INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team moais

Multi-programmation et Ordonnancement pour les Applications Interactives de Simulation

Grenoble - Rhône-Alpes

Theme : Distributed and High Performance Computing

Table of contents

The MOAIS project-team is supported by the INRIA and LIG lab (UMR 5217 - CNRS, Grenoble Universities: Grenoble-INP, UJF, UPMF).

1. Team

Research Scientist

Thierry Gautier [Research Scientist CR1]
Bruno Raffin [Research Scientist CR1, HdR]

Faculty Member

Jean-Louis Roch [Team leader, Associate Professor]
Vincent Danjean [Assistant Professor]
Pierre-François Dutot [Assistant Professor]
Guillaume Huard [Assistant Professor]
Grégory Mounié [Assistant Professor]
Denis Trystram [Professor, HdR]
Frédéric Wagner [Assistant Professor]
Clément Pernet [Assistant Professor]

Technical Staff

Daouda Traore [2009, CHOC and SCEPTRE Contract]
Christophe Laferrière [2009, CHOC contract]
Fabien Lementec [2009, Engineer ADT Kaapi]
Emilie Morel [2008]
Antoine Vanel [2008, Engineer ANR FVNano]

PhD Student

Sami Achour [2006, co-tutelle ESST Tunis, Tunisia (Mohamed Jemni), EGIDE scholarship]
Julien Bernard [2005, ATER Ensimag]
Xavier Besseron [2006, MRNT scholarship]
Marin Bougeret [2007, BDI CNRS / DGA scholarship]
Mohamed-Slim Bouguerra [2008, INRIA Cordi]
Daniel Cordeiro [2007, Alban scholarship]
Adel Essafi [2006, co-tutelle ESST Tunis, Tunisia (Mohamed Jemni), EGIDE scholarship]
Everton Hermann [2006, INRIA Cordi]
Ludovic Jacquin [2009, common to PLANETE and MOAIS]
Jean-Denis Lesage [2006, MRNT scholarship]
Yanik N'Goko [2006, co-tutelle Univ. Yaoundé, Cameroon, SARIMA scholarship]
Swann Perarnau [2008, MRNT scholarship]
Benjamin Petit [2007, common to PERCEPTION and MOAIS]
Jean-Noel Quintin [2008, CILOE constract scholarship]
Thomas Roche [2007, common to UJF-Institut Fourier and MOAIS, CIFRE C-S scholarship]
Lucas Schnorr [2007, co-tutelle INPG – UFRGS Brazil, CAPES COFECUB scholarship]
Marc Tchiboukdjian [2007, BDI CNRS / CEA DAM scholarship]
Gérald Vaisman [2006, DCN contract]
Haifeng Xu [2007, co-tutelle INPG – Zhejiang University, Hangzhou, China (Guochuan Zhang)]

Post-Doctoral Fellow

Ingo Assenmacher [Post-Doctoral Fellow]
Veronika Sonigo [Post-Doctoral Fellow]

Visiting Scientist

Jacek Blazewicz [Poznan University of Technology, 2.5 months]
Alfredo Goldman [USP Sao Paulo Brazil, 1 month]
Nicolas Maillard [UFRGS Porto Alegre, 1 month]

Administrative Assistant
Ahlem Zammit-Boubaker [INRIA Administrative Assistant, 50%]

2. Overall Objectives

2.1. Introduction

The objective of the MOAIS team-project is to develop the scientific and technological foundations for parallel programming that enable to achieve provable performances on distributed parallel architectures, from multi-processor systems on chips to computational grids and global computing platforms. Beyond the optimization of the application itself, the effective use of a larger number of resources is expected to enhance the performance. This encompasses large scale scientific interactive simulations (such as immersive virtual reality) that involve various resources: input (sensors, cameras, ...), computing units (processors, memory), output (videoprojectors, images wall) that play a prominent role in the development of high performance parallel computing.
The research directions of the MOAIS team are focused on the scheduling problem with a multi-criteria performance objective: precision, reactivity, resources comsuption, reliability, ... The originality of the MOAIS approach is to use the application's adaptability to enable its control by the scheduling. The critical points concern designing adaptive malleable algorithms and coupling the various components of the application to reach interactivity with performance guarantees.

The originality of the MOAIS approach is to use the application's adaptability to control its scheduling:

- the application describes synchronization conditions;
- the scheduler computes a schedule that verifies those conditions on the available resources;
- each resource behaves independently and performs the decision of the scheduler.

To enable the scheduler to drive the execution, the application is modeled by a macro data flow graph, a popular bridging model for parallel programming (BSP, Nesl, Earth, Jade, Cilk, Athapascan, Smarts, Satin, ...) and scheduling. A node represents the state transition of a given component; edges represent synchronizations between components. However, the application is malleable and this macro data flow is dynamic and recursive: depending on the available resources and/or the required precision, it may be unrolled to increase precision (e.g. zooming on parts of simulation) or enrolled to increase reactivity (e.g. respecting latency constraints). The decision of unrolling/enrolling is taken by the scheduler; the execution of this decision is performed by the application.

The MOAIS project-team is structured around four axis:

- **Scheduling**: To formalize and study the related scheduling problems, the critical points are: the modeling of an adaptive application; the formalization and the optimization of the multi-objective problems; the design of scalable scheduling algorithms. We are interested in classical combinatorial optimization methods (approximation algorithms, theoretical bounds and complexity analysis), and also in non-standard methods such as Game Theory.
- **Adaptive parallel and distributed algorithms**: To design and analyze algorithms that may adapt their execution under the control of the scheduling, the critical point is that algorithms are either parallel or distributed; then, adaptation should be performed locally while ensuring the coherency of results.
- **Programming interfaces and tools for coordination and execution**: To specify and implement interfaces that express coupling of components with various synchronization constraints, the critical point is to enable an efficient control of the coupling while ensuring coherency. We develop the **Kaapi** runtime software that manages the scheduling of multithreaded computations with billions of threads on a virtual architecture with an arbitrary number of resources; Kaapi supports node additions and resilience. Kaapi manages the *fine grain* scheduling of the computation part of the

application. To enable parallel application execution and analysis. We develop runtime tools that support large scale and fault tolerant processes deployment (**TakTuk**), visualization of parallel executions on heterogeneous platforms (**Triva**), reproducible CPU load generation on many-cores machines (**KRASH**).

- **Interactivity**: To improve interactivity, the critical point is scalability. The number of resources (including input and output devices) should be adapted without modification of the application. We develop the **FlowVR** middleware that enables to configure an application on a cluster with a fixed set of input and output resources. FlowVR manages the *coarse grain* scheduling of the whole application and the latency to produce outputs from the inputs.

Often, computing platforms have a dynamic behavior. The dataflow model of computation directly enables to take into account addition of resources. To deal with resilience, we develop softwares that provide **fault-tolerance** to dataflow computations. We distinguish non-malicious faults from malicious intrusions. Our approach is based on a checkpoint of the dataflow with bounded and amortized overhead.

For those themes, the scientific methodology of MOAIS consists in:

- designing algorithms with provable performance on generic theoretical models;
- implementing and evaluating those algorithms with our main softwares:
  - Kaapi for fine grain scheduling of compute-intensive applications;
  - FlowVR for coarse-grain scheduling of interactive applications;
  - TakTuk, a tool for large scale remote executions deployment.
  - Triva, for the visualization of heterogeneous parallel executions.
  - KRASH, to generate reproducible CPU load on many-cores machines.
- customizing our softwares for their use in real applications studied and developed by other partners. Applications are essential to the validation and further development of MOAIS results. Application fields are: virtual reality and scientific computing (simulation, visualization, combinatorial optimization, biology, computer algebra). Depending on the application the target architecture ranges from MPSoCs (multi-processor system on chips), multicore and GPU units to clusters and heterogeneous grids. In all cases, the performance is related to the efficient use of the available, often heterogeneous, parallel resources.

MOAIS research is not only oriented towards theory but also focuses on applicative software and hardware platforms developed with external partners. Significant efforts are made to build, manage and maintain these platforms. We are involved with other teams in four main platforms:

- SOFA, a real-time physics simulation engine http://www.sofa-framework.org/;
- Grimage, a 3D modeling and high performance 3D rendering platform http://www.inrialpes.fr/grimage);
- Digitalis, a 780 core cluster based on Intel Nehalem processors and Infiniband network. Digitalis is used both for batch computations and interactive applications;
- Grid'5000, the exprimental national grid (https://www.grid5000.fr/).

2.2. Highlights of the year

- The Moais, Perception and Evasion project-teams, in collaboration with the 4DViews Company, France and the Crescent Company, Japan, demonstrated the v-gate full-body immersive environment at the SIGGRAPH 2009 conference, New Orléans. This demo has been shown at the SIGGRAPH Emerging Technologies showcase during 4 days, after a selection by an international committee.

3. Scientific Foundations

3.1. Scheduling

Participants: Pierre-François Dutot, Thierry Gautier, Guillaume Huard, Grégory Mounié, Jean-Louis Roch, Denis Trystram, Frédéric Wagner.

The goal of this theme is to determine adequate multi-criteria objectives which are efficient (precision, reactivity, speed) and to study scheduling algorithms to reach these objectives.

In the context of parallel and distributed processing, the term *scheduling* is used with many acceptations. In general, scheduling means assigning tasks of a program (or processes) to the various components of a system (processors, communication links).

Researchers within MOAIS have been working on this subject for many years. They are known for their multiple contributions for determining the target dates and processors the tasks of a parallel program should be executed; especially regarding execution models (taking into account inter-task communications or any other system features) and the design of efficient algorithms (for which there exists a performance guarantee relative to the optimal scheduling).

Parallel tasks model and extensions. We have contributed to the definition and promotion of modern task models: parallel moldable tasks and divisible load. For both models, we have developed new techniques to derive efficient scheduling algorithms (with a good performance guaranty). We proposed recently some extensions taking into account machine unavailabilities (reservations).

Multi-objective Optimization. A natural question while designing practical scheduling algorithms is "which criterion should be optimized ?". Most existing works have been developed for minimizing the *makespan* (time of the latest tasks to be executed). This objective corresponds to a system administrator view who wants to be able to complete all the waiting jobs as soon as possible. The user, from his-her point of view, would be more interested in minimizing the average of the completion times (called *minsum*) of the whole set of submitted jobs. There exist several other objectives which may be pertinent for specific use. We worked on the problem of designing scheduling algorithms that optimize simultaneously several objectives with a theoretical guarantee on each objective. The main issue is that most of the policies are good for one criterion but bad for another one.
We have proposed an algorithm that is guaranteed for both *makespan* and *minsum*. This algorithm has been implemented for managing the resources of a cluster of the regional grid CIMENT. More recently, we extended such analysis to other objectives (makespan and reliability). We concentrate now on finding good algorithms able to schedule a set of jobs with a large variety of objectives simultaneously. For hard problems, we propose approximation of Pareto curves (best compromizes).

Incertainties. Most of the new execution supports are characterized by a higher complexity in predicting the parameters (high versatility in desktop grids, machine crash, communication congestion, cache effects, etc.). We studied some time ago the impact of incertainties on the scheduling algorithms. There are several ways for dealing with this problem: First, it is possible to design robust algorithms that can optimized a problem over a set of scenarii, another solution is to design flexible algorithms. Finally, we promote semi on-line approaches that start from an optimized off-line solution computed on an initial data set and updated during the execution on the "perturbed" data (stability analysis).

Game Theory. Game Theory is a framework that can be used for obtaining good solution of both previous problems (multi-objective optimization and incertain data). On the first hand, it can be used as a complement of multi-objective analysis. On the other hand, it can take into account the incertainties. We are curently working at formalizing the concept of cooperation.

Scheduling for optimizing parallel time and memory space. It is well known that parallel time and memory space are two antagonists criteria. However, for many scientific computations, the use of parallel architectures is motivated by increasing both the computation power and the memory space. Also, scheduling for optimizing both parallel time and memory space targets an important multicriteria objective. Based on the analysis of the dataflow related to the execution, we have proposed a scheduling algorithm with provable performance.

Coarse-grain scheduling of fine grain multithreaded computations on heterogeneous platforms. Designing multi-objective scheduling algorithms is a transversal problem. Work-stealing scheduling is well studied for fine grain multithreaded computations with a small critical time: the speed-up is asymptotically optimal. However, since the number of tasks to manage is huge, the control of the scheduling is expensive. We proposed a generalized lock-free cactus stack execution mechanism, to extend previous results, mainly from Cilk, based

on the *work-first principle* for strict multi-threaded computations on SMPs to general multithreaded computations with dataflow dependencies. The main result is that optimizing the sequential local executions of tasks enables to amortize the overhead of scheduling. This distributed work-stealing scheduling algorithm has been implemented in **Kaapi**

3.2. Adaptive Parallel and Distributed Algorithms Design

Participants: Pierre-François Dutot, Thierry Gautier, Guillaume Huard, Bruno Raffin, Jean-Louis Roch, Denis Trystram, Frédéric Wagner.

This theme deals with the analysis and the design of algorithmic schemes that control (statically or dynamically) the grain of interactive applications.

The classical approach consists in setting in advance the number of processors for an application, the execution being limited to the use of these processors. This approach is restricted to a constant number of identical resources and for regular computations. To deal with irregularity (data and/or computations on the one hand; heterogeneous and/or dynamical resources on the other hand), an alternate approach consists in adapting the potential parallelism degree to the one suited to the resources. Two cases are distinguished:

- in the classical bottom-up approach, the application provides fine grain tasks; then those tasks are clustered to obtain a minimal parallel degree.
- the top-down approach (Cilk, Hood, Athapascan) is based on a work-stealing scheduling driven by idle resources. A local sequential depth-first execution of tasks is favored when recursive parallelism is available.

Ideally, a good parallel execution can be viewed as a flow of computations flowing through resources with no control overhead. To minimize control overhead, the application has to be adapted: a parallel algorithm on p resources is not efficient on $q < p$ resources. On one processor, the scheduler should execute a sequential algorithm instead of emulating a parallel one. Then, the scheduler should adapt to resource availability by changing its underlying algorithm. This first way of adapting granularity is implemented by Kaapi (default work-stealing schedule based on work-first principle); an implementation of Athapascan, the parallel programming interface developed by the APACHE project, is available on top of Kaapi.

However, this adaptation is restrictive. More generally, the algorithm should adapt itself at runtime to improve its performance by decreasing the overheads induced by parallelism, namely the arithmetic operations and communications. This motivates the development of new parallel algorithmic schemes that enable the scheduler to control the distribution between computation and communication (grain) in the application to find the good balance between parallelism and synchronizations. MOAIS has exhibited several techniques to manage adaptivity from an algorithmic point of view:

- amortization of the number of global synchronizations required in an iteration (for the evaluation of a stopping criterion);
- adaptive deployment of an application based on on-line discovery and performance measurements of communication links;
- generic recursive cascading of two kind of algorithms: a sequential one, to provide efficient executions on the local resource, and a parallel one that enables an idle resource to extract parallelism to dynamically suit the degree of parallelism to the available resources.

The generic underlying approach consists in finding a good mix of various algorithms, what is often called a "poly-algorithm". Particular instances of this approach are Atlas library (performance benchmark are used to decide at compile time the best block size and instruction interleaving for sequential matrix product) and FFTW library (at run time, the best recursive splitting of the FFT butterfly scheme is precomputed by dynamic programming). Both cases rely on pre-benchmarking of the algorithms. Our approach is more general in the sense that it also enables to tune the granularity at any time during execution. The objective is to develop processor oblivious algorithms: similarly to cache oblivious algorithms, we define a parallel algorithm as *processor-oblivious* if no program variable that depends on architecture parameters, such as the number or processors or their respective speeds, needs to be tuned to minimize the algorithm runtime.

We have applied this technique to develop processor oblivious algorithms for several applications with provable performance: iterated and prefix sum (partial sums) computations, stream computations (cipher and hd-video transformation), 3D image reconstruction (based on the concurrent usage of multi-core and GPU), loop computations with early termination. Finally, to validate these novel parallel computation schemes, we developed a tool named **KRASH**. This tool is able to generate dynamic CPU load in a reproducible way on many-cores machines. Thus, by providing the same experimental conditions to several parallel applications, it enables users to evaluate the efficiency of resource uses for each approach.

This adaptation technique is now integrated in softwares that we are developing with external partners within contracts. In particular, in partnership with STM within the Minalogic SCEPTRE contract (completed in Q4 2009), we have developed a specific optimized C interface, dedicated to stream computation for multi-processor system on chips (MPSoC); this interface is named AWS (Adaptive Work-Stealing).

Besides, we developed a parallel implementation of the C++ Standard Template Library STL on top of Kaapi; this library, named KaSTL, provides adaptive parallel algorithms for distributed containers (such as transform, foreach and findif on vectors). By optimizing the work-stealing to our adaptive algorithm scheme, a new non-blocking (wait-free) implementation of Kaapi has been designed. A first prototype of this C library, named X-Kaapi, will be delivered to STM in Q4 2009. The benchmarks experimented on SMPs and NUMAs architectures provides good performances with respect to concurrent libraries MCSTL, PaSTL, Intel TBB, and Cilk++, while improving the grain where parallelism can be exploited.

Extensions concern the development of algorithms that are both cache and processor oblivious. The processor algorithms proposed for prefix sums and segmentation of an array are cache oblivious too. We are currently working on sorting and mesh partitioning within a collaboration with the CEA.

3.3. Interactivity

Participants: Vincent Danjean, Pierre-François Dutot, Thierry Gautier, Bruno Raffin, Jean-Louis Roch.

The goal of this theme is to develop approaches to tackle interactivity in the context of large scale distributed applications.

We distinguish 2 types of interactions. A user can interact with an application having only little insight about the internal details of the program running. This is typically the case for a virtual reality application where the user just manipulates 3D objects. We have a "user-in-the-loop". In opposite, we have an "expert -in-the-loop" if the user is an expert that knows the limits of the progam that is being executed and that he can interacts with it to steer the execution. This is the case for instance when the user can change some parameters during the execution to improve the convergence of a computation.

3.3.1. User-in-the-loop

Some applications, like virtual reality applications, must comply with interactivity constraints. The user should be able to observe and interact with the application with an acceptable reaction delay. To reach this goal the user is often ready to accept a lower level of details. To execute such application on a distributed architecture requires to balance the workload and activation frequency of the different tasks. The goal is to optimize CPU and network resource use to get as close as possible to the reactivity/level of detail the user expect.

Virtual reality environments significantly improve the quality of the interaction by providing advanced interfaces. The display surface provided by multiple projectors in CAVE -like systems for instance, allows a high resolution rendering on a large surface. Stereoscopic visualization gives an information of depth. Sound and haptic systems (force feedback) can provide extra information in addition to visualized data. However driving such an environment requires an important computation power and raises difficult issues of synchronization to maintain the overall application coherent while guaranteeing a good latency, bandwidth (or refresh rate) and level of details. We define the coherency as the fact that the information provided to the different user senses at a given moment are related to the same simulated time.

Today's availability of high performance commodity components including networks, CPUs as well as graphics or sound cards make it possible to build large clusters or grid environments providing the necessary resources to enlarge the class of applications that can aspire to an interactive execution. However the approaches usually used for mid size parallel machines are not adapted. Typically, there exist two different approaches to handle data exchange between the processes (or threads). The synchronous (or FIFO) approach ensures all messages sent are received in the order they were sent. In this case, a process cannot compute a new state if all incoming buffers do not store at least one message each. As a consequence, the application refresh rate is driven by the slowest process. This can be improved if the user knows the relative speed of each module and specify a read frequency on each of the incoming buffers. This approach ensures a strong coherency but impact on latency. This is the approach commonly used to ensure the global coherency of the images displayed in multi-projector environments.The other approach, the asynchronous one, comes from sampling systems. The producer updates data in a shared buffer asynchronously read by the consumer. Some updates may be lost if the consumer is slower than the producer. The process refresh rates are therefore totally independent. Latency is improved as produced data are consumed as soon as possible, but no coherency is ensured. This approach is commonly used when coupling haptic and visualization systems. A fine tuning of the application usually leads to satisfactory results where the user does not experience major incoherences. However, in both cases, increasing the number of computing nodes quickly makes infeasible hand tuning to keep coherency and good performance.

We propose to develop techniques to manage a distributed interactive application regarding the following criteria :

- latency (the application reactivity);
- refresh rate (the application continuity);
- coherency (between the different components);
- level of detail (the precision of computations).

We developed a programming environment, called FlowVR, that enables the expression and realization of loosen but controlled coherency policies between data flows. The goal is to give users the possibility to express a large variety of coherency policies from a strong coherency based on a synchronous approach to an uncontrolled coherency based on an asynchronous approach. It enables the user to loosen coherency where it is acceptable, to improve asynchronism and thus performance. This approach maximizes the refresh rate and minimizes the latency given the coherency policy and a fixed level of details. It still requires the user to tune many parameters. In a second step, we are planning to explore auto-adaptive techniques that enable to decrease the number of parameters that must be user tuned. The goal is to take into account (possibly dynamically) user specified high level parameters like target latencies, bandwidths and levels of details, and to have the system automatically adapt to reach a trade-off given the user wishes and the resources available. Issues include multi-criterion optimizations, adaptive algorithmic schemes, distributed decision making, global stability and balance of the regulation effort.

3.3.2. Expert-in-the-loop

Some applications can be interactively guided by an expert who may give advices or answer specific questions to hasten a problem resolution. A theoretical framework has been developed in the last decade to define precisely the complexity of a problem when interactions with an expert is allowed. We are studying these interactive proof systems and interactive complexity classes in order to define efficient interactive algorithms dedicated to scheduling problems. This, in particular, applies to load-balancing of interactive simulations when a user interaction can generate a sudden surge of imbalance which could be easily predicted by an operator.

3.4. Adaptive middleware for code coupling and data movements

Participants: Vincent Danjean, Thierry Gautier, Bruno Raffin, Jean-Louis Roch, Frédéric Wagner.

This theme deals with the design and implementation of programming interfaces in order to achieve an efficient coupling of distributed components.

The implementation of interactive simulation application requires to assemble together various software components and to ensure a semantic on the displayed result. To take into account functional aspects of the computation (inputs, outputs) as well as non functional aspects (bandwidth, latency, persistence), elementary actions (method invocation, communication) have to be coordinated in order to meet some performance objective (precision, quality, fluidity, *etc*). In such a context the scheduling algorithm plays an important role to adapt the computational power of a cluster architecture to the dynamic behavior due to the interactivity. Whatever the scheduling algorithm is, it is fundamental to enable the control of the simulation. The purpose of this research theme is to specify the semantics of the operators that perform components assembling and to develop a prototype to experiment our proposals on real architectures and applications.

3.4.1. Application Programming Interface

The specification of an API to compose interactive simulation application requires to characterize the components and the interaction between components.The respect of causality between elementary events ensures, at the application level, that a reader will see the *last* write with respect to an order. Such a consistency should be defined at the level of the application to control the events ordered by a chain of causality. For instance, one of the result of Athapascan was to prove that a data flow consistency is more efficient than other ones because it generates fewer messages. Beyond causality based interactions, new models of interaction should be studied to capture non predictable events (delay of communication, capture of image) while ensuring a semantic.

Our methodology is based on the characterization of interactions required between components in the context of an interactive simulation application. For instance, criteria could be coherency of visualization, degree of interactivity. Beyond such characterization we hope to provide an operational semantic of interactions (at least well suited and understood by usage) and a cost model. Moreover they should be preserved by composition to predict the cost of an execution for part of the application.

This work is based on the experience of the APACHE project and the collaborative research actions ARC SIMBIO and ARC COUPLAGE. The main result relies on a computable representation of the future of an execution; representations such as macro data flow are well suited because they explicit which data are required by a task. Such a representation can be built at runtime by an interpretation technique: the execution of a function call is differed by computing beforehand at runtime a graph of tasks that represents the (future) calls to execute. Based on this technique, Athapascan, the language developed by the APACHE project, enables to write a single program for both the code to execute and the description of the future of the execution.

3.4.2. Kernel for Asynchronous, Adaptive, Parallel and Interactive Application

Managing the complexity related to fine grain components and reaching high efficiency on a cluster architecture require to consider a dynamic behavior. Also, the runtime kernel is based on a representation of the execution: data flow graph with attributes for each node and efficient operators will be the basis for our software. This kernel has to be specialized for the considered applications. The low layer of the kernel has features to transfer data and to perform remote signalization efficiently. Well known techniques and legacy code have to be reused. For instance, multithreading, asynchronous invocation, overlapping of latency by computing, parallel communication and parallel algorithms for collective operations are fundamental techniques to reach performance. Because the choice of the scheduling algorithm depends on the application and the architecture, the kernel will provide an *causally connected representation* of the system that is running. This allows to specialize the computation of a good schedule of the data flow graph by providing algorithms (scheduling algorithms for instance) that compute on this (causally connected) representation: any modification of the representation is turned into a modification on the system (the parallel program under execution). Moreover, the kernel provides a set of basic operators to manipulate the graph (*e.g.* computes a partition from a schedule, remapping tasks, ...) to allow to control a distributed execution. A new non-blocking implementation of Kaapi is currently developed; avoiding locks and heavy atomic operations such as compare-and-swap, it is an almost wait-free implementation of adaptive work-stealing.

4. Application Domains

4.1. Virtual Reality

Participants: Thierry Gautier, Bruno Raffin, Jean-Louis Roch.

We are pursuing and extending existing collaborations to develop virtual reality applications on PC clusters and grid environments:

- Real time 3D modeling. An on-going collaboration with the PERCEPTION project focuses on developing solutions to enable real time 3D modeling from multiple cameras using a PC cluster. An operational code base was transferred to the 4DViews Start-up in September 2007. 4DViews is now selling turn key solutions for real-time 3D modeling. Recent developments take two main directions:
  - Using a HMD (Head Mounted Display) and a Head Mounted Camera to provide the user a high level of interaction and immersion in the mixed reality environment. Having a mobile camera raises several concerns. The camera position and orientation need to be precisely known at anytime, requiring to develop on-line calibration approaches. The background subtraction cannot anymore be based on a static background learning for the mobile camera, required here too new algorithms.
  - Distributed collaboration across distant sites. In the context of the ANR DALIA we are developing a collaborative application where a user at Bordeaux (iParla project-team) using a real time 3D modeling platform can meet in a virtual world with a user in Grenoble also using a similar platform. We rely on the Grid'5000 dedicated 10 Gbits/s network to enable a low latency. The main issues are related to data transfers that need to be carefully managed to ensure a good latency while keeping a good quality, and the development of new interaction paradigms.

  On these issues, Benjamin Petit started a Ph.D. in October 2007, co-advised by Edmond Boyer (PERCEPTION) and Bruno Raffin.
- Real time physical simulation. We are collaborating with the EVASION project on the SOFA simulation framework. Everton Hermann, a Ph.D. co-advised by François Faure (EVASION) and Bruno Raffin, works on parallelizing SOFA using the KAAPI programming environment. The challenge is to provide SOFA with a parallelization that is efficient (real-time) while not being invasive for SOFA programmers (usually not parallel programmer). We developed a first version using the Kaapi environment for SMP machines that relies on a mix of work-stealing and dependency graph analysis and partitioning. A second version targets machines with multiples CPUs and multiple GPUs. We extended the initial framework to support a work stealing based load balancing between CPUs and GPUs. It required to extend Kaapi to support heterogeneous tasks (GPU and CPU ones) and to adapt the work stealing strategy to limit data transfers between CPUs and GPUs (the main bottleneck for GPU computing).
- Distant collaborative work. We conduct experiments using FlowVR for running applications on Grid environments. Two kinds of experiments will be considered: collaborative work by coupling two or more distant VR environments ; large scale interactive simulation using computing resources from the grid. For these experiments, we are collaborating with the LIFO and the LABRI.
- Parallel cache-oblivious algorithms for scientific visualization. In collaboration with the CEA DAM, we have developed a cache-oblivious algorithm with provable performance for irregulars meshes. Based on this work, we are studying parallel algorithms that take advantage of the shared cache usually encountered on multi-core architectures (L3 shared cache). The goal is to have the cores collaborating to efficiently share the L3 cache for a better performance than with a more traditional approach that leads to split the L3 cache between the cores. We are obtaining good performance gains with a parallel iso-surface extraction algorithm. This work is the main focus of Marc Tchiboukdjian Ph.D.

4.2. Code Coupling and Grid Programming

Participants: Thierry Gautier, Jean-Louis Roch, Vincent Danjean, Frédéric Wagner.

Code coupling aim is to assemble component to build distributed applications by reusing legacy code. The objective here is to build high performance applications for cluster and grid infrastructures.

- **Grid programming model and runtime support.** Programming the grid is a challenging problem. The MOAIS Team has a strong knowledge in parallel algorithms and develop a runtime support for scheduling grid program written in a very high level interface. The parallelism from recursive divide and conquer applications and those from iterative simulation are studied. Scheduling heuristics are based on online work stealing for the former class of applications, and on hierarchical partitioning for the latter. The runtime support provides capabilities to hide latency by computation thanks to a non-blocking one-side communication protocol and by re-ordering computational tasks.
- **Grid application deployment.** To test grid applications, we need to deploy and start programs on all used computers. This can become difficult if the real topology involves several clusters with firewall, different runtime environments, etc. The MOAIS Team designed and implemented a new tool called `karun` that allows a user to easily deploy a parallel application wrote with the KAAPI software. This KAAPI tool relies on the `TakTuk` software to quickly launch programs on all nodes. The user only needs to describe the hierarchical networks/clusters involved in the experiment with their firewall if any.
- **Visualization of grid applications execution.** The analysis of applications execution on the grid is challenging both because of the large scale of the platform and because of the heterogeneous topology of the interconnections. To help users to understand their application behavior and to detect potential bottleneck or load unbalance, the MOAIS team designed and implemented a tool named **Triva**. This tool proposes a new three dimensional visualization model that combines topological information to space time data collected during the execution. It also proposes an aggregation mechanism that eases the detection of application load unbalance.

4.3. Safe Distributed Computations

Participants: Vincent Danjean, Thierry Gautier, Clément Pernet, Jean-Louis Roch.

Large scale distributed platforms, such as the GRID and Peer-to-Peer computing systems, gather thousands of nodes for computing parallel applications. At this scale, component failures, disconnections (fail-stop faults) or results modifications (malicious faults) are part of operation, and applications have to deal directly with repeated failures during program runs. Indeed, since failure rate in such platform is proportional to the number of involved resources, the mean time between failure is dramatically decreased on very large size architectures. Moreover, even if a middleware is used to secure the communications and to manage the resources, the computational nodes operate in an unbounded environment and are subject to a wide range of attacks able to break confidentiality or to alter the resources or the computed results. Beyond fault-tolerancy, yet the possibility of massive attacks resulting in an error rate larger than tolerable by the application has to be considered. Such massive attacks are especially of concern due to Distributed Denial of Service, virus or Trojan attacks, and more generally orchestrated attacks against widespread vulnerabilities of a specific operating system that may result in the corruption of a large number of resources. The challenge is then to provide confidence to the parties about the use of such an unbound infrastructure. The MOAIS team addresses two issues:

- fault tolerance (node failures and disconnections): based on a global distributed consistent state , for the sake of scalability;
- security aspects: confidentiality, authentication and integrity of the computations.

Our approach to solve those problems is based on the efficient checkpointing of the dataflow that described the computation at coarse-grain. This distributed checkpoint, based on the local stack of each work-stealer process, provides a causally linked representation of the state. It is used for a scalable checkpoint/restart protocol and for probabilistic detection of massive attacks.

Moreover, we study the scalability of security protocols on large scale infrastructures. To open the grid usage to commercial applications from small-size companies (namely in the field of micro and nano-technology within the global competitiveness cluster Minalogic in Grenoble), we are currently studying the scalability issues related to systematic ciphering of all components of a distributed application in relation with CS Group (thesis of Thomas Roche, CIFRE scholarship). Dedicated to multicore architectures, an adaptive parallelization of a block cipher (based on counter mode) has been evaluated. Within the SHIVA contract and the Ph.D. of Ludovic Jacquin (coadvised with the PLANETE EPI), we develop a high-rate systematic ciphering architecture based on the coupling of a multicore architecture with security components (FPGA and smart card).

4.4. Embedded Systems

Participants: Jean-Louis Roch, Guillaume Huard, Denis Trystram, Vincent Danjean.

To improve the performance of current embedded systems, Multiprocessor System-on-Chip (MPSoC) offers many advantages, especially in terms of flexibility and low cost. Multimedia applications, such as video encoding, require more and more intensive computations. The system should be able to exploit the resources as much as possible to save power and time. This challenge may be addressed by parallel computing coupled with performant scheduling. On-going work focuses on reusing the scheduling technologies developed in MOAIS for embedded systems.

In the framework of our cooperation with STM (Serge de Paoli, Miguel Santana) and within the SCEPTRE project (global competitiveness cluster MINALOGIC/EMSOC), Julien Bernard in his thesis (grant cofunded by STM and CNRS) provides a specialized version of Kaapi for adaptive stream computations, named AWS, on MPSoCs platforms. AWS has been implemented and is being evaluated on two platforms: STM-8010 (3 processors on chip) and a cycle-approximate simulation (TIMA, Frédéric Pétrot). We are also studying self-specialized implementation of work-stealing from an abstract description (from SPIRIT standard) of the MPSoC architecture. Since those applications are developed based on component models, we are developing adaptive schedules for such component applications within the Nano2012 HiPeCoMP contract.

We are also considering adaptive algorithms to take advantage of the new trend of computers to integrate several computing units that may have different computing abilities. For instance today machines can be built with several dual-core processors and graphical processing units. New architectures, like the Cell processors, also integrate several computing units. First works concern balancing work load on multi GPU and CPU architectures workload balancing for scientific visualization problems.

5. Software

5.1. FlowVR

Participants: Jean-Denis Lesage, Bruno Raffin [correspondant].

The goal of the **FlowVR** library is to provide users with the necessary tools to develop and run high performance interactive applications on PC clusters and Grids. The main target applications include virtual reality and scientific visualization. FlowVR enforces a modular programming that leverages software engineering issues while enabling high performance executions on distributed and parallel architectures.

The FlowVR software suite has today 3 main components:

- **FlowVR**: The core middleware library. FlowVR relies on data-flow hierarchical component model. Developing a FlowVR application is a two step process. First, modules are developed. Modules encapsulate a piece of code, imported from an existing application or developed from scratch. The code can be a multi-threaded or parallel, as FlowVR enables parallel code coupling. In a second step, modules are mapped on the target architecture and assembled into a network to define how data are exchanged. This network can make use of advanced features, from simple routing operations to complex message filtering or synchronization operations.

- **FlowVR Render**: A parallel rendering library. FlowVR Render proposes a framework to take advantage of the power offered by graphics clusters to drive display walls or immersive multi-projector environments like Caves. It relies on an original approach making an intensive use of hardware shaders. FlowVR Render comes with a port of the MPlayer Movie Player. This enables to play movies on a multi display environment. This application also a good example of the potential of FlowVR and FlowVR Render.
- **VTK FlowVR**: a VTK / FlowVR / FlowVR Render coupling library. VTK FlowVR enables to render VTK applications using FlowVR Render with minimal modifications of the original code. VTK FlowVR enables to encapsulate VTK code into FlowVR modules to get access to the FlowVR capabilities for modularizing and distributing VTK processings.

The FlowVR suite is freely available under a GP/LGPL licence at http://flowvr.sf.net with a full documentation and related publications.

5.2. Kaapi - Kernel for Asynchronous, Adaptive, Parallel and Interactive Application

Participants: Vincent Danjean, Thierry Gautier [correspondant], Frédéric Wagner.

Kaapi is an efficient fine grain multithreaded runtime that runs on more than 1000 processors and supports addition/resilience of resources. Kaapi means *Kernel for Asynchronous, Adaptive, Parallel and Interactive Application*. Kaapi runtime support uses a macro data flow representation to build, schedule and execute programs on distributed architectures. Kaapi allows the programmer to tune the scheduling algorithm used to execute its application. Currently, Kaapi only considers data dependencies between multiple producers and multiple consumers. A high level C++ API, called Athapascan and developed by the APACHE project, is implemented on top of Kaapi. Kaapi provides methods to schedule a data flow on multiple processors and then to evaluate it on a parallel architecture. The important key point is the way communications are handled. At a low level of implementation, Kaapi uses an active message protocol to perform very high performance remote write and remote signalization operations. This protocol has been ported on top of various networks (Ethernet/Socket, Myrinet/GM). Moreover, Kaapi is able to generate broadcasts and reductions that are critical for efficiency.

The performance of applications on top of Kaapi scales on clusters and large SMP machines (Symmetric Multi Processors): the kernel is developed using distributed algorithms to reduce synchronizations between threads and UNIX processes. Kaapi, through the use of the Athapascan interface, has been used to compute combinatorial optimization problems on the French Grid Etoile and Grid5000. SOFA is today key target application for Kaapi software.

X-Kaapi is the basic C library that implements non-blocking adaptive work-stealing. The work stealing algorithm implemented in Kaapi has a predictive cost model. Kaapi is able to report important measures to capture the parallel complexity or parallel bottleneck of an application.

Kaapi is developed for UNIX platform and has been ported on most of the UNIX systems (LINUX, IRIX, Mac OS X); it is compliant with both 32 bits and 64 bits architectures (IA32, G4, IA64, G5, MIPS). All Kaapi related material are available at https://gforge.inria.fr/projects/kaapi/ under CeCILL licence.

5.3. TakTuk - Adaptive large scale remote execution deployment

Participant: Guillaume Huard [corespondant].

TakTuk is a tool for deploying remote execution commands to a potentially large set of remote nodes. It spreads itself using an adaptive algorithm and set up an interconnection network to transport commands and perform I/Os multiplexing/demultiplexing. The TakTuk algorithms dynamically adapt to environment (machine performance and current load, network contention) by using a reactive algorithm that mix local parallelization and work distribution.

Characteristics:

- adaptivity: efficient work distribution is achieved even on heterogeneous platforms thanks to an adaptive work-stealing algorithm
- scalability TakTuk has been tested to perform large size deployments (hundreds of nodes), either on SMPs, regular clusters or clusters of SMPs
- portability: TakTuk is architecture independent (tested on x86, PPC, IA-64) and distinct instances can communicate whatever the machine they're running on
- configurability: mechanics are configurable (deployment window size, timeouts, ...) and TakTuk outputs can be suppressed/formatted using I/O templates

Outstanding features:

- auto-propagation: the engine can spread its own code to remote nodes in order to deploy itself
- communication layer: nodes successfully deployed are numbered and perl scripts executed by TakTuk can send multicast communications to other nodes using this logical number
- information redirection: I/O and commands status are multiplexed from/to the root node.

http://taktuk.gforge.inria.fr under GNU GPL licence.

5.4. Triva - Three dimension-Interactive Visualization Analysis

Participants: Lucas Schnorr [corespondant], Guillaume Huard.

Parallel applications use grid infrastructures to obtain more performance during their execution. The successful result of these executions depends directly on a performance analysis that takes into account the grid characteristics, such as the network topology and resources location. Triva is a software analysis tool that implements a novel technique to visualize the behavior of parallel applications. The proposed technique explores 3D graphics in order to show the application behavior together with a description of the resources, highlighting communication patterns, the network topology and a visual representation of a logical organization of the resources. We have used a real grid infrastructure in order to execute and trace applications composed of thousands of processes. http://triva.gforge.inria.fr under GNU GPL licence.

5.5. KRASH - Kernel for Reproduction and Analysis of System Heterogeneity

Participants: Swann Perarnau [corespondant], Guillaume Huard.

KRASH is a tool for reproducible generation of system-level CPU load. This tool is intended for use in shared memory machines equipped with multiple CPU cores that are usually exploited concurrently by several users. The objective of KRASH is to enable parallel application developers to validate their resources use strategies on a partially loaded machine by *replaying* an observed load in concurrence with their application. To reach this objective, KRASH relies on a method for CPU load generation which behaves as realistically as possible: the resulting load is similar to the load that would be produced by concurrent processes run by other users. Nevertheless, contrary to a simple run of a CPU-intensive application, KRASH is not sensitive to system scheduling decisions. The main benefit brought by KRASH is this reproducibility: no matter how many processes are present in the system the load generated by our tool strictly respects a given load profile. This last characteristic proves to be hard to achieve using simple methods because the system scheduler is supposed to share the resources fairly among running processes. http://krash.ligforge.imag.fr under GNU GPL licence.

6. New Results

6.1. Parallel algorithms, complexity and scheduling

6.1.1. Scheduling

The work on scheduling mainly concerns multi-objective optimization and jobs scheduling on resources grid. We have exhibited techniques to find good trade-off between criteria. Mainly we achieved two main results. First, we characterized a multi-user problem with an algorithm achieving a constant approximation to the pareto curve. This part is related to our previous works related to results of the game theory. Secondly, we extended the spectrum of multi-criteria results to include either numerous criteria or new and radically different criteria like reliability or memory consumption versus execution time.

Regarding work-stealing and greedy scheduling, new analysis are developed that enables the development of several variants of work-stealing that achieve provable performances. An important motivation was to relax the classical restrictions. Yet we have considered both following extensions: tasks with bounded fan-out; considering push greedy scheduling strategy coupled with standard work-stealing (poll); taking into account other criteria then depth (eg critical height and locality).

6.1.2. Adaptive algorithm

New results, both theoretical and experimental, have been obtained with respect to the bi-criteria work/depth threshold in order to reach asymptotic optimal running time on distributed architectures with processors of heterogeneous frequencies.

Provable work-optimal parallelizations of STL (Standard Template Library) algorithms based on the work-stealing technique has been achieved. Build on top of Kaapi work-stealing, the KaSTL library provides adaptive implementations of the STL C++ library. Unlike previous approaches where a deque for each processor is typically used to locally store ready tasks and where a processor that runs out of work steals a ready task from the deque of a randomly selected processor, overhead for task creations is reduced based on an original implementation of work-stealing without using any deque but a distributed list.

Based on our results on the adaptive coupling of GPU and CPU parallelism for interactive 3D modeling, another perspective is to take benefit of Lastly to provide fine grain adaptive parallelization of a part of the SOFA library.

6.1.3. Safe distributed computation

In the fail-silent model, an efficient coordinated checkpoint mechanism of the dataflow that described the computation at coarse-grain has been developed and integrated into Kaapi (Xavier Besseron PhD Thesis). It extends the IEEE TDSC paper for iterative applications to take into account the knowledge of the dependencies among processors to speedup restart time after a failure.

With respect to malicious faults (byzantine errors), a probabilistic certification platform has been designed that includes hardware crypto-processors. This work has been performed within the ANR SAFESCALE project. Using a macro-data flow representation of the program execution, a complementary work, jointly developed with Paris team-project, is based on work-stealing scheduling to dynamically adapt the execution to sabotage while keeping a reasonable slowdown rate. Unlike static adaptation or adaptation at the source code level, a dynamic adaptation at the middleware level is proposed, enforcing separation of concepts and programming transparency. We are still extending algorithm-based fault-tolerance schemes for probabilistic certification in a more general context. We have developed a byzantine fault-tolerant interpolation algorithm suited to integral or polynomial modular computations. A first prototype has been developed and evaluated for the computation of the determinant of a matrix with integral coefficients distributed over GRID'5000 in order to simulate a global computing environment.

Finally, considering cryptographic primitives, we have proposed a new way to bound the probability of occurrence of an n-round differential in the context of differential cryptanalysis. Hence this new model allows us to claim proof of resistance against impossible differential cryptanalysis, as initially defined by Bi- ham in 1999. This work is applied to CS-Cipher, to which, assuming some non-trivial hypothesis, provable security against impossible differential cryptanalysis is obtained.

6.1.4. Cache-Oblivious Mesh Layout

One important bottleneck when visualizing large data sets is the data transfer between processor and memory. Cache-aware (CA) and cache-oblivious (CO) algorithms take into consideration the memory hierarchy to design cache efficient algorithms. CO approaches have the advantage to adapt to unknown and varying memory hierarchies. Recent CA and CO algorithms developed for 3D mesh layouts significantly improve performance of previous approaches, but they lack of theoretical performance guarantees. We developped a $O(N \log N)$ algorithm, called FastCOL, to compute a CO layout for unstructured but well shaped meshes. We proved that a coherent traversal of a N-size mesh in dimension d induces less than $N/B + O(N/M^{1/d})$ cache-misses where B and M are the block size and the cache size, respectively. Experiments show that our layout computation is faster and significantly less memory consuming than the best known CO algorithm (OpenCCL). The FastCOL performance is comparable to the OpenCCL algorithm for classical visualization algorithm access patterns, or better when the BSP tree produced while computing the layout is used as an acceleration data structure adjusted to the layout. We also show that cache oblivious approaches lead to significant performance increases on recent GPU architectures. This algorithm will be published in IEEE TVCG, 2010.

6.2. Software

6.2.1. Fault-tolerance in KAAPI

We have developed a new algorithm to have a high performance fault tolerant mechanism in KAAPI. The protocol is based on coordinated checkpointing. The algorithm is well suited for iterative parallel application. The originality of our protocol is to allows partial restart of processes after detection of a fault.

6.2.2. Scalability of KAAPI

KAAPI software has been tested on two grid platforms, the French National Grid50000 and the Japanese Intrigger, during the 5th PLUGTEST event organised by ETSI and project OASIS at Sophia-Antipolis, France, Octobre, 20th - October, 24th, 2008. The KAAPI team took part of the Super Quant Monte Carlo contest during the PLUGTEST event and was the winner in front of 7 teams. Runs have shown the ability to fully exploit machines geographically distributed among France and Japan which demonstrates concrete communication between processes behind firewalls.

6.2.3. GRID5000: scheduling algorithm for OAR and authentication

OAR is a batch scheduler developed by Mescal team. The MOAIS team develops the central automata and the scheduling module that includes successive evolutions and improvements of the policy.OAR is used to schedule jobs both on the CiGri (Grenoble region) and Grid50000 (France) grids. CiGri is a production grid that federates about 500 heterogeneous resources of various Grenoble laboratories to perform computations in physics. MOAIS has also developed the distributed authentication for access to Grid5000.

6.2.4. FlowVR

We introduced in FlowVR a hierarchical component model for the description of the application. This hierarchy of components strongly enforces the modularity of applications and eases debugging, maintenance and application development. The hierarchy is processed before to start the application by traversing the description and applying a sequence of controllers. The three main controllers take care of building the component content, mapping components on the target architecture hosts,and extracting the low level description required to execute the application.

2009 was also a year of consolidation for FlowVR. Part of the code was deeply reorganized. We fixed numerous bugs, improved the installation procedure and reduced the number of dependencies. The latest version, FlowVR 1.7 was released in December 2009.

7. Contracts and Grants with Industry

7.1. BDI funded by C-S, 07-10

C-S is funding a PhD thesis in joined collaboration with MOAIS and Institute Fourier (Roland Gillard). This PhD is focused on the dimensioning and the integration of a symmetric cipher in the context of a large scale distributed infrastructure. The first objective is to design efficient extensions and integration of the cipher CS (initially designed by C-S group) in order to exploit parallelism (based on parallel mode of operations). The second one concerns the design of scalable protocols to provide confidence and security in a large scale infrastructure.

7.2. BDI co-funded CNRS and CEA/DIF, 07-10

CEA/DIF is cofunding a PhD these in collaboration with MOAIS. This PhD is focused on cache and processor oblivious approaches applied to high performance visualization. The goal is to study rendering algorithms (mainly volume rendering and isosurface extraction) for large meshes (irregular and adaptive) that are proven efficient without requiring the mesh layout or the algorithm to actually know the memory hierarchy of the target architecture or the number of processor available. We will conduct experiments rendering large data sets provided by the CEA/DIF on NUMA machines. We will also study the benefits of such approaches for programming GPUs.

7.3. Contract with DCN, 05-09

The objective of the contract was to provide an efficient evaluation and planification of actions with real-time reactivity constraints and multicriteria performance guarantees. This contract is joined with POPART INRIA team (realtime aspects) and ProBayes company (probabilistic inference engine ProBT). MOAIS was in charge of the planification. The contract funded a phD thesis, co-advised by Moais and PopArt.

8. Other Grants and Activities

8.1. Regional initiatives

- *SCEPTRE*, 06-09, Minalogic: Started in 10/2006, SCEPTRE is a joint project with ST (coordinator), INRIA Rhône-Alpes (MOAIS, MESCAL, ARENAIRE, COMPSYS), IRISA (CAPS), TIMA-IMAG and VERIMAG. Within the SCEPTRE project, MOAIS has validated its technology of fine grain work-stealing to support adaptive multimedia applications on MPSoCs that include about 10 units on a single chip (general purpose units, DSP, ...).
- *CILOE*, 2008-2011, Minalogic: This project is to develop tools and high level interfaces for compute-intensive applications for nano and micro-electronic design and optimizations. The partners are: two large companies CS-SI (leader), Bull; three small size companies EDXACT, INFINISCALE, PROBAYES; and four research units INRIA, CEA-LETI, GIPSA-LAB, TIMA. For Moais, the contract funds the phD thesis of Jean-Noel Quiintin.
- *HiPeComp*, NANO 2008-2012 contract. The project HiPeCoMP (High Performance Components for MPSoC) consists in the development an coupling of: on the one hand, wait-free scheduling techniques (pre-partitioning and mapping, on-line work stealing) of component based multimedia applications on MPSoC architectures; and on the other hand, monitoring, debug and performance software tools for the programming of MPSoC with provable performances. For Moais, the contract funds the phD thesis of Christophe Laferrière who started on 1/9/2009.

- *SHIVA*, Minalogic 2009-2012 contract. This project aims at the development of a high throughput backbone ciphering that ensures a high level of security for intranet and extranet communications over internet. The partners are: CS-SI (leader); 1 small size companies: Easii-IC (support for Xilinx FPGA) IWall-Mataru (key management), Netheos (customizable FPGA for ciphering); INRIA; CEA-LETI (security certification); Grenoble-INP (TIMA lab, integration of cryptography on FPGA); UJF (LJK and Institut Fourier: open cryptographic protocols and handshake; VERIMAG: provable security). Within INRIA, the MOAIS and the PLANET teams provide the parallel implementation on a multicore pltaform of IP-Sec and coordination with hardware accelerators (Frog's and GPUs). The contract funds the phD thesis of Ludovic Jacquin, coadvised by PLANET and MOAIS.
- *GRIMDEV*, 2009-2010, ADT INRIA. This project brings engineering support for maintaining, operating and developing new experimentson the Grimage platform: Hervé Mathieu, INRIA SED, part time, Nicolas Turro, INRIA SED, part time and IJD Thomas Dupeux, INRIA, full time. The partners are the EPI MOAIS and PERCEPTION.

8.2. National initiatives

- *FVNANO*, 07-10, ANR-CIS: the project focuses on developing a framework for the interactive manipulation of nano objects. FlowVR is the core middleware used to build interactive applications coupling nano simulations, visualization and haptic force feedback. Partners : projects MOAIS (INRIA Rhône-Alpes), the CEA/DIF, the Laboratoire de Biochimie Théorique (LBT) and the LIFO (Université d'Orléans).
- *Vulcain*, 07-10, ANR Programme Génie Civil et Urbain: the project focuses on studying industrial structure reliability under dynamic constrinats (explosions, impacts). The role of the INRIA projects MOAIS and EVASION in this project is to provide a parallel framework based on SOFA for fast dynamic simulations. Partners: projects EVASION and MOAIS (INRIA Rhône-Alpes), 3S-R, IPSC-ELSA, CEG-DGA, LEES, LaM, INERIS, IRSN, CEA, SME Environnement, Phimeca, Bull.
- *DALIA*, 06-10, ARA Masse de Données: the project deals with multi-site interactive applications involving from handheld devices up to large multi-camera and multi-projector platforms. Partners : projects PERCEPTION, MOAIS (INRIA Rhône-Alpes), project I-parla (Bordeaux, INRIA Futurs) and the LIFO (Université d'Orléans).
- *CHOC*, 06-09, ANR Grid. The project deals with combinatorial problems and software to compute exact and approximate solutions over a grid. Partners: PRiSM (Versailles), LIFL (Lille), GILCO (Grenoble), MOAIS (Grenoble)
- *DISCO*, 06-09, ANR Grid. The project deals with evaluating middleware to do scientific computation over computational grid. Partners: CAIMAN (Sophia-Antipolis), OASIS (Sophia-Antipolis), SMASH (Rennes), PARIS(Rennes), LABRI (Bordeaux), EAD (Toulouse), MOAIS (Grenoble)
- *GRID'5000*, the french grid platform. MOAIS has participated to the development of the middleware stack used in Grid5000 (namely deployment with TakTuk, scheduling policies in OAR and distributed authentication based on LDAP).

8.3. International initiatives

8.3.1. Brazil

- We have a long term and strong collaboration with the Universities of Rio Grande do Sul, Brazil, and in particular with UFRGS, Porto Alegre. This collaboration is funded in 2009 by 4 different grants:
  - Capes/cofecub (2007-2009).
  - INRIA/Cnpq (2008-2010).
  - Equipe associée INRIA Diode-A (2006-2011).
  - Stic AmSud (2008-2009)

- USP-COFECUB project with the universities of Sao Paulo and Fortaleza, Brazil, focused on the impact of communications on parallel task scheduling. One year funding.

8.3.2. USA

LinBox project with the university of Delaware (Dave Saunders) , NCSU (Erich Kaltofen), LJK lab (Grenoble, Jean-Guillaume Dumas), ARENAIRE team project (LIP-ENSL, Lyon, Gilles Villard)) and LIRMM 5Montpelier Univ, Pascal Giorgi). The parallelization of the LinBox is in progress based on X-Kaapi library (C Pernet, T Gautier). LinBox is packaged in the SAGE software.

8.4. Hardware Platforms

8.4.1. The GRIMAGE platform

The GrImage platform (http://grimage.inrialpes.fr) gathers a network of cameras and a PC cluster. It is dedicated to interactive applications. GrImage is co-leaded by the Moais and Perception projects . It is the milestone of a strong and fruitful collaboration between Moais and Perception (common publications, software and application development).

GrImage (Grid and Image) aggregates commodity components for high performance video acquisition, computation and graphics rendering. Computing power is provided by a PC cluster, with some PCs dedicated to video acquisition and others to graphics rendering. A set of digital cameras enables real time video acquisition. The main goal is to rebuild in real time a 3D model of a scene shot from different points of view. Visualization can be performed using a head mounted display for first-person interactions or on a multi-projector display-wall for high resolution rendering.

Since July 2009, the computing cluster was upgraded through grants from INRIA and CNRS-LIG. Grimage uses some specific nodes from the Digitalis machine capable of hosting several daughter boards (mainly video acquisition and graphics cards). It relies on Intel Nehalem processors and a high speed Infiniband network. This integrated approach will enable to test interactive applications using a very high number of processing resources as other nodes from the Digitalis machine can be reserved if needed.

8.4.2. The Digitalis machine

Digitalis is a 780 cores cluster based on Intel Nehalem processors and Infiniband network located at INRIA Rhône-Alpes. Digitalis has been designed to suit both the needs for batch computations and interactive applications. As mentioned before, one rack is dedicated to nodes hosting video acquisition boards and graphics cards. These nodes are mainly used for the Grimage platform, but can also be used for batch computing. Additional nodes with Nvidia Tesla GPUs will be installed during the first quarter of 2010.

By having a single unified machine for batch and interactive computing we expect to better use the available resources, favor the emergence of high performance applications integrating interactive steering and vice versa enable the development of a new generation of interactive 3D applications using a significantly larger number of CPUs and GPUs that what has been done so far on the Grimage platform.

8.4.3. SMP Machines

MOAIS invested in 2006 on two SMP architectures:

- A 8-way SMP machine equipped with Itanium processors.
- A 8-way SMP machine equipped with dual core processors (total of 16 cores) and 2 GPUs.

These machines enables us to keep-up with the evolution of parallel architectures and in particular today's availability of large multi-core machines. They are used to develop and test parallel adaptive algorithms taking advantage of the processing power provided by the multiple CPUs and GPUs available. We expect to have a new large SMP architecture by the end of 2010 or the beginning of 2011 to replace these machines.

8.4.4. MPSoC

ST Microelectronics provided us a STM8010 machine for experimenting parallel adaptive algorithms on MPSoC.

9. Dissemination

9.1. Leadership within scientific community

- Program committees :
  - HCW'2009 (18th IEEE Heterogeneous Computing Workshop), 25 may 2009, Roma, Italy
  - chair of ASTEC, 2-5 june 2009, Les Plantiers, France
  - Workshop on large scale system and application performances, 9-10 june, Munchen, Germany
  - ISPDC 2009, 30 june - 4 july, Lisboa, Portugal
  - 4th MISTA 2009, 10-12 august 2009, Dublin
  - SPAA'2009, 11-13 august 2009, Calgary, Canada
  - 7th HeteroPar 2009, 22-25 august 2009, Delft, The Netherlands
  - ParCo'2009, september 2009, Lyon, France
  - PPAM'2009, 13-16 september 2009, Wroclaw, Poland
  - co-chair of Parallel Bio-Computing workshop, 13 september 2009, Wroclaw, Poland
  - RenPar'2009, 9-11 september, Toulouse, France
  - 21st SBAC-PAD, 28-31 october 2009, Sao Paulo, Brazil
  - 24th IEEE symposium on Computer Science and Information Systems, 14-16 dec., Cyprus
  - HiPC'2009, 17-20 december 2009, Cochin, India
  - Demo Co-chair and program committee member of JVRC 2009 (Joint Virtual Reality Conference of EGVE - ICAT - EuroVR) , Lyon, France.
  - Tutorial co-chair and program committee member of IEEE VR 2009, Lafayette,USA.
  - Commitee member of ISVC 2009 (International Symposium on Visual Computing).
  - Committee member of SVR 2009 (Symposium on Virtual and Augmented Reality), Brazil.
  - Committee member of PAPP 2009 (International Workshop on aPpplications of declArative and object-oriented Parallel Programming), Baton Rouge, USA.
  - Committy member of CLCAR 2009 (Conferencia Latinamericana de Computatición de Alto Rendimiento), Venezuela.
- Members of editorial board : Calculateurs Parallèles, collection *Studies in Computer and Communications Systems*-IOS Press;*Handbook on Parallel and Distributed Processing, Springer Verlag*; *Parallel Computing Journal, series Advances in parallel processing,Elsevier Press*; ARIMA Journal; Parallel Computing Journal. IEEE Transactions on Parallel and Distributed Systems (TPDS).
- Member of the steering board of the EGPGV workshop (Eurographics Symposium on Parallel Graphics and Visualization).

10. Bibliography

Major publications by the team in recent years

[1] P.-F. DUTOT, L. EYRAUD, G. MOUNIÉ, D. TRYSTRAM. *Scheduling on large scale distributed platforms: from models to implementations*, in "Internat. Journal of Foundations of Computer Science", vol. 16, n^{o} 2, World Scientific, april 2005, p. 217-237.

[2] S. JAFAR, A. W. KRINGS, T. GAUTIER. *Flexible Rollback Recovery in Dynamic Heterogeneous Grid Computing*, in "IEEE Transactions on Dependable and Secure Computing", 2008.

[3] J.-D. LESAGE, B. RAFFIN. *A Hierarchical Component Model for Large Parallel Interactive Applications*, in "Journal of Supercomputing", Extended version of NPC 2007 article., July 2008, http://dx.doi.org/10.1007/s11227-008-0228-7.

[4] G. MOUNIÉ, C. RAPINE, D. TRYSTRAM. *A 3/2-Dual Approximation Algorithm for Scheduling Independent Monotonic Malleable Tasks*, in "SIAM Journal on Computing", vol. 37, n° 2, 2007, p. 401–412, http://hal.archives-ouvertes.fr/hal-00002166/en/.

[5] D. TRAORE, J.-L. ROCH, N. MAILLARD, T. GAUTIER, J. BERNARD. *Deque-free work-optimal parallel STL algorithms*, in "EUROPAR 2008, Las Palmas, Spain", Springer-Verlag, Aug 2008, http://www-id.imag.fr/Laboratoire/Membres/Roch_Jean-Louis/perso_html/papers/2008-europar-adaptSTL.pdf.

Year Publications

Doctoral Dissertations and Habilitation Theses

[6] J.-D. LESAGE. *Couplage dans les applications interactives de grande taille*, Ph. D. Thesis, Grenoble INP, December 2009.

[7] B. RAFFIN. *Calcul interactif haute performance*, Habilitation à diriger des recherches (HDR), Grenoble INP, March 2009.

[8] L. M. SCHNORR. *Some Visualization Models applied to the Analysis of Parallel Applications*, Ph. D. Thesis, Institut National Polytechnique de Grenoble, October 2009.

Articles in International Peer-Reviewed Journal

[9] F. DIEDRICH, K. JANSEN, F. PASCUAL, D. TRYSTRAM. *Approximation algorithms for scheduling with reservations*, in "Algorithmica", Springer, 2009.

[10] P.-F. DUTOT, T. N'TAKPÉ, F. SUTER, H. CASANOVA. *Scheduling Parallel Task Graphs on (Almost) Homogeneous Multicluster Platforms*, in "IEEE Transactions on Parallel and Distributed Systems", vol. 20, n° 7, IEEE Computer Society, Los Alamitos, CA, USA, 2009, p. 940-952.

[11] A. GIRAULT, É. SAULE, D. TRYSTRAM. *Reliability versus performance for critical applications*, in "Journal of Parallel and Distributed Computing", vol. 69, n° 3, Elsevier, 2009, p. 326-336.

[12] W. NASRI, L. STEFFENEL, D. TRYSTRAM. *Adaptive approaches for efficient parallel algorithms on cluster-based systems*, in "International Journal of Grid and Utility Computing", vol. 1, n° 2, InderScience Pub., 2009, p. 98-108.

[13] F. PASCUAL, K. RZADCA, D. TRYSTRAM. *Cooperation in Multi-Organization Scheduling*, in "Concurency and Computation: Practice and Experience", n° 21, Wiley and sons, 2009, p. 905-921.

[14] K. RZADCA, D. TRYSTRAM. *Promoting Cooperation in Selfish Computational Grids*, in "European Journal of Operational Research", n° 199, Elsevier, 2009, p. 647-657.

[15] É. SAULE, D. TRYSTRAM. *Analyzing scheduling with transient failures*, in "Information Processing Letters", vol. 109, n^{o} 11, Elsevier, 2009, p. 539-542.

[16] A. TCHERNYKH, D. TRYSTRAM, C. BRIZUELA, I. SCHERSON. *Idle regulation in non-clairvoyant scheduling of parallel jobs*, in "Discrete Applied Mathematics", n^{o} 157, Elsevier, 2009, p. 364-376.

International Peer-Reviewed Conference/Proceedings

[17] I. ASSENMACHER, B. RAFFIN. *Short Paper: A Modular Framework for Distributed VR Interaction Processing*, in "Proceedings of the Joint Virtual Reality Conference of EGVE - ICAT - EuroVR (JVRC'09)", Eurographics, December 2009.

[18] M. BOUGERET, P.-F. DUTOT, A. GOLDMAN, Y. NGOKO, D. TRYSTRAM. *Combining multiple heuristics on dicrete resources*, in "APDCM, 11th workshop on advances in parallel and distributed computational models, Roma, Italy", IEEE, 2009.

[19] M. BOUGERET, P.-F. DUTOT, D. TRYSTRAM. *The guess approximation technique and its application to the discrete resource sharing scheduling problem*, in "MAPSP, (the 9th workshop on Models and Algorithms for Planning and Scheduling Problems), The Netherlands", 2009.

[20] M. S. BOUGUERRA, T. GAUTIER, D. TRYSTRAM, J.-M. VINCENT. *A flexible checkpoint-restart model in distributed systems*, in "Proceedings of PPAM 2009, the 8th International Conference on Parallel Processing and Applied Mathematics, Wroclaw, Poland", spetember 2009.

[21] B. BOYER, J.-G. DUMAS, C. PERNET, W. ZHOU. *Memory efficient scheduling of Strassen-Winograd's matrix multiplication algorithm*, in "ISSAC '09: Proceedings of the 2009 international symposium on symbolic and algebraic computation, Seoul, Korea", ACM, 2009.

[22] B. CLAUDEL, G. HUARD, O. RICHARD. *TakTuk, Adaptive Deployment of Remote Executions*, in "Proceedings of the International Symposium on High Performance Distributed Computing (HPDC)", June 2009.

[23] J.-G. DUMAS, C. PERNET, D. B. SAUNDERS. *On finding multiplicities of characteristic polynomial factors of black-box matrices*, in "ISSAC '09: Proceedings of the 2009 international symposium on symbolic and algebraic computation, Seoul, Korea", ACM, 2009.

[24] A. ESSAFI, A. MAHJOUB, G. MOUNIÉ, D. TRYSTRAM. *Implementation issues of scheduling with reservations on desktop grids*, in "Proceedings of Parco 2009, Lyon, France", Sept 2009.

[25] E. HERMANN, B. RAFFIN, F. FAURE. *Interactive Physical Simulation on Multicore Architectures*, in "Eurographics 2009 Symposium on Parallel Graphics and Visualization (EGPGV'09), Munich, Germany", March 2009, p. 1–8.

[26] Y. NGOKO, D. TRYSTRAM. *Combining SAT solvers on dicrete resources*, in "HPCS, International Conference on high performance computing and simulation, Leipzig, Germany", 2009.

[27] J. PECERO, D. TRYSTRAM, A. ZOMAYA. *A new genetic algorithm for scheduling for large communication delays*, in "EuroPar 2009, Delft, The Netherlands", LNCS, Springer, 2009.

[28] B. PETIT, J.-D. LESAGE, E. BOYER, J.-S. FRANCO, B. RAFFIN. *Remote and Collaborative 3D Interactions*, in "Proceedings of the 3DTV Conference (3DTV-CON 2009)", May 2009.

[29] B. PETIT, J.-D. LESAGE, E. BOYER, B. RAFFIN. *Virtualization Gate*, in "SIGGRAPH'09: ACM SIGGRAPH 2009 Emerging Technologies, New York, NY, USA", ACM, August 2009, p. 1–1.

[30] T. ROCHE, J.-L. ROCH, M. CUNCHE. *Algorithm-based fault tolerance applied to P2P computing networks.*, in "The First International Conference on Advances in P2P Systems, Sliema, Malta", IEEE, October 2009, p. 144–129, http://www.iaria.org/conferences2009/AP2PS09.html.

[31] É. SAULE, D. TRYSTRAM. *Multi-Users Scheduling in Parallel Systems*, in "Proceedings of IPDPS 2009, the 23rd International Parallel and Distributed Processing Symposium, Roma, Italy", IEEE, may 2009.

[32] L. M. SCHNORR, G. HUARD, P. O. A. NAVAUX. *Towards Visualization Scalability through Time Intervals and Hierarchical Organization of Monitoring Data*, in "Cluster Computing and Grid 2009 (CCGrid'09) Proceedings", May 2009.

[33] L. M. SCHNORR, G. HUARD, P. O. A. NAVAUX. *Visual Mapping of Program Components to Resources Representation: a 3D Analysis of Grid Parallel Applications*, in "Proceedings of the 21st International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)", October 2009.

National Peer-Reviewed Conference/Proceedings

[34] D. CORDEIRO, G. MOUNIÉ, S. PÉRARNAU, D. TRYSTRAM, J.-M. VINCENT, F. WAGNER. *Comment rater la validation de votre algorithme d'ordonnancement*, in "Proceedings of Renpar 19, Toulouse, France", Poster, Sept 2009, p. 1-2.

Workshops without Proceedings

[35] G. HUARD. *Ressources management and runtime environments in the exascale computing era*, in "French-Japanese workshop on Petascale Applications, Algorithms and Programming (PAAP)", 2009.

Scientific Books (or Scientific Book chapters)

[36] X. BESSERON, M. S. BOUGUERRA, T. GAUTIER, É. SAULE, D. TRYSTRAM. *Fault tolerance and availability awarness in computational grids*, in "Fundamentals of Grid Computing", F. MAGOULES (editor), Numerical Analysis and Scientific Computing, ISBN: 978-1439803677, chap. 5, Chapman and Hall/CRC Press, 2009.

[37] F. DIEDRICH, K. JANSEN, U. SCHWARTZ, D. TRYSTRAM. *A survey on approximation algorithms for scheduling with machine unavailability*, in "Algorithmics of Large and Complex Networks", J. LERNER, D. WAGNER, K. ZWEIG (editors), LNCS, n^{o} 5515, Springer, 2009, 50,64.

[38] P.-F. DUTOT, K. RZADCA, É. SAULE, D. TRYSTRAM. *Multi-objective scheduling*, in "Introduction to scheduling", Y. ROBERT, F. VIVIEN (editors), ISBN: 978-1420072730, chap. 9, Chapman and Hall/CRC Press, 2009.