Fault Tolerance Techniques for High-Performance Computing’. In: *Fault-Tolerance Techniques for High-Performance Computing*. Ed. by T. Hérault and Y. Robert. Springer, May 2015, p. 83. URL: https://hal.inria.fr/hal-01200488.

[4] F. Dufossé and B. Uçar. ‘Notes on Birkhoff-von Neumann decomposition of doubly stochastic matrices’. In: *Linear Algebra and its Applications* 497 (Feb. 2016), pp. 108–115. DOI: 10.1016/j.laa.2016.02.023. URL: https://hal.inria.fr/hal-01270331.

[5] L. Eyraud-Dubois, L. Marchal, O. Sinnen and F. Vivien. ‘Parallel scheduling of task trees with limited memory’. In: *ACM Transactions on Parallel Computing* 2.2 (July 2015), p. 36. DOI: 10.1145/2779052. URL: https://hal.inria.fr/hal-01160118.

[6] L. Marchal, B. Simon and F. Vivien. ‘Limiting the memory footprint when dynamically scheduling DAGs on shared-memory platforms’. In: *Journal of Parallel and Distributed Computing* 128 (Feb. 2019), pp. 30–42. DOI: 10.1016/j.jpdc.2019.01.009. URL: https://hal.inria.fr/hal-02025521.

10.2 Publications of the year

International journals

[7] G. Bathie, L. Marchal, Y. Robert and S. Thibault. ‘Dynamic DAG Scheduling Under Memory Constraints for Shared-Memory Platforms’. In: *International Journal of Networking and Computing* (Jan. 2021), pp. 1–29. DOI: 10.15803/ijnc.11.1_27. URL: https://hal.inria.fr/hal-03029847.

[8] A. Benoit, V. Le Fèvre, L. Perotin, P. Raghavan, Y. Robert and H. Sun. 'Resilient Scheduling of Moldable Parallel Jobs to Cope with Silent Errors'. In: *IEEE Transactions on Computers* (Aug. 2021), pp. 1–14. DOI: 10.1109/TC.2021.3104747. URL: https://hal.inria.fr/hal-03509760.

[9] A. Benoit, V. Le Fèvre, P. Raghavan, Y. Robert and H. Sun. 'Resilient Scheduling Heuristics for Rigid Parallel Jobs'. In: *International Journal of Networking and Computing* 11.1 (2021), pp. 1–25. URL: https://hal.inria.fr/hal-03508937.

[10] Y. Caniou, E. Caron, A. Kong Win Chang and Y. Robert. 'Budget-aware scheduling algorithms for scientific workflows with stochastic task weights on IaaS Cloud platforms'. In: *Concurrency and Computation: Practice and Experience* 33.17 (2021), pp. 1–25. URL: https://hal.inria.fr/hal-03508925.

[11] Y. Du, L. Marchal, G. Pallez and Y. Robert. 'Optimal Checkpointing Strategies for Iterative Applications'. In: *IEEE Transactions on Parallel and Distributed Systems* 33.3 (1st Mar. 2022), pp. 507–522. DOI: 10.1109/TPDS.2021.3099440. URL: https://hal.inria.fr/hal-03338278.

[12] F. Dufossé, K. Kaya, I. Panagiotas and B. Uçar. 'Scaling matrices and counting the perfect matchings in graphs'. In: *Discrete Applied Mathematics*. 2021st ser. 308 (Feb. 2022), pp. 130–146. URL: https://hal.inria.fr/hal-01743802.

[13] L. Marchal, T. Marette, G. Pichon and F. Vivien. 'Trading Performance for Memory in Sparse Direct Solvers using Low-rank Compression'. In: *Future Generation Computer Systems* (2022). URL: https://hal.inria.fr/hal-03517124.

International peer-reviewed conferences

[14] E. Angriman, H. Meyerhenke, C. Schulz and B. Uçar. 'Fully-dynamic Weighted Matching Approximation in Practice'. In: SIAM Conference on Applied and Computational Discrete Algorithms (ACDA21). Virtual, France, 2021. URL: https://hal.inria.fr/hal-03210915.

[15] A. Benoit, L.-C. Canon, R. Elghazi and P.-C. Heam. 'Shelf schedules for independent moldable tasks to minimize the energy consumption'. In: SBAC-PAD 2021 - IEEE 33rd International Symposium on Computer Architecture and High Performance Computing. Belo Horizonte, Brazil, Oct. 2021, pp. 1–11. URL: https://hal.inria.fr/hal-03509709.

[16] A. Benoit, L.-C. Canon, R. Elghazi and P.-C. Heam. 'Update on the Asymptotic Optimality of LPT'. In: Euro-Par 2021 - 27th International European Conference on Parallel and Distributed Computing. Lisbon, Portugal, Aug. 2021, pp. 1–14. URL: https://hal.inria.fr/hal-03509666.

[17] A. Benoit, R. Elghazi and Y. Robert. 'Max-stretch minimization on an edge-cloud platform'. In: IPDPS 2021 - IEEE International Parallel and Distributed Processing Symposium. Portland, Oregon, United States: IEEE, Oct. 2020, pp. 1–10. URL: https://hal.inria.fr/hal-03509637.

[18] Y. Caniou, E. Caron, A. Kong Win Chang and Y. Robert. 'Budget-aware Static Scheduling of Stochastic Workflows with DIET'. In: ADVCOMP 2021 - Fifteenth International Conference on Advanced Engineering Computing and Applications in Sciences. Barcelona, Spain, 3rd Oct. 2021, pp. 1–8. URL: https://hal.inria.fr/hal-03332601.

[19] Y. Gao, G. Pallez, Y. Robert and F. Vivien. 'Work-in-Progress: Evaluating Task Dropping Strategies for Overloaded Real-Time Systems'. In: RTSS 2021 - 42nd IEEE Real-Time Systems Symposium. Dortmund, Germany: IEEE, 7th Dec. 2021, pp. 1–4. URL: https://hal.inria.fr/hal-03357422.

[20] M. Gonthier, L. Marchal and S. Thibault. 'Locality-Aware Scheduling of Independent Tasks for Runtime Systems'. In: COLOC - 5th workshop on data locality - 27th International European Conference on Parallel and Distributed Computing. Lisbon, Portugal: Springer, 30th Aug. 2021, pp. 1–12. URL: https://hal.archives-ouvertes.fr/hal-03290998.

[21] T. Herault, Y. Robert, G. Bosilca, R. J. Harrison, C. A. Lewis, E. F. Valeev and J. J. Dongarra. 'Distributed-memory multi-GPU block-sparse tensor contraction for electronic structure'. In: IPDPS 2021 - IEEE International Parallel and Distributed Processing Symposium. Portland, OR, United States: IEEE, 17th May 2021, pp. 1–10. URL: https://hal.inria.fr/hal-03508930.

[22] E. Korkmaz, M. Faverge, G. Pichon and P. Ramet. 'Deciding Non-Compressible Blocks in Sparse Direct Solvers using Incomplete Factorization'. In: HiPC 2021 - 28th IEEE International Conference on High Performance Computing, Data, and Analytics. Bangalore, India: IEEE, 17th Dec. 2021, pp. 1–10. URL: https://hal.inria.fr/hal-03361299.

[23] J. Langguth, I. Panagiotas and B. Uçar. 'Shared-memory implementation of the Karp-Sipser kernelization process'. In: HiPC 2021 - 28th edition of the IEEE International Conference on High Performance Computing, Data, and Analytics. Bangalore, India: IEEE, 17th Dec. 2021, pp. 71–80. URL: https://hal.inria.fr/hal-03404798.

[24] S. B. Mokhtar, L.-C. Canon, A. Dugois, L. Marchal and E. Rivière. 'Taming Tail Latency in Key-Value Stores: a Scheduling Perspective'. In: *Euro-Par 2021: Parallel Processing*. Euro-Par 2021: Parallel Processing. Vol. 12820. Lecture Notes in Computer Science. Lisbon (virtual), Portugal: Springer International Publishing, 25th Aug. 2021, pp. 136–150. DOI: 10.1007/978-3-030-85665-6_9. URL: https://hal.inria.fr/hal-03424040.

[25] F. Taşyaran, B. Demireller, K. Kaya and B. Uçar. 'Streaming Hypergraph Partitioning Algorithms on Limited Memory Environments'. In: HPCS 2020 - International Conference on High Performance Computing & Simulation. Virtual online, Spain: IEEE, 22nd Mar. 2021, pp. 1–8. URL: https://hal.archives-ouvertes.fr/hal-03182122.

Doctoral dissertations and habilitation theses

[26] Y. Gao. 'Scheduling independent tasks under budget and time constraints'. Université de Lyon, 29th Sept. 2021. URL: https://tel.archives-ouvertes.fr/tel-03412631.

Reports & preprints

[27] P. Amestoy, A. Buttari, N. J. Higham, J.-Y. L'Excellent, T. Mary and B. Vieuble. *Combining sparse approximate factorizations with mixed precision iterative refinement*. 19th Jan. 2022. URL: https://hal.archives-ouvertes.fr/hal-03536031.

[28] S. Ben Mokhtar, L.-C. Canon, A. Dugois, L. Marchal and E. Rivière. *Taming Tail Latency in Key-Value Stores: a Scheduling Perspective (extended version)*. 12th Mar. 2021. DOI: 10.6084/m9.figshare.13114196. URL: https://hal.inria.fr/hal-03144818.

[29] A. Benoit, L.-C. Canon, R. Elghazi and P.-C. Heam. *Shelf schedules for independent moldable tasks to minimize the energy consumption*. RR-9436. Institut National de Recherche en Informatique et en Automatique (INRIA), Nov. 2021, p. 19. URL: https://hal.inria.fr/hal-03447266.

[30] A. Benoit, L.-C. Canon, R. Elghazi and P.-C. Heam. *Update on the Asymptotic Optimality of LPT*. RR-9397. Inria Grenoble - Rhône-Alpes, Feb. 2021, p. 23. URL: https://hal.inria.fr/hal-03159022.

[31] A. Benoit, V. Le Fèvre, L. Perotin, P. Raghavan, Y. Robert and H. Sun. *Resilient Scheduling of Moldable Parallel Jobs to Cope with Silent Errors*. RR-9340. Inria - Research Centre Grenoble – Rhône-Alpes, Jan. 2021. URL: https://hal.inria.fr/hal-02614215.

[32] A. Benoit, L. Perotin, Y. Robert and H. Sun. *Checkpointing Workflows à la Young/Daly Is Not Good Enough*. RR-9413. Inria - Research Centre Grenoble – Rhône-Alpes, June 2021, p. 54. URL: https://hal.inria.fr/hal-03264047.

[33] J. Bertrand, F. Dufossé and B. Uçar. *Algorithms and data structures for hyperedge queries*. RR-9390. Inria Grenoble Rhône-Alpes, 1st Feb. 2021, p. 25. URL: https://hal.inria.fr/hal-03127673.

[34] Y. Gao, L. Han, J. Liu, Y. Robert and F. Vivien. *Minimizing energy consumption for real-time tasks on heterogeneous platforms under deadline and reliability constraints*. RR-9403. Inria - Research Centre Grenoble – Rhône-Alpes, Apr. 2021, p. 417. URL: https://hal.inria.fr/hal-03202996.

[35] M. Gonthier, L. Marchal and S. Thibault. *Locality-Aware Scheduling of Independant Tasks for Runtime Systems*. RR-9394. Inria Grenoble -Rhône-Alpes, 2021, p. 21. URL: https://hal.inria.fr/hal-03144290.

[36] E. Korkmaz, M. Faverge, G. Pichon and P. Ramet. *Deciding Non-Compressible Blocks in Sparse Direct Solvers using Incomplete Factorization*. RR-9396. Inria Bordeaux - Sud Ouest, 2021, p. 16. URL: https://hal.inria.fr/hal-03152932.

10.3 Cited publications

[37] A. Benoit, V. Le Fèvre, L. Perotin, P. Raghavan, Y. Robert and H. Sun. 'Resilient Scheduling of Moldable Jobs on Failure-Prone Platforms'. In: *CLUSTER 2020 - IEEE International Conference on Cluster Computing*. Kobe, Japan: IEEE, Sept. 2020, pp. 1–29. URL: https://hal.inria.fr/hal-03028773.