INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Team AlGorille

Algorithms for the Grid

Lorraine

THEME NUM

Table of contents

1. Team

Team Leader
Jens Gustedt [research director, INRIA]

Administrative Assistant
Josiane Reffort [UHP]

Staff Members
Emmanuel Jeannot [Maître de Conférences, UHP, until 31/08/05. Temporary assignment as Chargé de Recherche, INRIA, since then]
Martin Quinson [Maître de Conférences, UHP/ESIAL, since 01/02/05]
Frédéric Suter [Maître de Conférences, UHP]

External Collaborator
Stéphane Vialle [professor, Supélec Metz Campus]

Engineer
Xavier Delaruelle [Ingénieur associé, INRIA, since 01/10/05]

Teaching Assistant
Luiz Angelo Barchet-Esteffenel [Nancy 2, since 01/09/05]
Pierre-François Dutot [UHP/ESIAL, since 01/09/05]
Flavien Vernier [UHP, since 01/09/05]

Ph. D. Students
Frédéric Wagner [joint regional/INRIA grant until 30/8/05]
Tchimou N'Takpé [joint regional/Ivorian goverment grant since 01/10/05]

Student Intern
Tchimou N'Takpé [INPL/ENSEM]
Gaurav Goel [IIT, Dehli, joint regional/INRIA grant]
Louis-Claude Canon [ESEO, Angers]

2. Overall Objectives

2.1. Overall Objectives

Keywords: *Grid computing, algorithms, data redistribution, data distribution, experiments, parallel and distributed computing, scheduling.*

The possible access to distributed computing resources over the Internet allows a new type of applications that use the power of the machines and the network. The transparent and efficient access to these distributed resources that form *the Grid* is one of the major challenges of information technology. It needs the implementation of specific techniques and algorithms to make computers communicate with each other, let applications work together, allocate resources and improve the quality of service and the security of the transactions.

Challenge: We tackle several problems related to the first of the major challenges that INRIA has identified in its strategic plan:
Design and master the future network infrastructures and communication services platforms.

Originality: Our approach emphasizes on *algorithmic aspects* of grid computing, in particular it addresses the problems of organizing the computation *efficiently*, be it on the side of a service provider, be it within the application program of a customer.

Research themes:

- Structuring of applications for scalability: modeling of size, locality and granularity of computation and data.

– Transparent resource management: sequential and parallel task scheduling; migration of computations; data exchange; distribution and redistribution of data.
– Experimental validation: reproducibility, extendability and applicability of simulations, emulations and *in situ* experiments.

Methods: Our methodology is based upon three points (1) modeling, (2) design and (3) engineering of algorithms. These three points interact strongly to form a feedback loop.

1. With models we obtain an abstraction of the physical, technical or social reality.
2. This abstraction allows us to design techniques for the resolution of specific problems.
3. These techniques are implemented to validate the models with experiments and by applying them to real world problems.

3. Scientific Foundations

3.1. Structuring of Applications for Scalability

Keywords: *message passing*, *models for parallel and distributed computing*, *performance evaluation*, *shared memory*.

Participants: Jens Gustedt, Frédéric Suter, Stéphane Vialle, Gaurav Goel.

Our approach is based on a "good" separation of the different problem levels that we encounter with Grid problems. Simultaneously this has to ensure a good data locality (a computation will use data that are "close") and a good granularity (the computation is divided into non preemptive tasks of reasonable size). For problems for which there is no natural data parallelism or control parallelism such a division (into data and tasks) is indispensable when tackling the issues related to spatial and temporal distances as we encounter them in the Grid.

Several parallel models offering simplified frameworks that ease the design and the implementation of algorithms have been proposed. The best known of these provide a modeling that is called "*fined grained*", *i.e.*, at the instruction level. Their lack of realism with respect to the existing parallel architectures and their inability to predict the behavior of implementations, has triggered the development of new models that allow a switch to a *coarse grained* paradigm. In the framework of parallel and distributed (but homogeneous) computing they started with the fundamental work of Valiant [39]. Their common characteristics are:

- to maximally exploit the data that is located on a particular node by a local computation,
- to collect all requests for other nodes during the computation, and
- to only transmit these requests if the computation can't progress anymore.

The coarse grained models aim at being realistic with regard to two different aspects: algorithms and architectures. In fact, the coarseness of these models uses the common characteristic of today's parallel settings: the size of the input is orders of magnitude larger than the number of processors that are available. In contrast to the PRAM (Parallel Random Access Machine) model, the coarse grained models are able to integrate the cost of communications between different processors. This allows them to give realistic predictions about the overall execution time of a parallel program. As examples we refer to BSP (Bulk Synchronous Parallel model) [39], LogP (Latency overhead gap Procs) [33], CGM (Coarse Grained Multicomputer) [35] and *PRO* (Parallel Resource Optimal Model) [27].

The assumptions on the architecture are very similar: p homogeneous processors with local memory distributed on a point-to-point interconnection network. They also have similar models for program execution that are based on *supersteps*; an alternation of computation and communication phases. For the algorithmics,

this takes the distribution of the data on the different processors into account. But, all the mentioned models do not allow the design of algorithms for the Grid since they all assume homogeneity, for the processors as well as for the interconnection network.

Our approach is algorithmic. We try to provide a modeling of a computation on grids that allows an easy design of algorithms and realistic performing implementations. Even if there are problems for which the existing sequential algorithms may be easily parallelized, an extension to other more complex problems such as computing on large discrete structures (*e.g.*, web graphs or social networks) is desirable. Such an extension will only be possible if we accept a paradigm change. We have to explicitly decompose data and tasks.

We are convinced that this new paradigm should:

1. be guided by the idea of **supersteps** (BSP). This is to enforce a concentration of the computation to the local data,
2. ensure an economic use of all available resources.

On the other hand, we have to be careful that the model (and the design of algorithms) remains simple. The number of supersteps and the minimization thereof should by themselves not be a goal. It has to be constraint by other more "*natural*" parameters coming from the architecture and the problem instance.

A first solution that uses (1) to combine these objectives for homogeneous environments has been given in [27] with PRO.

In a complementary approach we have addressed (2) to develop a simple interface that gives a consistent view of the data services that are exported to an application, see [6].

Starting from this model, we try to design high level algorithms for grids. It will be based upon an abstract view of the architecture and as far as possible be independent of the intermediate levels. It aims at being robust with regard to the different hardware constraints and should be sufficiently expressive. The applications for which our approach will be feasible are those that fulfill certain constraints:

- they need a lot of computing power
- they need a lot of data that is distributed upon several resources, or,
- they need a lot of temporary storage exceeding the capacity of a single machine.

To become useful on grids, coarse grained models (and the algorithms designed for them) must first of all overcome a principle constraint: the assumption of homogeneity of the processors and connections. The long term goal should be arbitrarily mixed architectures but it would not be realistic to assume to be able to achieve this in one step.

3.2. Transparent Resource Management

Keywords: *approximating algorithms, data redistribution, parallel and distributed computing, scheduling.*

Participants: Louis-Claude Canon, Emmanuel Jeannot, Frédéric Suter, Frédéric Wagner.

We think of the future Grid as of a medium to access resources. This access has to be as transparent as possible to a user of such a Grid and the management of these resources has not to be imposed to him/her, but entirely done by a "system", so called middleware. This middleware has to be able to manage all resources in a satisfactory way. Currently, numerous algorithmic problems hinder such an efficient resource management and thus the transparent use of the Grid.

By their nature, distributed applications use different types of resources; the most important being these of computing power and network connections. The management and optimization of those resources is essential for networking and computing on Grids. This optimization may be necessary at the level of the computation of the application, of the organization of the underlying interconnection network or for the organization of the messages between the different parts of the application. Managing these resources relates to a set of policies to optimize their use and allow an application to be executed under favorable circumstances.

Our approach consists of the tuning of techniques and algorithms for a transparent management of resources, be they data, computations, networks, ...This approach has to be clearly distinguished from others which are more focused on applications and middlewares. We aim at proposing new algorithms (or improve the exiting ones) *for* the resource management in middlewares. Our objective is to provide these algorithms in libraries so that they may be easily integrated. For instance we will propose algorithms to efficiently transfer data (data compression, distribution or redistribution of data) or schedule sequential or parallel tasks.

The problems that we are aiming at solving are quite complex. Therefore they often translate into combinatorial or graph theoretical problems where the identification of an optimal solution is known to be hard. But, the classical measures of complexity (polynomial versus NP-hard) are not very satisfactory for really large problems: even if a problem has a polynomial solution it is often infeasible in reality whereas on the other hand NP-hard problems may allow a quite efficient resolution with results close to optimality.

Consequently it is mandatory to study approximation techniques where the objective is not to impose global optimality constraints but to relax them in favor of a compromise. Thereby we hope to find *good* solutions at a *reasonable* price. But, these can only be useful if we know how to analyze and evaluate them.

3.3. Experimental Validation

Keywords: *applicability, emulations, experiments in situ, extendability, reproductibility, simulations.*

Participants: Louis-Claude Canon, Emmanuel Jeannot, Martin Quinson, Frédéric Wagner.

An important issue for the research on complex systems such as grids is to validate the obtained results. This validation constitutes a scientific challenge by itself since we have to validate models, their adequation to reality and the algorithms that we design inside these models. Whereas mathematical proofs establish soundness within such a context, the overall validation must be done by experiments. A successful experiment shows the validity of both the algorithm and the modeling at the same time. But, if the algorithm does not provide the expected result or performance, this might be due to several factors: a faulty modeling, a weak design, or a bad implementation.

Experimental validation of grid systems is a particularly challenging issue. Such systems will be large, rapidly changing, shared and severely protected. Naive experiments on real platforms will usually not be reproducible, while the extensibility and applicability of simulations and emulations will be very difficult to achieve. These difficulties imply the study phases through modeling, algorithm design, implementation, tests and experiments. The test results will reveal precious for a subsequent modeling phase, complementing the process into a feedback loop.

In addition to this idea of validating the whole (modeling, design and implementation) in our research we are often restricted by a lack of knowledge: the systems that we want to describe might be too complex;

some components or aspects might be unknown or the theoretical investigations might not yet be sufficiently advanced to allow for provable satisfactory solutions of problems.

We think that an experimental validation is a valuable completion of theoretical results on protocol and algorithm behavior. The focus of algorithmic research being upon performance, the main experiments we are concerned with are performance evaluation. To our opinion, such experiments should fulfill the following properties:

reproducibility: Experimental settings must be designed and described such that they are reproducible by others and must give the same result with the same input.

extensibility: A report on a scientific experiment concerning performance of an implementation on a particular system is only of marginal interest if it is simply descriptive and does not point beyond the particular setting in which it is performed. Therefore, the design of an experiment *must* target possible comparisons with passed and (in particular) future work, extensions to more and other processors, larger data sets, different architectures and alike. Several dimensions have to be taken into account: scalability, portability, prediction and realism.

applicability: Performance evaluation should not be a goal *in fine* but should result in concrete predictions of the behavior of programs in the real world. However, as the set of parameters and conditions is potentially infinite, a good experiment campaign must define realistic parameters for platforms, data sets, programs, applications, *etc.* and must allow for an easy calibration.

revisability: When an implementation does not perform as expected, it should be possible to identify the reasons, be they caused by the modeling, the algorithmic design, the particular implementation and/or the experimental environment. Methodologies that help to explain mispredictions and to indicate improvements have to be developed.

4. Application Domains

4.1. High Performance Computing

Participants: Jens Gustedt, Gaurav Goel.

4.1.1. Models and Algorithms for Coarse Grained Computation

With this work we aim at extending the coarse grained modeling (and the resulting algorithms) to hierarchically composed machines such as clusters of clusters or clusters of multiprocessors.

To be usable in a Grid context this modeling has first of all to overcome a principal constraint of the existing models: the idea of an homogeneity of the processors and the interconnection network. Even if the long term goal is to target arbitrary architectures it would not be realistic to think to achieve this directly, but in different steps:

- Hierarchical but homogeneous architectures: these are composed of an homogeneous set of processors (or of the same computing power) interconnected with a non-uniform network or bus which is hierarchic (CC-Numa, clusters of SMPs).
- Hierarchical heterogeneous architectures: there is no established measurable notion of efficiency or speedup. Also most certainly not any arbitrary collection of processors will be useful for computation on the Grid. Our aim is to be able to give a set of concrete indications of how to construct an extensible Grid.

In parallel, we have to work upon the characterization of architecture-robust efficient algorithms, *i.e.*, algorithms that are independent, up to a certain degree, of low-level components or the underlying middleware.

The literature about fine grained parallel algorithms is quite exhaustive. It contains a lot of examples of algorithms that could be translated to our setting, and we will look for systematic descriptions of such a translation.

List ranking, tree contraction and graph coloring algorithms already have been designed following the coarse grained setting given by the model *PRO* [27].

To work in the direction of understanding of what problems might be "*hard*" we tackled a problem that is known to be P-complete in the PRAM/NC framework, but for which not much had been known when only imposing the use of relatively few processors: the *lexicographic first maximal independent set* (LFMIS) problem [7].

We already are able to give a work optimal algorithm in case we have about $\log n$ processors and thus to prove that the NC classification is not necessarily appropriate for today's parallel environments which consist of few processors (up to some thousands) and large amount of data (up to some terabytes).

4.1.2. External Memory Computation

In the mid-nineties several authors [32], [34] developed a connection between two different types of computation models: BSP-like models of parallel computation and IO efficient external memory algorithms. Their main idea is to enforce data locality during the execution of a program by simulating a parallel computation of several processors on one single processor.

Whereas such an approach is convincing on a theoretical level, its efficient and competitive implementation is quite challenging in practice. In particular, it needs software that induces as little computational overhead as possible by itself. Up to now, it seems that this has only been provided by software specialized in IO efficient implementations.

In fact, the stability of our library *SSCRAP*, see Section 5.1, also showed in its extension towards external memory computing [5]. With some relatively small add-ons to *SSCRAP* we were able to provide such a framework. It was tested successfully on some typical hardware, PC with some gigabytes of free disk.

The main add-on that was integrated into *SSCRAP* was a consequent implementation of an abstraction between the *data* of a process execution and the memory of a processor. The programmer acts upon these on two different levels:

- with a sort of *handle* on some data array which is an abstract object that is common to all *SSCRAP* processors.
- with a map of its (local) part of that data into the address space of the *SSCRAP* processor, accessible as a conventional pointer.

Another add-on was the possibility to fix a maximal number of processors (*i.e.*, threads) that should be executed concurrently. With these add-ons, simple environment variables `SSCRAP_MAP_MEMORY` and `SSCRAP_SERIALIZE` allow a runtime control of the program behavior.

4.1.3. Irregular problems

Irregular data structures like sparse graphs and matrices are in wide use in scientific computing and discrete optimization. The importance and the variety of application domains are the main motivation for the study of efficient methods on such type of objects. The main approaches to obtain good results are parallel, distributed and out-of-core computation.

We follow several tracks to tackle irregular problems: automatic parallelization, design of coarse grained algorithms and the extension of these to external memory settings.

In particular we study the possible management of very large graphs, as they occur in reality. Here, the notion of "*networks*" appears twofold: on one side many of these graphs originate from networks that we use or encounter (Internet, Web, peer-to-peer, social networks) and on the other the handling of these graphs has to take place in a distributed Grid environment. The principal techniques to handle these large graphs will be provided by the coarse grained models. With the PRO model [27] and the *SSCRAP* library we already provide tools to better design algorithms (and implement them afterwards) that are adapted to these irregular problems.

In addition we will be able to rely on certain structural properties of the relevant graphs (short diameter, small clustering coefficient, power laws). This will help to design data structures that will have good locality properties and algorithms that compute invariants of these graphs efficiently.

4.2. Evolution of Scheduling Policies and Network Protocols

Participants: Emmanuel Jeannot, Frédéric Suter, Frédéric Wagner.

4.2.1. Scheduling on the Grid

Our work deals with algorithms that allocate applications divided into tasks onto remote compute servers in an client-agent-server model. A good scheduling of these tasks is a primal requirement to achieve good performance.

We have investigated the limits of the greedy algorithm MCT (Minimum Completion Time), used in the NetSolve middleware for instance. To improve it, we introduced the notion of an "*history*" allowing a better prediction of the execution time of a task on a particular server. On the basis of real experimentations, we shown the interest of heuristics relying on the Historical Trace Manager (HTM) to dynamically schedule independent tasks on a grid platform. The HTM is a time-shared predicting module. We have revisited different heuristics when scheduling an application with precedence constraints, or mixed submissions of such constraints and some independent tasks. Many experiments corresponding to many scenarios have been executed on a real testbed and present a large gain on the *makespan*, the *sumflow* and on the quality of service over the MCT heuristic. Moreover, we study the accuracy of the HTM, observed from all undertaken experiments. We show that the HTM is able to provide very accurate and useful information, and allows a good environment management.

We also proposed algorithms minimizing the perturbation caused by the allocation of some task to a server. This optimization is done by still enforcing a good performance (response time) for that task. This system-oriented approach has first been tested through simulations. The best among all the heuristics that we studied have been integrated into NetSolve and studied on a broader scale [1]. The thesis of Yves Caniou, defended in December 2004, was centered around this subject.

4.2.2. Parallel Task Scheduling

Two kinds of parallelism can be exploited in most scientific applications: data- and task-parallelism. One way to maximize the degree of parallelism of a given application is to combine both kinds of parallelism. This approach is called *mixed data and task parallelism* or *mixed parallelism*. In mixed-parallel applications, several data-parallel computations can be executed concurrently in a task-parallel way. This increases scalability as more parallelism can be exploited when the maximal amount of either data- or task-parallelism has been achieved.

This capability is a key advantage for today's parallel computing platforms. Indeed, to face the increasing computation and memory demands of parallel scientific applications, a recent approach has been to aggregate

multiple compute clusters either within or across institutions. Typically, clusters of various sizes are used, and different clusters contain nodes with different capabilities depending on the technology available at the time each cluster was assembled. Therefore, the computing environment is at the same time attractive because of the large computing power, and challenging because it is heterogeneous.

A number of researchers have explored mixed-parallel application *scheduling* in the context of homogeneous platforms. However, heterogeneous platforms have become prevalent and are extremely attractive for deploying applications at unprecedented scales. We propose to build on existing scheduling algorithms for heterogeneous platforms (*i.e.*, specifically designed for task-parallelism) to develop scheduling algorithms for mixed-parallelism on heterogeneous platforms.

For a certain class of scheduling heuristics (list scheduling heuristics) a generic adaptation can be used. While in task scheduling the smallest computational element is a processor, in mixed parallel scheduling the smallest element is a *configuration*. A configuration is defined as a subset of the set of the processors available within, and only within, a cluster and gives information about the size and shape of the virtual grid it represents. The generic method consists in adapting different functions of an heuristic to handle the allocation of tasks on sets of processors.

4.2.3. Data Redistribution Between Clusters

During computations performed on clusters of machines it occurs that data has to be shifted from one cluster to an other. For instance, these two clusters may differ in the resources they offer (specific hardware, computing power, available software) and each cluster may be more adequate for a certain phase of the computation. Then the data have to be redistributed from the first cluster to the second. Such a redistribution should use the capacities of the underlying network in an efficient way.

This problem of redistribution between clusters generalizes the redistribution problem inside a parallel machine, which already is highly non trivial.

We modeled this problem by a decomposition of the underlying bipartite graph into certain types of matching. In general, this problem is NP-hard, as we have been able to show in [4]. Then we have to study lower bounds, approximation algorithms and heuristics. We already obtained some results on heuristics that show a good practical behavior.

We have proposed and studied two fast and efficient algorithms for this problem. We prove that these algorithms are $\frac{8}{3}$-approximation algorithms. Simulation results show that both algorithms perform very well compared to the optimal solution. These algorithms have been implemented using MPI. Experimental results show that both algorithms outperform a brute-force TCP based solution, when no scheduling of the messages is performed [9].

4.2.4. Dynamic and Adaptive Compression of Network Streams

A commonly used technique to speed up transfer of large data over networks with restricted capacity during a distributed computation is data compression. But such an approach fails to be efficient if we switch to a high speed network, since here the time to compress and decompress the data dominates the transfer time. Then a programmer wanting to be efficient in both cases, would have to provide two different implementations of the network layer of his code, and a user of this program would have to determine which of the variants he/she has to run to be efficient in a particular case.

In [8] we presented an algorithm that avoids such an expensive and error-prone setting and provides a technique to compress data on the fly, as necessity of a particular execution requires. It overlaps the compression and communication and automatically adapts the effort for compression to the available resources (network and CPU). It also includes another compression algorithm which favors speed against compression ratio. This allows very good performance when dealing with fast networks.

These algorithms are implemented in our library ADOC, "*Adaptive Online Compression*" which has been deposed at the *Agence de Protection des Programmes*. The ADOC library is known to be portable on the Linux, FreeBSD, MAC OS X, Solaris, AIX, IRIX operating systems.

Several performance issues such as running on very fast network, avoiding Compression level divergence, dealing with small messages and compressed or random data have been recently solved [20].

4.3. Providing environments for experiments

Participants: Jens Gustedt, Emmanuel Jeannot, Martin Quinson.

4.3.1. Emulating Heterogeneity

We have designed a tool called *Wrekavoc*. The goal of Wrekavoc is to define and control the heterogeneity of a given platform by degrading CPU, network or memory capabilities of each node composing this platform. Our current implementation of Wrekavoc have been tested on an homogeneous cluster. We have shown that configuring a set of nodes is very fast. Micro-benchmarks show that we are able to independently degrade CPU, bandwidth and latency to the desired values. Tests on algorithms of the literature (load balancing and matrix multiplication) confirm the previous results and show that Wrekavoc is a suitable tool for developing, studying and comparing algorithms in heterogeneous environments.

4.3.2. Simulating Grid Platforms

We participate to the development of the SIMGRID tool. It enables the simulation of distributed applications in distributed computing environments for the specific purpose of developing and evaluating scheduling algorithms. Simulations not only allow repeatable results (what is hard to achieve on shared resources) but also make it possible to explore wide ranges of platform and application scenarios. SIMGRID implements realistic fluid network models that result in very fast yet precise simulations. SIMGRID also enables the simulation of distributed scheduling agents, which has become critical for current scheduling research in large-scale platforms. This tool is being used by several groups in the Grid Scheduling literature.

5. Software

5.1. SSCRAP

Participant: Jens Gustedt.

SSCRAP is developed to ease the implementation, test and benchmarking of algorithms that are written for the model PRO.

SSCRAP is the prototype of a C++-library that was initially developed together with Isabelle Guérin Lassous from the project-team Ares.

This library takes the requirements of *PRO*, see Section 3.1, into account, *i.e.*, the design of algorithms in alternating computation and communication steps. It realizes an abstraction layer between the algorithm as it was designed and its realization on different architectures and different modes of communication. The current version of this library is now able to integrate:

- a layer for message passing with MPI,
- a layer for shared memory with POSIX threads, and,
- a layer for out-of-core management with file mapping (system call *mmap*).

All three different realizations of the communication layers are quite efficient. They let us execute programs that are otherwise unchanged within the three different contexts such that they reach or maybe outperform programs that are directly written for them.

In the future, the different parts of *SSCRAP* will be integrated together with PARCEL-6 into a common library suite, `par::xxx`.

5.2. AdOC

Participant: Emmanuel Jeannot.

The ADOC *(Adaptive Online Compression) library implements the* ADOC *algorithm for dynamic adaptive compression of network streams.*

ADOC is written in C and uses the standard library *zlib* for the compression part. It is realized as an additional layer above TCP and offers a service of adaptive compression for the transmission of program buffers or files. Compression is only used if it doesn't generate an additional cost, typically if the network is slow or the sending processor is not charged too much. It integrates overlap techniques between compression and communication as well as mechanisms that avoid superfluous copy operations. The send and receive functions have exactly the same semantics as the system calls `read` and `write` so the integration of ADOC into existing libraries and application software is straightforward. Moreover, ADOC is thread-safe.

5.3. Wrekavoc

Participants: Louis-Claude Canon, Emmanuel Jeannot.

Wrekavoc addresses the problem of controlling the heterogeneity of a cluster. Our objective is to have a configurable environment that allows for reproducible experiments on large set of configurations using real applications with no emulation of the code. Given an homogeneous cluster Wrekavoc degrades the performance of nodes and network links independently in order to build a new heterogeneous cluster. Then, any application can be run on this new cluster without modifications.

Wrekavoc is implemented using the client-server model. A server, with administrator privilege, is deployed on each node one wants to configure. The client reads a configuration file and sends orders to each node in the configuration. The client can also order the nodes to recover the original state.

CPU Degradation. We have implemented several methods for degrading CPU performance. The first approach consists in managing the frequency of the CPU through the kernel CPU-Freq interface. We propose two other solutions in case CPU-Freq is not available. One is based on CPU burning. A program that runs under real-time scheduling policy burns a constant portion of the CPU, whatever the number of processes currently running. The other is based on user-level process scheduling called CPU-lim. A CPU limiter is a program that supervises processes of a given user. Using the `/proc` pseudo-filesystem, it suspends the processes when they have used more than the required fraction of the CPU.

Network Limitation. Limiting latency and bandwidth is done using *tc* (traffic controller) based on *Iproute2* a program that allows advanced IP routing. With this tools it is possible to control both incoming and outgoing traffic. Furthermore, the latest versions (above 2.6.8.1) allows to control the latency of the network interface.

Memory Limitation. Wrekavoc is able to limit the largest malloc a user can make. This is possible through the security module PAM. However, we have not been able to limit the whole memory usable by all the processes yet.

Configuring and Controlling Nodes and Links. The configuration of a homogeneous cluster is made through the notion of islet. An islet is a set of nodes that share similar limitation. Two islets are linked together by a virtual network which can also be limited. An islet is defined as a union of IP addresses (or machine names) intervals.

Each islet configuration is stored into a configuration file. At the end of this file is described the network connection (bandwidth and latency) between each islet.

5.4. GRAS

Participant: Martin Quinson.

The GRAS *(Grid Reality And Simulation) framework eases the development of Grid services and infrastructures.*

GRAS provides a C ANSI interface to build distributed services and infrastructure for the Grid. Two implementations of this API are provided: the first one (called Grid R&D Kit) lets the developers experiment, test and debug their work within the SimGrid simulator. The other implementation (called Grid Runtime Environment) allows the resulting programs to run efficiently on real systems.

The simulator thus greatly eases the research and development of Grid services (such as for example monitoring infrastructure or distributed storage systems). In addition, the Grid Runtime Environment is ported to Linux, Solaris, Mac OS X, AIX and IRIX operating systems, and to 9 hardware architectures. Services built on top of this achieve better communication performance than heterogeneous implementations of the MPI protocol.

This tool is now integrated into the SIMGRID framework, and available from http://gforge.inria.fr/projects/simgrid/.

6. New Results

6.1. Large scale experiments

Participant: Jens Gustedt.

Now that the communication layer of *SSCRAP* can handle large numbers of POSIX threads (shared memory) or distributed processes (MPI), we were able to run large scale experiments on mainframes and clusters. These have proved the scalability of our approach as a whole, including engineering, modeling and algorithmic aspects: the algorithms that are implemented and tested show a speedup that is very close to the best possible theoretical values, and these speedups are reproducible on a large variety of platforms. The thesis of Mohamed Essaïdi defended in 2004, was centered around this subject. It will be published in [18].

A lot of the code of *SSCRAP* has been rewritten this year to allow for the separation into different toolboxes, `par::cpp` (interfaces for the C++ language), `par::sys` (interfacing POSIX systems), `par::mem` (tools for managing memory), `par::step` (manage supersteps). This will in particular allow an integration of PARCEL-6 (as `par::cell`) and the different parts of *SSCRAP* into a common library suite, `par::xxx`. PARCEL-6 is a parallel cellular language designed for complex neural networks (cortical systems) and some physical system simulations (based on local equations) that is developed by Stéphane Vialle at Supélec, Metz campus. The integration allows to validate *SSCRAP* on a wide range of fine grained applications and problems. For PARCEL-6 the advantage of clarifying the mapping from the fine-grained formulation (cells) to a coarse-grained real live execution and in addition in achieving portability to distributed environments. The integration is a joint effort together with Amélia De Vivo, university of Basilicata, Ponteza, Italy.

6.2. Models and algorithms for coarse grained computation

Participants: Jens Gustedt, Gaurav Goel.

We continued the design of algorithms in the coarse grained setting as given by the model *PRO* [27]. In particular the internship of Gaurav Goel aimed for the design of an algorithm that takes advantage of the structure commonly encountered with massive graphs, namely the fact that they usually have a bounded arboricity. There we gave algorithms for computing probability vectors that can be used for the clustering of communities, see [26].

The on-going research and discussion on *PRO* and BSP-like computation in general as well as our implementation of *SSCRAP* clearly showed that there is a need for tools that are simple to use and that enforce efficiency at the same time. In [6], we present a programming paradigm and interface (called "*data handover*")

that aims to handle data between parallel or distributed processes and that mixes aspects of message passing and shared memory. In 2005 we started the implementation of a reference library for this API.

6.3. Overlapping Computations and Communications with I/O

Participant: Frédéric Suter.

Several numerical computation algorithms exhibit dependencies that lead to a wavefront in the computation. Depending on the data distribution chosen, pipelining communication and computation can be the only way to avoid a sequential execution of the parallel code. The computation grain has to be wisely chosen to obtain at the same time a maximum parallelism and a small communication overhead.

On the other hand, when the size of data exceeds the memory capacity of the target platform, data have to be stored on disk. The concept of Out-of-Core computation aims at minimizing the impact of the I/O needed to compute on such data. It has been applied successfully on several linear algebra applications.

In [13], [17], we apply Out-of-Core techniques to wavefront algorithms. The originality of our approach is to overlap computation, communication, and I/O. An original strategy is proposed using several memory blocks accessed in a cyclic manner. The resulting pipeline algorithm achieves a saturation of the disk resource which is the bottleneck in Out-of-Core algorithms.

6.4. Scheduling Stochastic Workload

Participant: Emmanuel Jeannot.

Scheduling stochastic workloads is a difficult task. In order to design efficient scheduling algorithms for such workloads it is required to have a good in-depth knowledge of basic random scheduling strategies. We have analyzed the way sequential jobs are distributed and the system behaves in a heterogeneous computational grid environments. In this case, the brokering is done in such a way that each Computing Element has a probability to be chosen proportional to its number of CPUs and its relative speed and its number of CPU. We have given the asymptotic behavior for several metrics (queue sizes, slowdowns,...) in several cases, or, in some case, an approximation of this behavior. We study those metrics in several workload specifications: for various loads (saturated or non saturated), with several distributions,...We have compared our probabilistic analysis to simulations in order to validate our results. These results provide a good understanding of the behavior of the proposed metrics. This will enable to design advanced and efficient algorithms for more complex cases.

6.5. Parallel Task Scheduling

Participants: Tchimou N'Takpé, Frédéric Suter.

There are two ways to produce a parallel task scheduling algorithm on heterogeneous environments starting from existing work of the literature. The former consists in adapting sequential task scheduling algorithms specifically designed for heterogeneous resources in order to introduce parallel task handling. This year, we explored the latter approach, starting from parallel task scheduling algorithms designed for homogeneous environments in which we introduce the handling of heterogeneity.

In [29] we addressed the problem of scheduling applications represented by such DAGs is to find the number of processors and the cluster on which execute each task of the graph while satisfying resource constraints and task dependencies. The objectives are to minimize the total execution time of the application while keeping a "good" balance between execution time and resource consumption, thereby achieving a "good" efficiency. We proposed a novel "virtual" cluster methodology for handling platform heterogeneity combined to an existing algorithm [37] to produced two original algorithm HCPA and S-HCPA. The latter differs by its placement rule based on the sufferage [31] idea. Finally we gave an empirical evaluation in a wide range of platform and application scenarios resorting on simulation.

6.6. Data Redistribution

Participants: Louis-Claude Canon, Emmanuel Jeannot, Frédéric Wagner.

Various redistribution algorithm of the literature have been implemented and tested this year. Indeed, many algorithms have been proposed to redistribute data on a same cluster. However, no fair comparison exists between these algorithms. Moreover, some of them can easily be extended to solve the KPBS problem that consists in redistributing data between clusters over a backbone.

We have carried out experiments on the grid explorer cluster and on grid 5000 between the Orsay site and the Rennes site. Results are currently under analysis.

In 2005 this activity gave rise to various publications, [14], [22], [23], and to the thesis of Frédéric Wagner on December 14.

6.7. Emulation of Heterogeneity

Participants: Louis-Claude Canon, Emmanuel Jeannot.

We have worked on improving the speed and efficiency of Wrekavoc. We mainly focused on CPU degradation. The original CPU degradation mechanism was based on CPU-freq, a kernel module. It was designed to limit CPU frequency in order to save the power on laptops. It is based on several CPU technologies (such as *powernow*) which are not always available on cluster nodes. However, only 10 different frequencies are available through CPU-Freq. Therefore, if the required CPU technologies are not available on the nodes or if the discretization is too coarse we have proposed two other solutions. One is based on CPU burning. A program that runs under real-time scheduling policy burns a constant portion of the CPU, whatever the number of processes currently running. Thanks to the Unix `sched_setaffinity` system call, each CPU burner is tight to a given processor of a multi-processor node. The main drawbacks of this approach is that the CPU limitation occurs for every processes whatever its mode (kernel or user) and therefore, the network bandwidth is limited by the same fraction than the CPU. When an independent limitation of the CPU and the network is required, we have proposed a third alternative based on user-level process scheduling called CPU-lim. A CPU limiter is a program that supervises processes of a given user. Using the `/proc` pseudo-filesystem, it suspends the processes when they have used more than the required fraction of the CPU.

We have also tested Wrekavoc on the Grid-explorer cluster with 216 nodes. Each node has two 2 GHz AMD Opteron 246 with 2 GB of RAM. It runs under Linux Debian 3.1 with kernel 2.6.8.

Results show that we are able to independently degrades each resources and that the obtained degradation is very close to the desired one.

6.8. Grid R&D assisted by simulator

Participant: Martin Quinson.

The goal of the SIMGRID tool suite is to allow the study and development of distributed application on modern platforms. It is the result of a collaboration with Henri Casanova (Univ. of Hawaii, Manoa) and Arnaud Legrand (MESCAL team, INRIA Rhône-Alpes, France).

Simulation is a common answer to the grid specific challenges such as scale and heterogeneity. Unfortunately, the resulting implementations are typically confined to simple proof-of-concept prototypes, needing a complete rewrite for use in real applications.

The *Grid Reality And Simulation* (GRAS) aims at filling this gap by providing two implementations of the same API: the client applications can be run on top of the simulator using the first implementation while the other permits the application deployment on real platforms. This setting allows developers to first implement and experiment with distributed heterogeneous environment in simulation, benefiting from a controlled and fast environment. The applications can then be deployed in the real-world without code modification.

The GRAS framework emphasizes on the characteristics such as Simplicity (both of development framework use and of *in-situ* deployment), Portability (5 UNIX variants, 9 hardware architectures and progress toward Windows port) and Performance (comparable to homogeneous MPI implementations).

GRAS is the outcome of a fruitful collaboration with Rich Wolski (Univ. of California, Santa Barbara) initiated in 2004. This year, we have been continuing research and development of GRAS following the same

direction, resulting in its integration into the SIMGRID project. It is now freely downloadable [30] and enjoys a growing user base as it provides users with critical functionality for the modeling and development and evaluation of distributed applications [36], [38].

7. Other Grants and Activities

7.1. Bilateral international relations and European initiatives

We take part in the NoE "*CoreGrid*" lead by Thierry Priol from INRIA Rennes. More precisely we are part of the work package 6 on scheduling. Emmanuel Jeannot is the leader for CNRS of task 6.5: evaluation and benchmarking.

We maintain several international collaborations with other research teams. The two most fruitful are with the team of Jan Arne Telle from Bergen University, Norway, and with the team of Jack Dongarra at the University of Tennessee, Knoxville.

The collaboration with Bergen has been financed by a bilateral French-Norwegian grant and by some regional visiting grants for Jan Arne Telle.

We collaborate with Vandy Berten and Joël Goossens of the Université Libre de Bruxelles on scheduling problems under stochastic models.

The collaboration with Jack Dongarra of the University of Tennessee, Knoxville and the GRAAL project of INRIA, has been formalized in an INRIA-NSF project which handles the aspects of the integration of our scheduling algorithms into NetSolve.

We collaborate with Henri Casanova of University of Hawaii at Manoa on parallel task scheduling heuristics for heterogeneous environments as well as on the simulation of grid platforms within the SimGrid project.

We collaborate with Prof. Rich Wolski of University of California at Santa Barbara on grid platforms monitoring and characteristics discovering within the NWS project.

7.2. National initiatives

7.2.1. CNRS initiatives, GDR-ARP and specific initiatives

We participate at numerous national initiatives. In the GDR-ARP (architecture, networks and parallelism) we take part in TAROT[1], Grappes[2], and RGE[3].

The support for the latter has been augmented in 2001 by a project called ARGE in the national grid initiative. ARGE had first been guided by André Schaff, and was recently handed over to Jens Gustedt and Stéphane Vialle (Supélec, Metz Campus).

Furthermore, we participate in two AS (actions spécifiques – specific initiatives) *Enabling Grid 5000* and *Programming methods for the Grid* The first is a program that studies the possibilities of enabling a large Grid of several thousand CPUs in France. The second studies more fundamental questions related to Grid computing.

7.2.2. ACI initiatives of the French Research Ministry

We are partners in several projects of different ACI initiatives:

- Grid Explorer. We participate with a joint proposition together with Stéphane Vialle from Supélec, Metz Campus, which concerns the integration tests of *SSCRAP* and Parcel-6 as described in Section 6.1. We also work on designing a set of emulation tools for transforming an homogeneous platform into an heterogeneous one, see Section 6.7.
- In the 2004 initiative ACI AGIR we participate in the definition and design of a set of services for medical image processing on the grid. More precisely we are in charge of transfer with compression task and the evaluation of grid middleware.

[1] *Techniques algorithmiques, réseaux et d'optimisation pour les télécommunications*
[2] *Architecture, systèmes, outils et applications pour réseaux de stations de travail hautes performances*
[3] *Réseau Grand Est*

7.2.3. INRIA New Investigation Grant

The goal of the ARC OTaPHe is to federate conceptual and experimental researches around parallel task scheduling conducted by the AlGorille, GRAAL, MESCAL and MOAIS INRIA teams. Our approach consists in defining models taking computational grid heterogeneity into account. These models however have to remain simple. From those models guaranteed heuristics will be designed and implemented into the DIET and OAR middlewares in order to validate them.

8. Dissemination

8.1. Dissemination

8.1.1. Leadership within the scientific community

On a national level, Jens Gustedt is elected member of INRIA scientific board and was a member of the INRIA steering committee VISON[4] until June 2005.

Emmanuel Jeannot have been an elected member of the computer science hiring committee of UHP up to Sept. 2005. He is also member of the steering committee of the réseau thématique pluridisciplinaire (RTP) (Pluri-disciplinary Thematic Network) "Calcul à hautes performances et calcul réparti" (High Performance and Distributed Computing) of the CNRS STIC Department.

8.1.2. Scientific Expertise

In 2005, Jens Gustedt was a member of the thesis committee of Gérald Oster, University Henri Poincaré, Nancy.

8.1.3. Teaching activities

Emmanuel Jeannot is teaching in the *Algorithme et programmation des systèmes distribués* module of the DEA at Henri Poincaré University . He has taught computer science (System, Java, Data Base, C) in the IUT of Henri Poincaré University up to June 2005.

Frédéric Suter is teaching *Algorithmique et programmation* (L1 and L3) and *Réseaux et Internet* (M2-IMOI) at Henri Poincaré University.

Martin Quinson is teaching the following modules at ÉSIAL (University Henri Poincaré): *C et Shell* (1A), *Réseaux et système* (2A) and *Programmation d'applications réparties* (3A) (third year). He also participates to the following modules: *Informatique de base* (1A), *Algorithmique Parallèle et Distribuée* (3A).

8.1.4. Editorial activities

Since October 2001, Jens Gustedt is Editor-in-Chief of the journal *Discrete Mathematics and Theoretical Computer Science* (DMTCS). DMTCS is an journal that is published electronically by an independent association under French law. Based on a contract with INRIA, its main web-server is located at the LORIA. DMTCS has acquired a good visibility within the concerned domains of Computer Science and Mathematics. In 2005, DMTCS has enlarged its field of activity in publishing two important conference proceedings.

In 2005, Jens Gustedt has served as program committee member of three conferences and workshops:

- The *Rencontres Francophones sur les aspects Algorithmiques des Télécommunications*, Algotel 2005 is the annual French meeting of the community interested in distributed algorithms.
- The *Annual IEEE International Conference on High Performance Computing* (HiPC 2005) is a conference of mainly Indian and Far East radiation.
- The *ACM-SIAM Symposium on Discrete Algorithms* (SODA 2006) is *the* major annual international conference on algorithms.

Emmanuel Jeannot was member of the program committee of RenPar 2005 (16ème rencontre du paralélisme), and IEEE Grid (workshop of Supercomputing 2005) 2005. He is member of the Technical Program Committee of IEEE IPDPS (International Parallel and Distributed Processing Symposium) 2006. Inside LORIA he has participated in the board of "*lettre du* LORIA" up to June 2005.

[4] VISON: Vers un Intranet Sécurisé Ouvert au Nomadisme, towards an secured intranet open to nomadism

8.1.5. Refereeing

In 2005, members of the team served as referees for the following journals and conferences:

Journals: Discrete Applied Mathematics, IEEE Transactions on Parallel and Distributed Systems, Information Processing Letters, International Journal of High Performance Computing, Journal of Parallel Distributed Computing, Theoretical Computer Science,

Conferences: Algotel 2005, ESA 2005, EuroPar 2005, EuroComb 2005, Grid 2005, HCW 2006, HiPC 2005, ICCS 2005, INFOCOM 2006, IPDPS 2005, IPDPS 2006, RenPar 2005, SODA 2006

9. Bibliography

Major publications by the team in recent years

[1] Y. CANIOU, E. JEANNOT. *New Dynamic Heuristics in the Client-Agent-Server Model*, in "IEEE Heterogeneous Computing Workshop - HCW'03, Nice, France", April 2003.

[2] E. CARON, F. SUTER. *Parallel Extension of a Dynamic Performance Forecasting Tool*, in "Accepted for publication in Parallel and Distributed Computing Practice (PDCP)", Special issue on selected papers of ISPDC'02, 2004.

[3] H. CASANOVA, F. DESPREZ, F. SUTER. *From Heterogeneous Task Scheduling to Heterogeneous Mixed Parallel Scheduling*, in "Proceedings of the 10th International Euro-Par Conference (Euro-Par'04), Pisa, Italy", M. DANELUTTO, D. LAFORENZA, M. VANNESCHI (editors). , Lecture Notes in Computer Science, vol. 3149, Springer, August/September 2004, p. 230–237.

[4] J. COHEN, E. JEANNOT, N. PADOY. *Messages Scheduling for Data Redistribution between Clusters*, in "Algorithms, models and tools for parallel computing on heterogeneous network - HeteroPar'03, workshop of SIAM PPAM 2003, Czestochowa, Poland", September 2003.

[5] J. GUSTEDT. *Towards Realistic Implementations of External Memory Algorithms using a Coarse Grained Paradigm*, in "International Conference on Computer Science and its Applications - ICCSA'2003, Montréal, Canada", Lecture Notes in Computer Science, vol. 2668, Springer, February 2003, p. 269-278.

[6] J. GUSTEDT. *Data Handover : Reconciling Message Passing and Shared Memory*, Technical report, n° RR-5383, INRIA, Nov 2004, http://www.inria.fr/rrrt/rr-5383.html.

[7] J. GUSTEDT, J. A. TELLE. *A work-optimal coarse-grained PRAM algorithm for Lexicographically First Maximal Independent Set*, in "Italian Conference on Theoretical Computer Science - ICTCS'03, Bertinoro, Italy", C. BLUNDO, C. LANEVE (editors). , Lecture notes in Computer Science, vol. 2841, Springer, EATCS, October 2003, p. 125-136.

[8] E. JEANNOT, B. KNUTTSON, M. BJORKMAN. *Adaptive Online Data Compression*, in "Eleventh IEEE International Symposium on High Performance Distributed Computing - HPDC 11, Edinburgh, Scotland", IEEE, July 2002.

[9] E. JEANNOT, F. WAGNER. *Two fast and efficient message scheduling algorithms for data redistribution through a backbone*, in "18th International Parallel and Distributed Processing Symposium - IPDPS'04, Santa Fe, New Mexico", IEEE, Apr 2004.

Articles in refereed journals and book chapters

[10] V. BERTEN, J. GOOSSENS, E. JEANNOT. *On the Distribution of Sequential Jobs in Random Brokering For Heterogeneous Computational Grids*, in "IEEE Transaction on Parallel and Distributed Systems", Special Issue on Algorithm Design and Scheduling Techniques (Realistic Platform Models) for Heterogeneous Clusters, to appear, 2005.

[11] Y. CANIOU, E. JEANNOT. *Multi-Criteria Scheduling Heuristics for GridRPC Systems*, in "International Journal of High Performance Computing Applications", to appear, 2005.

[12] E. CARON, B. DELFABBRO, F. DESPREZ, E. JEANNOT, J.-M. NICOD. *Managing Data Persistence in Network Enabled Servers*, in "Scientific Programming Journal", Special Issue on Dynamic Grids and Worldwide Computing, to appear, 2005.

[13] E. CARON, F. DESPREZ, F. SUTER. *Overlapping Communications and Computations with I/O in Wavefront Algorithms*, in "Concurrency and Computation: Practice and Experience", Accepted for publication, 2005.

[14] J. COHEN, E. JEANNOT, N. PADOY, F. WAGNER. *Messages Scheduling for Parallel Data Redistribution between Clusters*, in "IEEE Transaction on Parallel and Distributed Systems", to appear, 2005.

Publications in Conferences and Workshops

[15] Y. CANIOU, E. JEANNOT. *Le HTM : un module de prédiction de performance non-intrusif pour l'ordonnancement de tâches sur plate-forme de meta-computing*, in "16 ème Rencontres Francophones du Parallèlisme (RENPAR 2005), Le Croisic, France", April 2005.

[16] F. CAPPELLO, F. DESPREZ, M. DAYDE, E. JEANNOT, Y. JEGOU, S. LANTERI, N. MELAB, P. NAMYST, O. RICHARD, E. CARON, J. LEDUC, G. MORNET. *Grid'5000: a large scale, reconfigurable, controlable and monitorable Grid platform*, in "6th IEEE/ACM International Workshop on Grid Computing (GRID 2005), Seattle, WA, USA", to appear, November 2005.

[17] E. CARON, F. DESPREZ, F. SUTER. *Out-of-Core and Pipeline Techniques for Wavefront Algorithms*, in "Proceedings of the 19th International Parallel and Distributed Processing Symposium (IPDPS'05), Denver, CO", April 2005.

[18] M. ESSAÏDI, J. GUSTEDT. *An experimental validation of the PRO model for parallel and distributed computation*, in "14th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP 2006)", B. DI MARTINO (editor). , 2005, http://hal.inria.fr/inria-00000612/en/.

[19] C. GERMAIN, V. BRETON, P. CLARYSSE, Y. GAUDEAU, T. GLATARD, E. JEANNOT, Y. LEGRÉ, C. LOOMIS, J. MONTAGNAT, J.-M. MOUREAUX, A. OSORIO, X. PENNEC, R. TEXIER. *Grid-Enabling Medical Image Analysis*, in "Third International Workshop on Biomedical Computations on the Grid (Bio-Grid 2005), Cardiff, UK", May 2005.

[20] E. JEANNOT. *Improving Middleware Performance with AdOC: an Adaptive Online Compression Library for Data Transfer*, in "International Parallel and Distributed Processing Symposium 2005 (IPDPS'05), Denver, Colorado, USA", April 2005.

[21] E. JEANNOT, G. MONARD. *Computing Molecular Potential Energy Surface with DIET*, in "International Conference on Information Technology (ITCC2005), Las-Vegas, Nevada, USA", April 2005.

[22] E. JEANNOT, F. WAGNER. *Messages Scheduling for data Redistribution between Heterogeneous Clusters*, in "IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS 2005), Phoenix, AZ, USA", Best paper award, November 2005.

[23] F. WAGNER. *Redistribution de données en régime permanent*, in "16 ème Rencontres Francophones du Parallèlisme (RENPAR), Le Croisic, France", April 2005.

Internal Reports

[24] V. BERTEN, J. GOOSSENS, E. JEANNOT. *On the Distribution of Sequential Jobs in Random Brokering For Heterogeneous Computational Grids*, Technical report, n° RR-5499, INRIA, February 2005, http://www.inria.fr/rrrt/rr-5499.html.

[25] E. CARON, B. DELFABBRO, F. DESPREZ, E. JEANNOT, J.-M. NICOD. *Managing Data Persistence in Network Enabled Servers*, Technical report, n° RR-5725, INRIA, October 2005, http://www.inria.fr/rrrt/rr-5725.html.

[26] G. GOEL. *Computing the Probability Vectors for Random Walks on Graphs with Bounded Arboricity*, 2005, http://hal.inria.fr/inria-00000578/en/.

Miscellaneous

[27] A. GEBREMEDHIN, I. GUÉRIN LASSOUS, J. GUSTEDT, J. A. TELLE. *PRO: A Model for the Design and Analysis of Efficient and Scalable Parallel Algorithms*, 2005, http://hal.inria.fr/inria-00000899/en/.

[28] J. GUSTEDT. *Efficient Sampling of Random Permutations*, 2005, http://hal.inria.fr/inria-00000900/en/.

[29] T. N'TAKPÉ. *Adaptation d'un algorithme d'ordonnancement de tâches parallèles sur plates-formes homogènes aux systèmes hétérogènes*, Technical report, INPL-ENSEM / LORIA, June 2005.

Bibliography in notes

[30] H. CASANOVA, A. LEGAND, L. MARCHAL. *Scheduling Distributed Applications: the SimGrid Simulation Framework*, in "Proceedings of the third IEEE International Symposium on Cluster Computing and the Grid (CCGrid'03)", may 2003.

[31] H. CASANOVA, A. LEGRAND, D. ZAGORODNOV, F. BERMAN. *Heuristics for Scheduling Parameter Sweep Applications in Grid Environments*, in "Proc. of the 9th Heterogeneous Computing Workshop (HCW'00), Cancun", 2000, p. 349–363.

[32] T. H. CORMEN, M. T. GOODRICH. *A Bridging Model for Parallel Computation, Communication, and I/O*, in "ACM Computing Surveys", vol. 28A, n° 4, 1996.

[33] D. CULLER, R. KARP, D. PATTERSON, A. SAHAY, K. SCHAUSER, E. SANTOS, R. SUBRAMONIAN, T. VON

EICKEN. *LogP: Towards a Realistic Model of Parallel Computation*, in "Proceeding of 4-th ACM SIGPLAN Symp. on Principles and Practises of Parallel Programming", 1993, p. 1-12.

[34] F. DEHNE, W. DITTRICH, D. HUTCHINSON. *Efficient external memory algorithms by simulating coarsegrained parallel algorithms*, in "ACM Symposium on Parallel Algorithms and Architectures", 1997, p. 106-115.

[35] F. DEHNE, A. FABRI, A. RAU-CHAPLIN. *Scalable parallel computational geometry for coarse grained multicomputers*, in "International Journal on Computational Geometry", vol. 6, n° 3, 1996, p. 379-400.

[36] V. GARONNE, A. TSAREGORODTSEV, E. CARON. *A study of meta-scheduling architectures for high throughput computing: Pull vs. Push*, in "ISPDC'05", July 2005.

[37] A. RADULESCU, A. VAN GEMUND. *A Low-Cost Approach towards Mixed Task and Data Parallel Scheduling*, in "Proc of the 15th International Conference on Parallel Processing (ICPP), Valencia, Spain", September 2001.

[38] E. SANTOS-NETO, W. CIRNE, F. BRASILEIRO, A. LIMA. *Exploiting Replication and Data Reuse to Efficiently Schedule Data-intensive Applications on Grids*, in "Proceedings of 10th Job Scheduling Strategies for Parallel Processing", June 2004.

[39] L. G. VALIANT. *A bridging model for parallel computation*, in "Communications of the ACM", vol. 33, n° 8, 1990, p. 103-111.