

Algorithms and Scheduling Techniques for Exascale Systems

Edited by

Henri Casanova¹, Yves Robert², and Uwe Schwiegelshohn³

1 University of Hawaii at Manoa, US, henric@hawaii.edu

2 ENS – Lyon, FR, Yves.Robert@ens-lyon.fr

3 TU Dortmund, DE, uwe.schwiegelshohn@udo.edu

Abstract

Exascale systems to be deployed in the near future will come with deep hierarchical parallelism, will exhibit various levels of heterogeneity, will be prone to frequent component failures, and will face tight power consumption constraints. The notion of application performance in these systems becomes multi-criteria, with fault-tolerance and power consumption metrics to be considered in addition to sheer compute speed. As a result, many of the proven algorithmic techniques used in parallel computing for decades will not be effective in Exascale systems unless they are adapted or in some cases radically changed. The Dagstuhl seminar “Algorithms and Scheduling Techniques for Exascale Systems” was aimed at sharing open problems, new results, and prospective ideas broadly connected to the Exascale problem. This report provides a brief executive summary of the seminar and lists all the presented material.

Seminar 15.–20. September, 2013 – www.dagstuhl.de/13381

1998 ACM Subject Classification D.4 Operating Systems, C.1 Processor Architectures, F.2 Analysis of Algorithms and Problem Complexity, D.1.3 Concurrent Programming, F.2.2 Nonnumerical Algorithms and Problems, I.2.8 Problem Solving, Control Methods, and Search

Keywords and phrases Exascale computing, high-performance computing and networking, fault-tolerance, power management, scheduling, numerical linear algebra

Digital Object Identifier 10.4230/DagRep.3.9.106

1 Executive Summary

Henri Casanova

Yves Robert

Uwe Schwiegelshohn

License  Creative Commons BY 3.0 Unported license
© Henri Casanova, Yves Robert, and Uwe Schwiegelshohn

Hardware manufacturers are currently deploying machines with sustained petascale performance while already looking forward to produce Exascale machines. Exascale systems are likely to contain 10^5 to 10^6 processors, each processor itself being equipped with more than 100 cores, and possibly 10^3 to 10^4 GPU cores. These systems already reach such a degree of sophistication and complexity that the conventional approach of hardware goes first and applications follow is likely to fail. Furthermore, application performance is no longer solely defined by time-to-solution but also by power consumption and resilience to fault. Many conferences and workshops are dedicated to the architecture and systems issues pertaining to Exascale computing. Instead, in this seminar we have discussed algorithmic



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 3.0 Unported license

Algorithms and Scheduling Techniques for Exascale Systems, *Dagstuhl Reports*, Vol. 1, Issue 1, pp. 106–129
Editors: Henri Casanova, Yves Robert, and Uwe Schwiegelshohn



DAGSTUHL
REPORTS

Dagstuhl Reports
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

issues (application parallelization, application scheduling, resource management, etc.) that must be addressed to make Exascale computing a tenable proposition. As seen in many of the presentations during the seminar, core elements or principles of existing applications must be modified so that they can form the building blocks of new Exascale applications while new methods specifically targeting Exascale systems must be developed for new application areas.

The presentations during the seminar covered a wide range of topics. Some of these topics were directly targeted to various aspects of “the Exascale problem”. Some topics were targeted to components of the problem, e.g., efficient execution of application kernels on a heterogeneous many-core node. Finally, yet other topics were in broader, and often more theoretical, parallel and distributed computing contexts with less immediate but possibly large impact on the future of Exascale computing. Overall, the topics presented and discussed during the workshop can be roughly categorized as follows, noting that at least half the presentations spanned more than one of these topics:

- **Fault-tolerance.** Fault-tolerance is a major concern at large scale and several presentations focused on the limitations of current checkpoint-restart fault-tolerance techniques, providing analytical studies to quantify expected performance of these solutions and comparing them to proposed new solutions. These new solutions included, for instance, the use of algorithm-specific checkpointing combined with system-level checkpointing, or the use of imperfect fault predictors.
- **Multi-criteria optimization.** A large number of presentations presented multi-criteria optimization problems, including one traditional performance metric (throughput, makespan) and one (2-criteria) or two (3-criteria) metrics relating to power consumption and/or reliability. Several works studied the use of techniques such as DVFS to trade-off performance for a lower power consumption. These multi-criteria problems were formalized, and various theoretical and practical results were obtained in attempts to solve these problems. Two main approaches were followed: (i) optimizing one metric w.r.t. constraints on the other metric(s); or (ii) obtaining Pareto-optimal solutions or determining the entire Pareto front.
- **Multiple cores.** A handful of presentations focused on the above optimization problems not on large-scale platforms but on many-core nodes with shared memory, i.e., the intended individual components of future Exascale platforms. These nodes consist of possibly heterogeneous cores, accelerators (GPUs, etc.) connected via busses and on-chip networks.
- **Novel scheduling results.** A large number of presentations included novel findings regarding the complexity of scheduling problems. These scheduling problems are of general interest for various models of parallel computation, as motivated by the above topics. Results consisted of p-time optimal algorithms, new NP-completeness results, approximation algorithms, and efficient heuristics.
- **Exascale scientific computing.** Several presentations focused on particular scientific applications (e.g., PDE solvers) or scientific kernels (e.g., matrix multiplication), and discussed how age-old algorithms should be adapted to exploit Exascale platforms with heterogeneous components, hierarchical networks, and the need to have both efficient and rare communication primitive invocations. One presentation presented recent experience with scalable performance monitoring and performance debugging, capabilities that will be crucial in the practice of Exascale computing.
- **Programming models for Exascale.** A handful of presentations spoke to the need for novel programming models at large scale. These presentations spanned the spectrum from very (e.g., actual implementations of programming models usable today, proposals to

enhance current programming standards) to theoretical (e.g., a new theoretical approach to assess the efficiency of techniques such as work stealing and least-loaded-machine-first scheduling when the number of compute nodes tends to infinity).

- **Resource and application management.** A handful of presentations discussed Exascale computing in the context of cloud computing. In other words, these presenters made a case for applying/evolving some of the concepts currently applied in cloud deployments to future Exascale platforms (e.g., service level agreements, virtualization, resource economy). A number of open problems were identified when trying to make these two “worlds” collide.

Although the presentations at the seminar were very diverse in scope, ranging from practice to theory, an interesting observation is that many works do establish strong links between practice (e.g., particular applications, programming models) and theory (e.g., abstract scheduling problems and results). For instance, it was found that the age-old numerical linear algebra topic, far from being well-understood, in fact gives rise to a range of interrelated and interesting practical and theoretical problems that must be solved conjointly to achieve efficiency at large scale. Such observations make it plain that forums that blends practice and theory, as is the case with this seminar, are very much needed.

The seminar brought together 41 researchers from Austria, France, Germany, Ireland, Morocco, Netherlands, New Zealand, Switzerland, U.K, and U.S.A. with interests and expertise in different aspect of parallel and distributed computing. Among participants there was a good mix of senior researchers, junior researchers, postdoctoral researchers, and Ph.D. students. Altogether there were 36 presentations over the 5 days of the seminar, organized in morning and late-afternoon sessions, plus an open problem session. The program was as usual a compromise between allowing sufficient time for participants to present their work, while also providing unstructured periods that were used by participants to pursue ongoing collaborations as well as to foster new ones. The feedback provided by the participants show that the goals of the seminar, namely to circulate new ideas and create new collaborations, were met to a large extent.

The organizers and participants wish to thank the staff and the management of Schloss Dagstuhl for their assistance and support in the arrangement of a very successful and productive event.

2 Table of Contents

Executive Summary

Henri Casanova, Yves Robert, and Uwe Schwiegelshohn 106

Overview of Talks

Adaptive Energy and Throughput Scheduling for Dynamic Systems

Ismail Assayad 111

Approximation algorithms for energy, reliability and makespan optimization problems

Guillaume Aupy 111

What Makes Affinity-Based Schedulers So Efficient?

Olivier Beaumont 112

Energy-aware scheduling

Anne Benoit 112

Using Runtimes to ease the way to Exascale

George Bosilca 112

Enabling the production deployment of advanced Fault Tolerance techniques: Fault Tolerant MPI

Aurélien Bouteiller 113

Scheduling Data Sensor Retrieval for Boolean Tree Query Processing

Henri Casanova 113

Does the cache matter? When? Why? and How Much?

Anthony Danalis 114

Comparison of push and pull strategies in server farms

Bruno Gaujal 114

Exascale techniques for Numerics for PDEs (Part II)

Dominik Goeddeke 115

Sparse QR factorization for multicore architectures over runtime systems

Abdou Guermouche 115

Unified Model for Assessing Checkpointing Protocols at Extreme-Scale

Amina Guermouche 116

ABFT & Checkpoint-Composite Strategy

Thomas Héroult 116

Tree traversals with task-memory affinities

Julien Herrmann 117

On the Scalability of Moldable Task Scheduling Algorithms

Sascha Hunold 117

Communication and Topology-aware Load Balancing in Charm++ with TreeMatch

Emmanuel Jeannot 118

Many-task computing for the rest of us

Thilo Kielmann 118

On Variants of the Hierarchically Structured Bin Packing Problem <i>Thomas Lambert</i>	119
Hierarchical Approach to Improve Performance of Legacy Scientific Applications on Large-Scale Platforms <i>Alexey Lastovetsky</i>	119
A greedy algorithm for optimally pipelining a reduction <i>Bradley Lowery</i>	120
Efficient Bi-objective Optimization of Energy and Makespan for Exascale Heteroge- neous Computing Systems <i>Anthony A. Maciejewski</i>	120
Scheduling tree-shaped task graphs to minimize memory and makespan <i>Loris Marchal</i>	121
Liveness Evaluation for Cyclo-Static DataFlow Graphs <i>Alix Munier</i>	121
Planning for ExaScale Systems: The Challenge to be Prepared <i>Wolfgang E. Nagel</i>	122
Task Scheduling to Extend Platform Useful Life using Prognostics <i>Jean-Marc Nicod</i>	122
Energy-aware Execution of Fork-Join-based Task Parallelism <i>Thomas Rauber</i>	123
Reliable Service Allocation in Clouds with Memory and Capacity Constraints <i>Paul Renaud-Goud</i>	123
Some thoughts on exascale and scheduling <i>Rizos Sakellariou</i>	124
Parallel Dataflow Graph Coloring <i>Eric Saule</i>	124
Resource Management in an IaaS Environment <i>Uwe Schwiegelshohn</i>	124
Stochastic-Based Deadline-Aware and Energy-Constrained Robust Static Resource Management <i>Howard Jay Siegel</i>	125
Tools and Resources for Task Scheduling Research <i>Oliver Sinnen</i>	125
Job Scheduling Using Successive Linear Approximations of a Sparse Model <i>Veronika Sonigo</i>	126
Exascale techniques for Numerics for PDEs (Part I) <i>Stefan Turek</i>	126
Semi-matching algorithms for scheduling parallel tasks under resource constraints <i>Bora Ucar</i>	127
Checkpointing and fault prediction <i>Frédéric Vivien</i>	127
Open Problems	127
Participants	129

3 Overview of Talks

3.1 Adaptive Energy and Throughput Scheduling for Dynamic Systems

Ismail Assayad (University Hassan II – Casablanca, MA)

License © Creative Commons BY 3.0 Unported license
© Ismail Assayad

Joint work of Assayad, Ismail; Girault, Alain

Owing to unpredictable changes of the workload variation, optimally running multiple applications in terms of throughput and energy consumption on multi-core architectures is a huge challenge. We propose a novel adaptive approach capable of handling dynamism of the set of applications. For each such application, a set of non-dominated (Pareto) schedules are computed at design-time. Then, upon the starting or ending of an application, a lightweight adaptive run-time scheduler re-configures the mapping of the live applications according to the available resources (i.e., the available cores of the multi-core chip). This novel scheduling approach is adaptive, and thus able to change the mapping of applications during their execution, under throughput and power-constrained modes. This is a work in progress.

3.2 Approximation algorithms for energy, reliability and makespan optimization problems

Guillaume Aupy (ENS – Lyon, FR)

License © Creative Commons BY 3.0 Unported license
© Guillaume Aupy

Joint work of Aupy, Guillaume; Benoit, Anne

Main reference G. Aupy, A. Benoit, “Approximation algorithms for energy, reliability and makespan optimization problems,” INRIA Research report RR-8107, October 2012.

URL <http://hal.inria.fr/hal-00742754>

In this paper, we consider the problem of scheduling an application on a parallel computational platform. The application is a particular task graph, either a linear chain of tasks, or a set of independent tasks. The platform is made of identical processors, whose speed can be dynamically modified. It is also subject to failures: if a processor is slowed down to decrease the energy consumption, it has a higher chance to fail. Therefore, the scheduling problem requires to re-execute or replicate tasks (i.e., execute twice a same task, either on the same processor, or on two distinct processors), in order to increase the reliability. It is a tri-criteria problem: the goal is to minimize the energy consumption, while enforcing a bound on the total execution time (the makespan), and a constraint on the reliability of each task.

Our main contribution is to propose approximation algorithms for these particular classes of task graphs. For linear chains, we design a fully polynomial time approximation scheme. However, we show that there exists no constant factor approximation algorithm for independent tasks, unless $P=NP$, and we are able in this case to propose an approximation algorithm with a relaxation on the makespan constraint.

3.3 What Makes Affinity-Based Schedulers So Efficient?

Olivier Beaumont (INRIA – Bordeaux, FR)

License © Creative Commons BY 3.0 Unported license
© Olivier Beaumont

Joint work of Beaumont, Olivier; Marchal, Loris

Main reference O. Beaumont, L. Marchal, “What Makes Affinity-Based Schedulers So Efficient?,” hal-00875487, 2013.

URL <http://hal.inria.fr/hal-00875487>

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

3.4 Energy-aware scheduling

Anne Benoit (ENS – Lyon, FR)

License © Creative Commons BY 3.0 Unported license
© Anne Benoit

Joint work of Aupy, Guillaume; Benoit, Anne; Renaud-Goud, Paul; Robert, Yves

In this talk, I survey recent works on energy-efficient scheduling. The goal is to minimize the energy consumption of a schedule, given some performance constraints, for instance a bound on the total execution time. I first revisit the greedy algorithm for independent tasks in this context. Then I present problems accounting for the reliability of a schedule: if a failure may occur, then replication or checkpoint is used to achieve a reliable schedule. The goal remains the same, i.e., minimize the energy consumption under performance constraints.

3.5 Using Runtimes to ease the way to Exascale

George Bosilca (University of Tennessee, US)

License © Creative Commons BY 3.0 Unported license
© George Bosilca

Joint work of Bosilca, G. and Bouteiller, A. and Danalis, A. and Hérault, T. and Luszczek, P. and Dongarra, J.

URL <http://icl.utk.edu/~bosilca/>

Novel system designs with steeply escalating resource counts, burgeoning heterogeneity, and increasing memory accesses unpredictability, calls for novel programming paradigms.

PaRSEC, our answer to this challenge, is a layered approach that provides a clear separation of concerns between architecture, algorithm, and data distribution by allowing the algorithm to be decoupled from the data distribution and the underlying hardware. Developers can focus solely on the algorithmic level without the constraints involved with programming for hardware trends.

References

- 1 Bosilca, G. and Bouteiller, A. and Danalis, A. and Hérault, T. and Luszczek, P. and Don-garra, J., *Dense Linear Algebra on Distributed Heterogeneous Hardware with a Symbolic DAG Approach*, in *Scalable Computing and Communications: Theory and Practice*, John Wiley & Sons, ISBN 978-1-118-16265-1, March 2013

3.6 Enabling the production deployment of advanced Fault Tolerance techniques: Fault Tolerant MPI

Aurélien Bouteiller (University of Tennessee, US)

License © Creative Commons BY 3.0 Unported license
© Aurélien Bouteiller

As Supercomputers are entering an era of massive parallelism where the frequency of faults is increasing, the MPI Standard remains distressingly vague on the consequence of failures on MPI communications. Advanced fault tolerance techniques have the potential to prevent full scale application restart and therefore lower the cost incurred for each failure, but they demand from MPI the capability to detect failures and resume communications afterward. In this talk, we present a set of extensions to MPI that allow restoring communication capabilities, while maintaining the extreme level of performance to which MPI users have become accustomed. The general design is to avoid impacting existing interface behavior, and constrain recovery actions to a limited number of new interfaces. The performance impact (or lack thereof) on MPI communication performance is outlined before we use the Wang-Landau protein folding application as an example to illustrate how to employ the new interfaces to effectively express advanced fault tolerance techniques.

3.7 Scheduling Data Sensor Retrieval for Boolean Tree Query Processing

Henri Casanova (University of Hawaii at Manoa, US)

License © Creative Commons BY 3.0 Unported license
© Henri Casanova

The processing of queries expressed as trees of boolean operators applied to predicates on sensor data streams has several applications in mobile computing. Sensor data must be retrieved from the sensors, which incurs a cost, e.g., an energy expense that depletes the battery of the mobile query processing device. The objective is to determine the order in which predicates should be evaluated so as to shortcut part of the query evaluation and minimize the expected cost. This problem has been studied assuming that each data stream occurs at a single predicate. In this work we remove this assumption since it does not necessarily hold for real-world queries. Our main results are a novel optimal algorithm for

single-level trees and a proof of NP-completeness for DNF trees. For DNF trees, however, we show that there is an optimal predicate evaluation order that corresponds to a depth-first traversal. This result provides inspiration for a class of heuristics. We show that one of these heuristics largely outperforms other sensible heuristics, including a heuristic proposed in previous work.

3.8 Does the cache matter? When? Why? and How Much?

Anthony Danalis (University of Tennessee, US)

License  Creative Commons BY 3.0 Unported license
© Anthony Danalis

It is a well known fact that the cache hierarchy can have a profound effect on the performance of codes with heavy memory usage. However, it seems that several misconceptions regarding the effects of the cache on application performance are deeply rooted in the community. In the first part of this talk I will examine the effects of the cache hierarchy on two extreme cases of applications, in terms of memory access patterns. On one extreme we have memory bound codes with no, or little data reuse, such as simple buffer traversals. On the other extreme we have codes with heavy computation and good reuse patterns, such as matrix- matrix multiply. Using experimental results, I will first demonstrate that the cache is neither very easy to utilize, nor very easy to trick into pathological behavior. In the remainder of the talk I will use parallel codes to examine the interaction between the cache hierarchy and the remainder of the memory subsystem – i.e., NUMA memory bus. The algorithm used in this part of the talk will be matrix-matrix multiply and specifically the highly sophisticated implementation that exists in the Intel MKL library. Using experimental results collected from custom parallel benchmarks built on top of this matrix multiply kernel I will demonstrate that the dominant factor affecting the performance of such parallel codes is not the cache friendliness of the scheduling, or the size of the data with respect to the cache hierarchy, but rather the load that the scheduling is putting on the memory bus and the cross-traffic generated.

3.9 Comparison of push and pull strategies in server farms

Bruno Gaujal (INRIA Rhône-Alpes, FR)

License  Creative Commons BY 3.0 Unported license
© Bruno Gaujal

Joint work of Gast, Nicolas; Gaujal, Bruno;
Main reference N. Gast, B. Gaujal, “Markov chains with discontinuous drifts have differential inclusion limits,”
Performance Evaluation, 69(12):623–642, 2012.
URL <http://dx.doi.org/10.1016/j.peva.2012.07.003>

Let us consider a model of a server farm composed of N identical servers. Jobs arrive according to a Poisson process with rate Nr and have a size exponentially distributed with mean 1. Each server can buffer up to B jobs.

We consider two strategies that improve load balancing:

- pull strategy – we add a centralized server that serves jobs at rate Np . It chooses to serve a job from the cluster with longest queue first. For fair comparison, we consider that the total computing capacity remains N (the new speed of the servers is set to $1 - p$). This model is similar to the model studied in by Tsitsiklis & Xu in 2011.
- push strategy – with probability q , a job is pushed to the cluster with the shortest queue. With probability $1 - q$, it is routed to a server at random.

We use an asymptotic model by letting N go to infinity (mean field approximation) to measure the performance of the system under the three strategies (push, pull or none).

When the load of the system approaches one (r goes to one), the expected response time of jobs for the fully distributed system (with no pull nor push) verifies: $E[W] = O(1/(1-r))$.

With an arbitrary small degree of pull ($p > 0$), then the response times are drastically smaller: $E[W] = O(\log 1/(1-r))$.

With a small degree of push, ($q > 0$), the expected response times remain bounded: $E[W] < 1/q$.

3.10 Exascale techniques for Numerics for PDEs (Part II)

Dominik Goeddeke (TU Dortmund, DE)

License © Creative Commons BY 3.0 Unported license
© Dominik Goeddeke

We motivate the concept of hardware-oriented numerics to address the challenges imposed on the simulation of real-life phenomena by increasingly heterogeneous and parallel hardware: Only by integrating hardware aspects directly in the design of novel numerical solution methods for PDEs, the conflicting goals of scalability and total efficiency can be reached. Our examples include CFD problems, low-power architectures, exposing parallelism and resilience.

3.11 Sparse QR factorization for multicore architectures over runtime systems

Abdou Guermouche (University of Bordeaux, FR)

License © Creative Commons BY 3.0 Unported license
© Abdou Guermouche

Joint work of E. Agullo, A. Buttari, A. Guermouche and F. Lopez

Main reference E. Agullo, A. Buttari, A. Guermouche, F. Lopez, “Multifrontal QR Factorization for Multicore Architectures over Runtime Systems,” in Proc. of the 19th Int’l Conf. on Parallel Processing (Euro-Par’13), LNCS, Vol. 8097, pp. 521–532, Springer, 2013.

URL http://dx.doi.org/10.1007/978-3-642-40047-6_53

To face the advent of multicore processors and the ever increasing complexity of hardware architectures, programming models based on DAG parallelism regained popularity in the high performance, scientific computing community. Modern runtime systems offer a programming interface that complies with this paradigm and powerful engines for scheduling the tasks into which the application is decomposed. These tools have already proved their effectiveness on a number of dense linear algebra applications. This paper evaluates the usability of runtime systems for complex applications, namely, sparse matrix multifrontal factorizations which constitute extremely irregular workloads, with tasks of different granularities and characteristics and with a variable memory consumption. Experimental results on real-life matrices show that it is possible to achieve the same efficiency as with an ad hoc scheduler which relies on the knowledge of the algorithm. A detailed analysis shows the performance behavior of the resulting code and possible ways of improving the effectiveness of runtime systems.

3.12 Unified Model for Assessing Checkpointing Protocols at Extreme-Scale

Amina Guermouche (University of Versailles, FR)

License © Creative Commons BY 3.0 Unported license
© Amina Guermouche

This talk presents 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.

3.13 ABFT & Checkpoint-Composite Strategy

Thomas Héroult (University of Tennessee, US)

License © Creative Commons BY 3.0 Unported license
© Thomas Héroult

Joint work of Héroult, Thomas; Bosilca, George; Bouteiller, Aurélien; Robert, Yves; Dongarra, Jack
Main reference G. Bosilca, A. Bouteiller, T. Héroult, Y. Robert, J. Dongarra, “Assessing the impact of ABFT and checkpoint composite strategies,” Research Report ICL-UT-13-03, University of Tennessee, 2013.
URL http://icl.cs.utk.edu/news_pub/submissions/brainstorm.pdf

Algorithm-specific fault tolerant approaches promise unparalleled scalability and performance in failure-prone environments. With the advances in the theoretical and practical understanding of algorithmic traits enabling such approaches, a growing number of frequently used algorithms (including all widely used factorization kernels) have been proven capable of such properties. These algorithms provide a temporal section of the execution when the data is protected by its own intrinsic properties, and can be algorithmically recomputed without the need of checkpoints. However, while typical scientific applications spend a significant fraction of their execution time in library calls that can be ABFT-protected, they interleave sections that are difficult or even impossible to protect with ABFT. As a consequence, the only fault-tolerance approach that is currently used for these applications is checkpoint/restart. In this presentation, I presented an algorithm, a model and a simulator to investigate the behavior of a composite protocol, that alternates between ABFT and checkpoint/restart protection for effective protection of each phase of an iterative application composed of ABFT-aware and ABFT-unaware sections. In this work, we highlight this approach drastically increases the performance delivered by the system, especially at scale, by providing means to rarefy the checkpoints while simultaneously decreasing the volume of data needed to be saved in the checkpoints.

3.14 Tree traversals with task-memory affinities

Julien Herrmann (ENS – Lyon, FR)

License © Creative Commons BY 3.0 Unported license
© Julien Herrmann

Joint work of Herrmann, Julien; Robert, Yves; Marchal, Loris

Main reference J. Herrmann, Y. Robert, L. Marchal, “Model and Complexity Results for Tree Traversals on Hybrid Platforms,” in Proc. of the 19th Int’l Conf. on Parallel Processing (Euro-Par’13), LNCS, Vol. 8097, pp. 647–658, Springer, 2013.

URL http://dx.doi.org/10.1007/978-3-642-40047-6_65

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

3.15 On the Scalability of Moldable Task Scheduling Algorithms

Sascha Hunold (TU Wien, AT)

License © Creative Commons BY 3.0 Unported license
© Sascha Hunold

Parallel task scheduling is now more relevant than ever, especially under the moldable task model, in which tasks are allocated a fixed number of processors before execution. A common assumption for scheduling algorithms is that the speedup function of moldable tasks is either non-decreasing, sub-linear or concave. However, in practice the resulting speedup of parallel programs on current hardware with deep memory hierarchies is most often neither non-decreasing nor concave. We present an algorithm for scheduling moldable tasks with precedence constraints for the makespan objective and arbitrary speedup functions. We show through simulation that the algorithm not only creates competitive schedules for moldable tasks with arbitrary speedup functions, but also outperforms other published heuristics and approximation algorithms for non-decreasing speedup functions.

3.16 Communication and Topology-aware Load Balancing in Charm++ with TreeMatch

Emmanuel Jeannot (INRIA – Bordeaux, FR)

License © Creative Commons BY 3.0 Unported license
© Emmanuel Jeannot

Joint work of Jeannot, Emmanuel; Meneses-Rojas, Esteban; Mercier, Guillaume; Tessier, François; Zheng, Gengbin

Main reference E. Jeannot, E. Meneses, G. Mercier, F. Tessier, G. Zheng, “Communication and Topology-aware Load Balancing in Charm++ with TreeMatch,” in Proc. of IEEE Cluster 2013, IEEE, 2013.

Programming multicore or manycore architectures is a hard challenge particularly if one wants to fully take advantage of their computing power. Moreover, a hierarchical topology implies that communication performance is heterogeneous and this characteristic should also be exploited. We developed two load balancers for Charm++ that take into account both aspects depending on the fact that the application is compute-bound or communication-bound. This work is based on our TREEMATCH library that compute process placement in order to reduce an application communication cost based on the hardware topology. We show that the proposed load-balancing scheme manages to improve the execution times for the two classes of parallel applications.

3.17 Many-task computing for the rest of us

Thilo Kielmann (Vrije Universiteit Amsterdam, NL)

License © Creative Commons BY 3.0 Unported license
© Thilo Kielmann

Joint work of Oprescu, Ana-Maria Oprescu; Kielmann, Thilo; Leahu, Haralambie

Main reference A.-M. Oprescu, T. Kielmann, H. Leahu, “Budget Estimation and Control for Bag-of-Tasks Scheduling in Clouds,” Parallel Processing Letters, Vol. 21, No. 2, pp. 219–243, June 2011.

URL <http://dx.doi.org/10.1142/S0129626411000175>

Exascale machines come with unprecedented challenges for applications that wish to exploit their potential. Efficient communication for tightly-coupled codes, energy efficiency, and fault tolerance are the ones that come to ones mind immediately. Whereas exascale machines still have to be built, there are already large-scale machines in daily commercial exploitation: the data centers of large cloud providers such as Google and Amazon. Can we learn something from these data centers for the exploitation of exascale supercomputers?

In this talk I outline the many-task computing (MTC) model that strives to bridge the gap between static, tightly-coupled applications (as with MPI) and the embarrassingly parallel model of high-throughput computing (as with Condor and BOINC). With MTC, individual tasks (both sequential and parallel) are orchestrated to form an application while typically communicating through shared file systems. I outline the task farming service of our ConPaaS platform (www.conpaas.eu) that, as an example of MTC, implements bag-of-tasks execution on cloud infrastructures, minimizing the overall makespan while respecting a user-selected monetary budget. It does so by sampling the bag of tasks on different cloud machine types, and by predicting the overall makespan and cost for different combinations of instances from available machine types.

The possibly provocative hypothesis of this talk is the applicability of such an execution model for exascale machines, as it provides a loosely-coupled, scalable, and fault-tolerant execution model, while introducing a financial incentive for application users to only use as many resources as their applications really need. Constructing a price model based on

energy consumption (and capabilities) of nodes in a supercomputer should give incentive to resource-conscious use of such supercomputers.

3.18 On Variants of the Hierarchically Structured Bin Packing Problem

Thomas Lambert (ENS – Lyon, FR)

License © Creative Commons BY 3.0 Unported license
© Thomas Lambert

Joint work of Lambert, Thomas ; Marchal, Loris; Ucar, Bora

We deal with a variant of the Bin Packing problem: the Hierarchically Structured Bin Packing. The inputs of this problem are a tree and an integer B and the output is a packing of the leaves of the tree into bins such that every bin has at most B leaves. The objective function is to minimize the total dispersal number. This number is equal to the sum, over all internal nodes, of the number of bins into which the descendants of an internal node are packed. We investigate four variants of this problem. We discuss inapproximability results, an exact algorithm with polynomial time complexity for constant arity trees, and simulation results. At the end, we identify a new variant of our problem with two trees and present two inapproximability results.

3.19 Hierarchical Approach to Improve Performance of Legacy Scientific Applications on Large-Scale Platforms

Alexey Lastovetsky (University College Dublin, IE)

License © Creative Commons BY 3.0 Unported license
© Alexey Lastovetsky

Joint work of Quintin, Jean-Noel; Hasanov, Khalid; Lastovetsky, Alexey;

Main reference J.-N. Quintin, K. Hasanov, A. Lastovetsky, “Hierarchical Parallel Matrix Multiplication on Large-Scale Distributed Memory Platforms,” arXiv:1306.4161v1 [cs.DC], 2013.

URL <http://arxiv.org/abs/1306.4161v1>

Many state-of-the-art parallel algorithms, which are widely used in scientific applications executed on high-end computing systems, were designed in 20th century with relatively small-scale parallelism in mind. Indeed, while in 1990s a system with few hundred cores was considered a powerful supercomputer, modern top supercomputers have millions of cores. In this talk, we present a hierarchical approach to optimization of message-passing parallel algorithms for execution on large-scale distributed-memory systems. The idea is to reduce the communication cost by introducing hierarchy and hence more parallelism in the communication scheme. We apply this approach to SUMMA, the state-of-the-art parallel algorithm for matrix-matrix multiplication, and demonstrate both theoretically and experimentally that the modified Hierarchical SUMMA significantly improves the communication cost and the overall performance on large-scale platforms.

3.20 A greedy algorithm for optimally pipelining a reduction

Bradley Lowery (University of Colorado, US)

License © Creative Commons BY 3.0 Unported license
© Bradley Lowery

Joint work of Lowery, Bradley; Langou, Julien;

Main reference B. Lowery, J. Langou, “A greedy algorithm for optimally pipelining a reduction,” arXiv:1310.4645v1 [cs.DC], 2013.

URL <http://arxiv.org/abs/1310.4645v1>

Collective communications are ubiquitous in parallel applications. We present two new algorithms for performing a reduction. The operation associated with our reduction needs to be associative and commutative. The two algorithms are developed under two different communication models (unidirectional and bidirectional). Both algorithms use a greedy scheduling scheme. For a unidirectional, fully connected network, we prove that our greedy algorithm is optimal when some realistic assumptions are respected. Previous algorithms fit the same assumptions and are only appropriate for some given configurations. Our algorithm is optimal for all configurations. We note that there are some configuration where our greedy algorithm significantly outperform any existing algorithms. This result represents a contribution to the state-of-the art. For a bidirectional, fully connected network, we present a different greedy algorithm. We verify by experimental simulations that our algorithm matches the time complexity of an optimal broadcast (with addition of the computation). Beside reversing an optimal broadcast algorithm, the greedy algorithm is the first known reduction algorithm to experimentally attain this time complexity. Simulations show that this greedy algorithm performs well in practice, outperforming any state-of-the-art reduction algorithms. Positive experiments on a parallel distributed machine are also presented.

References

- 1 Bradley Lowery and Julien Langou, “A greedy algorithm for optimally pipelining a reduction,” arXiv:1310.4645, 2013.

3.21 Efficient Bi-objective Optimization of Energy and Makespan for Exascale Heterogeneous Computing Systems

Anthony A. Maciejewski (Colorado State University, US)

License © Creative Commons BY 3.0 Unported license
© Anthony A. Maciejewski

Joint work of Tarplee, Kyle M.; Friese, Ryan; Maciejewski, Anthony A.; Siegel, Howard Jay

Main reference K. Tarplee, R. Friese, A. A. Maciejewski, H. J. Siegel, “Efficient and Scalable Computation of the Energy and Makespan Pareto Front for Heterogeneous Computing Systems,” in Proc. of the 6th Workshop on Computational Optimization (WCO’13)/Proceedings of the 2013 Federated Conf. on Computer Science and Information Systems (FedCSIS’13), pp. 401–408, Krakow, Poland, Sept. 8–11, 2013.

URL <http://www.engr.colostate.edu/~aam/pdf/conferences/152.pdf>

Future exascale systems will have to simultaneously reduce energy consumption while they strive to maximize performance. Because these are conflicting objectives, system administrators will need to evaluate the tradeoffs of different potential resource allocations. In this work we present a novel algorithm that provides an efficient method for computing the Pareto front of optimal solutions to the bi-objective problem of minimizing energy and makespan for a bag of tasks allocated to a set of heterogeneous compute resources.

3.22 Scheduling tree-shaped task graphs to minimize memory and makespan

Loris Marchal (ENS – Lyon, FR)

License © Creative Commons BY 3.0 Unported license
© Loris Marchal

Joint work of Eyraud-Dubois, Lionel; Marchal, Loris; Sinnen, Oliver; Vivien, Frédéric

Main reference L. Marchal, O. Sinnen, F. Vivien, “Scheduling Tree-Shaped Task Graphs to Minimize Memory and Makespan,” IPDPS 2013: 839–850; available as pre-print: arXiv:1210.2580v1 [cs.DC], 2012.

URL <http://arxiv.org/abs/1210.2580v1>

This work investigates the execution of tree-shaped task graphs using multiple processors. Each edge of such a tree represents a large data. A task can only be executed if all input and output data fit into memory. Such trees arise 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 problem of finding the tree traversal with minimum required memory was well studied and optimal polynomial algorithms has been proposed. Here, we extend the problem by considering multiple processors. With the multiple processors comes the additional objective to minimize the makespan. Not surprisingly, this problem proves to be much harder. We study its computational complexity and provide an inapproximability result even for unit weight trees. Several heuristics are proposed, especially for the realistic problem of minimizing the makespan under a strong memory constraint. They are analyzed in an extensive experimental evaluation using realistic trees.

3.23 Liveness Evaluation for Cyclo-Static DataFlow Graphs

Alix Munier (UPMC – Paris, FR)

License © Creative Commons BY 3.0 Unported license
© Alix Munier

Joint work of Nenazouz, Mohamed; Bodin, Bryon; Hujsa, Thomas; Munier-Kordon, Alix

Main reference M. Benazouz, B. Bodin, T. Hujsa, A. Munier-Kordon, “Liveness evaluation of a cyclo-static DataFlow graph,” in Proc. of the 50th Design Automation Conf. (DAC’13), ACM, 2013.

URL <http://dx.doi.org/10.1145/2463209.2488736>

Static DataFlow Graphs (CSDFG in short) is a formalism commonly used to model parallel applications composed by actors communicating through buffers. The liveness of a CSDFG ensures that all actors can be executed infinitely often. This property is clearly fundamental for the design of embedded applications. This talk aims to present first an original sufficient condition of liveness for a CSDFG. Two algorithms of polynomial-time for checking the liveness are then derived and compared to a symbolic execution of the graph. The performance of our methods are comparable to those existing in the literature for industrial applications. However, they are far more effective on randomly generated instances, ensuring their scalability for future more complex applications and their possible implementation in a compiler.

3.24 Planning for ExaScale Systems: The Challenge to be Prepared

Wolfgang E. Nagel (TU Dresden, DE)

License © Creative Commons BY 3.0 Unported license
© Wolfgang E. Nagel

Joint work of Nagel, Wolfgang E.; Hackenberg, Daniel; Juckeland, Guido; Brunst, Holger; Bungartz, Hans-Joachim

ExaScale systems are expected to arrive between 2018 and 2020, and they will have lots of cores, may be arranged in a heterogeneous way. The challenge will be to have programs that scale to hundreds of millions of parallel threads, and runtime environments that will efficiently support such a high level of parallelism with load imbalances, many parallel I/O requests, possibly heterogeneous cores and more often than today failing hardware components. There will be strong need to have software ready which provide solutions in such complex environments, and, at least in Germany, despite the activities from BMBF (HPC calls) and DFG (SPPEXA) we still invest much more in “racks” and not enough in “brains”.

The talk presents actual developments on the scalability of performance tools and describes research issues, which have to be solved to get solutions even on PFlops systems today. It also shows how algorithm development and tuning can benefit from these achievements. In a case study, results are presented for the code PIconGPU which has been optimized on Titan at ORNL. With more than 7 PFlops (64 Bit) and about 1,5 PFlops (32 Bit) measured on that machine, PIconGPU is one of the six finalists for the Gordon Bell Price Award in 2013.

3.25 Task Scheduling to Extend Platform Useful Life using Prognostics

Jean-Marc Nicod (FEMTO-ST Institute – Besancon, FR)

License © Creative Commons BY 3.0 Unported license
© Jean-Marc Nicod

Joint work of Nicod, Jean-Marc; Obrecht, Léo; Varnier, Christophe
Main reference J.-M. Nicod, L. Obrecht, C. Varnier, “Task scheduling to extend platform useful life using prognostics,” in Proc. of the International IFAC Conference on Manufacturing Modelling, Management and Control (MIM’2013), Russian Federation, 2013.
URL <http://hal.inria.fr/hal-00838457>

In this talk, we study the problem of optimizing the useful life of a heterogeneous distributed platform which has to produce a given production throughput rate. The machines perform independent tasks and may be configured with different profiles involving variable throughput rates. At each time the sum of the machine throughputs that are currently running determine the global throughput of the platform. Moreover, each machine of the platform is supposed to be monitored and a prognostic module gives a remaining useful life (RUL) depending on both its past and future usage (throughput rate). The problem that is tackled in this talk is to configure the platform so as to reach the demand as long as possible.

Our approach consists in discretizing the time in period and a configuration is chosen at the beginning of each period. We prove that the problem can be optimally solved in polynomial time in the homogeneous case and that this problem becomes NP-Complete since the the throughput is heterogeneous on the machine. We propose a Integer Linear Programming (ILP) model to find such a configuration for a fixed time horizon. Due to the ILP, the largest horizon can be computed for small instances of the problem. For larger instances, we propose polynomial time heuristics to reach a horizon. Exhaustive simulations show that the heuristic solutions are close to the optimal in particular case where the optimal

horizon can be computed (5% in average). For other platforms with a very large number of machines, simulations assess the efficiency of our heuristics. The distance to the theoretical maximal value is never up to 20%.

3.26 Energy-aware Execution of Fork-Join-based Task Parallelism

Thomas Rauber (Universität Bayreuth, DE)

License © Creative Commons BY 3.0 Unported license
© Thomas Rauber

Joint work of Rauber, Thomas; Rüniger, Gudula

Main reference T. Rauber, G. Rüniger, “Energy-aware Execution of Fork-Join-based Task Parallelism,” in Proc. of the 20th IEEE Int’l Symp. on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS’12), pp. 231–240, IEEE, 2012.

URL <http://dx.doi.org/10.1109/MASCOTS.2012.35>

In this talk, we use an analytical energy model based on frequency scaling to model the energy consumption of tasks in a fork-join pattern of parallelism. In particular, tasks that may be executed concurrently to each other are considered, and the resulting energy consumption for different processor assignments is investigated. Frequency scaling factors that lead to a minimum energy consumption are derived and used in task-based scheduling algorithms. An experimental evaluation provides simulations for a large number of randomly generated task sets as well as energy measurements on a Intel Sandy Bridge architecture using a complex application from numerical analysis.

3.27 Reliable Service Allocation in Clouds with Memory and Capacity Constraints

Paul Renaud-Goud (INRIA – Bordeaux, FR)

License © Creative Commons BY 3.0 Unported license
© Paul Renaud-Goud


Joint work of Beaumont, Olivier; Eyraud-Dubois, Lionel; Pesneau, Pierre; Renaud-Goud, Paul

Main reference O. Beaumont, L. Eyraud-Dubois, P. Pesneau, P. Renaud-Goud, “Reliable Service Allocation in Clouds with Memory and Capacity Constraints,” in Proc. of the 6th Workshop on Resiliency in High Performance Computing (Resilience) in Clusters, Clouds, and Grids in conjunction with the 19th Int’l European Conf. on Parallel and Distributed Computing (Euro-Par 2013), 2013.

We consider allocation problems that arise in the context of service allocation in Clouds. More specifically, on the one part we assume that each Physical Machine (denoted as PM) is offering resources (memory, CPU, disk, network). On the other part, we assume that each application in the IaaS Cloud comes as a set of services running as Virtual Machines (VMs) on top of the set of PMs. In turn, each service requires a given quantity of each resource on each machine where it runs (memory footprint, CPU, disk, network). Moreover, there exists a Service Level Agreement (SLA) between the Cloud provider and the client that can be expressed as follows: the client requires a minimal number of service instances which must be alive at the end of the day, with a given reliability (that can be converted into penalties paid by the provider). In this context, the goal for the Cloud provider is to find an allocation of VMs onto PMs so as to satisfy, at minimal cost, both capacity and reliability constraints for each service. In this paper, we propose a simple model for reliability constraints and we prove that it is possible to derive efficient heuristics.

3.28 Some thoughts on exascale and scheduling

Rizos Sakellariou (University of Manchester, GB)

License  Creative Commons BY 3.0 Unported license
© Rizos Sakellariou

The talk argues that past ways of thinking about High-Performance-Computing affect our way of viewing exascale computing. However, as progress is not necessarily linear we may need to focus more deeply on the key issues on exascale computing, such as fault-tolerance, communication and energy. The talk makes reference to recent results related to energy-aware resource provisioning [1] and argues that real energy measurements and a more holistic view may be necessary to understand energy consumption.

References

- 1 Ilia Pietri, Maciej Malawski, Gideon Juve, Ewa Deelman, Jarek Nabrzyski, Rizos Sakellariou. Energy Constrained Provisioning for Scientific Workflow Ensembles. In CGC'13 (Proceedings of the 2013 IEEE Third International Conference on Cloud and Green Computing, Karlsruhe, Germany, September 30-October 2), pp. 34-41.

3.29 Parallel Dataflow Graph Coloring

Erik Saule (University of North Carolina at Charlotte, US)

License  Creative Commons BY 3.0 Unported license
© Eric Saule

In this talk, we investigate the shared memory graph coloring problem. Existing methods such as speculative first fit coloring requires to make multiple passes on the graph so as to achieve pleasant parallelism. To reduce the execution to a single pass on the graph, a more synchronized algorithm is required. We present the dataflow coloring algorithm where a vertex v can only be colored once its neighbors $u < v$ have been colored. We show that the execution of this algorithm behaves like the output of a list scheduling algorithm on a DAG obtained by an orientation of the edges of the colored graph. However additional edges are added to the DAG in order to take the effect of the parallel runtime system into account. To solve the problem of bad orientation of the edges which can lead to a high critical path, we show experimentally that picking a randomized reordering of the vertices improves the amount of achieved concurrency.

3.30 Resource Management in an IaaS Environment

Uwe Schwiegelshohn (TU Dortmund, DE)

License  Creative Commons BY 3.0 Unported license
© Uwe Schwiegelshohn

This talk starts by describing constraints for resource management in an Infrastructure-as-a-Service (IaaS) market economy. Then user requests are divided into the categories interactive, user-facing, and batch. Also the slack factor is suggested as a quantitative metric for a service guarantee in this scenario. It is explained that an IaaS provider is particularly interested in optimizing the utilization of his computing resources while satisfying the expectations of

his users. As similar problems have already been addressed in the real-time scheduling area, relevant previous algorithmic results are presented. However, the application of these results in practice does not produce satisfactory results in general. Therefore, a new approach to generate provider offers is defined. This approach considers the previously discussed IaaS constraints and allows an efficient resource management. Theoretical analysis is used to determine the performance of the resulting scheduling methods and their variations. The methods include preemptive deadline scheduling on machines with different speed.

3.31 Stochastic-Based Deadline-Aware and Energy-Constrained Robust Static Resource Management

Howard Jay Siegel (Colorado State University, US)

License © Creative Commons BY 3.0 Unported license
© Howard Jay Siegel

Joint work of Mark Oxley, Sudeep Pasricha, Howard Jay Siegel, and Anthony A. Maciejewski

Main reference Mark Oxley, Sudeep Pasricha, Howard Jay Siegel, and Anthony A. Maciejewski, “Energy and Deadline Constrained Robust Stochastic Static Resource Allocation,” 1st Workshop on Power and Energy Aspects of Computation (PEAC’13), in the Proc. of the 10th Int’l Conference on Parallel Processing and Applied Mathematics (PPAM’13), 2013; to appear.

URL <http://www.engr.colostate.edu/~hj/conferences/338.pdf>

We study the problem of robust static (off-line) resource management for multicore-based heterogeneous parallel computing systems. We consider allocating resources to a known set of independent tasks (the “bag-of- tasks” workload model). We use a stochastic model (random variable) for the execution time of each task on each type of core in the system. Our goal is to design static resource management techniques that maximize the probability of finishing each task by a common deadline (our robustness metric) while ensuring the resource allocation meets a constraint on the probability of obeying a given energy budget. We describe our design and evaluation of a new resource allocation heuristic based on Tabu Search, where we exploit heterogeneity and employ dynamic voltage and frequency scaling (DVFS) by incorporating problem-domain specific local search techniques.

3.32 Tools and Resources for Task Scheduling Research

Oliver Sinnén (University of Auckland, NZ)

License © Creative Commons BY 3.0 Unported license
© Oliver Sinnén

Scheduling of task graphs (DAGs) on parallel systems with communication delay ($P|prec, c_{ij}|C_{max}$) is a classical scheduling problem. It is NP-hard in the strong sense and part of many computing problems. Given its importance, there is still a strong interest of the research community in this problem. In this talk tools and resources for research of this problem are presented. This includes a visual tool for the analysis and manual manipulation of schedules, optimal solvers for small to medium sized problem instances, and a database of optimal schedules for a large set of task graphs on different numbers of processors. The tools and resources are available at <http://www.ece.auckland.ac.nz/~parallel/OptimalTaskScheduling/>.

3.33 Job Scheduling Using Successive Linear Approximations of a Sparse Model

Veronika Sonigo (University of Franche-Comté – Besancon, FR)

License © Creative Commons BY 3.0 Unported license
© Veronika Sonigo

Joint work of Rehn-Sonigo, Veronika; Chrétien, Stéphane; Nicod, Jean Marc; Toch, Lamiel; Philippe, Laurent;
Main reference S. Chrétien, J.-M. Nicod, L. Philippe, V. Rehn-Sonigo, L. Toch, “Job Scheduling Using Successive Linear Programming Approximations of a Sparse Model,” in Proc. of the 18th Int’l Conf. on Parallel Processing (Euro-Par’12), LNCS, Vol. 7484, pp. 116–127, Springer, 2012.

URL http://dx.doi.org/10.1007/978-3-642-32820-6_14

URL <http://hal.inria.fr/hal-00789217>

In this talk we tackle the well-known problem of scheduling a collection of parallel jobs on a set of processors either in a cluster or in a multiprocessor computer. For the makespan objective, i.e., the completion time of the last job, this problem has been shown to be NP-Hard and several heuristics have already been proposed to minimize the execution time. We introduce a novel approach based on successive linear programming (LP) approximations of a sparse model. The idea is to relax an integer linear program and use lp norm-based operators to force the solver to find almost-integer solutions that can be assimilated to an integer solution. We consider the case where jobs are either rigid or moldable. A rigid parallel job is performed with a predefined number of processors while a moldable job can define the number of processors that it is using just before it starts its execution. We compare the scheduling approach with the classic Largest Task First list based algorithm and we show that our approach provides good results for small instances of the problem. The contributions of this paper are both the integration of mathematical methods in the scheduling world and the design of a promising approach which gives good results for scheduling problems with less than a hundred processors.

3.34 Exascale techniques for Numerics for PDEs (Part I)

Stefan Turek (TU Dortmund, DE)

License © Creative Commons BY 3.0 Unported license
© Stefan Turek

We motivate the concept of hardware-oriented numerics to address the challenges imposed on the simulation of real-life phenomena by increasingly heterogeneous and parallel hardware: Only by integrating hardware aspects directly in the design of novel numerical solution methods for PDEs, the conflicting goals of scalability and total efficiency can be reached. Our examples include CFD problems, low-power architectures, exposing parallelism and resilience.

3.35 Semi-matching algorithms for scheduling parallel tasks under resource constraints

Bora Ucar (ENS – Lyon, FR)

License © Creative Commons BY 3.0 Unported license
© Bora Ucar

Joint work of

Main reference A. Benoit, J. Langguth, B. Ucar, “Semi-matching algorithms for scheduling parallel tasks under resource constraints,” accepted to be published in IPDPSW 2013.

URL <http://hal.inria.fr/hal-00738393>

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 (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.

3.36 Checkpointing and fault prediction

Frédéric Vivien (ENS – Lyon, FR)

License © Creative Commons BY 3.0 Unported license
© Frédéric Vivien

Joint work of Aupy, Guillaume; Robert, Yves; Vivien, Frédéric; Zaidouni, Dounia

Main reference G. Aupy, Y. Robert, F. Vivien, D. Zaidouni, “Checkpointing strategies with prediction windows,” in Proc. of the IEEE Pacific Rim International Symposium on Dependable Computing (PRDC’13), 2013.

URL <http://hal.inria.fr/hal-00847622>

This work deals with the impact of fault prediction techniques on checkpointing strategies. We consider fault-prediction systems that do not provide exact prediction dates, but instead time intervals during which faults are predicted to strike. These intervals dramatically complicate the analysis of the checkpointing strategies. We propose a new approach based upon two periodic modes, a regular mode outside prediction windows, and a proactive mode inside prediction windows, whenever the size of these windows is large enough. We are able to compute the best period for any size of the prediction windows, thereby deriving the scheduling strategy that minimizes platform waste. In addition, the results of the analytical study are nicely corroborated by a comprehensive set of simulations, which demonstrate the validity of the model and the accuracy of the approach.

4 Open Problems

A short “Open Problems” session was held on the Thursday morning. Three open problems were introduced by three participants. The first two problems were theoretical in nature, i.e.,

mathematical problems directly connected to simple yet fundamental scheduling problems. Interestingly, one of these two problems was actually solved by yet another workshop participant that evening. The other problems received some attention but no solution was achieved and it is not clear that such a solution can be computed. The third problem was much larger in scope and targeted to the resource management models in cloud computing infrastructures. In other words, given that current resource abstractions provided by cloud infrastructures can be vastly sub-optimal especially given the heterogeneity of the underlying physical infrastructure, which new abstractions should be used and how? This problem led to a lively discussion that, it is fair to say, left the group with more (interesting) questions than answers.

Participants

- Ismail Assayad
University Hassan II –
Casablanca, MA
- Guillaume Aupy
ENS – Lyon, FR
- Olivier Beaumont
INRIA – Bordeaux, FR
- Anne Benoit
ENS – Lyon, FR
- George Bosilca
University of Tennessee, US
- Aurélien Bouteiller
University of Tennessee, US
- Heinrich Braun
SAP AG – Walldorf, DE
- Henri Casanova
Univ. of Hawaii at Manoa, US
- Anthony Danalis
University of Tennessee, US
- Carsten Franke
ABB – Baden-Dättwil, CH
- Bruno Gaujal
INRIA Rhône-Alpes, FR
- Dominik Göddeke
TU Dortmund, DE
- Christian Grimme
Universität Münster, DE
- Abdou Guermouche
University of Bordeaux, FR
- Amina Guermouche
University of Versailles, FR
- Thomas Hérault
University of Tennessee, US
- Julien Herrmann
ENS – Lyon, FR
- Sascha Humold
TU Wien, AT
- Emmanuel Jeannot
INRIA – Bordeaux, FR
- Thilo Kielmann
Vrije Univ. Amsterdam, NL
- Thomas Lambert
ENS – Lyon, FR
- Alexey Lastovetsky
University College Dublin, IE
- Bradley Lowery
University of Colorado, US
- Anthony A. Maciejewski
Colorado State University, US
- Loris Marchal
ENS – Lyon, FR
- Alix Munier
UPMC – Paris, FR
- Wolfgang E. Nagel
TU Dresden, DE
- Jean-Marc Nicod
FEMTO-ST Institute –
Besancon, FR
- Thomas Rauber
Universität Bayreuth, DE
- Paul Renaud-Goud
INRIA – Bordeaux, FR
- Yves Robert
ENS – Lyon, FR
- Gudula Rünger
TU Chemnitz, DE
- Rizos Sakellariou
University of Manchester, GB
- Erik Saule
University of North Carolina at
Charlotte, US
- Uwe Schwiegelshohn
TU Dortmund, DE
- Howard Jay Siegel
Colorado State University, US
- Oliver Sinnen
University of Auckland, NZ
- Veronika Sonigo
University of Franche-Comté –
Besancon, FR
- Stefan Turek
TU Dortmund, DE
- Bora Ucar
ENS – Lyon, FR
- Frédéric Vivien
ENS – Lyon, FR

