Project Proposal

(revision 2006-07-07)

Title:
Algorithms for the Grid — Algorithmes pour la Grille

Acronym: ALGORILLE

Scientific leader: Jens Gustedt

Proposed INRIA theme:
Grids and high-performance computing (Num B)

INRIA scientific and technological challenges:
Designing and mastering the future network and communication services infrastructures

Keywords: Grid Computing, Algorithms, Experiments, Scheduling, Data Management, Models of Computation

1 Project-team composition

Head		
Jens Gustedt	research director	INRIA Lorraine

Vice-Head		
Emmanuel Jeannot	research fellow	INRIA Lorraine

Administrative assistant		
Josiane Reffort	partial, 20%	Univ. Henri Poincaré

Scientific permanents		
Martin Quinson	assistant professor	Univ. Henri Poincaré
Frédéric Suter	assistant professor	Univ. Henri Poincaré

External collaborator		
Stéphane Vialle	professor	Supélec, Metz campus

Temporary staff		
X X	associate engineer	INRIA, SimGrid project
Xavier Delaruelle	associate engineer	INRIA, Grid5000 project

PhD Students		
Pierre-Nicolas Clauss		Univ. Henri Poincaré defense end 2009
Tchimou N'Takpé		Univ. Henri Poincaré defense end 2008

2 Overall objectives

The possible access to distributed computing resources over the Internet allows for 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 State of the art

The notion of *The Grid*[FK98b] appeared some years ago as an analogy of the electric power grid, where the power that an individual customer needs is provided automatically. The objective of *The Computing Grid* is to transparently *broker* access to computing resources between *customers* and *service providers* via the Internet.

A functional general *Grid* is actually far from reach, but several partial approaches have been developed that emphasize on different aspects of desired properties. These are for example peer to peer networks like Gnutella [gnu] or XtremWeb [xtr], Internet computing like SETI@home [set], web portals like PUNCH [pun] or Océano [oce], Application Service Providers like NetSolve [AD00,net], NINF [HN99,nin] or DIET [die,CLQS02], and virtual super-computers like Globus [FK98a] or UniCore [uni].

Most of these projects address the difficult engineering tasks that are related to such systems. But many problems that are of a more fundamental nature remain to be solved before we can expect substantial progress towards a unified, general *grid*. In our project, we aim at tackling the *algorithmic* issues for computing on the grid, that is, we want to design efficient algorithms for which we will give evidence on how to control the use of the involved resources. To have a potential impact, the evidence must go beyond classical proofs of complexity, *e.g.,* in terms of orders of magnitudes, and must provide hard and reproducible experimental data that validates our modeling, design and implementations.
These algorithmic issues originate in several fundamental properties of grids:

genericity: In contrast to a classical distributed environment, a grid environment will provide very few knowledge about the effective resources to a user. The user in turn, will not provide much information about her/his application to the executing environment. This lack of knowledge will have implications on both sides:

[FK98b] I. FOSTER, C. KESSELMAN, *The Grid: Blueprint for a New Computing Infrastructure*, Morgan-Kaufmann, 1998.

[gnu] "Gnutella.com", `http://www.gnutella.com/`.

[xtr] "XtremWeb", `http://www.xtremweb.org/`.

[set] "Seti@Home", `http://setiathome.ssl.berkeley.edu/`.

[pun] "The PUNCH project", `http://punch.purdue.edu`.

[oce] "The Océano project", `http://www.research.ibm.com/oceanoproject/`.

[AD00] D.-C. ARNOLD, J. DONGARRA, "The NetSolve Environment: Progressing Towards the Seamless Grid", *in : International Conference on Parallel Processing (ICPP-2000)*, Toronto Canada, August 2000.

[net] "NetSolve", `http://icl.cs.utk.edu/netsolve`.

[HN99] S. S. H. NAKADA, M. SATO, "Design and Implementations of Ninf: Towards a Global Computing Infrastructure", *Future Generation Computing Systems, Metacomputing Issue 15*, 1999, p. 649–658.

[nin] "The NINF project", `http://ninf.apgrid.org/`.

[die] "DIET (Distributed Interactive Engineering Toolbox)", `http://graal.ens-lyon.fr/DIET`.

[CLQS02] P. COMBES, F. LOMBARD, M. QUINSON, F. SUTER, "A Scalable Approach to Network Enabled Servers", *in : Advances in Computing Science - ASIAN 2002. Internet Computing and Modeling, Grid Computing, Peer-to-Peer Computing, and Cluster Computing. Seventh Asian Computing Science Conference*, A. Jean-Marie (editor), *Lecture Notes in Computer Science*, *2550*, Springer-Verlag, p. 110–124, Hanoï, Vietnam, December 2002.

[FK98a] I. FOSTER, C. KESSELMAN, "The Globus Project: A Progress Report", *in : Heterogeneous Computing Workshop*, March 1998.

[uni] "The UniCore project", `http://www.unicore.org`.

applications will have to be written differently and service providers will have to react differently.

heterogeneity: Up to a certain degree, the execution environment for an application will be heterogeneous. Applications and environments must be able to predict behavior with some simple factors that describe resource utilization, such as CPU, memory and bandwidth demands. These factors must be predictable, measurable (benchmarked) and controllable (tunable and accounted).

scale: Grids will allow us to scale applications up to orders of magnitude which until now had simply not been possible or had been reserved to very special circumstances.

irregularity: Grid computing will only be a success if it will target new *types* of applications than previously. So far, many of the applications that scaled up to a large amount of processors have been "embarrassingly parallel" or were handling data that was regularly structured. In particular, problems that exploit highly irregular data such as networks are hit by latency problems that will become more and more severe as the hardware evolves.

There exist a lot of ongoing efforts that try to cope with each of these properties as they are listed above, but we are convinced that none of them can be handled isolated. Looking at the interplay of them is what we hope constitutes one of the originalities of our approach. To emphasize on that, for three particular challenges we will go a little more into details: the connection between spatial and temporal distances, the interplay between different objective functions corresponding to the roles of different grid-actors, and, the challenge of performance evaluation.

Other important concerns for grids will have less focus in our project:

security Important issues here are authentication (users, hosts and components), authorization and verification (protocols, data and results). According to Chakrabarti[Cha05], issues in security in the area of grid computing can be broadly classified into **system level**, **architectural**, and **interoperability** issues and thus do not fall within the main scope of our project. We will assume that these problems can be clearly identified and that the corresponding solutions can properly interfaced such that their impact on performance is minor.

robustness When combining more and more heterogeneous components, the possibilities of failures or unavailabilities of parts or the whole of a system increases. This is the case for grid systems that build upon a large number of inexpensive resources that are highly distributed. The proposed solutions for these problems may or may not have an impact on performance. For instance the impact of check pointing techniques on performance should be controllable, whereas introducing redundancy of executions augments the resource usage substantially. Being an algorithmic project, our contribution to this domain will mainly concern the trade-off between performance and the gain in robustness that is achieved via redundant program execution.

[Cha05] A. CHAKRABARTI, "Grid Computing Security – Issues, Concerns and Counter-measures", Tutorial held at CCGrid'05, May 2005, http://www.cs.cf.ac.uk/ccgrid2005/tutorials/TutorialCCGrid-Chakrabarti.htm.

Challenges

Our project will focus on three different challenges, giving rise to three different research directions:

- Spatial and temporal distances will be the major challenge to structure applications such that they may scale to the grid,
- reconciling different objectives challenges the transparent management of grid resources, and
- faithful performance evaluation of grid systems is a research challenge and research direction of its own rights.

The algorithmic challenge: overcome spatial and temporal distances

Grids are distributed environments for which we will have to organize

$$\left.\begin{array}{l}\text{tasks}\\ \textit{and}\\ \text{data}\end{array}\right\} \text{ within } \left\{\begin{array}{l}\text{time}\\ \textit{and}\\ \text{space.}\end{array}\right.$$

This organization can not be left to the duty of only one side, service provider or customer, since both only will have partial knowledge of the situation. The service provider may not foresee all future requirements that an application will express, the customer may not know much about the details of the different platforms, their composition and their interconnection on which her/his application will finally be executed.

Because of this obvious difficulty, current solutions rarely address the whole problem. Merely on the service provider side, *Scheduling* [ERLA94,Leu04] asks for organization of tasks in time, *Job Placement* allocates physical processors to the tasks to which *Job Balancing* adds a dynamic (temporal) component, *Data Distribution* allocates memory locations to data, whereas *Data Redistribution* [BPR98,DDP+98] takes the data movement into account. On the side of the algorithm designer a lot of attempts have been made to provide abstract models that allow to conceive applications independently of the platform. But most of them remain tied to one of the paradigms of parallel or distributed computing and inherit the respective difficulties of these: the false illusion of shared memory in a distributed setting calls for fine grained programming with severe latency problems, the lack of bindingness between distributed processes calls for data consistency and data persistence problems.

[ERLA94] H. EL-REWINI, T. LEWIS, H. ALI, *Task Scheduling in Parallel and Distributed Systems*, Prentice Hall, 1994.

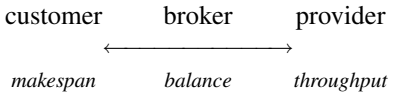
[Leu04] J. Y.-T. LEUNG (editor), *Handbook of Scheduling*, Chapman & Hall/CCR, 2004.

[BPR98] P. BHAT, V. PRASANNA, C. RAGHAVENDRA, "Block Cyclic Redistribution over Heterogeneous Networks", *in : 11th Int. Conf. on Parallel and Distrinuted Computing Systems (PDCS 1998)*, 1998.

[DDP+98] F. DESPREZ, J. DONGARRA, A. PETITET, C. RANDRIAMARO, Y. ROBERT, "Scheduling Block-Cyclic Array Redistribution", *IEEE Transaction on Parallel and Distributed Systems 9*, 2, 1998, p. 192–205.

The conceptual challenge: reconcile differing objectives

There is no common agreement on what the correct measures for the performance and cost of an application in an grid environment are: service provider, resource broker and customer do not necessarily have the same objectives. The service provider most likely wants to squeeze as much computation "power" out of the investment and thus will privilege high throughput.



In contrast to that, the customer usually wants her/his tasks done as soon as possible thus will privilege the use of the most effective resources. She/He needs algorithmic methods that help to structure her/his application to reduce its cost and to impose less constraints on the provider and other customers that concurrently compute on the same environment.

Then, the resource broker has to organize the resources at best, and, at the same time, by minimizing the impact on all applications that are executed. Thus she/he will privilege schemes that balance resource usage. This introduces a third objective to those of the customer and the provider and it is not clear at all how it relates to them.

Current research in *e.g.,* job scheduling mostly privileges the users perspective and concentrates on the completion time of individual tasks or jobs.

The methodological challenge: performance evaluation

On top of all these difficulties for the design and running of applications on grid systems stacks another fundamental difficulty: the problem of *performance evaluation*. We have to compare different algorithms reliably and thereby prove scientific progress for our research. In particular we have to *predict*, *control* and *evaluate* the performance in a setting that is as wide as possible.

Nowadays, we only know how to describe the performance of very restricted parts of the platform: there are hardware oriented benchmarks for computing environments (SPEC [spe], NAS [NAS] or LinPack [lin]), simulators (NS [ns], SIMGRID [sim]), system sensors (NWS [nws]) or application profiling tools (gprof [gpr], *etc.*).

[spe] "Standard Performance Evaluation Corporation (SPEC)", `http://www.spec.org/`.

[NAS] "NAS Parallel Benchmark", `http://www.nas.nasa.gov/Software/NPB/`.

[lin] "LinPack", .

[ns] "Network Simulator (NS)", `http://www.isi.edu/nsnam/ns/`.

[sim] "SimGrid", `http://simgrid.gforge.inria.fr/`.

[nws] "Network Weather Service (NWS)", `http://nws.cs.ucsb.edu`.

[gpr] "GNU Binutils", `http://sources.redhat.com/binutils/`.

4 Scientific foundations

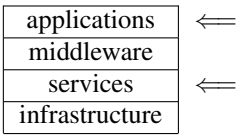


Figure 1: Layer model of a grid architecture

Given the complex nature of grid platforms, software systems targeting this kind of architecture have to rely on a layered model. Here, as a specific point of view for our project we will distinguish four layers as they are illustrated in Figure 1. The *infrastructure* encompasses both hardware and operating systems. *Services* abstract infrastructure into *functional units* (such as resource and data management, or authentication) and thus allow to cope with the heterogeneity and distribution of the infrastructure.

Services form grounding bricks that are aggregated into *middlewares*. Typically one particular service will be used by different middlewares, thus such a service must be sufficiently robust and generic, and the access to it should be standardized. Middlewares then offer a software infrastructure and programing model (data-parallel, client/server, peer-to-peer, *etc.*) to the user *applications*. Middlewares may be themselves generic (*e.g.,* Globus), specialized to specific programming models (*e.g.,* message passing libraries) or specific to certain types of applications.

To our opinion the algorithmic challenges of such a system are located at the *application* and *service* layers, hence in the following we will emphasize on these. In addition to these two types of challenges, we identify a third which consists in the evaluation of models, algorithms and implementations for grid systems. To summarize, the three research areas that we address are:

applications: We have to organize the application and its access to the middleware in a way that is convenient for both. Therefore the middleware should export an interface that offers enough potential for an efficient implementation of an application and such that it constrains it to a reasonable set of actions. The application should restrict itself to a usage of the middleware that remains sensible and makes the least assumptions about the other underlying (and hidden) layers.

services: The service layer has to organize the infrastructure in a convenient way such that resources are used efficiently and such that the applications receive a good service quality. Each service has to provide an API and a precise semantic. The middleware call a given service to solve a given problem.

performance evaluation: To be able to make a statement about the quality of computational models and algorithms that we develop within such a paradigm, we have to compare algorithms and program executions amongst each other. As of today there remain a lot of challenges here, on how to organize reproducible experiments that permit to extrapolate to new environments and to new scales in the number of processors or the input data size.

4.1 Modeling and structuring of complex applications

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 or control parallelism, such a division (into data and tasks) is necessary to overcome spatial and temporal distances such as those we encounter in grids.

Several parallel models offering simplified frameworks that ease the design of algorithms and their implementation 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 regard 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. These organize their computation in *supersteps*; *i.e.,* an alternation of computation and communication phases. 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 cannot 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 input size is orders of magnitude larger than the number of available 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)[Val90], LOGP (Latency overhead gap Procs)[CKP+93], CGM (Coarse Grained Multicomputer)[DFRC96] and PRO (Parallel Resource Optimal) [53] models.

In fact, this organization in supersteps has another interesting property in that it allows for a good control of the data *locality*. By simulating a coarse grained computation on a sequential machine this can then be used to tackle problem sizes that are otherwise intractable. Computing on large datasets implies to store the considered data on auxiliary memory supports such as disks when the data size exceed the memory capacity of the computing platform (out-of-core computing). The main objective in the classical model[VS94] for this computation is to minimize the disk I/O needed to perform such a computation. Its main disadvantage is that it makes no distinction between the latency of an individual disk access and the bandwidth capacities of the disk.

All the mentioned models do not allow the design of algorithms for grids since they all assume homogeneity, for the processors, for the storage capacity and for the interconnection network. Hence, an important issue of this proposal will be to work on extensions of the known coarse grained setting to heterogeneous platforms and to integrate storage requirements into the models and the algorithms.

[Val90] L. G. VALIANT, "A bridging model for parallel computation", *Communications of the ACM 33*, 8, 1990, p. 103–111.

[CKP+93] 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*, p. 1–12, 1993.

[DFRC96] F. DEHNE, A. FABRI, A. RAU-CHAPLIN, "Scalable parallel computational geometry for coarse grained multicomputers", *International Journal on Computational Geometry 6*, 3, 1996, p. 379–400.

[VS94] J. S. VITTER, E. A. M. SHRIVER, "Algorithms for Parallel Memory I: Two-Level Memories", *Algorithmica 12*, 2-3, 1994, p. 110–147.

4.2 Resource Management

We think of the grids as of a medium to access resources. This access has to be as transparent as possible to the users and the management of these resources has not to be imposed to them, but entirely done by a *system*, perceived through the *middleware* in our layered model. This middleware in turn has to be able to build upon a *service layer* that manages all the resources of a grid infrastructure in a satisfactory way. Currently, numerous *algorithmic* problems hinder such an efficient resource management and thus the transparent use of grids.

By their nature, distributed applications use different types of resources; the most important being computing power, storage capacity 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.

Most of the existing work [Leu04] concentrates to solve one of the difficult aspects at a time, *e.g.,* heterogeneity (processors, OS, network), dynamics (volatility, network jitter), resource sharing (CPU, network) and resource distribution. This strongly restricts the current usage of these technologies. Tackling different challenges at the same time would constitute an important step towards transparent resource management.

Another important restriction of algorithmic research on resource management is its focus on user/customer centered cost functions. These correspond more to a world were the big investment in a costly infrastructure has already been made and the user governs the scheduling strategy or the bandwidth allocation. But as we mentioned above, on a grid we encounter different actors who have different objectives. Hence, different criteria have to be taken into account (*e.g.,* not only the makespan for scheduling but also time, load balance and throughput).

There is a rich toolbox of algorithmic approaches to solve resource management problems. In terms of algorithms design, there are heuristic approaches, metaheuristics, approximation schemes and randomized algorithms. Measurements of the performance of individual parts may be intrusive into the lower levels of a system or just satisfy themselves with some global observations of CPU load or bandwidth usage. In terms of evaluation of solutions we work with probability models (for computing bounds) or statistics (performing simulations).

The ability to evaluate and predict the state of the environment is crucial to many service components we plan to integrate into run-time middlewares. Several approaches can be used to provide this ability. The *Network Weather Service* (NWS [WSH99]) relies on periodic measurements to acquire system availabilities and on statistical methods on measures to interpolate future variations. On the other hand, the *Historical Trace Manager* (HTM), [3, 32] is informed by the scheduler of the task placement and their estimated run time. It can then predict the system availability by simulation without any active measurement. This approach is less intrusive than the former one and allows to predict availability changes even before they happen, but this is also less robust to external load. To our opinion, a new service combining the advantages of both approaches is still to be developed.

[Leu04] J. Y.-T. LEUNG (editor), *Handbook of Scheduling*, Chapman & Hall/CCR, 2004.

[WSH99] R. WOLSKI, N. T. SPRING, J. HAYES, "The network weather service: a distributed resource performance forecasting service for metacomputing", *Future Generation Computer Systems 15*, 5-6, 1999, p. 757–768.

4.3 Performance Evaluation and Prediction

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.

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. A brief look into the literature shows that for performance experiments in our scientific community this is not as straightforward and by no means obvious.

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.

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 that 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.

5 Application domains

Our project is not application-driven but focused on algorithmic foundations and experimental validation. Therefore our results will often not be directly applicable but be the input to other, more applied, research about grid technology. Nevertheless we will interact with some chosen applications to ensure that our research keeps track of reality and stays sufficiently grounded.

According to our layered model of the grid, we distinguish two different kinds of *consumers* for our research:

- our work in the service layer interacts and is included in existing middlewares
- our work in the application layer interacts with highlevel libraries and APIs.

For the service layer we target the design of new resource management strategies and therefore the middleware can be seen as the application of these new services. We aim at designing such algorithms for different kinds of middleware and provide implementations that may integrate independently into them. Because of the emergence of the standardized GridRPC API we will be able to ingrate our libraries into middleware that use this interface, such as DIET [die] or NetSolve [net]. An important task will also be to introduce our scheduling strategies into the meta-scheduler OAR [oar].

Concerning the application layer we participate in the design of scientific applications in a interdisciplinary context. For instance in the ACI *Masse de données* AGIR, we work on grid applications for medical images processing. More precisely we tackle the problem of software infrastructure for interactive medical image processing on a grid. In the ACI GRID ARC, we designed an application called HSEP for computing molecular potential Energy Hyper-Surface using DIET.

The close integration of SSCRAP and PARCEL into PARXXL (see the following Section 7 on software) offers the advantages of coarse grained parallel computing to applications that are programmed within the neural network paradigm. Recently, together with the institute of theoretical physics (LPMI) we have proposed an ANR[1] project that combines modeling of nuclear fusion on the atom level by cellular networks with the aim of scaling to simulations of very large test sets. At the time of writing the acceptance of this project is pending.

6 Research directions

Based on the layered model presented page 8, we propose to address issues at the application layer and at the service layer. We believe that these two layers set important algorithmic challenges and that our team combines the required skills to tackle them. As a third research direction, we want to address experiments for validating the proposed models and solutions. Thus we have three main directions of research:

1. In the application layer, we will provide algorithmic models that are adapted to the nature of grid environments as well as algorithms that take benefit of these new models.

[1] Agence Nationale de la Recherche

[die] "DIET (Distributed Interactive Engineering Toolbox)", `http://graal.ens-lyon.fr/DIET`.

[net] "NetSolve", `http://icl.cs.utk.edu/netsolve`.

[oar] "OAR", `http://oar.imag.fr`.

2. In the service layer, we want to design algorithms and tools in order to efficiently manage the different resources that make-up a grid (mainly CPU, storage, and network).

3. Concerning experiments we aim at designing methodologies and tools for simulation, emulation and real-scale tests. We will also participate in the building of large-scale experimental testbeds.

These three research directions will be detailed in the following sections.

6.1 Structuring of Applications for Scalability

Participants: Jens Gustedt, Frédéric Suter, Stéphane Vialle.

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.

In particular 1 is implicitly or explicitly present in many large scale applications,*e.g.,* from theoretical physics or chemistry[HLSØ06].

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 [53] 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 [83].

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.

[HLSØ06] K. HINSEN, H. P. LANGTANGEN, O. SKAVHAUG, Å. ØDEGÅRD, "Using BSP and Python to Simplify Parallel Programming", *Future Generation Computer Systems 22*, 1-2, 2006, p. 123–157.

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.

6.1.1 Modeling a grid environment

We plan to extend the coarse grained architecture model in several steps such that it will better cover the heterogeneous aspects that we encounter in an *a priori* unknown grid environment. The first two steps will involve the following types of architectures:

"*Hierarchical and homogeneous*" architectures. These are architectures composed of homogeneous processors (at least of same performance) but that are equipped with an interconnection that is non-uniform but hierarchical (CC-NUMA machines, clusters of SMP, *etc.*)

"*Hierarchical and heterogeneous*" architectures. Other than for the previous case, the loosening of the homogeneity condition makes it difficult to even define the speedup of a parallel code since it is not even clear what should be taken as a sequential reference (time on the fastest processor, the slowest, some (weighted?) average?)

A possibility to model heterogeneous systems with heterogeneous processor power is to simulate a certain amount of virtual processors on a physical processor. Then we may allocate a number of virtual processors to physical processors that is proportional to their capacity. The resulting (virtual) architecture then remains hierarchical, the virtual processors just add another level to the physical hierarchy. Hence, such an approach reduces to the case of homogeneous processors but has its own difficulties:

- The communication capacities of the physical processors is not necessarily linked to the processing power. Thus a strong processor with a weak communication link will slow down the processing.
- The communication capacity of a processor may be constrained by the memory bandwidth that a computation uses. Thus communication between the virtual processors may slow down communication on the physical network.
- The algorithmic overhead imposed by the different levels of the hierarchies must remain controllable.

6.1.2 Algorithm Design

Pertinent models of Grid environments have to be accompanied by algorithms that use them adequately. In many cases existing algorithms can be adapted or conveniently restricted such that they fit into such a context, but often this will not suffice and we will have to develop new ones. Sometimes we hope that we will even be able to give efficient algorithms for problems for which in other models there are impossibility results.

According to our expertise we focus on algorithms for specific fields:

1. the design of algorithms for large graphs,

2. algorithms for problems that are hard in a fine-grained setting, and

3. out-of-core algorithms that circumvent IO-latency problems.

New algorithms for large graphs. Certain problems require algorithms and data structures that are well adapted to a grid context. We focus in particular on the study of very large graphs as they appear in multiple contexts such as the graph of the Internet, the Web-graph, peer-to-peer graphs or social networks.

Their connection to the grid is double. On one hand they will be needed to model certain dependencies between grid components, tasks of an execution or data. On the other hand we will process these graphs in a grid context. The technique for this processing will be provided by the coarse grained models since they show a good realism concerning the architecture and the performance analysis.

Recently it has been observed that most of these large graphs which appear in real world have quite special properties:

- Their average degree is low, but there are substantially many vertices of high degree.
- The degree distribution decreases polynomially.
- The average distance in these graphs is low, *i.e.,* in general there are short paths between all pairs of vertices.
- The clustering coefficient (the probability that two neighbors of a vertex are neighbors as well) is high.

Our objective will be to design algorithms and data structures that are well adapted to exploit these structural properties:

- The low average degree helps to structure the graphs and their distributed representation and to maintain a good data locality.
- The similarity of the vertex neighborhoods allows for efficient pre-computation.
- A hierarchic treatment with respect to the vertex degrees will allow to define and then compute new global properties and to find clusters, *i.e.,* components with a weak connection to the exterior.

Since they form the base of any efficient handling of large graphs, other graph problems will be important to study such as search strategies, connected components or coloring problems.

Re-use of fine grained algorithms and models. Many algorithms in the rich literature on fine grained algorithms could be adapted to our context. We are looking for descriptive tools that help for an easy translation of these algorithms. For instance we try to describe a large class of PRAM algorithms that translates more or less directly into our PRO model.

For other more specialized algorithms other models could be used. This applies in particular to algorithms that use fixed data structures such as vectors or matrices and for which the communication pattern is known in advance.

Another emphasize in this context will be given to the mapping of other models of computation to the coarse grained setting. This concerns in particular models of

neuronal and cellular networks. Due to their proceeding in rounds between local computations and inter-cellular updates they fit quite well into the context of coarse grained computing. Nevertheless, many issues have to be resolved before such an integration (or mapping) can be successful and transparent to the designer of such a network of cells. The most important being the problem of data consistency and update periods that in many practical applications directly influence the convergence of the computation.

Limits of fine grained complexity results. In addition to re-using existing algorithms it will be important to know more about the extensibility of complexity bounds that have been proved for the PRAM. Because of their very different nature, none of the completeness results translates directly into the coarse grained setting and even more there are indications that they might simply not hold in general: as a first result we already have shown that there is an algorithm for a P-complete problem that uses few processors ($\approx \log n$) and that is work optimal, see [58].

Out-of-core computing Recently we studied the application of out-of-core techniques to a specific class of algorithms, so-called wavefront algorithms [38]. These algorithms (Column-Sort, ADI, Gauss-Seidel, SOR, or Sweep3D) rely on macro-pipelining techniques (loop reordering, pipeline loops, appropriate data distribution, *etc.*) to overlap communications with computations where possible through asynchronous communications. We proposed an out-of-core wavefront algorithm in which communications and computations are overlapped with disk I/O. The main issue with out-of-core computing is indeed that disk I/O costs are an order of magnitude higher than other communication or computation costs. In particular, the latency of out-of-core storage is usually very high and it is very important to hide it to only encounter the device bandwidth as dominating cost factor.

We will study an alternative approach in which computational grids are used in order to gain access to more memory and thereby be able to perform computations in memory. This approach will be applied to pipelined computations on out-of-core data. According to this approach the bottleneck is not totally removed but reduced to a communication bottleneck: besides latency issues, inter-cluster and intra-cluster communications are in general of the same order of magnitude.

We will propose a method to determine the appropriate amount of data distributed to each processor according to multiple optimization criteria: (i) available memory space of each machine, (ii) relative computing power of a given cluster with regard to the rest of the platform, and (iii) its connectivity.

We will then design and structure algorithms such that they achieve high performance and scale well. This design step will put more emphasis on particular phases of pipelined algorithms where data has to be communicated. The modified algorithms may indeed introduce new communications between processors of different clusters due to the new data distribution. The workflow of the algorithm will have to be respected and the communications efficiently scheduled.

To validate the alternative approach of this research project, grid implementations of pipelined algorithms will eventually be compared to classical out-of-core implementations. The goal of such a comparison will be to characterize in what cases it is more efficient to keep data at a cluster scale even if the available memory space is not sufficient to store all data, and those where targeting grids achieves better performance.

6.2 Transparent Resource Management

Participants: Jens Gustedt, Emmanuel Jeannot, Martin Quinson and Frédéric Suter.

Our approach consists in the tuning of techniques and algorithms for a transparent management of resources, be they data, computations, networks, *etc.* 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 that are merely located at the service layer than in the 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 (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. However, it is sometimes very hard to find approximation schemes for some problems. In that case, designing heuristics remains a good alternative.

Therefore, in order to validate the proposed solutions, we will follow two complementary directions. The former deals with experimental validation of heuristics while the latter deals with designing approximation/exact algorithm.

In approximation techniques the objective is not to impose global optimality constraints but to relax them to find "*good*" solutions at a "*reasonable*" price. But, these can only be useful if we know how to analyze and evaluate them.

When we are not able to find an algorithm (either approximation or exact) for a given problem a solution to validate the approach is to experiment the heuristic on different applications and infrastructures. The experiments can be performed by simulation, emulation or on real platforms, see Section 6.3 below.

In the following we propose a set of services and algorithms that deal with resources management for grids.

6.2.1 Scheduling and Placement for Computing Resource Management

Concerning computing resources our goal is to design scheduling strategies for different kind of models of applications or architecture. We shall use standard models such as task graph or new models such as workflow (that generalizes the task graph model). However, it should be noted that fundamentally the workflow model leads to few differences in term of designing scheduling algorithms.

Multi-criteria scheduling within the GridRPC model. The GridRPC model defines an heterogeneous architecture composed of three parts: a set of *clients*, a set of *servers* and a middleware composed of an *agent* (also called a registry) or a hierarchy of agents. The agent has in charge to map a client request to a server. Several middlewares instantiate the GridRPC model DIET [die] or NetSolve [net], NINF [nin], *etc.*). The

[die] "DIET (Distributed Interactive Engineering Toolbox)", `http://graal.ens-lyon.fr/DIET`.

[net] "NetSolve", `http://icl.cs.utk.edu/netsolve`.

[nin] "The NINF project", `http://ninf.apgrid.org/`.

performance of the system depends on how efficiently the agent schedules the client requests on the resources (the servers). Moreover several criteria have to be taken into account while allocating the tasks. As shown in Section 3, these criteria can be the service time, the flow (resource utilization), the application makespan, *etc.* In this project we want to design multi-criteria algorithms or heuristics for such environments. This work will be based on our previous results on this subject [3, 32, 33]. We want to extend our results into the following direction:

1. Taking into account data locality. In some GridRPC environments it is possible to manage data by allocating it to some resources and redistribute it between servers. A good scheduling strategy is therefore needed to take into account the location of the data before scheduling the requests and to allocate the data in order to optimize the scheduling strategy.

2. Taking into account the dynamic nature of the environment. Most scheduling strategies assume that the environment will not change during the execution of the application. This simplistic assumption helped to designed powerful scheduling techniques. In this project we will extend such techniques for the case where the availability of the resources dynamically change with the time. In this case we will have to designed fault tolerant strategies based either on duplication or on resubmission. The scheduling algorithm would be able to find a trade-off between the makespan and the reliability and resource utilization according to the user and provider goals.

Scheduling parallel tasks. When computations (or tasks) exhibit a certain amount of parallelism, it may become interesting to execute them onto multiple processors. This kind of tasks is then denoted as *parallel tasks*. A major issue of grid computing is to efficiently schedule this kind of applications under the mentioned constraints of dependence and heterogeneity. Two approaches can be considered to design original parallel task scheduling algorithms. The former introduces the handling of parallelism into existing sequential task scheduling algorithms designed for heterogeneous platforms while the latter injects heterogeneity into parallel task scheduling algorithms targeting homogeneous platforms. In previous work [41] we proposed an algorithm designed following the first approach. First we aim at extending our research to the second approach.

In a second step we will use the acquired knowledge to design original algorithms taking explicitly into account the features of a grid: heterogeneous and hierarchical network, homogeneous processors sets, resource sharing, *etc.*

This last point on resource sharing opens a even larger research field. Grids are indeed intrinsically shared among multiple applications and accessed through various middlewares and resource managers. Reducing the completion time of a given application might then no more be the most prevalent optimization criterion, as said in Section 3. In order to perform multi-criteria scheduling of parallel tasks, we will have to model how parallel tasks share computing and network resources.

There are two ways to express such a sharing which directly depends on the use of a resource manager. If not, tasks share computing resources in a time-shared way. Thus we have to determine the perturbation resulting of the scheduling of a new parallel task on a given resource and taking this perturbation into account in the scheduling process. This has been studied for concurrent sequential tasks [3, 32, 33], but is more complex in the case of parallel tasks. Indeed if a new task shares a subset of the resources allocated

to another task, the perturbation will propagate to every allocated resources. Modeling these perturbations is still an open problem that we propose to address in this research project.

Scheduling stochastic workload. Most of the work that is done about scheduling task on heterogeneous platforms assumes the full knowledge of both the application and the environment. However, due to the dynamic nature of many applications executed on computational grids, it is not possible to know the arrival date and the duration of each request in advance. Therefore, in this context it is neither possible to design a static algorithm that assumes the full knowledge of the application nor a dynamic one that only assumes the knowledge of the task duration.

In our previous work [77, 27] we have proposed and studied a randomized resource broker for heterogeneous multi-level architectures where the arrival date and duration of the requests follow probabilistic distributions with fixed mean. This approach, which is complementary with the static one enables us to schedule stochastic workloads.

We want to extend randomized resource brokering algorithms to the case where tasks may fail during their execution. The failure rate will also follow probabilistic laws. We want to study duplication-based policies in this context and see how it improves the probability to correctly execute the whole application without delaying it too much.

6.2.2 Network Resource Management

Concerning network resource management our goal is to design algorithms and protocols that improve several objectives such as the communication time, the resource utilization and a good balance in the use of the resources. Moreover, this has to be done while managing the heterogeneity and the distributed nature of the target environments. Here follow examples of work we want to extend in this project.

Adaptive Online Compression (ADOC). Quickly transmitting large datasets in the context of distributed computing on wide area networks can be achieved by compressing data before transmission. However, such an approach is not efficient when dealing with higher speed networks. Indeed, the time to compress a large file and to send might be greater than the time to send the uncompressed file. With ADOC (Adaptive online compression), we have explored an algorithm that allows us to overlap communications with compression and to automatically adapt the compression effort to currently available network and processor resources.

This algorithm has been implemented into a library that is described in Section 7 on software. It features portability and efficiency without performance degradation on broad range of networks.

In this project we want to extend this work into two directions:

1. Improve performance. In the case of very fast networks we see a slight increase of latency due to the ADOC protocol overhead. We think we might be able to reduce this latency by caching some information between two corresponding send/receive calls. In the case of very slow networks, we can improve the compression by using very efficient but slow algorithms like the Burrows Wheeler algorithm used in bzip2.

2. Lossy compression for images. In some applications lossless compression is not mandatory. This is the case for some kind of images. We want to work on this

feature by incorporating lossy image compression algorithms in ADOC. In this case we have to find a tradeoff between compression rate, compression speed and image degradation.

Data Redistribution. Parallel redistribution of distributed data between clusters interconnected by a backbone appears in many applications such as code coupling, data management in GridRPC environments or parallel tasks execution.

This problem is a generalization of the well-known redistribution problem that occurs in parallelism [DDP+98].

We use the knowledge of the application in order to schedule messages and directly control the congestion. Preliminary results [43] show that this problem is NP-hard. We have proposed approximation algorithms that solve this problem in several cases (homogeneous or heterogeneous environments, with or without using the local network of the cluster, *etc.*).

We have implemented this algorithm with both MPI and POSIX sockets. Experimental results show that our algorithms outperform a brute-force TCP based solution, where no scheduling of the messages is performed.

In the project we want to extend these algorithms to the dynamic case and to complex topologies. For the dynamic case the problem consists in scheduling messages when the communication pattern is not known statically. Concerning complex topologies, the goal is to tackle the situation where more than two distributed computing centers cooperate at given moment.

Network Mapping. Information about the network is naturally mandatory to efficient network-aware applications. Unfortunately, such information in grid environment is much more complicated to gather than in other environments. Since those platforms result from the sharing and aggregation of resources between several organizations, nobody has a complete control over the inter-connexion scheme. In particular, the usual network administration tools cannot be used because nobody is granted with the required privileges over the whole network.

In [86], we present a theoretical framework and the corresponding tool for the automatic discovery of network topology. The goal is not to discover the physical layout of machines interconnections, but to construct a synthetic view of the effects of the topology as perceived by an application. Among other things, this requires that performance of concurrent transfers can be assessed.

This work contributes an original mathematical model that arises from the formalization of the topology discovery problem, for which we propose an algorithm. We prove that when this problem accepts a constellation of trees (*i.e.,* a set of trees whose roots are interconnected by a complete clique) as a solution, our algorithm finds it.

In this project, we want to extend the applicability of this algorithm. We are currently working on an extension handling the cases where the introduction of a cycle in the produced graph is required. We also plan to implement this algorithm and provide an actual topology discovery tool to grid users.

[DDP+98] F. DESPREZ, J. DONGARRA, A. PETITET, C. RANDRIAMARO, Y. ROBERT, "Scheduling Block-Cyclic Array Redistribution", *IEEE Transaction on Parallel and Distributed Systems 9*, 2, 1998, p. 192–205.

6.2.3 Data and Memory resource management.

The great success of the message passing paradigm in recent years is certainly due to a multitude of factors, *e.g.,* its standardization through MPI and the efficiency of the existing implementations. One aspect that seems to be particularly interesting for program designers is that data control within message passing environments is conceptually simple. The programmer controls the buffers that hold the data and has precise synchronization points after which he may assume that the data has been successfully transmitted. The programmer completely controls the consistency of the data.

The message passing libraries also have improved a lot in recent years on their efficiency when executed on parallel machines. But inherently due to the message passing paradigm itself they *must* suffer from the following two closely related problems:

Memory blowup: For messages, memory for the data is allocated twice, at the sending side and at the receiving side.

Extra copy operations: To perform the data transfer, the data must be copied from the sender to the recipient.

In mixed environments that are composed of clusters of SMP machines these problems will lead to a sub-optimal usage of the memory resource. On the other hand, the shared memory paradigm is not easily extendable to grid environments.

Numerous proposals have been made to extend each of the two paradigms (message passing and shared memory) to the respective other setting.

Distributed Shared Memory [dsm] systems (DSM) extend the shared memory paradigm into a distributed setting in that they provide an intermediate layer to the application such that memory access is handled transparently in one (emulated) address space. Generally, these distributed shared memory systems impose a very close cooperation between processes, are intrusive with regard to the hosting system and are optimized on a fine grained level of control. These properties make them very effective in physically close, homogeneous clusters of machines that are within the same administrative domain. We think that it is unlikely that this will scale to the opaque, heterogeneous and bandwidth oriented world of the grid.

For the message passing paradigm, the extension from MPI v. 1.2 to MPI 2 [mpib] includes so-called *one sided communication*. They allow for *Remote Memory Access*, RMA. Perhaps for many of the potential users and implementers the benefits that this paradigm offers have not outperformed these difficulties: usage of the features of MPI 2 is minor compared to v. 1.2 and other than for MPI v. 1.2 a complete reference implementation of MPI 2 in the follow up of MPICH 1.2 took several years and has only been completed recently [mpic].

As a first step to overcome the problems mentioned above, we will propose a programming paradigm and interface that aims at handling data between parallel or distributed processes that mix aspects of message passing and shared memory. This paradigm has to respect the following properties:

- It must be simple and easy to understand.

[dsm] "DSM", `http://www.ics.uci.edu/~javid/dsm.html`.

[mpib] "MPI-2: Extensions to the Message-Passing Interface", `http://www.mpi-forum.org/docs/mpi-20-html/mpi2-report.html`.

[mpic] "MPICH-2", `http://www-unix.mcs.anl.gov/mpi/mpich2/`.

- It should content itself with just a handful of functions to cover the main aspects of coarse grained inter-operations upon data.
- It must be as concrete as possible for all aspects of data control and must easily guarantee data consistency.
- It must be independent of middlewares and enclose all protocols to access data (`http`, `ftp`, `scp`, `nfs`, *etc.*) that are commonly used.

Such a paradigm will then be used to improve and simplify the algorithms that are designed for Section 6.1 and it will be integrated in our programming libraries, see Section 7.

6.3 Experimental Validation

Participants: Emmanuel Jeannot and Martin Quinson.

6.3.1 Methodologies for Experimentation

As we already have emphasized above, we believe that an experimental component for research in computer science and especially in the case of distributed and grid systems is crucial. We have to validate all components of a given setting (modeling, design, implementation) despite the imprecision about the elements and their interactions, which hinder complete predictions. In addition, the dynamics of the target platforms prevents reproducible experiments and thus makes algorithm comparisons very difficult.

As a result, several experimental paradigms are used in the community depending on the desired level of realism and the accepted amount of efforts to setup the environment. *Simulations* are rather easy to setup, but the results sometimes lack of accuracy and applicability. On the other hand, *in situ* experimentations naturally alleviate model validation issues, but require inordinate amount of time and energy to establish stable development and evaluation environments. *Emulation* solutions aim at constituting a tradeoff between the two extremes, being easier to setup and use than real platform while providing more accurate information than simulations.

Concerning simulation, we contribute to the SIMGRID toolkit which enables the simulation of distributed applications in distributed computing environments for the purpose of both algorithmic studies and software development. As detailed in Section 7.3.1, this software is the result of collaborations between several laboratories both in France and in the US. Our participation consists in exploring how to simplify the large scale distributed algorithm study with SIMGRID. In particular, we investigate automatic solutions to change the prototypes developed within the framework into ready-to-use applications.

Concerning emulation, we work in the ACI *Masse de Données* GRID-Explorer at designing an environment to emulate heterogeneity. GRID-Explorer is indeed an homogeneous cluster designed to perform grid experiment. As distributed systems are mainly heterogeneous it is required to transform this homogeneous environment into an heterogeneous one. Our tool, called Wrekavoc [35], is doing that by degrading the performance of nodes and the network. We target mainly the CPU power, the available memory, the network bandwidth and latency. The scientific issues are how to control the degradation, how scalable is the proposed solution, and how the emulated environment compared with regard to a real one. With Wrekavoc we will have a tool that:

- helps in testing algorithms designed for heterogeneous environments,
- controls the heterogeneity,
- provides reproducible experiments,
- allows quantitative comparison of algorithms and real applications.

Concerning real-world platform and *in situ* experimentations, we are involved in GRID5000 [36] which aims at designing an distributed instrument for grid experiments. More precisely, nine sites spread across France (Lille, Rennes, Orsay, Bordeaux, Lyon, Grenoble, Toulouse, Nice and lastly Nancy), are hosting clusters linked together by a high-speed backbone. These nine clusters form a grid on which reproducible experiments can be conducted. ALGORILLE, is responsible of the Nancy site for acquiring, maintaining and providing to the community the cluster and the experimental environment. With GRID'5000 we have a platform for experiments in real life condition:

- Address critical issues of Grid system/middleware: programming, scalability, fault tolerance, scheduling, etc.
- Address critical issues of Grid Networking: High performance transport protocols, QoS
- Port and test applications.
- Investigate original mechanisms: P2P resources discovery, Desktop Grids.

6.3.2 Modeling experimental conditions

The border between simulation and emulation is often very difficult to draw. Some emulation solutions rely on a network simulator to mediate the communications in the system. On the other hand, the SIMGRID simulator is able to actually run some parts of code to benchmark their performance, just like in an emulator.

Any model of a computational system that aims at building a simulator or an emulator can be decomposed in several interacting sub-models:

Task model It describes how a sequential application uses the local resources such as processor and memory as well as how these resources get shared between concurrent applications. Numerous models are presented in the literature. The simplest one is to consider that machines deliver a fixed computational power (given in Mflops per second) and that tasks require a given amount of computation to get achieved. It neglect memory and cache effects, and as well as any technical details that allow for obtaining more computation power from a given machine by low-level optimizations. On the other end of the scope, one can completely emulate a given architecture, *i.e.,* computing exactly the actions a given processor would have to do to run the target application. Similarly, virtualization consists in running the target application on the physical computational facilities after some alteration to make them compatible.

Network model It describes how data is exchanged between machines. The traditional approach consists in simulating the flow of data packets along the wire, using a realistic networking stack such as TCP/IP. Other approaches are possible, and SIMGRID models data flows in an abstraction close to fluid mechanics.

Application model It describes the behavior of the applications run on top of the simulated system, and the interactions between the sequential computations and the communication. It can range from a theoretical abstraction such as Direct Acyclic Graphs (DAGs) of tasks to the code of an actual implementation.

Platform model It describes the hardware elements of the system, and their interactions. Grids entail specific challenges to platform modeling because of their heterogeneity. In contrary to classical platforms, it is unrealistic to assume that the operator knows every details of their running machines. Because of the scale of target platforms, it is also impossible to describe the platform in every details since the resulting model would be intractable in practice.

Usage scenario The performance delivered by a given algorithm naturally depends on the use conditions. The scalability of an algorithm captures one part of this idea, where a program behave correctly under some conditions (N requests per second, N clients for one master, N participating nodes, *etc.*) but suffer of a dramatically performance decrease when reaching a given threshold.

Through projects like SIMGRID, our team is deeply involved in this modeling effort. The main challenge is to come up with *realistic* models, to *validate* them and to *instantiate* them.

Given the scales we target, SIMGRID relies on rather simplistic models such as fixed power for computational resources and fluid flows for the network. The application can be either modeled through an actual implementation or a DAG of tasks. The platform is represented as a graph interconnecting computational nodes with links. A routing function is also associated to that graph to capture the fact that the routing on real platforms do not follow a globally consistent schema.

We are currently working in collaboration with the university of Hawai'i at Manoa on the validation of the networking model by comparing the results for a given simulation obtained in SIMGRID with the ones obtained with packet-level simulator such as NS2 or GTNets.

On the instantiation axis, the main difficulty is about platforms. In [87], we present an original platform discovery algorithm allowing to capture the possible contention caused by concurrent data flows. We are currently implementing this algorithm. Afterward, we will use the resulting tool to capture the topology of several existing platforms. We expect this to give us more insight into the structure of *realistic* platforms, such that we will also be able to propose a generator of realistic synthetic platforms.

7 Software

To be able to validate our research experimentally it is crucial for us to have toolboxes that allow for a quick implementation of prototypes and their subsequent evaluation. For the application layer this is PARXXL, which allows for an implementation of coarse grained algorithms that is independent from the middleware. For the service layer this is (or will be) GRAS that allows an easy assemblage and evaluation of services.

The layers of both libraries (the application layer for PARXXL and the service layer for GRAS) are separated by the middleware layer. As a consequence these two libraries do not interact directly and need not to be integrated closely in the near future. This picture might slightly change, if interfaces that we develop for the service layer (such as DHO, see below) prove to be interesting also for the applications.

In addition to these two general toolboxes we develop more specific software to provide tools for specific application domains (`par::cell` and `par::cellnet` layers of PARXXL for neural networks, and for large graphs) and for specific services (PATATE for scheduling, ADOC for network compression, DHO for data abstraction).

7.1 `par::step` – Implementing Coarse Grained Algorithms

Participant: Jens Gustedt and Stéphane Vialle.

Our previous library SSCRAP was developed to ease the implementation, test and benchmark algorithms that are written for the PRO model. This prototype of a C++-library that was initially developed together with Isabelle Guérin Lassous from the project-team Ares [are]. SSCRAP is now fully integrated in a new library called PARXXL, that combines it with PARCEL, see below. Here, the interface layer `par::step` offers a comfortable interface for implementing coarse grained algorithms.

This library takes the requirements of PRO, see Section 6.1.1, 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. PARXXL is scheduled to be publicly available in autumn 2006[par]. The latest version of SSCRAP can be fount at [ssc]. PARXXL integrates

- a layer for message passing with MPI [mpia],
- a layer for shared memory with POSIX threads [pth], and,
- a layer for out-of-core management with file mapping (`mmap` [mma] system call).

[are] "ARES project", `http://citi.insa-lyon.fr/jsp/site/Portal.jsp?page_id=10`.

[par] "ParXXL", `http://parxxl.gforge.inria.fr/`.

[ssc] "SSCRAP", `http://www.loria.fr/~gustedt/sscrap/`.

[mpia] "Message Passing Interface (MPI)", `http://http://www-unix.mcs.anl.gov/mpi/`.

[pth] "POSIX thread", `http://www.humanfactor.com/pthreads/pthreadlinks.html`.

[mma] "Map pages of memory (mmap)", `http://www.opengroup.org/onlinepubs/007908799/xsh/mmap.html`.

All three different realizations of the communication layer 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, see [52, 59, 92].

Data Handover

Due to the instability of the systems that we considered for passing over to heterogeneous environments, we are not yet able to use message passing and shared memory simultaneously. In fact we realized that this difficulty is symptomatic for the lack of a uniform treatment of data on the different platforms. They put different emphasis on efficiency, robustness or data consistency and therefore cannot easily be combined.

To address this issue we will separate the parts that handle data from SSCRAP. They will be collected into a new library, DHO, that consistently handles data *Data Handover* (DHO), see [83]. This is a programming paradigm and interface that aims to handle data between parallel or distributed processes that mixes aspects of message passing and shared memory. It is designed to overcome the potential problems in terms of efficiency of both: (1) memory blowup and forced copies for message passing and (2) data consistency and latency problems for shared memory. Our approach attempts to be simple and easy to understand. It contents itself with just a handful of functions to cover the main aspects of coarse grained inter-operation upon data.

`par::cell`: programming cellular networks

Because of its interesting properties concerning the modeling of small entities, distributed computation within the cellular framework is nowadays adopted within many application domains. These include for example simulations in physics to resolve local equations, multi-agent systems to model colonies of insects or cortical systems. Such simulated systems usually operate on thousands of interconnected entities.

PARCEL was initially jointly developed by Supélec and the Cortex project-team at LORIA. Its actual development is mostly done by Stèphane Vialle and his group at Supélec. PARCEL is a modeling framework and library suite that allows the creation and efficient execution of such large systems on mainframe multi-processors. The fact that PARCEL can not be executed in a distributed environment constitutes an important constraint for its usefulness: applications want to go beyond the limits in size that are imposed by installed mainframe hardware and to be able to execute on cheap and widely available resources.

Achieving good performance for such fine grained systems in a distributed context such as clusters and grids is a major challenge and needs a careful design and integration. These must in particular address the network latency problem which, if not handled, would in most cases make such a system useless for practical purposes.

To tackle these difficulties, PARCEL now is merged together with SSCRAP into our new library PARXXL, layers `par::cell` and `par::cellnet`. First benchmarks show that PARXXL can be used to address these issues, see [59]. Thereby we provide a scalable and portable base for programming cellular networks and obtain valuable feedback for the BSP oriented modeling and design approach.

Graph Algorithms

We will also use SSCRAP as a testbed to implement different types of new graph algorithms, basic primitives as well as algorithms that are more application driven. The primitives concern things such as list ranking, maximal independent sets, coloring heuristics or recursive shelling. For these more and broader experiments are needed to be capable to compute on very large data sets. In addition, with our software we want to be able to take advantage of the well known structural properties of graphs that occur in applications.

Algorithms that are more application driven then concern *e.g.,* clustering, which tries to structure a graph in local neighborhoods of vertices that have similar properties. Being practicably able to cluster large graphs is highly demanded by real life applications (social networks, web graphs). It also can be an important subroutine for parallelizing other programs efficiently: good parallelizations often need a good domain decomposition.

7.2 ADOC– Transport Compression on the Fly

Participant: Emmanuel Jeannot.

Data transfer is a key feature for computational and data grids. Such grids have to rely on efficient data transmission services that are able to provide fast transfer rate. Compression is one mean to increase the bandwidth seen by the application. However, the heterogeneous and dynamic nature of the grids required to adapt the compression to the environment.

We have designed a library called ADOC (Adaptive Online Compression). The main features of the ADOC library are:

- The compression level is adapted according to the environment (current speed of the network and CPUs) and the data. The compression is lossless.
- It provides compression and communication overlap. ADOC is able to compress some part of the data while sending compressed or uncompressed packets.
- It works on a broad range of networks (up to Gbit LAN)
- It is easy to incorporate into any existing software. ADOC is thread-safe and its API is very close to the read/write POSIX system calls and respects their semantics.
- It is ported on many UNIX like systems (LINUX (32/64 bits), SunOS, Darwin, Cygwin, *etc.*)
- It has a low latency: for small messages ADOC gives the same performance as POSIX read/write (up to 100 MBit LAN).

We have tested this library under various conditions with various data types. The performance gain obtained using ADOC depends on the data itself and the environment and can be very important (up to 6 times faster).

This library is intended to be used in any middleware that performs data transfer. We have incorporated the library into NetSolve. This was done easily thanks to the API close to the read and write system call. The performance of NetSolve with ADOC is never worst than NetSolve alone. Most of the results show an increase of performance for NetSolve with ADOC.

We also have faced some performance issues that have been solved has follow:

Very fast networks: We have previously incorporated LZF (Lev-Zimpel-Free) into ADOC in order to speedup compression for fast Ethernet networks. For gigabit networks, we test the bandwidth, and disable the compression for such network.

Compression level divergence: If the receiver is very slow compared to the sender, the size of the fifo queue will increase. This will lead to increase the compression ratio. Then the receiver will take more time to decompress the data, which will lead in an increase of the size of its fifo queue, and so on. As we want to respect the read/write system call semantic, the receiver cannot send a message back to the sender. Our solution is to record visible bandwidth for each compression level and disable those that have a lower bandwidth than the lowest compression level.

Small messages: The ADOC mechanism (thread, mutexes, protocol overhead, *etc.*) increases the latency. This becomes critical for small messages. Therefore we have a simpler (no cost) protocol, without compression for small messages.

Compressed or random data: For compressed or random data, compression is not useful as it degrades the performance. When we detect such data, we disable the compression for the next second.

We want to implement and test the work described page 19 on the ADOC algorithm. It concerns both performance improvement and lossy compression for images.

7.3 Simgrid

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). The complete SIMGRID suite is freely downloadable [sim]

7.3.1 GRAS – Grid R&D assisted by simulator

Participants: Martin Quinson and Jens Gustedt.

As mentioned in Section 6.3, simulation is a common answer to the grid specific challenges. 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.

GRAS is the result of a collaboration with Rich Wolski (Univ. of California, Santa Barbara) and is now included in the SIMGRID tool suite.

The GRAS framework emphasizes on the following characteristics:

[sim] "SimGrid", `http://simgrid.gforge.inria.fr/`.

Simplicity Our goal is to provide a high level interface masking common implementation details to the users and letting them concentrate on the algorithmic difficulties of their applications. To that extend, an high level message passing interface is offered. Applications can attach callbacks to specified message types. The system pools the network for incoming messages and dispatches them.

In contrary to most existing systems in that context, the messages are strongly typed, and all messages of a given type will have the same payload structure. The payload can be any data type representable in C (scalar, array, structures, enumeration, unions) and can contain pointers of any kind. Data are then automatically passed by deep-copy from one host to another.

To keep the system simple to use, a C data type parser is provided, sparing the users from manipulating error-prone meta-data. Some rudimentary tools for source code deployment, remote compilation and distributed management are also provided.

Portability To address the system heterogeneity presented by the grid, GRAS uses a plain ANSI C implementation, modern compilation configuration tools like `autotool` and `automake` and has no uncommon dependency on other tools or libraries. Thanks to this approach, GRAS is known to work out of the box on the following platforms: LINUX (X86, IA64, AMD64 ALPHA, SPARC, HPPA and PPC), SOLARIS (SPARC and X86), Mac OSX, IRIX, AIX and WINDOWS.

Performance On real platforms, a particular attention was given to structured data exchange efficiency. In the literature, high performance communication libraries do not address the costs due to conversions between hardware architectures (little- vs big-endian or 32 vs 64 bits) and the existing solutions such as XDR offer poor performance despite the quality of the transfer solutions.

Our approach consists in transferring the data in the sender native format and do only one conversion (on the receiver side) when both architectures differ. Underway experimentations show that this allows resulting applications to run almost as fast as the MPI cluster-world standard. This is encouraging since our framework also transparently address hardware heterogeneity, in contrary to most common MPI implementations.

Concerning the simulation part of the GRAS framework, performance is achieved by the SIMGRID simulator kernel using very simple and fast models, along with trace-based simulation. This approach enables to have much better acceleration factors (hence a better scalability) than other grid emulation projects like MicroGrid [SLJ+00].

We plan to improve the provided tools for distributed application management and to contribute several grounding bricks of distributed applications, such as network mapping (see page 20) and system monitoring; lightweight distributed database; remote resource management; security, *etc.* This list is of course ambitious and we only plan to provide some provision for each of these problems.

We also plan to work further on the framework itself on two axis: performance could be further improved by an adaptive compression schema such as ADOC while

[SLJ+00] H. J. SONG, X. LIU, D. JAKOBSEN, R. BHAGWAN, X. ZHANG, K. TAURA, A. A. CHIEN, "The MicroGrid: a Scientific Tool for Modeling Computational Grids", *in : Supercomputing*, 2000.

interoperability with standards such as CORBA could be achieved by multiplexing the protocols depending on the messages first bytes.

7.3.2 PATATE – Parallel Task Simulation

Participants: Frédéric Suter and Tchimou N'Takpé.

The objective of the SIMGRID *Parallel Task Toolkit Extension* (PATATE) is to provide a simulation framework specific to parallel computations and resources. This software development project is strongly connected to researches on parallel task scheduling presented in Section 6.2.1.

To be able to simulate the execution of multi-processors tasks on parallel resources, we have to address several issues while designing this software:

Parallel computing resource model. As for now, SIMGRID computing resources are sequential and represent single *hosts*. There are two main approaches in order to provide a SIMGRID resource representing a parallel computing resources. The former is to model such a resource as a batch scheduler queue involving a fixed number of hosts, while the latter offers the capacity to aggregate any number of sequential resources.

Parallel task and data movement model A SIMGRID task stands either for a sequential computation or a point-to-point communication. A SIMGRID parallel task will thus represent a parallel computation or a data redistribution. Different models of parallel computation exist such as a DAG decomposition (each task of the graph being sequential), coarse grain model (concurrent sequential operations within a step) or speedup model (*e.g.,* inspired from Amdahl's law) even if this last option implies to ignore intra-task communication. Concerning data redistribution, a bipartite graph representation as proposed in Section 6.2.2 could be used.

Resource sharing model The way how parallel tasks share parallel computing resources directly depends on the parallel resource model. With the batch scheduler queue model, computing resources are not shared as only one task can be scheduled on a host. But network resources are still shared at a cluster scale. With the aggregation model, both network and computing resources are shared. There we have to determine the perturbation of a new task on those already executing on a given resource, as mentioned in Section 6.2.1.

8 Expected results and criteria of success

Widen the field of applications and algorithms and ease their design. The part of the ALGORILLE project concerning the structuring of applications aims at defining models of infrastructures and applications in order to ease the design of new algorithms and their implementations in the context of grids. We expect to design new algorithms (*e.g.,* for clustering and classification) for large graphs which are defined by interaction networks such as the web graph, peer-to-peer graphs, etc. We will extend the PRO model to hierarchical and then heterogeneous environments. We expect to provide a SSCRAP interface for the cellular computation library PARCEL.

Efficient combination of different resources and their managers. Concerning the resource management part of this project, we aim at designing and implement new algorithms for managing the three resource types CPU, network and storage. We target different kinds of infrastructure and middleware, and we expect to have new efficient algorithms (scheduling, transfer, *etc.*) in this context.

Moreover, we expect to be able to integrate the solutions for the different resources into a more unified framework. In the case of GridRPC environments, we will target an efficient scheduling algorithm that will combine the available computing resources with an ability of the environment to ensure persistence (which requires storage management) and to perform data redistribution (which requires network management).

Scalability to large data sets and environments. Our solutions will work for large environment; for each of these we will define and validate their application scale. Concerning large data sets we expect to provide algorithms handling out-of-core data that are able to efficiently use remote memory resources.

Sound scientific evaluation and reproducibility of grid experiments. The experimental part of this project will contribute to define the methodology and to implement tools in order to be able to tests and validate services and algorithms for grid middleware. Our experiments will be designed to fulfill well specified requirement such as reproducibility and extensibility.

This experimental validation will include simulations, emulations, and experiments on real scale platform. We will contribute on each of these cases by building/enhancing tools (SIMGRID, GRAS, etc.) and by participating in community efforts, be they national (Grid5000, Grid Explorer) or European (CoreGrid).

Criteria of success. In addition to the specific goals that are given above, the principal criterion of success for our project will be the acceptance of our work by the community. This concerns all different facets of our work: modeling, algorithm design, implementations as well as experiments. We would highly appreciate if our layered view of a grid environment (in particular the separation of service layer and middleware) would be more widely accepted, and, if in general we would be able to sensitize for the inherent conceptual problems of grid computing that exist beyond the severe engineering difficulties as they have mainly been investigated up to now.

9 Project-team positioning

Nowadays many teams (INRIA, national and internal) do research on grid computing. Up to our knowledge most of them are guided by applications or by the aim to create or consolidate middlewares.

Obviously the different research subjects of our proposal are also investigated by others, but we think that our approach is original *in its entirety* in that it studies the potential of grids in its foundations and to propose algorithmic solutions that are verified by experiments.

9.1 French teams studying grids

9.1.1 Other INRIA teams

Several INRIA teams are interested in grids, especially in grid middlewares, *i.e.,* they design realize different software layers for the deployment of tasks on a grid. Often these middlewares are driven by specific applications that strongly influence the design of the middleware. In general their work is complementary to ours, but they give rise to punctual collaborations.

Most of the teams are organized in the same INRIA theme as ALGORILLE, "Num B"[2]. Two (*Oasis* and *Sardes*) are in the theme "Com A"[3]. We may roughly put these teams into four groups:

- *GRAAL*, *MESCAL* and *MOAIS* emphasizes on grid middlewares and components and also handle algorithmic aspects of these.
- *Grand Large*, *Paris* and *Runtime* center around low-level environments (system layer) and their efficiency and robustness. Two of them (Grand Large and Paris) also have a major interest in peer-to-peer networks.
- *ScAlApplix* and *SAGE* are centered around scientific computing.
- *Oasis* and *Sardes* are interested in software components of grids.

In particular, none of these teams has a focus on the modeling of algorithms for large scale applications that could be compared to our first theme of research, see Section 6.1.

In the following we provide some brief details on these teams.

GRAAL If the GRAAL project studies scheduling strategies and algorithm design for heterogeneous platforms as ALGORILLE does, it mainly focuses on parallel applications composed of sequential tasks. Members of this project thus have interest in steady state scheduling and divisible load applications. In our project we take a complementary approach by studying moldable tasks, *i.e.,* parallel tasks alloted on a flexible number of processors and investigating stochastic scheduling strategies.

Another research topic of GRAAL focuses on environments and tools for the deployment of applications in a client-server mode, mainly with the design and development of the DIET middleware. Our work at the service layer level is and will continue to be integrated into that middleware as strong connections

[2] Grids and high-performance computing
[3] Distributed systems and software architecture

exist between the two teams. GRAAL is very close of our research interests as shown by our numerous collaborations and joint papers. For instance GRAAL is involved in the ARC INRIA OTaPHe in which we aim at including moldable task scheduling algorithms into the DIET middleware.

MESCAL The MESCAL project aims at designing and validating middleware and services in order to efficiently exploit large distributed infrastructures. Their applications are intensive scientific computations and their methodology is based on the design of building bricks that scale efficiently, using modeling and performance evaluation of target architectures, software layers and applications. The research directions of MESCAL include modeling, simulation, evaluation and optimization of computation grids and more generally, of large discrete event systems, using deterministic as well as stochastic techniques. The design of the SIMGRID toolkit is a joint effort of the two teams. The MESCAL project develops the simulation core of the toolkit, while the ALGORILLE team is more involved in the design of GRAS, a abstraction layer allowing to seamlessly target either simulated and real world.

MOAIS The MOAIS project focuses on the programming of applications where the effective use of a larger number of resources is expected to enhance the performance. This encompasses large scale scientific interactive simulations that involve various resources. The research axes of the MOAIS project are focused on the scheduling problem with a multi-criteria performance objective. The MOAIS approach is to use the application's adaptability to enable its control by the scheduling. The critical points concern designing adaptive moldable algorithms and coupling the various components of the application to reach interactivity with performance guarantees. This last research topic is connected to our work on parallel task scheduling and is the subject of ARC OTaPHe. But where MOAIS focuses on the theoretical design of guaranteed scheduling algorithms, ALGORILLE develops more pragmatic heuristic and validates their efficiency through simulation and experimentation.

Grand Large's approach concerns middleware and low-level programming environments, between low-level system mechanisms and high-level programming environments. Grand Large has three main axis that are quite complementary to the ones of ALGORILLE. They provide the opportunity of many collaborations, for the time being mainly via the projects GridExplorer and Grid5000, see Section 10.2 below.

Paris The Paris project-team focuses on resource management but more for clusters than for grids, with the development of Kerrighed. Collaborations exist between Paris and ALGORILLE as results on data redistribution have been integrated into the software tools for distributed numerical simulation developed in this project.

Runtime The Runtime project aims at mastering large scale heterogeneous configurations. Their focus is on the infrastructure level and complementary to our algorithmic approach.

ScAlApplix is a project in which high performance algorithms are designed and implemented but with a particular focus on distributed numerical simulations.

SAGE is centered around scientific and numerical computing and its transpose on grids, with a particular focus on environmental applications.

Oasis focuses its activities on distributed (Grid) computing and more specifically on the development of secure and reliable distributed systems using distributed asynchronous object systems.

SARDES aims to develop dependable and highly adaptable software infrastructures, by systematically leveraging component-based and reflective programming techniques.

9.1.2 French Teams not connected to INRIA

There are a lot of non-INRIA laboratories in France that cover some aspects of our research themes. The following list is far from being exhaustive: LIFC (Franche Comté), IRIT (Toulouse), LSIIT (Strasbourg), LaRIA (Amiens), LIPN (Paris XIII), LIFL (Lille), LIFO (Orleans).

9.2 International Scientific Context

Research on grids is very active. Initiated in the United States by projects like Globus [glo], they spread worldwide.

A list of large grid projects can be found here [ggf] This list is not meant to be exhaustive and deals mainly with production grids. Among these projects let us cite the following:

- The TeraGRID [ter] project (USA) aims to federate large computing centers within the US for executing large-scale scientific applications.
- The NAREGI [nar] initiative (Japan) aims to build a nationwide grid infrastructure to conduct research and development on grid software and network technology.
- The EGEE [ege] project (EU) goal is to develop a service grid infrastructure for scientific computation.
- The GRIBUS [gri] project (Australia and South Asia), is engaged in the creation of open-source specifications, architecture for eScience and eBusiness applications.

These projects are very large and involves a lot of scientists and engineers. They also raise very interesting algorithmic challenges that (for some of them) will be tackled by the ALGORILLE project.

Among the few research grids let us cite the DAS-2 project [das] (Netherlands) which aims to give grid researcher an infrastructure to develop and experiment innovative solutions before bringing them to production.

Several smaller-size international projects are also related to ALGORILLE: research on NetSolve with the Univ. of Tennessee, Knoxville; scheduling and SIMGRID with the Univ. of Hawaii, Manoa; and NWS with the Univ. of Santa-Barbara.

[glo] "The Globus Alliance", `http://www.globus.org/`.

[ggf] "Grid projects", `http://www.gridforum.org/ggf_grid_projects.htm`.

[ter] "The Teragrid project", `http://www.teragrid.org`.

[nar] "National Research GRID Initiative (NAREGI)", `http://http://www.naregi.org/index_e.html`.

[ege] "EGEE Homepage", `http://public.eu-egee.org/`.

[gri] "The Griibus project", `http://www.gridbus.org/`.

[das] "The Distributed ASCI Supercomputer 2 (DAS-2)", `http://www.cs.vu.nl/das2/`.

Where there is a great effort in the international community for middleware aspects and resource management on grids, much less is apparent for the theoretical foundation and modeling of grid algorithms, applications and architectures. The worldwide research on BSP-related algorithms and other models for coarse grained architectures, programming models and libraries is summarized on the BSP Worldwide page [bwp]). It mentions several implementations of the BSP library standard, but which does not seem to be followed very actively anymore. In particular there is no collection of reproducible benchmarks that would make a comparison between existing implementations possible. As such the predictions of running times for which the BSP model allows are mostly carried out to prove the relative speedup (ideally doubling the number of processors halves the running time). Such predictions remain theoretical as long as they don't prove to be competitive to ad-hoc code on realistic platforms.

Therefore, the BSP programs that are currently developed often seem to be based directly on standards that are more widely accepted, in particular MPI. The most noticeable exception is a BSP-toolbox for the Python language, that was developed in the context of scientific applications[HLSØ06], where one of the main goals is the shortening of the development cycle.

Generally MPI based BSP programs achieve good performances when executed in cluster environments, *i.e.,* homogeneous environments that are geographically close. Their extension to the context of heterogeneous and distributed grids remains a major challenge.

10 Collaborations

10.1 Collaborations with other INRIA project teams

ARC OTaPHe 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.

ODL SimGrid The INRIA funded an ODL (Opération de Développement Logiciel – Software development action) on the SimGrid project. A full-time engineer will be hired to improve and sanitize the codebase and make sure that it becomes a widely used tool for other teams of the research area.

10.2 Collaborations with other French research groups

ACIs *GRID* and *Masses de Données*

A great part of the French grid activities had been kicked off by the initiative ACI GRID of the French Ministry of Research. This program, to which we participated

[bwp] "BSP Worldwide", `http://www.bsp-worldwide.org/`.

[HLSØ06] K. HINSEN, H. P. LANGTANGEN, O. SKAVHAUG, Å. ØDEGÅRD, "Using BSP and Python to Simplify Parallel Programming", *Future Generation Computer Systems 22*, 1-2, 2006, p. 123–157.

within several projects has been followed by ACI *Masses de Données* on large data sets.

The following is a list of the ACI/ANR projects to which we participate or have participated.

GridExplorer This project aims to construct an instrument for the emulation of grids. It is now (march 2005) in its test phase and will finally consist of about 1000 CPUs.

Grid5000 This project [36] aims to construct a realistic grid testbed that is distribute all over France. Its goal is to group together 5000 CPUs. Emmanuel Jeannot and Martin Quinson are the project leaders for the Nancy site. The Nancy site is connected since December 2005.

AGIR AGIR is a multi-thematic research aiming at developing research in the area of grid calculation in order to implement new medical images analysis algorithm.

ALPAGE ALPAGE is a research project seeking for efficient solutions for large scale systems design following four complementary axes: Large scale distributed platform modeling; Overlay networks topologies; Scheduling for regular parallel applications; Scheduling for file-sharing applications. Martin Quinson participates more precisely on the first axis.

10.3 Collaborations with foreign research groups

CoreGrid We participate at the European CoreGRID network of excellence. We are involved in the work package 6 : "resource management and scheduling".

Univ. of Brussels We collaborate with the Université Libre de Bruxelles (Prof. J. Goossens) on scheduling stochastic workload and scheduling real-time system with fault-tolerance.

Univ. of Bergen We collaborate closely with the group of Prof. Jan Arne Telle. This collaboration already produced a lot of joint publications. The collaboration has already been financed by several means, in particular visiting grants from the Lorraine Region and by to AURORA projects.

Univ. Tennessee We participate in a joint NSF/INRIA program about NetSolve between the Univ. of Tennessee (Prof. Dongarra) and INRIA (teams GRAAL and ALGORILLE).

Univ. of California, Santa Barbara We collaborate with Prof. Rich Wolski on grid platforms monitoring and characteristics discovery within the NWS project.

Univ. of Hawaii, Manoa We collaborate with Prof. Henri Casanova on the simulation of grid platforms within the SimGrid project, as well as on the design of heuristics for parallel task scheduling in heterogeneous environment.

Project-team bibliography

The following is a complete bibliography of the members of the team starting in 2001. In particular, it includes publications that are in not affiliated with INRIA. By doing so we aim to give a complete picture of the scientific achievement of our team and this does not constitute a claim for the respective publications by INRIA. The affiliation of an author of these papers appear in the publications themselves and remain unchanged.

Books and Monographs

[1] J. GUSTEDT, E. JEANNOT, J.-L. PAZAT, S. VIALLE (editors), *École GRID 2002*, INRIA, December 2002. École thématique sur la globalisation des ressources et des données, Aussois, France.

[2] S. VIALLE, J. GUSTEDT, E. JEANNOT (editors), *GridUSe-2004 : École thématique sur la Globalisation des Ressources Informatiques et des Données : Utilisation et Services*, Supelec, June 2004.

Doctoral dissertations and "*Habilitation*" thesis

[3] Y. CANIOU, *Ordonnancement sur une plate-forme de métacomputing*, PhD thesis, Université Henri Poincaré, December 2004.

[4] M. ESSAÏDI, *Echange de données pour le parallélisme à gros grain*, PhD thesis, Université Henri Poincaré, February 2004.

[5] M. QUINSON, *Découverte automatique des caractéristiques et capacités d'une plate-forme de calcul distribué*, PhD thesis, École Normale Supérieure de Lyon, December 2003.

[6] F. SUTER, *Parallélisme mixte et prédiction de performances sur réseaux hétérogènes de machines parallèles*, PhD thesis, École Normale Supérieure de Lyon, November 2002.

[7] S. VIALLE, *Synthèse des recherches et perspectives en : Parallélisation de Systémes de Calculs Distribués d'Orientation Cellulaire, sur Architectures MIMD*, Habilitation à diriger des recherches, Univ. Henri-Poincaré – Nancy 1, 2002, in French.

Articles in referred journals and book chapters

[8] V. BERTEN, J. GOOSSENS, E. JEANNOT, "*On the Distribution of Sequential Jobs in Random Brokering For Heterogeneous Computational Grids*", *IEEE Transactions on Parallel and Distributed Systems 17*, 2, 2006, p. 113–124.

[9] K. BERTET, J. GUSTEDT, M. MORVAN, "*Weak-Order Extensions of an Order*", *Theoretical Computer Science 304*, 1–3, July 2003, p. 249–268.

[10] M. BOUZID, V. CHEVRIER, S. VIALLE, F. CHARPILLET, "*Parallel Simulation of a Stochastic Agent/Environment Interaction*", *Integrated Computer-Aided Engineering 8*, 3, 2001.

[11] Y. CANIOU, E. JEANNOT, "*Multi-Criteria Scheduling Heuristics for GridRPC Systems*", *International Journal of High Performance Computing Applications 20*, 1, spring 2006, p. 61–76.

[12] E. CARON, S. CHAUMETTE, S. CONTASSOT-VIVIER, F. DESPREZ, E. FLEURY, C. GOMEZ, M. GOURSAT, E. JEANNOT, D. LAZURE, F. LOMBARD, J.-M. NICOD, L. PHILIPPE, M. QUINSON, P. RAMET, J. ROMAN, F. RUBI, S. STEER, F. SUTER, G. UTARD, "*Scilab to Scilab//, the OURAGAN Project*", *Parallel Computing 27*, 11, October 2001, p. 1497–1519.

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

[14] E. CARON, F. DESPREZ, E. FLEURY, F. LOMBARD, J.-M. NICOD, M. QUINSON, F. SUTER, *Calcul réparti à grande échelle*, Hermès Science Paris, 2002, ch. Une approche hiérarchique des serveurs de calculs, ISBN 2-7462-0472-X.

[15] E. CARON, F. DESPREZ, M. QUINSON, F. SUTER, "*Performance Evaluation of Linear Algebra Routines*", *International Journal of High Performance Computing Applications 18*, 3, 2004, p. 373–390, Special issue on Clusters and Computational Grids for Scientific Computing (CCGSC'02).

[16] E. CARON, F. DESPREZ, F. SUTER, "*Parallel Extension of a Dynamic Performance Forecasting Tool*", *Scalable Computing: Practice and Experience 6*, 1, March 2003, p. 57–69, Special issue on selected papers of ISPDC'02.

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

[18] M. COSNARD, E. JEANNOT, T. YANG, "*Compact Dag Representation and its Symbolic Scheduling*", *Journal of Parallel and Distributed Computing 64*, 8, August 2004, p. 921 – 935.

[19] M. COSNARD, E. JEANNOT, "*Automatic Parallelization Techniques Based on Compact DAG Extraction and Symbolic Scheduling*", *Parallel Processing Letters 11*, 1, 2001, p. 151–168.

[20] E. DAHLHAUS, J. GUSTEDT, R. M. MCCONNELL, "*Efficient and Pratical Algorithms for Sequential Modular Decomposition*", *Journal of Algorithms 41*, 2, November 2001, p. 360–387.

[21] E. DAHLHAUS, J. GUSTEDT, R. M. MCCONNELL, "*Partially Complemented Representations of Digraphs*", *Discrete Mathematics and Theoretical Computer Science 5*, 1, 2002, p. 147–168, `http://dmtcs.loria.fr/volumes/abstracts/dm050110.abs.html`.

[22] F. DESPREZ, F. SUTER, "*Impact of Mixed-Parallelism on Parallel Implementations of Strassen and Winograd Matrix Multiplication Algorithms*", *Concurrency and Computation:Practice and Experience 16*, 8, July 2004, p. 771–797.

[23] A. H. GEBREMEDHIN, I. GUÉRIN LASSOUS, J. GUSTEDT, J. A. TELLE, "*Graph Coloring on a Coarse Grained Multicomputers*", *Discrete Applied Mathematics 131*, 1, September 2003, p. 179–198.

[24] I. GUÉRIN LASSOUS, J. GUSTEDT, "*Portable List Ranking: an Experimental Study*", *ACM Journal of Experimental Algorithmics 7*, 7, 2002, `http://www.jea.acm.org/2002/GuerinRanking/`.

[25] M. QUINSON, "*Un outil de prédiction dynamique de performances dans un environnement de metacomputing*", *Technique et Science Informatiques 21*, 5, 2002, p. 685–710, Numéro spécial RenPar'13.

Publications in Conferences and Workshops

[26] W. BEN FRAJ, M. ESSAÏDI, J. GUSTEDT, "*Performance Implications by the Hierarchical Design of Clusters*", *in : The 7th world multiconference on Systemics, Cybernetics and Informatics (SCI'2003)*, Orlando, FL, July 2003.

[27] V. BERTEN, J. GOOSSENS, E. JEANNOT, "*A Probabilistic Approach for Fault Tolerant Multiprocessor Real-time Scheduling*", *in : 14th Workshop on Parallel and Distributed Real-Time Systems*, Island of Rhodes, Greece, April 2006.

[28] V. BOUDET, F. DESPREZ, F. SUTER, "*One-Step Algorithm for Mixed Data and Task Parallel Scheduling Without Data Replication*", *in : Proceedings of the 17th International Parallel and Distributed Processing Symposium (IPDPS'03)*, April 2003.

[29] V. BOUDET, F. SUTER, "*Algorithme d'ordonnancement mixte à étape unique sans réplication de données*", *in : Quinzièmes Rencontres Francophones du Parallélisme*, La Colle sur Loup, October 2003.

[30] Y. CANIOU, E. JEANNOT, "*Un nouvel algorithme d'ordonnancement pour NetSolve*", *in : Quatorzièmes Rencontres Francophones du Parallélisme (RenPar'14)*, Hammamet, Tunisie, April 2002.

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

[32] Y. CANIOU, E. JEANNOT, "*Efficient Scheduling Heuristics for GridRPC Systems*", *in : QOS and Dynamic System Workshop of ICPADS'2004 - 10th International Conference on Parallel and Distributed Systems*, IEEE, p. 621–630, New-Port Beach, CA, July 2004.

[33] Y. CANIOU, E. JEANNOT, "*Experimental Study of Multi-Criteria Scheduling Heuristics for GridRPC Systems*", *in : Proceedings of the 10th International Euro-Par Conference (Euro-Par'04)*, M. Danelutto, D. Laforenza, M. Vanneschi (editors), *Lecture Notes in Computer Science*, *3149*, Springer, p. 1048–1055, Pisa, Italy, August/September 2004.

[34] 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.

[35] L.-C. CANON, E. JEANNOT, "*Wrekavoc a Tool for Hemulating Heterogeneity*", *in : 15th IEEE Heterogeneous Computing Workshop (HCW'06)*, Island of Rhodes, Greece, April 2006.

[36] F. CAPPELLO, F. DESPREZ, M. DAYDE, E. JEANNOT, Y. JEGOU, S. LANTERI, N. MELAB, P. NAMYST, RAYMOND PRIMET, 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, November 2005. to appear.

[37] E. CARON, F. DESPREZ, F. LOMBARD, J.-M. NICOD, M. QUINSON, F. SUTER, "*A Scalable Approach to Network Enabled Servers*", *in : Proceedings of the 8th International EuroPar Conference (Research Note)*, B. Monien, R. Feldmann (editors), *Lecture Notes in Computer Science*, *2400*, Springer-Verlag, p. 907–910, Paderborn, Germany, August 2002.

[38] 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.

[39] E. CARON, F. SUTER, "*Extension parallèle d'un outil de prédiction dynamique de performances*", *in : Quatorzièmes Rencontres Francophones du Parallélisme (RenPar'14)*, p. 69–74, Hammamet, Tunisie, April 2002.

[40] E. CARON, F. SUTER, "*Parallel Extension of a Dynamic Performance Forecasting Tool*", *in : Proceedings of the International Symposium on Parallel and Distributed Computing (ISPDC'02)*, p. 80–93, Iasi, Romania, July 2002.

[41] 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)*, M. Danelutto, D. Laforenza, M. Vanneschi (editors), *Lecture Notes in Computer Science*, *3149*, Springer, p. 230–237, Pisa, Italy, August/September 2004.

[42] B. CIROU, E. JEANNOT, "*Tripet: a Clustering Scheduling Algorithm for Heterogeneous Systems*", *in : IEEE ICPP International Workshop on Metacomputing Systems and Applications (MSA'2001)*, p. 231–236, Valencia, Spain, September 2001.

[43] J. COHEN, E. JEANNOT, N. PADOY, "*Messages Scheduling for Data Redistribution between Clusters*", *in : Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks (HeteroPar'03), Workshop of SIAM PPAM 2003*, Czestochowa, Poland, September 2003.

[44] P. COMBES, F. LOMBARD, M. QUINSON, F. SUTER, "*A Scalable Approach to Network Enabled Servers*", *in : Advances in Computing Science - ASIAN 2002. Internet Computing and Modeling, Grid Computing, Peer-to-Peer Computing, and Cluster Computing. Seventh Asian Computing Science Conference*, A. Jean-Marie (editor), *Lecture Notes in Computer Science*, *2550*, Springer-Verlag, p. 110–124, Hanoï, Vietnam, December 2002.

[45] E. DEDU, S. VIALLE, C. TIMSIT, "*Parallelisation of wave propagation algorithms for odour propagation in multi-agent systems*", *in : Advanced Environments, Tools and Applications for Cluster Computing (IWCC)*, D. Grigoras, A. Nicolau, B. Toursel, B. Folliot (editors), *LNCS*, *2326*, Springer Verlag, p. 92–102, 2001.

[46] F. DESPREZ, E. JEANNOT, "*Improving the GridRPC Model with Data Persistence and Redistribution*", *in : Third International Symposium on Parallel and Distributed Computing (ISPDC'2004)*, IEEE, p. 193–200, Cork, Ireland, July 2004. held with Third International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks - HeteroPar'04.

[47] F. DESPREZ, M. QUINSON, F. SUTER, "*Dynamic Performance Forecasting for Network-Enabled Servers in a Heterogeneous Environment*", *in : Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, H. Arabnia (editor), *III*, CSREA Press, p. 1421–1427, Las Vegas, June 2001. ISBN: 1-892512-69-6.

[48] F. DESPREZ, F. SUTER, "*Mixed Parallel Implementations of the Top Level of Strassen and Winograd Matrix Multiplication Algorithms*", *in : Proceedings of the 15th International Parallel and Distributed Processing Symposium (IPDPS'01)*, San Francisco, April 2001.

[49] F. DESPREZ, F. SUTER, "*Produit de matrices, Strassen et parallélisme mixte*", *in : Treizièmes Rencontres Francophones du Parallélisme des Architectures et des Systèmes (RenPar'13)*, p. 25–30, Paris, La Villette, April 2001.

[50] M. ESSAÏDI, I. GUÉRIN LASSOUS, J. GUSTEDT, "*SSCRAP : environnement de développement pour les modèles à gros grain*", *in : Quartozièmes Rencontres Francophones Du Parallélisme (RenPar'14)*, p. 9–16, Hammamet, Tunisie, April 2002.

[51] M. ESSAÏDI, I. GUÉRIN LASSOUS, J. GUSTEDT, "*SSCRAP: An Environment for Coarse Grained Algorithms*", *in : Fourteenth International Conference on Parallel and Distributed Computing and Systems (PDCS 2002)*, S. G. Akl, T. F. Gonzalez (editors), IASTED/ACTA Press, p. 398–403, Cambridge, MA, November 2002.

[52] M. ESSAÏDI, J. GUSTEDT, "*An Experimental Validation of the PRO Model for Parallel and Distributed Computation*", *in : 14^{th} Euromicro International Conference on Parallel, Distributed, and Network-Based Processing PDP 2006*, B. Di Martino, S. Venticinque (editors), IEEE Computer Society Press, p. 449–456, 2006.

[53] A. H. GEBREMEDHIN, I. GUÉRIN LASSOUS, J. GUSTEDT, J. A. TELLE, "*PRO: a Model for Parallel Resource-Optimal Computation*", *in : 16th Annual International Symposium on High Performance Computing Systems and Applications (HPCS'02)*, IEEE, p. 106–113, Moncton, Canada, June 2002.

[54] 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.

[55] G. GOEL, J. GUSTEDT, "*Bounded Arboricity to Determine the Local Structure of Sparse Graphs*", *in : 32nd International Workshop on Graph-Theoretic Concepts in Computer Science*, F. V. Fomin *et al.* (editors), *Lecture Notes in Computer Science*, Springer Verlag, 2006. accepted for publication.

[56] I. GUÉRIN LASSOUS, J. GUSTEDT, "*Portable List Ranking: an Experimental Study*", *in : Workshop on Algorithm Engineering (WAE 2000)*, *Lecture Notes in Computer Science*, *1982*, 2409, Springer Verlag, p. 111–122, Saarbrücken, Germany, September 2001.

[57] J. GUSTEDT, O. A. MÆHLE, J. A. TELLE, "*The Treewidth of Java Programs*", *in : Algorithm Engeneering and Experimentation, ALENEX 2002*, D. M. Mount, C. Stein (editors), *Lecture Notes in Computer Science*, *2409*, Springer Verlag, p. 86–97, San Francisco, CA, January 2002.

[58] 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)*, C. Blundo, C. Laneve (editors), *Lecture notes in Computer Science*, *2841*, EATCS, Springer, p. 125–136, Bertinoro, Italy, October 2003.

[59] J. GUSTEDT, S. VIALLE, A. DE VIVO, "*parXXL: A Fine Grained Development Environment on Coarse Grained Architectures*", *in : Workshop on state-of-the-art in Scientific and Parallel Computing PARA'06*, B. Kågström *et al.* (editors), Umeå, Sweden, June 2006.

[60] J. GUSTEDT, "*Randomized Permutations in a Coarse Grained Parallel Environment [extended abstract]*", *in : Fifteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA'03)*, F. Meyer auf der Heide (editor), ACM, ACM Press, p. 248–249, San Diego, CA, June 2003.

[61] 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)*, *Lecture Notes in Computer Science*, *2668*, Springer, p. 269–278, Montréal, Canada, February 2003.

[62] E. JEANNOT, B. KNUTSSON, M. BJÖRKMAN, "*Adaptive Online Data Compression*", *in : IEEE High Performance Distributed Computing (HPDC'11)*, p. 379–388, Edinburgh, Scotland, July 2002.

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

[64] E. JEANNOT, F. WAGNER, "*Two Fast and Efficient Message Scheduling Algorithms for Data Redistribution Through a Backbone*", *in : Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS'04)*, IEEE Computer Society Press, Santa Fe, NM, April 2004.

[65] 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, November 2005. to appear.

[66] E. JEANNOT, "*Automatic Multithreaded Parallel Program Generation for Message Passing Multiprocessors Using Parameterized Task Graphs*", *in : International Conference on Parallel Computing (ParCo2001)*, Naples, Italy, September 2001.

[67] 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.

[68] A. LEGRAND, M. QUINSON, "*Automatic Deployment of the Network Weather Service Using the Effective Network View*", *in : 18th International Parallel and Distributed Processing Symposium (IPDPS'04)*, IEEE Computer Society Press, Santa Fe, NM, April 2004.

[69] F. LOMBARD, M. QUINSON, F. SUTER, "*Une approche extensible des serveurs de calcul*", *in : Treizièmes Rencontres Francophones du Parallélisme des Architectures et des Systèmes (RenPar'13)*, p. 79–84, Paris, La Villette, April 2001.

[70] O. MENARD, S. VIALLE, H. FREZZA-BUET, "*Making Cortically-Inspired Sensorimotor Control Realistic for Robotics: Design of an Extended Parallel Cellular Programming Models*", *in : Advances in Intelligent Systems - Theory and Applications (AISTA-04)*, 2004.

[71] T. N'TAKPÉ, F. SUTER, "*Critical Path and Area Based Scheduling of Parallel Task Graphs on Heterogeneous Platforms*", *in : Proceedings of the Twelfth International Conference on Parallel and Distributed Systems (ICPADS)*, Minneapolis, MN, Jul 2006.

[72] T. N'TAKPÉ, "*Algorithmes d'ordonnancement de graphes de tâches parallèles sur plates-formes hétérogènes*", *in : Dix-septièmes Rencontres Francophones du Parallélisme (RenPar'17)*, Perpignan, France, October 2006.

[73] M. QUINSON, "*Un outil de modélisation de performances dans un environnement de metacomputing*", *in : Treizièmes Rencontres Francophones du Parallélisme des Architectures et des Systèmes (RenPar'13)*, Paris, La Villette, April 2001.

[74] M. QUINSON, "*Dynamic Performance Forecasting for Network-Enabled Servers in a Metacomputing Environment*", *in : International Workshop on Performance Modeling, Evaluation, and Optimization of Parallel and Distributed Systems (PMEO-PDS'02), in conjunction with IPDPS'02*, Fort Lauderdale, FL, April 2002.

[75] Z. SHI, E. JEANNOT, J. J. DONGARRA, "*Robust Task Scheduling in Non-Deterministic Heterogeneous Systems*", *in : Proceedings of IEEE International Conference on Cluster Computing*, IEEE, p. To appear, Barcelona, Spain, 2006.

[76] S. VIALLE, E. DEDU, C. TIMSIT, "*ParCeL-5/ParSSAP: A parallel programming model and library for easy and fast execution of simulations of situated multi-agent systems*", *in : Software Engineering Applied to Networking & Parallel/Distributed Computing (SNPD),*, Association for Computer and Information Science, p. 115–122, June 2002.

Internal Reports

[77] V. BERTEN, J. GOOSSENS, E. JEANNOT, "*On the Distribution of Sequential Jobs in Random Brokering For Heterogeneous Computational Grids*", *rapport de recherche RR-5499*, INRIA, February 2005.

[78] Y. CANIOU, E. JEANNOT, "*Scheduling on the Grid: Historical Trace and Dynamic Heuristics*", *Rapport de recherche RR-4620*, Institut National de Recherche en Informatique et en Automatique (INRIA), November 2002.

[79] Y. CANIOU, E. JEANNOT, "*Improvements and Study of the Accuracy of the Tasks Duration Predictor, New Heuristics*", *Rapport de recherche*, INRIA, May 2004.

[80] Y. CANIOU, E. JEANNOT, "*Study of the Behaviour of Heuristics Relying on the Historical Trace*", *Rapport de recherche*, INRIA, April 2004.

[81] F. DESPREZ, E. JEANNOT, "*Adding Data Persistence and Redistribution to NetSolve*", *rapport de recherche RR2001-39*, Laboratoire de l'Informatique du Parallélisme, École Normale Supérieure de Lyon, France, 2001.

[82] M. ESSAÏDI, I. GUÉRIN LASSOUS, J. GUSTEDT, "*SSCRAP: Soft Synchronized Computing in Rounds for Adequate Parallelization*", *Rapport de recherche*, Institut National de Recherche en Informatique et en Automatique (INRIA), May 2004.

[83] J. GUSTEDT, "*Data Handover: Reconciling Message Passing and Shared Memory*", *Rapport de recherche*, INRIA, November 2004.

[84] J. GUSTEDT, "*External Memory Algorithms using a Coarse Grained Paradigm*", *Rapport de recherche*, INRIA, March 2004.

[85] E. JEANNOT, F. WAGNER, "*Modelizing, Predicting and Optimizing Redistribution between Clusters on Low Latency Networks*", *Rapport de recherche RR-5361*, Institut National de Recherche en Informatique et en Automatique (INRIA), November 2004.

[86] A. LEGRAND, F. MAZOIT, M. QUINSON, "*An Application-Level Network Mapper*", *rapport de recherche*, INRIA, January 2006.

[87] M. QUINSON, "*GRAS: a Research and Development framework for Grid services*", *rapport de recherche*, INRIA, January 2006.

[88] F. WAGNER, E. JEANNOT, "*Message Scheduling for Data Redistribution Through High Performance Networks*", *Rapport de recherche*, INRIA, April 2004.

Miscellaneous

[89] Y. CANIOU, E. JEANNOT, "*Limitation des études validées par Simulation*", Metz, France, *in : École thématique sur la Globalisation des Ressources Informatique et des Données : Utilisation et Services (GridUSe-2004)*, June 2004.

[90] Y. CANIOU, "*Ordonnancement pour le modèle temps partagé*", Aussois, France, *in : École d'hiver Grid 2002*, December 2002.

[91] L. CASSE, *Extension d'une bibliothèque parallèle de calcul cellulaire à des clusters de PC et à des grilles de calcul*, master thesis, Univ. Nancy-I & Supélec, 2004, in French. *(Improvement of a parallel library for cellular computing to run on PC clusters).*

[92] P.-N. CLAUSS, *Algorithme à front d'onde et pipeline out-of-core sur architecture à mémoire partagée*, master thesis, Univ. Nancy-I, June 2006, in French. *(A wave-front algorithm and out-of-core pipelining on shared memory architectures)*.

[93] E. JEANNOT, F. WAGNER, "*Message Scheduling for Data Redistribution through High Performance Networks*", Le Croisic, France, *in : DistRibUtIon de Données à grande Echelle (DRUIDE'2004)*, May 2004.

[94] E. JEANNOT, "*Compression adatative et dynamique de donnés*", INRIA, Aussois, France, *in : École thématique GRID 2002*, December 2002.

[95] T. N'TAKPÉ, *Adaptation d'un algorithme d'ordonnancement de tâches parallèles sur plates-formes homogènes aux systèmes hétérogènes*, master thesis, INPL-ENSEM / LORIA, June 2005, in French. *(Adjustment of a parallel task scheduling algorithm from homogeneous to heterogeneous platforms)*.

[96] D. VAGNER, *Conception d'une bibliothèque de calcul cellulaire étendu sur cluster de PC, disposant de modes de communication hybrides*, master thesis, Univ. Nancy-I & Supélec, 2005, in French, *(Design of an extended cellular computing library on a PC cluster, including hybrid communication modes)*.