IN PARTNERSHIP WITH:
CNRS

Université de Lorraine

Activity Report 2011

Project-Team ALGORILLE

Algorithms for the Grid

IN COLLABORATION WITH: Laboratoire lorrain de recherche en informatique et ses applications (LORIA)

RESEARCH CENTER
Nancy - Grand Est

THEME
Distributed and High Performance Computing

Table of contents

Project-Team ALGORILLE

Keywords: Distributed System, Parallel Algorithms, Performance, Experimentation, High Performance Computing, Simulation

1. Members

Research Scientist
Jens Gustedt [Team leader, Senior Researcher, INRIA, HdR]

Faculty Members
Sylvain Contassot-Vivier [Professor, UHP, HdR]
Lucas Nussbaum [Associate Professor, Univ. Nancy 2]
Martin Quinson [Associate Professor, UHP/ÉSIAL; temporary assignment as INRIA junior researcher since Oct 2011]
Constantinos Makassikis [Temporary Assistant Professor, Univ. Nancy 2]

External Collaborators
Stéphane Genaud [Associate Professor, Univ. Strasbourg, HdR]
Stéphane Vialle [Professor, SUPÉLEC Metz Campus, HdR]

Technical Staff
Emmanuel Jeanvoine [Engineer, SED INRIA Nancy – Grand Est, ADT Solfége]
El Mehdi Fekari [Engineer, INRIA ADT SimGrid, until Jul 2011]
Sébastien Badia [engineer, INRIA, CPER MISN, thème EDGE]
Tinaherinantenaina Rakotoarivelo [engineer, INRIA ADT Aladdin-G5K]
Luc Sarzyniec [engineer, INRIA ADT Kadeploy since Oct 2011]

PhD Students
Soumeya Leila Hernane [teaching assistant UST Oran, Algeria, since Oct 2007]
Thomas Jost [since Oct 2009]
Cristian Rosa [ANR project grant, until Oct 2011]
Wilfried Kirschenmann [EDF R&D (Clamart, France) and SUPÉLEC, since Jan 2009]
Marion Guthmuller [since Oct 2011]
Tomasz Buchert [since Oct 2011]

Post-Doctoral Fellows
Pierre-Nicolas Clauss [Post-doc ANR project grant, until Aug 2011]
Christophe Thiéry [Post-Doc ANR project grant]

Administrative Assistant
Isabelle Herlich [INRIA]

2. Overall Objectives

2.1. Introduction

The common goal of the AlGorille team is to close up the gap between the modeling of large scale applications for the resolution of the corresponding challenging issues and their verification and testing through scientific experiments. Inside this large spectrum of scientific activities, it is often easier to reduce the scope of considerations and consider other parts as "providers" that help for the resolution of specific problems instead of viewing the progress of the modeling-development-experiment infrastructure as a whole. In contrast, our team composition gives us a unique chance of fertilization through the mutual feedback that the different parts can give to each other.

Both ends have already made substantial progress: on the application modeling side we are able to describe data and task dependencies that lead to efficient and consistent executions. On the experimenting side we are able to implement a large variety of models and frameworks (e.g MPI), to test and benchmark them through intensive and accurate simulations, emulations and *in situ* experiments.

2.2. Highlights

- ANR has granted the project SONGS (Simulation Of Next Generation Systems) with an attribution of 1.8 million euro. This project follows the project USS-SimGrid (Ultra Scalable Simulation with SimGrid). Martin Quinson is the national coordinator of both project. This acceptation confirms our leading position on the domain of experimental methodologies, and will open the door to future new collaborations. Without being direct members, IBM research and the CERN are associated to this project.
- Sébastien Badia and Lucas Nussbaum received the *Best Poster award* at *Rencontres France Grilles* for their work on the deployment of gLite on Grid'5000 [21]. This work is a demonstration of our mastering and our leadership role on the Grid'5000 testbed. It opens a path for further collaboration with the Production Grids community around experimentation on their software stack.

3. Scientific Foundations

3.1. Structuring of Applications for Scalability

Participants: Sylvain Contassot-Vivier, Jens Gustedt, Soumeya Leila Hernane, Thomas Jost, Wilfried Kirschenmann, Stéphane Vialle.

Large Scale Computing is a challenge under constant development that, almost by definition, will always try to reach the edge of what is possible at any given moment in time: in terms of the scale of the applications under consideration, in terms of the efficiency of implementations and in what concerns the optimized utilization of the resources that modern platforms provide or require. The complexity of all these aspects is currently increasing rapidly:

3.1.1. Diversity of platforms.

Design of processing hardware is diverging in many different directions. Nowadays we have SIMD registers inside processors, on-chip or off-chip accelerators (GPU, FPGA, vector-units), multi-cores and hyperthreading, multi-socket architectures, clusters, grids, clouds...The classical monolithic architecture of one-algorithm/one-implementation that solves a problem is obsolete in many cases. Algorithms (and the software that implements them) must deal with this variety of execution platforms robustly.

As we know, the "*free lunch*" for sequential algorithms provided by the increase of processor frequencies is over [41], we have to go parallel. But the "*free lunch*" is also over for many automatic or implicit adaption strategies between codes and platforms: e.g the best cache strategies can't help applications that accesses memory randomly, or algorithms written for "simple" CPU (von Neumann model) have to be adapted substantially to run efficiently on vector units.

3.1.2. The communication bottleneck.

Communication and processing capacities evolve at a different pace, thus the *communication bottleneck* is always narrowing. An efficient data management is becoming more and more crucial.

Not many implicit data models have yet found their place in the HPC domain, because of a simple observation: latency issues easily kill the performance of such tools. In the best case, they will be able to hide latency by doing some intelligent caching and delayed updating. But they can never hide the bottleneck for bandwidth.

HPC was previously able to cope with the communication bottleneck by using an explicit model of communication, namely MPI. It has the advantage of imposing explicit points in code where some guarantees about the state of data can be given. It has the clear disadvantage that coherence of data between different participating entities is difficult to manage and is completely left to the programmer.

Here, our approach is and will be to timely request explicit actions (like MPI) that mark the availability of (or need for) data. Such explicit actions ease the coordination between tasks (coherence management) and allow the platform underneath the program to perform a proactive resource management.

3.1.3. Models of interdependence and consistency.

Interdependence of data between different tasks of an application and components of hardware will be crucial to ensure that developments will possibly scale on the ever diverging architectures. We have up to now presented such models (PRO, DHO, ORWL) and their implementations, and proved their validity for the context of SPMD-type algorithms.

Over the next years we will have to enlarge their spectrum of application. On the algorithm side we will have to move to heterogeneous computations combining different types of tasks in one application. For the architectures we will have to take into account the fact of increased heterogeneity, processors of different speed, multi-cores, accelerators (FPU, GPU, vector units), communication links of different bandwidth and latency, memory and generally storage capacity of different size, speed and access characteristics. The first implementations using ORWL in that context look particularly promising.

The models themselves will have to evolve to be better suited for more types of applications, such that they allow for a more fine-grained partial locking and access of objects. They should handle e.g collaborative editing or the modification of just some fields in a data structure. This work has already started with DHO which allows the locking of *data ranges* inside an object. But a more structured approach would certainly be necessary here to be usable more comfortably in applications.

3.1.4. Algorithmic paradigms

Concerning asynchronous algorithms, we have developed several versions of implementations, allowing us to precisely study the impact of our design choices. However, we are still convinced that improvements are possible, especially concerning the global convergence detection strategies. Typically, the way that detection is performed must take into account the architecture of the parallel system. We have observed that switching between asynchronous and synchronous iterations during the process can be better on small/middle size systems with high performance networks whereas a completely asynchronous process should be used on (very) large systems or with low performance networks. We are currently working on the design of a generic and non-intrusive way of implementing such a procedure in any parallel iterative algorithm.

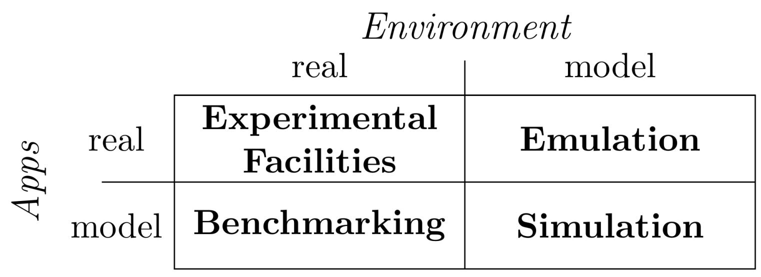
Also, we would like to compare other variants of asynchronous algorithms, such as waveform relaxation. In that context, computations are not performed for each time step of the simulation but for an entire time interval. Then, the evolutions of the elements at the frontiers between processors sub-domains are exchanged in an asynchronous way. Although we have already studied such schemes in the past, we would like to evaluate how they behave on recent architectures, and how the models and software for data consistency mentioned above can be helpful in that context.

3.1.5. Cost models and accelerators

We have already designed some models that relate computation power and energy consumption. Our next goal is to design and implement an auto-tuning system that controls the application according to user defined optimization criteria (computation and/or energy performance). This implies the insertion of multi-schemes and/or multi-kernels into the application such that it will able to adapt its behavior to the requirements.

3.2. Experimentation Methodology

Participants: Sébastien Badia, Pierre-Nicolas Clauss, El Mehdi Fekari, Stéphane Genaud, Jens Gustedt, Marion Guthmuller, Lucas Nussbaum, Martin Quinson, Cristian Rosa, Luc Sarzyniec, Christophe Thiéry.



		Environment real	*Environment* model
Apps	real	**Experimental Facilities**	**Emulation**
Apps	model	**Benchmarking**	**Simulation**

Figure 1. Main experimental methodologies.

As emphasized above, our scientific objects are constituted by modern distributed systems. The scientific questions that we address mainly concern the *performance* and *correction* of these systems. Their specificities are their ever increasing size, complexity and dynamics. This has reached a point where it becomes practically impossible to fully understand these systems through the fine description of their components' interaction. In that sense, computer science occupies a relatively unique position in epistemology since its scientific objects are human built artifacts, but become too complex to be directly understood by humans. This mandates a shift from the methodologies traditionally used in speculative sciences, where the knowledge is built *a priori* without experiments, to the ones of natural sciences, centered on the accumulation of knowledge through **experiments**.

These experiments can naturally be conducted on real platforms, but it is rarely practical to do so on real *production* platforms: the experiments can be disruptive to the production usage, and ongoing production can impose an uncontrollable bias on experiments. Direct execution on **experimental facilities** such as Grid'5000 alleviate these issues by providing a computing infrastructure similar to the targeted production platforms but dedicated to experiments and completely controllable. The obvious advantage of such direct execution is that it removes almost any experimental bias but it is naturally not applicable for some studies where the application to test or the target platform are not available. That is why **simulation** is very attractive to conduct *what if* analysis, for example to dimension a platform still to be built according to a given requirement, or to select the best algorithm for an application which is still to be written. Several intermediate steps exist between simulation and direct execution: **benchmarking** consists in experimenting with a real environment using synthetic applications while **emulation** consists in executing a real application on a platform model. The former is mainly used to devise some absolute insight about the environment without taking the application specificities into account. The latter is helpful to assess the behavior of an application on an environment that is not available [7].

These experimental methodologies are complementary, and the experimenters should combine them accordingly to benefit from their strengths in the process leading from an idea to a released product. Simulation is well adapted to the study of algorithms and helps in the realization of prototypes; direct execution on experimental facilities is an essential tool to change a prototype into an application; emulation permits to test the application in a wide range of conditions to transform it into a robust product.

Our team is active on most experimental methodologies (the only exception is benchmarking, since it consists in abstracting from the application to focus on the platforms while our whole approach is centered on the applications). This gives us an ideal position to attack the **larger challenge of improving experimental methodology** in distributed systems research, in order to put it on par with what is done in other sciences.

We will ensure that our methods and tools support a top-quality and unified experimental process by bridging *interfaces* and validating our work through series of experiments spanning all methodologies.

This axis and the *Structuring applications* axis benefit from each other by building a positive feedback loop: the *Structuring applications* axis provides first-class use cases, while the *Experimentation* axis helps to confirm the design choices made in the first axis. Other use cases are provided by our ongoing collaborations with the production grids community, with industry and with the Grid'5000 ecosystem (through *INRIA AEN Hemera*).

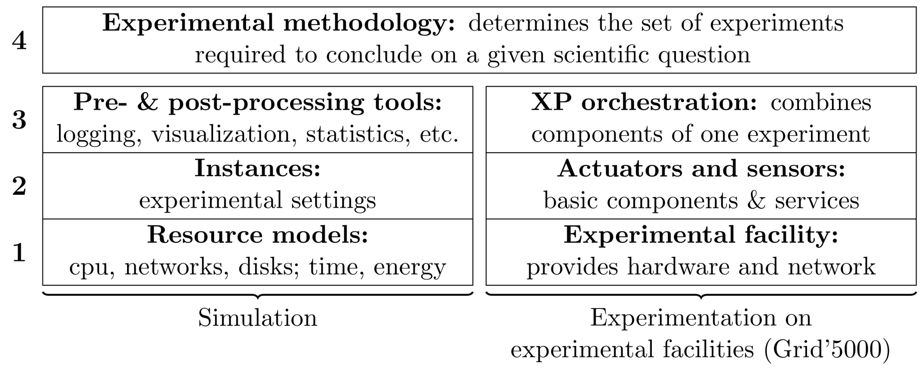

4
Experimental methodology: determines the set of experiments required to conclude on a given scientific question
3
Pre- & post-processing tools: logging, visualization, statistics, etc.
XP orchestration: combines components of one experiment
2
Instances: experimental settings
Actuators and sensors: basic components & services
1
Resource models: cpu, networks, disks; time, energy
Experimental facility: provides hardware and network
Simulation
Experimentation on experimental facilities (Grid'5000)

Figure 2. Our methodological approach, encompassing simulation and experimental facilities.

3.2.1. Simulation and dynamic verification.

Our team plays a key role in the SimGrid project, a mature simulation toolkit widely used in the distributed computing community. Since over ten years, we work on the validity, scalability and robustness of our tool. In the recent years, we increased its audience by targeting the P2P research community in addition to the one for grid scheduling. It now allows **precise simulations of millions of nodes** using a single computer.

In the future, we aim at extending further the applicability to **Clouds and Exascale systems**. Therefore, we will provide disk and memory models in addition to the already existing network and CPU models. We will also pursue our efforts on the tool's scalability and efficiency. **Interfaces** constitute another important work axis, with the addition of specific APIs on top of our simulation kernel. They will provide the "syntactic sugar" needed to express algorithms of these communities. For example, virtual machines will be handled explicitly in the interface provided for Cloud studies. Similarly, we will pursue our work on an implementation of the MPI standard allowing to study real applications using that interface. This work will also be extended to other interfaces such as OpenMP or the ones developed in our team to structure the applications, e.g ORWL. During the next evaluation period, we also consider using our toolbox to give **online performance predictions to the runtimes**. It would allow these systems to improve their adaptability to the changing performance conditions experienced on the platform.

We recently integrated a model checking kernel in our tool to enable **formal correction studies** in addition to the practical performance studies enabled by simulation. Being able to study these two fundamental aspects of distributed applications from the same tool constitute a major advantage. In the next evaluation period, we will further work on this convergence for the study of correctness and performance using the same tools. Ensuring that they become usable on real applications constitute another challenge that we will address.

3.2.2. Experimentation on experimental facilities

During the last years, we have played a **key role in the design and development of Grid'5000** by taking part in technical and design discussions and by managing several engineers working on the platform. We will pursue our involvement in the design of the testbed with a focus on ensuring that the testbed provides all the features needed for high-quality experimentation.

We will also work on **experiment-supporting software**, such as Kadeploy (software used to deploy user environments on Grid'5000) in *INRIA ADT Kadeploy (2011-2013)* and basic services that are common to most experiments (control of a large number of nodes, data management, etc.) in *INRIA ADT Solfege (2011-2013)*.

Specifically, we will pursue the development of our **emulation framework**, *Distem* (based on our previous emulation framework, Wrekavoc). We will address new challenges such as multi-core and heterogeneous systems, add load generation and fault injection features, and increase its usability, to get it widely accepted and used within the community.

Most experiments performed on Grid'5000 are still controlled manually or, in the best case, with quick and dirty scripts. This severely limits the quality of experimental results since it is hard to reproduce experiments many times to ensure statistical validity or to fully describe an experimental setup which was not created by an automated process. It also limits the scale and/or complexity of the possible experiments. Together with researchers from the field of Business Process Management (BPM), we will work on **enabling the use of Workflow Management Systems** to perform experiments on distributed systems.

Overall, we hope that our work in this research axis will bring major contributions to the **industrialization of experimentation** on parallel and distributed systems.

4. Application Domains

4.1. High Performance Computing

Participants: Pierre-Nicolas Clauss, Sylvain Contassot-Vivier, Jens Gustedt, Soumeya Leila Hernane, Emmanuel Jeanvoine, Thomas Jost, Wilfried Kirschenmann, Stéphane Vialle.

4.1.1. Models and Algorithms for Coarse Grained Computation

With this work we aim at extending the coarse grained modeling (and the resulting algorithms) that we provide previously, see [6], to hierarchically composed machines such as clusters of clusters or clusters of multiprocessors.

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

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

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

Asynchronous algorithms are very good candidates as they are robust to dynamic variations of the performances of the interconnection network used. Moreover, they are even tolerant to the loss of message related to the computations. However, as mentioned before they cannot be used in all cases. We will then focus on the feasibility to modify those schemes in order to widen their range of applicability while preserving a maximum of asynchronism.

Finally, as the number of components grows, so does the probability of having failures. Work has already been achieved to provide efficient fault tolerance solutions for some SPMD-with-communications and Master-Worker families of parallel applications (cf. Section 6.1.9). Being at the application level, these solutions seem suitable for the aforementioned heterogeneous architectures and may complement algorithmic-based fault tolerance such as the one naturally provided by asynchronous algorithms. We are currently investigating the compatibility of our fault tolerance solutions with some applications developed to run on clusters of GPGPUs (*e.g.*: American option pricer). We also see to extend our solutions to support and take advantage of asynchronous algorithms.

4.1.2. Irregular Problems

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

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

In particular we study the possible management of very large graphs, as they occur in reality. Here, the notion of "*networks*" appears twofold: on one side many of these graphs originate from networks that we use or encounter (Internet, Web, peer-to-peer, social networks) and on the other side the handling of these graphs has to take place in a distributed Grid environment. The principal techniques to handle these large graphs will be provided by the coarse grained models. With the PRO model [6] and the *parXXL* library we already provide tools to better design algorithms (and implement them afterward) that are adapted to these irregular problems.

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

4.1.3. Heterogeneous Architecture Programming

Clusters of heterogeneous nodes, composed of CPUs and GPUs, require complex multi-grain parallel algorithms: coarse grain to distribute tasks on cluster nodes and fine grain to run computations on each GPU. Algorithms implementation is achieved on these architectures using a multi-paradigm parallel development environment, typically composed of MPI and CUDA libraries (compiling with both gcc and nVIDIA nvcc compilers).

We investigate the design of multi-grain parallel algorithm and multi-paradigm parallel development environment for GPU clusters, in order to achieve both speedup and size up on different kinds of algorithms and applications. Our main application targets are: financial computations, PDE solvers, and relaxation methods.

4.1.4. Energy

Nowadays, society is getting more and more aware of the problem of energy supply and is therefore concerned with reducing energy consumption. Computer science is not an exception and a lot of effort has to be made in our domain in order to optimize the energetic efficiency of our systems and algorithms.

In that context, we investigate the potential benefit of using intensively parallel devices such as GPUs in addition to CPUs. Although such devices present quite high instantaneous energy consumptions, their energetic efficiency, that is to say their ratio of flops/Watt is often much better than the one of CPUs.

We have studied the potential energetic gain of GPUs in different kinds of applications (pricer, PDE solver,...). Our experiments have pointed out that there is, in most cases, a complex frontier between the best energetic solutions (CPU alone, CPU + GPU) according to the problem parameters (problem size,...) and architecture configuration (number of nodes, network...). Then, we have designed a first set of models that allows for predicting the best combination of compute kernels according to a given context of use. Further investigations will be done in order to enhance the models and try to design a dynamic adaptive scheme to make a bi-objective optimization (computing performance and energy consumption).

4.1.5. Load balancing

Although load balancing in parallel computing has been intensively studied, it is still an issue in the most recent parallel systems whose complexity and dynamic nature regularly increase. For the grid in particular, where the nodes or the links may be intermittent, the demand is stronger and stronger for non-centralized algorithms.

In a joint work with the University of Franche-Comté, we study the design and optimal tuning of a fully decentralized load balancing scheme (see 8.2.8). In particular, we study the optimal load amount to migrate between neighboring nodes. We have developed a SimGrid program to study the impact of the different strategies and we are currently adapting our load balancing scheme to a real application of neural learning (AdaBoost). This code was initially developed by S. Genaud and V. Galtier to compare JavaSpace and P2PMPI.

Another aspect of load-balancing is also addressed by our team in the context of the Neurad project. Neurad is a multi-disciplinary project involving our team and some computer scientists and physicists from the University of Franche-Comté to tackle the planning of external radiotherapy against cancer. In that work, we have already proposed an original approach in which a neural network is used inside a numerical algorithm to provide radiation dose deposits in any heterogeneous environments, see [42]. The interest of the Neurad software is to combine very small computation times (five minutes) with an accuracy close to the most accurate methods (Monte-Carlo) whereas these accurate methods take several hours to deliver their results.

In fact, in Neurad most of the computational cost is hidden in the learning of the internal neural network. This is why we work on the design of a parallel learning algorithm based on domain decomposition [25]. However, as learning the obtained sub-domains may take quite different times, a pertinent load-balancing is required in order to get approximately the same learning times for all the sub-domains. The work here is thus more focused on the decomposition strategy as well as the load estimator in the context of neural learning. We have recently proposed an efficient algorithm to perform the decomposition and the data selection of an initial learning set in order to obtain similar learning times of the induced sub-networks [20].

4.2. Providing Environments for Experiments

Participants: Sébastien Badia, Tomasz Buchert, Pierre-Nicolas Clauss, Sylvain Contassot-Vivier, El Mehdi Fekari, Jens Gustedt, Emmanuel Jeanvoine, Lucas Nussbaum, Martin Quinson, Tinaherinantenaina Rakotoarivelo, Cristian Rosa, Luc Sarzyniec, Christophe Thiéry, Stéphane Vialle.

4.2.1. Simulating Distributed Applications

We are major contributors to the SIMGrid framework (see 5.4 for the software description, and 6.2.1 for the new results of this year) a collaboration with the Univ. of Hawaii, Manoa, and INRIA Grenoble-Rhône-Alpes, France. It enables the simulation of distributed applications in large-scale settings for the specific purpose of evaluating and assessing algorithms. Simulations not only allow repeatable results (what is near to impossible when experimenting the applications on real experimental facilities) but also make it possible to explore wide ranges of platform and application scenarios. SIMGrid implements realistic fluid network models that result in very fast yet precise simulations. This is one of the main simulation tools used in the Grid Computing community.

4.2.2. Formally Assessing Distributed Algorithms

In joint research with Stephan Merz of the Veridis team of INRIA Nancy and LORIA, we are interested in the verification (essentially via model checking) of distributed and peer-to-peer algorithms. Whereas model checking is now routinely used for concurrent and embedded systems, existing algorithms and tools can rarely be effectively applied for the verification of asynchronous distributed algorithms and systems.

We are working on integrating these methods to the SIMGrid tool to make them more accessible to non-experts. The expected benefit of such an integration is that programmers can complement simulation runs by exhaustive state space exploration in order to detect defects that would be hard to reproduce by testing. Indeed, a simulation platform provides a controlled execution environment that mediates interactions between processes, and between processes and the environment, and thus provides the basic functionality for implementing a model checker. The principal challenge is the state explosion problem, as a naive approach to systematic generation of all possible process interleavings would be infeasible beyond the most trivial programs. Moreover, it is impractical to store the set of global system states that have already been visited: the programs under analysis are arbitrary C programs with full access to the heap, making even a hashed representation of system states very difficult and costly to implement.

4.2.3. Grid'5000

Grid'5000 is a scientific testbed for experimenting with a large variety of types of distributed systems, such as High Performance Computing, Clouds, P2P or Grids. It provides a unique combination of features to its users:

- deployment of user-provided operating system on bare hardware, with the Kadeploy tool developed in our team
- access to various technologies (CPUs, high performance networks, etc.) at a large scale
- dedicated network backbone, with monitoring and isolation features
- programmable API, for scripted experiments.

Grid'5000 is currently composed of 11 sites (one in Nancy, managed by our team, and two geographically close to Nancy, in Reims and Luxembourg). The Nancy site is one of the most important, both in terms of number of nodes and cores, and in terms of contribution to the technical team.

With this combination of features, Grid'5000 is a world-leading testbed for research in its field, and plays a central role in our work on experimentation methodologies.

4.2.4. Emulation

Experimental testbeds such as Grid'5000 provide a stable environment which is important to allow reproducible experiments. However, sometimes, the experimental conditions provided by the testbed do not match the conditions required by an experiment, in terms of computing power, network bandwidth, latency and topology, etc.

We are working on a software tool called *Distem* (see 5.2) based on another tool that we developed previously, Wrekavoc (see 5.3). The goal of Distem is to emulate a heterogeneous environment consisting of nodes of different compute and memory capacity and varying network bandwidth and latency. On such an emulated environment, it is possible to execute a real, unmodified application.

Distem uses homogeneous Linux clusters and achieves this emulation by controlling the heterogeneity of a given platform by degrading CPU and network of each node composing this platform.

4.2.5. InterCell

Intercell aims at setting up a cluster (256 PCs) for interactive fine grain computation. It is granted by the Lorraine Region (CPER 2007), and managed at the Metz campus of SUPÉLEC.

The purpose is to allow easy fine grain parallel design, providing interactive tools for the visualization and the management of the execution (debug, step by step, *etc*). The parallelization effort is not visible to the user, since InterCell relies on the dedicated *parXXL* framework, see 5.1 below. Among the applications that are tested is the interactive simulation of PDEs in physics, based on the Escapade project, see [29].

4.2.6. Experimental platform of GPU clusters

We participate in the scientific exploitation of two experimental 16-node clusters of GPUs that are installed at the SUPÉLEC Metz site. One cluster has already GPU with "FERMI" architecture, and the second should be updated at the beginning of 2012. This platform allows the experimentation of scientific programming on GPU ("GPGPU"), and to track computing and energetic performances, with specific monitoring hardware. Development environments available on these GPU clusters are mainly the gcc suite and its OpenMP library, OpenMPI and the CUDA environment of nVIDIA's nvcc compiler.

5. Software

5.1. parXXL

Participants: Jens Gustedt, Stéphane Vialle.

parXXL is a library for large scale computation and communication that executes fine grained algorithms (computation and communication are of the same order of magnitude) on coarse grained architectures (clusters, grids, mainframes). Historically, *parXXL* is the result of a collaboration between INRIA and SUPÉLEC. This library fulfills the requirements of our model *PRO*, *i.e.,*it uses an alternation of computation and communication steps. It realizes an abstraction layer between the algorithm as it was designed and its realization on different architectures and different modes of communication. The current version of this library has been registered at the *APP* and is available at http://parxxl.gforge.inria.fr/. It integrates a layer for message passing with MPI, a layer for shared memory with POSIX threads, a layer for out-of-core management with file mapping (system call *mmap*).

All three different realizations of the communication layers are quite efficient. They let us execute programs that are otherwise unchanged within the three different contexts. Usually, they reach the performance of programs that are directly written for a given context. Generally they outperform programs that are executed in a different context than they were written for, such as MPI programs that are executed on a shared memory mainframe, or such as multi-threaded programs that are executed on a distributed shared memory machine.

5.2. Distem

Participants: Tomasz Buchert, Emmanuel Jeanvoine, Lucas Nussbaum, Luc Sarzyniec.

Distem is a distributed systems emulator. In the context of research on Cloud, P2P, High Performance Computing or Grid systems, it can be used to transform an homogeneous cluster (composed of identical nodes) into an experimental platform where nodes have different performance, and are connected together through a complex network topology, thus facilitating the evaluation or benchmarking of applications targeting such environments.

Distem relies on modern Linux features (LXC, cgroups, cpufreq, iptables, traffic control) to *steal* resources from applications. At the node level, it provides the ability to introduce heterogeneity by splitting a multi-core node into several several virtual nodes of varying number of cores and CPU frequency. At the network level, it allows the user to describe and build virtual network topologies where each link has a given latency, and bandwidth limit.

Distem is controlled through a REST API to ease its integration into experiment scripts, but also provides a Ruby library and a command-line interface.

It has been registered with the APP, and is freely available under the GNU GPL.

More information is available from http://distem.gforge.inria.fr/.

5.3. Wrekavoc

Participants: Jens Gustedt, Lucas Nussbaum, Tomasz Buchert.

Wrekavoc addresses the problem of controlling the heterogeneity of a cluster to provide a configurable environment that allows for reproducible experiments on large sets of configurations using real applications with no emulation of the code. Work on Wrekavoc has stopped: current works are based on the *Distem* emulator.

5.4. SimGrid

Participants: Pierre-Nicolas Clauss, El Mehdi Fekari, Martin Quinson, Lucas Nussbaum, Cristian Rosa, Christophe Thiéry.

The SIMGrid framework aims at being a scientific instrument to the evaluation of algorithmic solutions for large-scale distributed experiments. It is the result of a collaboration with Henri Casanova (Univ. of Hawaii, Manoa) and Arnaud Legrand (MESCAL team, INRIA Grenoble-Rhône-Alpes, France). Simulation is a common answer to the grid specific challenges such as scale and heterogeneity. SIMGrid is one of the major simulators in the Grid community.

The main strong point of this is its carefully assessed **model validity**. To this end, the simulation kernel relies on a blend of analytical models and coarse-grain discrete event simulation. It proves several orders of magnitude faster than usual packet-level simulators used in the networking community (such as ns2 or GTNetS) while providing a good level of accuracy [43].

The SIMGrid framework is currently extremely **fast**. Independent authors demonstrated its superior scalability over its main concurrence [36], [38]. In addition to the efficiency of the simulation models, this **scalability** is ensured by a layered architecture, with a simulation kernel computing the time taken by *actions* which need to consume *resources* to complete. Another layer of abstraction introduces the notion of processes and network routing between hosts. On top of this come the user interfaces aiming at providing the syntactic sugar easing the tool usage.

Several such user interfaces exist, ensuring the **versatility** of the SIMGrid framework by adapting to the user goal: *MSG* helps the study of distributed heuristics. This is the historical interface of SIMGrid, and remains the most used one. *SMPI* is a new interface which allows the simulation of MPI programs designed for multi-processor systems on a single computer [4]. *SimDag* eases the study of scheduling heuristics for DAGs of (parallel) tasks, which helps the work on parallel task scheduling. *GRAS* (Grid Reality And Simulation) eases the development of Grid services and infrastructures [39] through a specific interface implemented twice: once on top of the simulator for the comfort of development, and once using regular sockets for live deployments.

SIMGrid can be freely downloaded from its web page and its user base is rapidly growing. Over the last decade, it grounded the experimental section of more than hundred scientific publications, not counting the ones being co-authored by members of the development team.

5.5. ORWL

Participant: Jens Gustedt.

ORWL is a reference implementation of the Ordered Read-Write Lock tools as described in [3]. It implements interfaces for locking and data management that easily allow to have an overlap between communication and computation. The main tool here is the introduction of a “handle” on a local or remote resource that can be used to trigger asynchronous prefetching of control and/or data. Also it implements a second layer of abstraction for the seamless programming of iterative tasks. With that layer iterative algorithms can be implemented that have guarantees for equity and deadlock-freeness.

ORWL is a standalone library that works on shared memory and in distributed settings. The implementation is uniquely based on C99 and POSIX. ORWL has already been registered at the *APP*. Final tests and benchmarks are on the way to ensure the quality of the implementation before it will be made publicly available.

5.6. P99

Participant: Jens Gustedt.

P99 is a toolbox of header files designated to ease programming in C, in particular modern C99. Originally, these macro definitions and tools for programming in C99 have been implemented for ORWL, but now they are separated out into a separate toolbox.

This toolbox allows *e.g* the simplified use of variable length argument list for macros and functions, default arguments of functions, compile time code unrolling, scope bound resource management, transparent allocation and initialization. It has been registered at the *APP* and is available at http://p99.gforge.inria.fr/.

6. New Results

6.1. Structuring of Applications for Scalability

Participants: Sylvain Contassot-Vivier, Thomas Jost, Jens Gustedt, Soumeya Leila Hernane, Constantinos Makassikis, Stéphane Vialle.

6.1.1. Large Scale and Interactive Fine Grained Simulations

Our library *parXXL* allows the validation of a wide range of fine grained applications and problems. We were able to test the interactive simulation of PDEs in physics, see [5], on a large scale. Also, biologically inspired neural networks have been investigated using *parXXL* and the InterCell software suite. The InterCell suite and these applicative results have been presented in [29].

6.1.2. Large Scale Models and Algorithms for Random Structures

A realistic generation of graphs is crucial as an input for testing large scale algorithms, theoretical graph algorithms as well as network algorithms, e.g platform generators.

Commonly used techniques for the random generation of graphs have two disadvantages, namely their lack of bias with respect to history of the evolution of the graph, and their incapability to produce families of graphs with non-vanishing prescribed clustering coefficient. In this work we propose a model for the genesis of graphs that tackles these two issues. When translated into random generation procedures it generalizes well-known procedures such as those of Erdős & Rény and Barabási & Albert. When just seen as composition schemes for graphs they generalize the perfect elimination schemes of chordal graphs. The model iteratively adds so-called *contexts* that introduce an explicit dependency to the previous evolution of the graph. Thereby they reflect a historical bias during this evolution that goes beyond the simple degree constraint of preference edge attachment. Fixing certain simple statical quantities during the genesis leads to families of random graphs with a clustering coefficient that can be bounded away from zero.

A journal article describing intensive simulations of these models that confirm the theoretical results and that show the ability of that approach to model the properties of graphs from application domains has been published as [13].

6.1.3. Development environment for co-processing units

In the framework of the PhD thesis of Wilfried Kirschenmann, co-supervised by Stéphane Vialle (SUPELEC & AlGorille team) and Laurent Plagne (EDF SINETICS team), we have designed and implemented a unified framework based on generic programming to achieve a development environment adapted both to multi-core CPUs, multi-core CPUS with SSE units, and GPUs, for linear algebra applied to neutronic computations. Our framework is composed of two layers: (1) MTPS is a low-level layer hiding the real parallel architecture used, and (2) Legolas++ is a high-level layer allowing the application developer to rapidly implement linear algebra operations. The Legolas++ layer aims at decreasing the development time, while the MTPS layer aims at automatically generating very optimized code for the target architecture, thus leading to decreased execution times. Experimental performances of the MTPS layer appeared very good, the same source code achieved performances close to 100% of the theoretical ones, on any of the supported target architectures. Our strategy is to generate optimized data storage and data access code for each target architecture, not just different computing codes.

A new version of Legolas++ is under development, and a minimal version has been implemented in 2011. It is optimized to use the MTPS layer: source code is generic while an optimized code is automatically generated to efficiently use all SSE/AVX vector units of a multicore CPU. An article on that work is accepted in the post-proceedings of PARA 2010 and will be published at the end of 2011; the thesis of Wilfried Kirschenmann will be defended in early 2012.

6.1.4. Structuring algorithms for co-processing units

Since 2009, we have designed and experimented several algorithms and applications, in the fields of option pricing for financial computations, generic relaxation methods, and PDE solving applied to a 3D transport model simulating chemical species in shallow waters. We aim at designing a large range of algorithms for GPU cluster architectures, to develop a real knowledge about mixed coarse and fine grained parallel algorithms, and to accumulate practical experience about heterogeneous cluster programming.

Our PDE solver on GPU cluster has been designed in the context of a larger project on the study of asynchronism (see 3.1 and 6.1.5). The iterations of the asynchronous parallel algorithm runs faster, but it requires more iterations and a more complex detection of convergence, see Section 6.1.5 below. We measured both computing and energy performances of our PDE solver in order to track the best solution, in function of the problem size, the cluster size and the features of the cluster nodes. We are tracking the most efficient solution for each configuration. It can be based on a CPU or a GPU computing kernel, and on a synchronous or asynchronous parallel algorithm. Moreover, the fastest solution is not always the less energy consuming. Our recent results are introduced in [26], and in an article accepted in the post-proceedings of PARA 2010. In 2011 we improved our asynchronous implementation. However, the most asynchronous version has led to significantly more complex code (with an increased probability of remaining bugs) but to similar performances. At the opposite, we designed and implemented different convergence detection mechanisms in our asynchronous version, and some versions seem to achieve really better performances. Execution time and energy consumption performances have now to be measured again for many configurations. We aim to get new complete performance evaluation at the beginning of 2012. Then we will design an automatic selection of the right kernel and the right algorithm, and we will implement an auto-setting application function of a global instruction of the user (to achieve a fast run, or a low consumption run, or a compromise...).

At last, we have continued to design option pricers on clusters of GPUs, with Lokman Abbas-Turki (PhD student at University of Marne-la-Valée) and some colleagues from financial computing. In the past we developed some European option pricers, distributing independent Monte-Carlo computations on the nodes of a GPU cluster. In 2010 we succeeded to develop an American Option pricer on our GPU clusters, distributing strongly coupled Monte-Carlo computations. The Monte-Carlo trajectories depend on each others, and lead to many data transfers between CPUs and GPUs, and to many communications between cluster nodes. First results were encouraging, we achieve speedup and size up. In 2011 we optimized a major step of our algorithm, consisting in a 4D to 2D reduction on GPU. Performances have increased, and are significantly easier to achieve. The configuration tuning of the application, function of the problem size and the number of computing nodes, has been simplified. Again, we investigate both computing and energy performances of our developments, in order to compare interests of CPU clusters and GPU clusters considering execution speed and the exploitation cost of our solution.

6.1.5. Asynchronism

In the previous paragraph is mentioned a project including the study of sparse linear solvers on GPU. That project deals with the study of asynchronism in hierarchical and hybrid clusters mentioned in 3.1.

In that context, we study the adaptation of asynchronous iterative algorithms on a cluster of GPUs for solving PDE problems. In our solver, the space is discretized by finite differences and all the derivatives are approximated by Euler equations. The inner computations of our PDE solver consist in solving linear equations (generally sparse). Thus, a linear solver is included in our solver. As this part is the most time consuming, to decrease the overall computation time it is essential to get a version that is as fast as possible. This is why we have decided to implement it on GPU, as discussed in the previous paragraph. Our parallel scheme uses the Multisplitting-Newton which is a more flexible kind of block decomposition. In particular, it allows for asynchronous iterations.

Our first experiments, conducted on an advection-diffusion problem, have shown very interesting results in terms of performances [8]. However, we investigate the possibility to insert periodic synchronous iterations inside the asynchronous scheme in order to improve the convergence detection delay. This is especially interesting on small/middle clusters with efficient networks.

Moreover, another aspect which is worth being studied is the full use of all the computational power present on each node, in particular the multiple cores, in conjunction with the GPU. This is still a work in progress.

6.1.6. New Control and Data Structures for Efficiently Overlapping Computations, Communications and I/O

With the thesis of Pierre-Nicolas Clauss we introduced the framework of *ordered read-write locks*, ORWL, see [3]. These are characterized by two main features: a strict FIFO policy for access and the attribution of access to *lock-handles* instead of processes or threads. These two properties allow applications to have a controlled pro-active access to resources and thereby to achieve a high degree of asynchronism between different tasks of the same application. For the case of iterative computations with many parallel tasks which access their resources in a cyclic pattern we provide a generic technique to implement them by means of ORWL. It was shown that the possible execution patterns for such a system correspond to a combinatorial lattice structure and that this lattice is finite iff the configuration contains a potential deadlock. In addition, we provide efficient algorithms: one that allows for a deadlock-free initialization of such a system and another one for the detection of deadlocks in an already initialized system.

We have developed a standalone distributed implementation of the API that is uniquely based on C and POSIX socket communications. Our goal is to simplify the usage of ORWL and to allow portability to a large variety of platforms. This implementation runs on different flavors of Linux and BSD, on different processor types Intel and ARM, and different compilers, gcc, clang, opencc and icc. An experimental evaluation of the performance is on its way. An engineering support from the local INRIA center has allowed to advance this implementation and to perform intensive benchmarks. The results have been presented in [28].

Data Handover, DHO, is a general purpose API that combines locking and mapping of data in a single interface. The access strategies are similar to ORWL, but locks and maps can also be hold only partially for a consecutive range of the data object. It is designed to ease the access to data for client code, by ensuring data consistency and efficiency at the same time.

In the thesis of Soumeya Hernane, we use the Grid Reality And Simulation (GRAS) environment of SimGrid, see 5.4, as a support for an implementation of DHO. GRAS has the advantage of allowing the execution in either the simulator or on a real platform. A first series of tests and benchmarks of that implementation demonstrates the ability of DHO to provide a robust and scalable framework, [18]. A step forward towards a distributed algorithm that allows distributed read-write locks with dynamic participation of process has been achieved in [30].

6.1.7. Energy performance measurement and optimization

Several experiments have been done on the GPU clusters of SUPÉLEC with different kinds of problems ranging from an embarrassingly parallel one to a strongly coupled one, via some intermediate levels. Our first results tend to confirm our first intuition that the GPUs are a good alternative to CPUs for problems which can be formulated in a SIMD or massively multi-threading way. However, when considering not embarrassingly parallel applications the supremacy of a GPU cluster tends to decrease when the number of nodes increases. This observation was the starting point of our participation to the COST-IC0804 about *energy efficiency in large scale distributed systems*, and an article accepted in the post-proceedings of PARA 2010 introduces our results achieved with our PDE solver distributed on our GPU clusters.

In 2011 we conducted new experiments and optimizations of our PDE solver and our American option pricer on the GPU clusters of SUPÉLEC. These experiments are still ongoing, and the optimization of this software should be achieved at the beginning of 2012. Simultaneously, we designed the foundations of a complete software architecture of self-configuring applications, choosing the right compute kernel and the right parallel algorithm to use, automatically. The global objectives to respect would either the overall speed, low energy consumption, or a speed-energy compromise. This global objective can be set by the user, or by an intelligent scheduler that aims to optimize a set of runs on a large cluster. This software architecture foundation has been introduced to a COST-IC0804 meeting in Budapest in June 2011.

In order to achieve this goal we need to establish some models of energy consumption of our applications on our CPU+GPU clusters, to be able to implement some heuristic of auto-setting. In 2011 we published a book chapter [26] introducing our first modeling strategies. Next step will be to implement a first auto-setting application, and to experiment its performances.

6.1.8. Load balancing

A load-balancing algorithm based on asynchronous diffusion with bounded delays has been designed to work on dynamical networks [34]. It is by nature iterative and we have provided a proof of its convergence in the context of load conservation. Also, we have given some constraints on the load migration ratios on the nodes in order to ensure the convergence. This work has been extended, especially with a detailed study of the imbalance of the system during the execution of a parallel algorithm simulated in the SimGrid platform.

The perspectives of that work are double. The first one concerns the internal functioning of our algorithm. There is an intrinsic parameter which tunes the load migration ratios and we would like to determine the optimal value of that ratio. The other aspect is on the application side in a real parallel environment. Indeed, we are currently applying this algorithm to a parallel version of the AdaBoost learning algorithm. This will allow us to study the best parameter to choose and to compare our load-balancing scheme to other existing ones.
Concerning the Neurad project, our parallel learning proceeds by decomposing the data-set to be learned. However, using a simple regular decomposition is not sufficient as the obtained sub-domains may have very different learning times. Thus, we have designed a first domain decomposition of the data set yielding subsets of similar learning times [40]. One of the main issue in this work has been the determination of the best estimator of the learning time of a sub-domain. As the learning time of a data set is directly linked to the complexity of the signal, several estimators taking into account that complexity have been tested, among which the entropy. However, the entropy is not the best estimator in that context, and we had to design a specific estimator. Also, we have optimized the decomposition process and added a selection phase that produces learning subsets of the same size [20]. Finally, we have also developed a parallel multi-threaded version of that decomposition/selection process.

6.1.9. Fault Tolerance

6.1.9.1. Application-level fault tolerance

Concerning fault tolerance, we have worked with Marc Sauget, from the University of Franche-Comté, on a parallel and robust algorithm for neural network learning in the context of the Neurad project [35]. A short description of that project is given in Section 4.1.5.

Our fault-tolerance strategy has shown to be rather efficient and robust in our different experiments performed with real data on a local cluster where faults were generated. Although those results are rather satisfying, we would like to investigate yet more reactive mechanisms as well as the insertion of robustness at the server level.

6.1.9.2. Programming model and frameworks for fault-tolerant applications

During the PhD thesis of Constantinos Makassikis [11], supervised by Stéphane Vialle, we have designed a new fault tolerance programming model (MoLoToF) to ease the development of fault-tolerant distributed applications. Main features of MoLoToF include so-called “fault-tolerant skeletons” to embed checkpoint-based fault tolerance within applications, and enable various collaborations, such as application-semantic knowledge supplied by users to the underlying system (*e.g.*: middleware), in order to fine tune fault tolerance.

Two development frameworks have been designed according to two different parallel programming paradigms: *ToMaWork* for *Master-Workers* applications [17] and *FT-GReLoSSS* (FTG) for some kind of *SPMD* applications including inter-node communications [10]. The programmer's task is limited. He only needs to supply some computing routines (functions of the application), has to add some extra code to specify a fault-tolerant parallel programming skeletons and then to tune the checkpointing frequency.

Our experiments have exhibited limited runtime overheads when no failure occurs and acceptable runtime overheads in the worst case failures. Observed runtime overheads are less than the ones obtained with all other system-level fault tolerance solutions we have experimented, while maintaining very limited development time overhead. Moreover, detailed experiments up to 256 nodes of our cluster have shown that it is possible to finely tune the checkpointing policies of the frameworks in order to implement different fault tolerance strategies, for example, according to cluster reliability.

In 2011, we have used the FTG framework to make fault-tolerant an existing parallel financial application [31] from EDF R&D, where it is used for gas storage valuation. The resulting application kept its initial runtime performance despite some source code modifications which are required in order to use FTG. As it was the case in earlier experiments with other applications, these modifications accounted for a limited development time overhead and fault tolerance remained more efficient than system-level fault tolerance solutions.

6.2. Experimentation Methodology

Participants: Tomasz Buchert, Sébastien Badia, Pierre-Nicolas Clauss, El Mehdi Fekari, Jens Gustedt, Lucas Nussbaum, Martin Quinson, Cristian Rosa, Luc Sarzyniec, Sylvain Contassot-Vivier.

6.2.1. Overall Improvements of the SimGrid Framework

See 4.2.1 for the scientific context of this result.

This year was the third year of the USS-SimGrid project on the simulation of distributed applications. We are principal investigator (see 8.2.7) of this project, funded by the ANR. It was prolonged until October 2012, giving us the ability to finish properly what was started. Several improvements have therefore been added to the framework, with numerous contributions from the ANR participants. This served as a flagship for the whole SIMGrid project and hosted several of our research efforts, detailed in the subsequent sections (up to 6.2.5). Also this year, the SONGS project got accepted by the ANR, paving the road for our research in this context for the next four years. Our team also coordinates this project, devoted to the "Simulation Of Next Generation Systems" (see 8.2.7).

In addition, the software quality efforts were pursued further through the second year of the INRIA ADT project (see 8.2.1) to maximize the impact of our research on our user community. First, we improved further our automated regression tests (by increasing the test coverage from below 60% to almost 80%) and by fixing the bugs found through the automated builds conducted on the INRIA pipol infrastructure. We also reduced the amount of possible configurations to reduce the test and maintenance burden. As usual, performance tuning deserved a lot of our attention this year. The bindings were solidified and improved. They are very well received by the user community. Finally, the port to the Mac architecture was improved while the experimental port to Windows was revived.

Finally, several operations were conducted to increase our user community. A publication summarizing all improvements made in the recent years were written and submitted [27]. The SimGrid team was represented at SuperComputing'11 (through our partners of Lyon) to meet potential users and distribute informative leaflets designed and printed to that extend.

6.2.2. Formal Verification of Distributed Applications

The context of this work is presented in 4.2.2.

In 2011, we started using the model-checker integrated last year into the SIMGrid framework with the goal to evaluate its limitations. Due to its generic design, it is able to verify protocols written using several APIs of SIMGrid. We tested it on both a MPI toy program written to that extend and on an implementation of the Chord P2P protocol. In this later case, the tested program was not written for the purpose of being model-checked but to assess the scalability limits of the simulator. The model-checker was used to track down a bug that was near to impossible to find with the simulator alone. This experiment and the formalism underlying our model-checker were described in the publication [19]. It is also described in further detail in Cristian Rosa's PhD, defended this year [12].

A second axis of our work this year consisted in extending the semantic power of the verified properties. In the work presented above, only local assertions and invariants can be verified. We started to investigate how to improve this during the internship of Marion Guthmuller. The major difficulty is that the reduction techniques based on the transition independences that we used so far are not sufficient for vivacity properties and must be extended to deal with the visibility of atomic properties [37]. One of the specificity of our work is the use of actual implementations were most of the literature uses handmade abstract models. This work continued in a PhD program, but didn't lead to any publication, yet.

6.2.3. Parallel Simulation within SimGrid

In addition to the software tuning and improvement described in 6.2.1, we tackled the issue of running SIMGrid simulators in parallel. Our work differs from the state of the art, because we do not aim to parallelize the simulation kernel itself but the execution of the user code processes running on top of the simulated system. Interestingly enough, this benefits greatly from the work on formal verification introduced in the previous section, and particularly of the new network abstraction layer that was added. It greatly reduced the code locations where the global state is modified, making the parallel execution possible.

This allowed for example the simulation of up to 2 million Chord hosts on a single computer. This work was described in [33] and a publication in a major conference is under preparation. Since the available memory constitutes the main scalability limit now, we will work on distributing the simulation to leverage the memory of several computers at the same time.

6.2.4. Simulating MPI Applications

The final goal of SMPI is to simulate a C/C++/FORTRAN MPI program designed for a multi-processor system on a single computer without any source code modification. This addresses one of the main limitation of SIMGrid, which requires the application to be written using one of the specific interfaces atop the simulator. New efforts have been put since July 2009 in this project, hereby continuing the work initiated by Henri Casanova and Mark Stilwell at University of Hawai'i at Manoa.

Previous work included a prototype implementation of various MPI primitives such as `send`, `recv`, `isend`, `irecv` and `wait`. Since the project's revival, many of the collective operations (such as `bcast`, `alltoall`, `reduce`) have been implemented. The standard network model used in SIMGrid has also been reworked to reach a higher precision in communication timings. Indeed, MPI programs are traditionally run on high performance computers such as clusters, and this requires to capture fine network details to correctly model the program behavior. Starting from the existing, validated network model of SimGrid, we have derived for SMPI a specific piece-wise linear model which closely fits real measurements. In particular, it enables to correctly model small messages and messages above the eager/rendezvous protocol limit. This work has been published at the IPDPS conference this year [15].

Ongoing work is now targeting a panel of MPI applications to have a better understanding of the applicability of our proposition. Pierre-Nicolas Clauss, who has been working full-time on the project between mid-2010 and mid-2011 has left, and we plan to put new workforce on SMPI with the support of the SONGS ANR project in 2012.

6.2.5. Simulating Real Applications

This work aims at providing a solution to simulate arbitrary applications on top of SIMGrid. The approach consists in intercepting the application actions at system level while they are executed on a test platform, and then replay these actions on top of the simulator.

Concerning trace capture, we continued our work on the Simterpose software, which intercepts the actions of the application and save them to file for further use by the simulator. This work, presented in a national conference [22], will be continued during the PhD work of Marion Guthmuller.

Concerning trace replay, we proposed a replay mechanism specific to MPI applications in collaboration with F. Suter from the Computing Center at IN2P3 together with F. Desprez and G. Markomanolis from the Graal team at INRIA Rhônes-Alpes. The originality is to rely on time-independent execution traces. This allows to completely decouple the acquisition process from the actual replay of the traces in a simulation context. We are able to acquire traces for large application instances without being limited to an execution on a single cluster. Finally, our replay framework is built directly on top of the SIMGrid simulation kernel. This work was published in [16].

6.2.6. Emulation & Distem

During the internship of Luc Sarzyniec, we re-implemented an emulator from scratch with the goal of having a more reliable basis for further developments. This new development, *Distem* (see 5.2), already includes support for CPU performance emulation (internship of Tomasz Buchert in 2010) and network emulation. We are currently preparing a first release of *Distem*, and are working on its validation.

6.2.7. Grid'5000 and ADT Aladdin-G5K

Grid'5000 is an experimental platform for research on distributed systems. Two new sites were added to Grid'5000 in 2011: Reims and Luxembourg. This should reinforce the impact of Grid'5000 in the east of France. It is worth noting that the system administrator of the Luxembourg Grid'5000 site was formerly a student in Nancy, and did a student project using Grid'5000 managed by Lucas Nussbaum. Also, more collaboration on technical aspects is expected thanks to this geographical proximity.

On the local level, power consumption sensors are being added to the graphene cluster, which will allow an accurate monitoring of energy consumption during experiments.

On the national level, Lucas Nussbaum is now mandated by the Grid'5000 executive committee to follow the work of the technical team. He contributed to two publications [23], [24] at *Journées Réseaux 2011* that describe the Grid'5000 software stack. He also gave invited talks during a Grid'5000 day at RenPar, and during the *Support for experimental computer science* workshop as SuperComputing'11.

Local scientific contributions include the automation of the deployment of the gLite middleware on Grid'5000. That work [21] was presented at *Rencontres France Grilles* and received the Best Poster award. We hope that this work will serve as a basis for further collaborations with the production grids community.

We also started the *ADT Kadeploy* project that will continue the development of the Kadeploy software, which already plays a key role on Grid'5000.

6.2.8. Experimental cluster of GPUs

The experimental platform of SUPÉLEC for "GPGPU", see Section 4.2.6, is composed of two GPU clusters, and its electrical line has been improved in 2011.

The first cluster is currently composed of 16 PCs, each one hosting a dual-core CPU and a GPU card: a nVIDIA GeForce GT285, with 1GB of RAM (on the GPU card). The 16 nodes are interconnected across a devoted Gigabit Ethernet switch. The second cluster has 16 more recent nodes, composed of an Intel Nehalem CPU with 4 hyper-threaded cores at 2.67GHz, and a nVIDIA GTX480 ("Fermi") GPU card with 1.5GB of memory. This cluster has a Gigabit Ethernet interconnection network too. These 2 clusters can been accessed and used like one 32-nodes heterogeneous cluster of hybrid nodes. This platform has allowed us to experiment different algorithms on an heterogeneous cluster of GPUs.

The energy consumption of each node of the cluster hosting the GTX285 GPUs is monitored by a Raritan DPXS20A-16 device that continuously measures the electric power consumption (in Watts). The nodes of the cluster hosting the GTX480 are monitored by two Raritan devices, because the energy consumed by this cluster exceeds the maximum energy supported by a Raritan DPXS20A-16 device.

A set of Perl and shell scripts, developed by our team, sample the electrical power (Watt) measured by the Raritan devices and compute the energy (Joule or Watt Hour) consumed by the computation on each node and on the complete cluster (including the interconnection switch).

In 2011 we have increased the amount of electrical energy supplied to these cluster, in order to support the experiments of our new distributed American option pricer. This application achieves high performances but consumes more energy on our GPU clusters than our previous codes, and exceeded the limit of our previous electrical line.

This platform has been intensively used to get experimental performance measures introduced in 2011 meetings of the COST IC0804 about *Energy efficiency in large scale distributed systems*, and published in a book chapter [26].

7. Contracts and Grants with Industry

7.1. Contracts with Industry

In 2011 we worked again with Quartet FS company, to design a parallel processing of continuous queries in a financial computing software suite. This applied research was a contract between SUPÉLEC and Quartet FS company, has been achieved with the help of two SUPÉLEC students, and is now included in the software products of Quartet FS.

At the end of 2011, we have established three agreements between SUPÉLEC and three different companies to make applied parallel computing research in the areas of finance (Quartet FS), energy (CGGVeritas) and defense (THALES). These contracts will start at the end of 2011 up to April 2012.

7.2. Grants with Industry

At the end of 2011, SUPÉLEC and Thales company planned to start a new PhD in 2012 about parallel computing on new hybrid architectures, and based on a Thales Grant.

8. Partnerships and Cooperations

8.1. Regional initiatives

Participants: Sylvain Contassot-Vivier, Lucas Nussbaum, Martin Quinson.

8.1.1. CPER MISN, EDGE project

Martin Quinson and Lucas Nussbaum are leading a project of the regional CPER contract, called *Expérimentations et calculs distribués à grande échelle* (EDGE). A cluster targeting large-scale experiments (144 single-CPU nodes) was bought in 2010 in that context. In 2011, we focused on maintaining and improving the local Grid'5000 infrastructure, and animating both the research on experimental grids and the research community using these facilities. More information is available at http://misn.loria.fr/spip.php?rubrique8.

8.1.2. Other regional grants

Martin Quinson received a grant from the Lorraine Region for two years (2010–2011) to fund our exploratory work on the possibility to use formal methods such as model-checking to ensure some properties (such as the lack of deadlocks in any case) of large-scale distributed algorithms. The results of this action are described in Section 6.2.2.

Sylvain Contassot-Vivier received a grant from the Lorraine Region for two years (2011–2013) to support a research project over dynamical systems: *Dynamical systems: theoretical study and application to parallel algorithmic for PDEs solving*.

8.2. National Initiatives

Participants: Sébastien Badia, Sylvain Contassot-Vivier, Stéphane Genaud, Jens Gustedt, Emmanuel Jeanvoine, Lucas Nussbaum, Martin Quinson, Tinaherinantenaina Rakotoarivelo, Luc Sarzyniec, Stéphane Vialle.

8.2.1. INRIA ADT SimGrid for human beings (2009–2011)

SimGrid for human beings is another INRIA Technological Development Action aiming at providing engineering manpower to the SIMGrid project to improve the documentation and to provide stock implementations of classical algorithms in order to ease its usage by the users. Mehdi Fekari was hired on this project, leading to the results described in Section 6.2.1.

8.2.2. INRIA ADT Aladdin-G5K (2007–2012?)

ADT Aladdin-G5K (A LArge-scale Distributed Deployable INfrastructure) is an INRIA Technological Development Action. It is a management structure for Grid'5000. The goal of Aladdin-G5K is to further develop the Grid'5000 testbed, and perform system administration and maintenance, as well as additional tasks such as maintaining the various tutorials. Three engineers from Nancy (two from AlGorille, one from SED) are contributing to Aladdin-G5K: Tina Rakotoarivelo (hired for the ADT), Sébastien Badia (IE CPER) and Benjamin Dexheimer (SED).

8.2.3. INRIA ADT Kadeploy (2011–2013)

ADT Kadeploy focuses on the Kadeploy software. Kadeploy is a tool for efficient, scalable and reliable cluster deployment, used on several clusters at INRIA and playing a key role on the Grid'5000 testbed. This ADT allows us to continue the development of Kadeploy, by improving its performance, its reliability and its security. We are also adding features that are required in some contexts so that it will be possible to further distribute Kadeploy and increase its usage. During the ADT, we are collaborating with INRIA platforms that do not use Kadeploy yet, but have similar needs, as well as with industry. Luc Sarzyniec was hired as *young engineer* (IJD) as part of this project.

8.2.4. INRIA ADT Solfege (2011–2013)

ADT Solfege (*Services et Outils Logiciels Facilitant l'Experimentation á Grande Échelle*) aims at developing or improving a tool suite for experimentation at large scale on testbeds such as Grid'5000. Specifically, we will work on control of a large number of nodes, on data management, and on changing experimental conditions with emulation. Emmanuel Jeanvoine (SED) is delegated in the AlGorille team for the duration of this project.

8.2.5. INRIA AEN HEMERA

Héméra is an INRIA Large Wingspan project, started in 2010, that aims at demonstrating ambitious upscaling techniques for large scale distributed computing by carrying out several dimensioning experiments on the Grid'5000 infrastructure, at animating the scientific community around Grid'5000 and at enlarging the Grid'5000 community by helping newcomers to make use of Grid'5000.

Within that project, Martin Quinson, Lucas Nussbaum and Stéphane Genaud lead three working groups, respectively on *simulating large-scale facilities*, on *conducting large and complex experimentations on real platforms*, and on *designing scientific applications for scalability*.

8.2.6. CNRS initiatives, GDR-ASR and specific initiatives

We participate at numerous national initiatives. In the GDR-ASR (architecture, systems, and networks) we take part in RGE action [1]. The finances of RGE, led by Stéphane Vialle at SUPÉLEC, are provided by the GDR ASR of CNRS and maintained by AlGorille. The RGE action organizes three meetings per year, and usually gathers 40-45 people per meeting.

[1] *Réseau Grand Est*

Sylvain Contassot-Vivier decided to stop his animation role in the Embedded Pole in 2011 in order to focus on his research activities. However, he continues his expert analysis for the MEI (Mission d'Expertises Internationales).

8.2.7. ANR USS-SimGrid (2009–2011) and ANR SONGS (2011–2015)

Martin Quinson is the principal investigator of one project of the ARPEGE call from the ANR (french funding agency), called USS-SimGrid (Ultra Scalable Simulation with SimGrid – 2009–2011). It aims at improving the scalability of the SIMGrid simulator to allow its use in Peer-to-Peer research in addition of Grid Computing research. The challenges to tackle include models being more scalable at the eventual price of slightly reduced accuracy, automatic instantiation of these models, tools to conduct experiments campaigns, as well as a partial parallelization of the simulator tool.

Martin Quinson is also the principal investigator of a project of the INFRA call from the ANR, called SONGS (Simulation Of Next Generation Systems – 2012-2016). It aims at increasing the target community of SIMGrid to two new research domains, namely Clouds (restricted to the *Infrastructure as a Service* context) and High Performance Computing. We will develop new models and interfaces to enable the use of SIMGrid for generic and specialized researches in these domains.

As project leading team, we are involved in most parts of these projects, which allows the improvement of our tool even further and set it as the reference in its domain (see Sections 6.2.1 through 6.2.5).

8.2.8. Bilateral Collaborations

With Arnaud Giersch from the University of Franche-Comté, we work on the design and implementation of a decentralized load-balancing algorithm, based on asynchronous diffusion, that works with dynamical networks. In such a context, we consider that the nodes are always available but the links between them may be intermittent. According to the load-balancing task, this is a rather difficult context of use.

Lucas Nussbaum and Martin Quinson are participating to a research effort lead by F. Suter from the Computing Center of IN2P3. This project is jointly funded by CNRS' Institut des Grilles and INRIA's ADT Aladdin in a program that aims at bridging the production grids and distributed systems research communities. The overall goal of the project is to explore ways to enable the scientific study and the evaluation of improvements in the context of the gLite grid middleware. New results in this project are described in Section 6.2.7.

8.3. European Initiatives

8.3.1. Collaborations in European Programs, except FP7

8.3.1.1. Energy efficiency in large scale distributed systems.

Stéphane Vialle, Sylvain Contassot-Vivier and Thomas Jost participate to the COST (European Cooperation in the field of Scientific and Technical Research) Action IC0804 (Energy efficiency in large scale distributed systems), started in 2010.

Our designs of PDE solvers using synchronous and asynchronous distributed algorithms, implemented and experimented both on CPU and GPU clusters, have led to the design of some performance models. Main results have been introduced in 2011 in a book chapter [26].

Moreover, in 2011 we achieved a first design of a software architecture to build self-configurating applications, in order to track a user instruction (to run fast, or to run consuming low energy, or to run achieving a compromise) in an execution environment imposing some energy constraints on application runs. This software architecture has been introduced to the COST IC0804 meeting of May 19-20, 2011 in Budapest. This software architecture is still under investigation, and its implementation is planned for 2012.

Finally, in 2011 we have achieved the implementation of an American option pricer on our GPU clusters, and we have run many experiments to measure its speedup and energy gain. A second version, more efficient, including an optimized 4D to 2D reduction on each GPU node, is under experimentation at the end of 2011. These research are conducted in collaboration with some colleagues of University of Marne-la-Vallée.

8.4. International Initiatives

8.4.1. Bilateral Collaborations

This year, we formalized the collaboration on the modeling of storage elements that we had with the team PH-ADP-DDM lead by V. Garone at CERN and with F. Suter from the Computing Center of IN2P3. This work will now be done in the context of the SONGS project, that got funded by the ANR this year.

We also started working with IBM France and IBM Haifa on the modeling of Cloud jobs and resources, also in the loose context of the SONGS project. This collaboration should take some more momentum in the next few years.

Finally, we are working with the team of Jan Broeckhove (Professor at university of Antwerp, Belgium), in a project funded (2010-2011) by the PHC Tournesol program. This project aims at exploring Large-Scale Discrete-Event Simulation of Distributed Systems.

We collaborate with Henri Casanova of University of Hawai'i at Manoa on the SimGrid framework, as detailed in 5.4. The result obtained this year on the simulation of MPI applications are detailed in Section 6.2.4.

We also collaborate with David Elizondo from the University of Leicester in Great Britain on the problem of linear separability determination. Our current work deals with the design and implementation of a fast algorithm to determine whether or not two or more sets of points in $\mathbb{R}^n$ are linearly separable. We have already obtained an interesting result in 2D whose publication is in preparation. The next step is the extension to $n > 2$ dimensions.

8.4.2. INRIA International Partners

8.4.2.1. Internships

Matías Ezequiel Vara

Subject: Ordered Read-Write Locks on Multicore Architectures

Institution: Universidad Nacional de La Plata (Argentina)

9. Dissemination

9.1. Animation of the scientific community

9.1.1. Editorial Activities

Since October 2001, Jens Gustedt is Editor-in-Chief of the journal *Discrete Mathematics and Theoretical Computer Science* (DMTCS). DMTCS is a journal that is published electronically by an independent association under French law. Based on a contract with INRIA, its main web-server is located at the LORIA. DMTCS has acquired a good visibility within the concerned domains of Computer Science and Mathematics, which is confirmed by a relatively high impact factor. This year, DMTCS published a special issue and proceedings volumes of noticeable events in the domain.

In 2011, Jens Gustedt has served as program committee member of the 20th Euromicro International Conference on Parallel, Distributed and network-based Processing PDP 2012, of the 3rd Workshop on Complex Networks CompleNet 2012 and of 20th RenPar (RenPar 2011).

Lucas Nussbaum was member of the organization and program committee for the Grid'5000 Spring School 2011.

Martin Quinson has served as program committee member of the 6th International Workshop on Modeling, Simulation, and Optimization of Peer-to-peer environments (MSOP2P 2012, associated to Euromicro PDP 2012).

Stéphane Genaud was member of the program committee of the 13th IEEE International Conference on High Performance Computing and Communications, HPCC 2011.

Stéphane Vialle was member of the twentieth RenPar conference committee ("Rencontres francophones du Parallélisme") (RenPar 2011). Stéphane Vialle was also member of the program committee of the international conference on Architecture of Computing Systems (ARCS 2011 and ARCS 2012).

9.1.2. Refereeing

In 2011, members of the team served as referees for the following journals and conferences:

Journals: Concurrency and Computation: Practice and Experience, Eurasip journal on Image and Video Processing, Journal of Parallel and Distributed Computing, Parallel Computing, Engineering Applications of Artificial Intelligence, Neural Networks

Conferences: CCGrid 2011, EuroPar 2011, Heteropar 2011, IPDPS 2012, PDP 2012, SuperComputing 2011, Solstice 2011

Projects: Sylvain Contassot-Vivier is expert for the MEI funding program.

9.1.3. Scientific Expertise

In 2011, our team members participated to following thesis committee:

Stéphane Genaud, rapporteur, Fabrice Dupros, University of Bordeaux 1, France.

Jens Gustedt, president, Tom Leclerc, University Henri Poincaré, France

Martin Quinson, member, Ahmed Harbaoui, University of Grenoble, France.

Martin Quinson, member, Bogdan Cornea, University of Franche-Comté, France.

Sylvain Contassot-Vivier, member, Lucian Aleçu, UHP University - Nancy 1, France.

Sylvain Contassot-Vivier, member, Rémy Laurent, University of Franche-Comté, France.

Sylvain Contassot-Vivier, member, Dawood Khan, INPL, France.

Sylvain Contassot-Vivier, member, Emmanuel Benard, University of Strasbourg, France.

Our members were also members of the following recruiting committees:

Jens Gustedt, full professor position, University of Franche-Compté, France.

Jens Gustedt, assistant professor position, University of Franche-Compté, France.

Martin Quinson, assistant professor position, University of Nancy I.

Martin Quinson, assistant professor position, University of Bordeaux.

9.1.4. Invitations and participations to scientific events

Stéphane Vialle was invited to participate to a panel about *Graphical Processing Units (GPUs): Opportunities and Challenges* at the 2011 International Conference on High Performance Computing & Simulation (HPCS'2011), Istanbul, Turkey.

Martin Quinson was invited to give a presentation entitled *H*C: Performance Everywhere (or, computing getting high)* at the workshop "Challenges & Pitfalls of Performance Assurance", associated to CECMG'11 in München, Germany in March 2011.

Lucas Nussbaum was invited to give the following talks:

- *Towards better experiments on Grid'5000* at *Journée Grid'5000 aux Rencontres francophones du Parallélisme (RenPar'20), Saint-Malo*
- *Grid'5000, a scientific instrument for experiment-driven research on parallel, large-scale and distributed systems* at *Support for Experimental Computer Science Workshop at SC 11, Seattle*
- *Towards better tools for experiments on distributed systems* at *Support for Experimental Computer Science Workshop at SC 11, Seattle*

Constantinos Makassikis was invited to give a presentation entitled *A Skeletal-Based Approach for the Development of Fault-Tolerant SPMD Applications* during the “Seconde journée du groupe de travail LaMHA (Langages et Modl`es de Haut-niveau pour la programmation parallèle, distribuée, de grilles de calcul et Applications)”, December 2010.

9.1.5. Scientific Mediation

Martin Quinson acts as a member to the INRIA-Nancy Grand Est committee for Scientific Mediation. He also participated alongside with Thomas Jost and Sébastien Badia to the LORIA and INRIA-Nancy Grand Est stand at the “Fête de la Science”, proposing several activities to demonstrate the *algorithmic thinking* at the core of the Computer Science without requiring any computer or even electric devices.

9.2. Teaching

Participants: Sylvain Contassot-Vivier, Jens Gustedt, Constantinos Makassikis, Lucas Nussbaum, Martin Quinson, Stéphane Vialle.

DUT Informatique (S1): System (40h), L1, Univ. Nancy 2, France

DUT Informatique (S2): Networks (30h), L1, Univ. Nancy 2, France

DUT Informatique (S1bis): Computer architecture (36h), L1, Univ. Nancy 2, France

DUT Informatique (S1): Introduction to algorithmic (40h), L1, Univ. Nancy 2, France

DUT Informatique (S1): Basic programming (48h), L1, Univ. Nancy 2, France

DUT Informatique (AS): Networks (70h), L2, Univ. Nancy 2, France

DUT Informatique (AS): Algorithmic (42h), L2, Univ. Nancy 2, France

DUT Informatique (S2): Object programming (32h), L1, Univ. Nancy 2, France

Licence pro ASRALL: Installation of Linux (20h), L3, Univ. Nancy 2, France

Licence pro ASRALL: Administration of applications (24h), L3, Univ. Nancy 2, France

Licence: Scientific computing (22h), L3, UHP University, France

Licence: Algorithmic and programming (40h), L3, UHP University, France

Licence (1st year engineer): Programming techniques and tools (44h), L3, ÉSIAL, UHP University, France

Licence (1st year engineer): C and Shell programming (44h), L3, ÉSIAL, UHP University, France

Master 1: Algorithmic and complexity (14H), M1, UHP University, France

Master 1: Algorithmic of distributed systems (24H), M1, UHP University, France

Master (2nd year engineer): Networks and systems, System part (25h), M1, UHP University, France

Master (2nd year engineer): Algorithmic of parallel and distributed systems (32H), M1, ÉSIAL, UHP University, France

Master (2nd year engineer): Information System, SUPELEC, Paris, France

Master projects (2nd year engineer): Software Development project, SUPELEC, Metz, France

Master 2: Communicating systems (24h), M2, UHP University, France

Master 2 IMOI: Networks (23H), M2, UHP University, France

Master 2 RISE: Distributed Systems and Grids, Strasbourg University, France

Master 2 ILC: Parallelism, Distributed Systems and Grids, Strasbourg University, France

Master (3rd year engineer): High Performance Computing, SUPELEC, Metz, France

Master (3rd year engineer): High Performance Computing, SUPELEC, Paris, France

Master projects: (14H), M2, UHP University, France

Master projects (3rd year engineer): Industrial project, SUPELEC, Metz, France

In addition, Martin Quinson authored a pedagogic platform in collaboration with Gérald Oster (Score team of INRIA Nancy Grand Est). This tool aims at providing an environment that is both appealing for the student, easy to use for the teacher, and efficient for the learning process. It is available from its page. Its advantages for the learning process are described in [32].

9.3. Advising

Participants: Sylvain Contassot-Vivier, Jens Gustedt, Lucas Nussbaum, Martin Quinson, Stéphane Vialle.

PhD: Constantinos Makassikis, *Conception d'un modèle et de frameworks de distribution d'applications sur grappes de PCs avec tolérance aux pannes à faible coût*, Université Henri Poincaré, defended on February, 2. 2011 (advisers: Stéphane Vialle and Virginie Galtier)

PhD: Cristian Rosa, *Performance & Correctness Assessment of Distributed Systems*, Université Henri Poincaré, defended on October, 24. 2011 (advisers: Stephan Merz and Martin Quinson).

PhD: Wilfried Kirschenmann *Toward sustainable intensive computing kernels*, Université Henri Poincaré, expected in 2012 (advisers: Stéphane Vialle and Laurent Plagne)

PhD: Soumeya Hernane, *Models and Algorithms of coherent data sharing for high performance parallel and distributed computing*, University of Oran, Algeria and University of Univ. Nancy 2, expected in 2012 (advisers: Jens Gustedt and Mohamed Benyettou)

PhD: Thomas Jost, *Sparse direct linear solver*, Université Henri Poincaré, expected in 2012 (advisers: Bruno Lévy and Sylvain Contassot-Vivier).

PhD: Marion Guthmuller, *Study of Legacy Distributed Applications through Simulation and Model-Checking for the Discovery and Validation of Behavioral Properties*, Université Nancy II, expected in 2014 (advisers: Sylvain Contassot-Vivier and Martin Quinson).

PhD: Tomasz Buchert, *Middleware for the orchestration of challenging large-scale experiments on distributed systems*, Université Nancy II, expected in 2014 (advisers: Jens Gustedt and Lucas Nussbaum).

master: Marion Guthmuller, *Vérification dynamique de propriétés de vivacité dans SimGrid*, ESIAL and Université Nancy I (adviser: Martin Quinson).

master: Luc Sarzyniec, *Accurate and efficient emulation of network topologies in Wrekavoc*, Université Nancy I (advisor: Lucas Nussbaum).

master: Yoanne Girardin, *Conducting experiments on a large-scale research testbed with a workflow management system*, ESIAL (advisor: Lucas Nussbaum).

engineering intern: Matías Ezequiel Vara *Ordered Read-Write Locks on Multicore Architectures* Universidad Nacional de La Plata (Argentina) (adviser Jens Gustedt)

10. Bibliography

Major publications by the team in recent years

[1] L.-C. CANON, O. DUBUISSON, J. GUSTEDT, E. JEANNOT. *Defining and Controlling the Heterogeneity of a Cluster: the Wrekavoc Tool*, in "Journal of Systems and Software", 2010, vol. 83, n^{o} 5, p. 786-802 [*DOI :* 10.1016/J.JSS.2009.11.734], http://hal.inria.fr/inria-00438616/en.

[2] H. CASANOVA, A. LEGRAND, M. QUINSON. *SimGrid: a Generic Framework for Large-Scale Distributed Experiments*, in "10th IEEE International Conference on Computer Modeling and Simulation - EUROSIM / UKSIM 2008", Royaume-Uni Cambrige, IEEE, 2008, http://hal.inria.fr/inria-00260697/en/.

[3] P.-N. Clauss, J. Gustedt. *Iterative Computations with Ordered Read-Write Locks*, in "Journal of Parallel and Distributed Computing", 2010, vol. 70, n^o 5, p. 496-504 [*DOI :* 10.1016/j.jpdc.2009.09.002], http://hal.inria.fr/inria-00330024/en.

[4] P.-N. Clauss, M. Stillwell, S. Genaud, F. Suter, H. Casanova, M. Quinson. *Single Node On-Line Simulation of MPI Applications with SMPI*, in "International Parallel & Distributed Processing Symposium", Anchorange (AK), États-Unis, IEEE, May 2011, http://hal.inria.fr/inria-00527150/en/.

[5] N. Fressengeas, H. Frezza-Buet, J. Gustedt, S. Vialle. *An Interactive Problem Modeller and PDE Solver, Distributed on Large Scale Architectures*, in "Third International Workshop on Distributed Frameworks for Multimedia Applications - DFMA '07", France Paris, IEEE, 2007, http://hal.inria.fr/inria-00139660/en/.

[6] A. H. Gebremedhin, J. Gustedt, M. Essaïdi, I. Guérin Lassous, J. A. Telle. *PRO: A Model for the Design and Analysis of Efficient and Scalable Parallel Algorithms*, in "Nordic Journal of Computing", 2006, vol. 13, p. 215-239, http://hal.inria.fr/inria-00000899/en/.

[7] J. Gustedt, E. Jeannot, M. Quinson. *Experimental Validation in Large-Scale Systems: a Survey of Methodologies*, in "Parallel Processing Letters", 2009, vol. 19, n^o 3, p. 399-418, RR-6859, http://hal.inria.fr/inria-00364180/en/.

[8] T. Jost, S. Contassot-Vivier, S. Vialle. *An efficient multi-algorithms sparse linear solver for GPUs*, in "Parallel Computing: From Multicores and GPU's to Petascale (Volume 19)", B. Chapman, F. Desprez, G. R. Joubert, A. Lichnewsky, F. Peters, T. Priol (editors), Advances in Parallel Computing, IOS Press, 2010, vol. 19, p. 546-553, Extended version of EuroGPU symposium article, in the International Conference on Parallel Computing (Parco) 2009 [*DOI :* 10.3233/978-1-60750-530-3-546], http://hal.inria.fr/hal-00485963/en.

[9] T. Kleinjung, L. Nussbaum, E. Thomé. *Using a grid platform for solving large sparse linear systems over GF(2)*, in "11th ACM/IEEE International Conference on Grid Computing (Grid 2010)", Brussels, Belgium, October 2010, http://hal.inria.fr/inria-00502899/en.

[10] C. Makassikis, V. Galtier, S. Vialle. *A Skeletal-Based Approach for the Development of Fault-Tolerant SPMD Applications*, in "The 11th International Conference on Parallel and Distributed Computing, Applications and Technologies - PDCAT 2010", Wuhan, China, December 2010 [*DOI :* 10.1109/PDCAT.2010.89], http://hal.inria.fr/inria-00548953/en.

Publications of the year

Doctoral Dissertations and Habilitation Theses

[11] C. Makassikis. *Conception d'un modèle et de frameworks de distribution d'applications sur grappes de PCs avec tolérance aux pannes à faible coût*, Université Henri Poincaré - Nancy I, February 2011, http://hal.inria.fr/tel-00591083/en.

[12] C. Rosa. *Performance & Correctness Assessment of Distributed Systems*, Université Henri Poincaré - Nancy I, October 2011, http://hal.inria.fr/.

Articles in International Peer-Reviewed Journal

[13] J. GUSTEDT, H. RAGHAVAN, P. SCHIMIT. *Exploring the random genesis of co-occurrence graphs*, in "Physica A: Statistical Mechanics and its Applications", March 2011, vol. 390, p. 1516 - 1528 [*DOI :* 10.1016/J.PHYSA.2010.12.036], http://hal.inria.fr/inria-00450684/en.

International Conferences with Proceedings

[14] T. BUCHERT, L. NUSSBAUM, J. GUSTEDT. *Methods for Emulation of Multi-Core CPU Performance*, in "13th IEEE International Conference on High Performance Computing and Communications (HPCC-2011)", Banff, Canada, IEEE, September 2011, p. 288 - 295 [*DOI :* 10.1109/HPCC.2011.45], http://hal.inria.fr/inria-00535534/en.

[15] P.-N. CLAUSS, M. STILLWELL, S. GENAUD, F. SUTER, H. CASANOVA, M. QUINSON. *Single Node On-Line Simulation of MPI Applications with SMPI*, in "International Parallel & Distributed Processing Symposium", Anchorange (AK), United States, IEEE, May 2011, http://hal.inria.fr/inria-00527150/en.

[16] F. DESPREZ, G. MARKOMANOLIS, M. QUINSON, F. SUTER. *Assessing the Performance of MPI Applications Through Time-Independent Trace Replay*, in "Second International Workshop on Parallel Software Tools and Tool Infrastructures (PSTI 2011) Held in conjunction with ICPP 2011, the 40th International Conference on Parallel Processing", Taipei, Taiwan, Province Of China, 2011, http://hal.inria.fr/inria-00546992/en.

[17] V. GALTIER, C. MAKASSIKIS, S. VIALLE. *A Javaspace-Based Framework for Efficient Fault-Tolerant Master-Worker Distributed Applications*, in "19th Euromicro International Conference on Parallel, Distributed and Network-Based Processing -PDP 2011", Ayia Napa, Cyprus, Y. COTRONIS, M. DANELUTTO, G. A. PAPADOPOULOS (editors), IEEE Computer Society, February 2011, p. 272-276 [*DOI :* 10.1109/PDP.2011.82], http://hal.inria.fr/hal-00618249/en.

[18] S. HERNANE, J. GUSTEDT, M. BENYETTOU. *Modeling and Experimental Validation of the Data Handover API*, in "Advances in Grid and Pervasive Computing", Oulu, Finland, J. RIEKKI, M. YLIANTTILA, M. GUO (editors), LNCS, Springer, May 2011, vol. 6646, p. 117-126, http://hal.inria.fr/inria-00547598/en.

[19] S. MERZ, M. QUINSON, C. ROSA. *SimGrid MC: Verification Support for a Multi-API Simulation Platform*, in "31st IFIP International Conference on Formal Techniques for Networked and Distributed Systems", Reykjavik, Iceland, R. BRUNI, J. DINGEL (editors), Lecture Notes in Computer Science, Springer, June 2011, vol. 6722, p. 274-288, The original publication is available at www.springerlink.com [*DOI :* 10.1007/978-3-642-21461-5_18], http://hal.inria.fr/inria-00593505/en.

[20] M. SAUGET, J. HENRIET, M. SALOMON, S. CONTASSOT-VIVIER. *Large datasets: a mixed method to adapt and improve their learning by neural networks used in regression contexts*, in "12th INNS EANN-SIG International Conference on Engineering Applications of Neural Networks - EANN 2011 and 7th IFIP WG 12.5 International Conference - AIAI 2011", Corfu, Greece, L. ILIADIS, C. JAYNE (editors), IFIP Advances in Information and Communication Technology, Springer, September 2011, vol. 363, p. 182–191 [*DOI :* 10.1007/978-3-642-23957-1_21], http://hal.inria.fr/hal-00643870/en.

National Conferences with Proceeding

[21] S. BADIA, L. NUSSBAUM. *gLite sur Grid'5000: vers une plate-forme d'expérimentation à taille réelle pour les grilles de production*, in "Rencontres scientifiques France Grilles", Lyon, France, September 2011, http://hal.inria.fr/inria-00626038/en.

[22] M. Guthmuller, L. Nussbaum, M. Quinson. *Émulation d'applications distribuées sur des plates-formes virtuelles simulées*, in "Rencontres francophones du Parallélisme (RenPar'20)", Saint Malo, France, May 2011, http://hal.inria.fr/inria-00565341/en.

[23] E. Jeanvoine, N. Niclausse, L. Nussbaum, D. Margery. *Déploiement et partitionnement dynamique de clusters avec Kadeploy et Kavlan*, in "JRES 2011", Toulouse, France, November 2011, 1, http://hal.inria.fr/hal-00640588/en.

[24] P. Morillon, L. Nussbaum, D. Margery. *Gestion d'une infrastructure expérimentale de grande échelle avec Puppet et Git*, in "JRES 2011", Toulouse, France, November 2011, 1, http://hal.inria.fr/hal-00640589/en.

Scientific Books (or Scientific Book chapters)

[25] M. Sauget, S. Contassot-Vivier, M. Salomon. *Parallelization of neural network building and training: an original decomposition method*, in "Horizons in Computer Science Research", T. S. Clary (editor), Nova Publishers, 2012, vol. 7, http://hal.inria.fr/hal-00643920/en.

[26] S. Vialle, S. Contassot-Vivier, T. Jost. *Optimizing computing and energy performances in heterogeneous clusters of CPUs and GPUs*, in "Handbook of Energy-Aware and Green Computing", I. Ahmad, S. Ranka (editors), Chapman & Hall/CRC, 2011, http://hal.inria.fr/hal-00643938/en.

Research Reports

[27] O. Beaumont, L. Bobelin, H. Casanova, P.-N. Clauss, B. Donassolo, L. Eyraud-Dubois, S. Genaud, S. Hunold, A. Legrand, M. Quinson, C. Rosa, L. Schnorr, M. Stillwell, F. Suter, C. Thiery, P. Velho, J.-M. Vincent, Y. Won. *Towards Scalable, Accurate, and Usable Simulations of Distributed Applications and Systems*, INRIA, October 2011, n^o RR-7761, http://hal.inria.fr/inria-00631141/en.

[28] J. Gustedt, E. Jeanvoine. *Relaxed Synchronization with Ordered Read-Write Locks*, INRIA, November 2011, n^o RR-7790, This article is accepted for publication in the post-proceedings of the Workshop on Algorithms and Programming Tools for Next-Generation High-Performance Scientific Software (HPSS) 2011, held in the context of Euro-Par 2011, August 29, 2011, Bordeaux, France, http://hal.inria.fr/hal-00639289/en.

[29] J. Gustedt, S. Vialle, H. Frezza-Buet, D. Boumba Sitou, N. Fressengeas, J. Fix. *InterCell: a Software Suite for Rapid Prototyping and Parallel Execution of Fine Grained Applications*, INRIA, November 2011, n^o RR-7814, accepted for publication in the post proceedings of PARA 2010, http://hal.inria.fr/hal-00645121/en.

[30] S. Hernane, J. Gustedt, M. Benyettou. *A Dynamic Distributed Algorithm for Read Write Locks (extended abstract)*, INRIA, November 2011, n^o RR-7798, as accepted for publication in PDP 2012, http://hal.inria.fr/hal-00641068/en.

[31] C. Makassikis, S. Vialle, X. Warin. *FT-GReLoSSS: a Skeletal-based approach towards application parallelization and low-overhead fault tolerance*, INRIA, November 2011, n^o RR-7797, http://hal.inria.fr/hal-00640558/en.

[32] M. Quinson, G. Oster. *The Java Learning Machine: A Learning Management System Dedicated To Computer Science Education*, INRIA, February 2011, n^o RR-7537, http://hal.inria.fr/inria-00565344/en.

[33] M. QUINSON, C. ROSA, C. THIERY. *Scalable and Fast Simulation of Peer-to-Peer Systems Using SimGrid*, INRIA, June 2011, n^o RR-7653, http://hal.inria.fr/inria-00602216/en.

References in notes

[34] J. M. BAHI, S. CONTASSOT-VIVIER, A. GIERSCH. *Load balancing in dynamic networks by bounded delays asynchronous diffusion*, in "10th International Meeting on High Performance Computing for Computational Science", Berkeley, États-Unis, 2010, Paper 31, An extended version is to appear in LNCS, http://hal.archives-ouvertes.fr/hal-00547300/en/.

[35] J. M. BAHI, S. CONTASSOT-VIVIER, M. SAUGET, A. VASSEUR. *A Parallel Incremental Learning Algorithm for Neural Networks with Fault Tolerance*, in "8th International Meeting on High Performance Computing for Computational Science, VECPAR'08", France Toulouse, 2008, p. 502–515, http://hal.inria.fr/inria-00336521/en/.

[36] M. BARISITS, W. BOYD. *MartinWilSim Grid Simulator*, Vienna UT and Georgia Tech, CERN, Switzerland, 2009, http://www.slideshare.net/wbinventor/slides-1884876.

[37] E. M. CLARKE, O. GRUMBERG, M. MINEA, D. PELED. *State Space Reduction Using Partial Order Techniques*, in "STTT", 1999, vol. 2, n^o 3, p. 279-287.

[38] W. DEPOORTER, N. MOOR, K. VANMECHELEN, J. BROECKHOVE. *Scalability of Grid Simulators: An Evaluation*, in "Proc. of the 14th Intl. Euro-Par Conf. on Parallel Processing", 2008, http://dx.doi.org/10.1007/978-3-540-85451-7_58.

[39] M. QUINSON. *GRAS: a Research and Development Framework for Grid and P2P Infrastructures*, in "The 18th IASTED International Conference on Parallel and Distributed Computing and Systems", IASTED, 2006, http://hal.inria.fr/inria-00108389.

[40] M. SAUGET, R. LAURENT, J. HENRIET, M. SALOMON, R. GSCHWIND, S. CONTASSOT-VIVIER, L. MAKOVICKA, C. SOUSSEN. *Efficient Domain Decomposition for a Neural Network Learning Algorithm, used for the Dose Evaluation in External Radiotherapy*, in "Artificial Neural Networks - ICANN 2010 20th International Conference Proceedings", K. DIAMANTARAS, W. DUCH, L. ILIADIS (editors), Lecture Notes in Computer Science, Volume 6352, Springer-Heidelberg, 2010, p. 261-266, ISBN-978-3-642-15818-6 [*DOI :* 10.1007/978-3-642-15819-3_34], http://hal.archives-ouvertes.fr/hal-00520154/en/.

[41] H. SUTTER. *The free lunch is over: A fundamental turn toward concurrency in software*, in "Dr. Dobb's Journal", 2005, vol. 30, n^o 3, http://www.gotw.ca/publications/concurrency-ddj.htm.

[42] A. VASSEUR, L. MAKOVICKA, É. MARTIN, M. SAUGET, S. CONTASSOT-VIVIER, J. M. BAHI. *Dose calculations using artificial neural networks: a feasibility study for photon beams*, in "Nucl. Instr. and Meth. in Phys. Res. B", 2008, vol. 266, n^o 7, p. 1085-1093.

[43] P. VELHO, A. LEGRAND. *Accuracy Study and Improvement of Network Simulation in the SimGrid Framework*, in "Proceedings of the 2nd International Conference on Simulation Tools and Techniques (SIMUTools'09)", Roma, Italia, March 2009.