



MOAIS

Multi-Programming and Scheduling Design for Applications of Interactive Simulation

Jean-Louis Roch & al.



Moais Transféré du Musée de l'Homme au Musée du Quai Branly.
Origin: Rapa Nui [Easter Island], Cook Bay, West coast
Date : between the XIst and the XVth century
Height: 1.85 m.

« Tête sculptée en tuf volcanique. Recueillie en 1872 (...) sur le navire "La Florel...". Ce buste a été détaché à l'époque du reste du corps devant l'impossibilité de transporter la pièce entière jusqu'au navire. »

<http://moais.imag.fr>

Visite AERES – 8-10 Février 2010



Staff

2

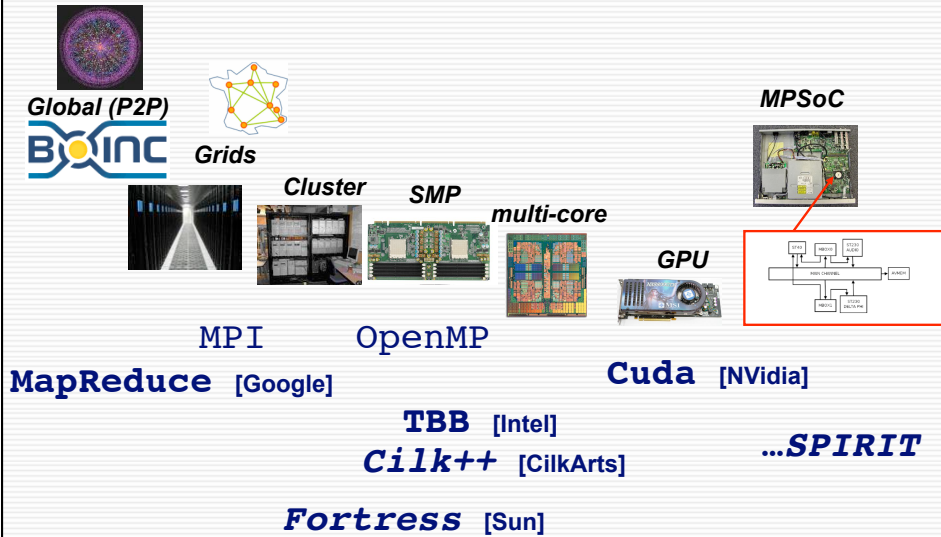
- 1/1/2005: Creation of MOAIS team -- 1/1/2006: INRIA team-project / LIG
27/3/2008: INRIA Evaluation

- Vincent Danjean [MdC, 9/05]
 - Pierre-François Dutot [MdC, 9/06]
 - Thierry Gautier [CR INRIA]
 - Guillaume Huard [MdC]
 - Grégory Mounié [MdC]
 - Clément Pernet [MdC, 12/08]
 - Bruno Raffin [CR INRIA, HDR]
 - Jean-Louis Roch [MdC, Team leader]
 - Denis Trystram [Prof]
 - Frédéric Wagner [MdC, 9/06]
- 2 Postdocs: Ingo Assenmacher, Veronika Sonigo
 - 12 PhD students, 2 engineers,
 - 17 PhDs defended (1/10/05->30/9/09) + 3 (L Schnorr, JD Lesage, T Roche)
 - Christian Seguy [computing facilities]
 - Ahlem Zammit-Boubaker, Annie-Claude Vial-Dallais [administration]

Parallelism everywhere

3

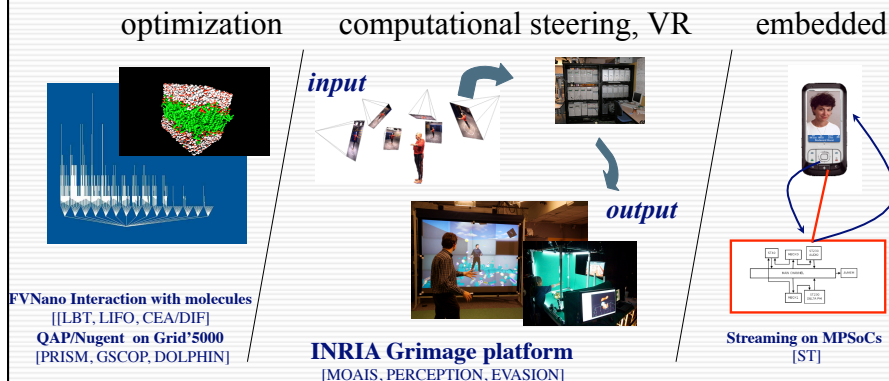
- Distributed, Heterogeneous



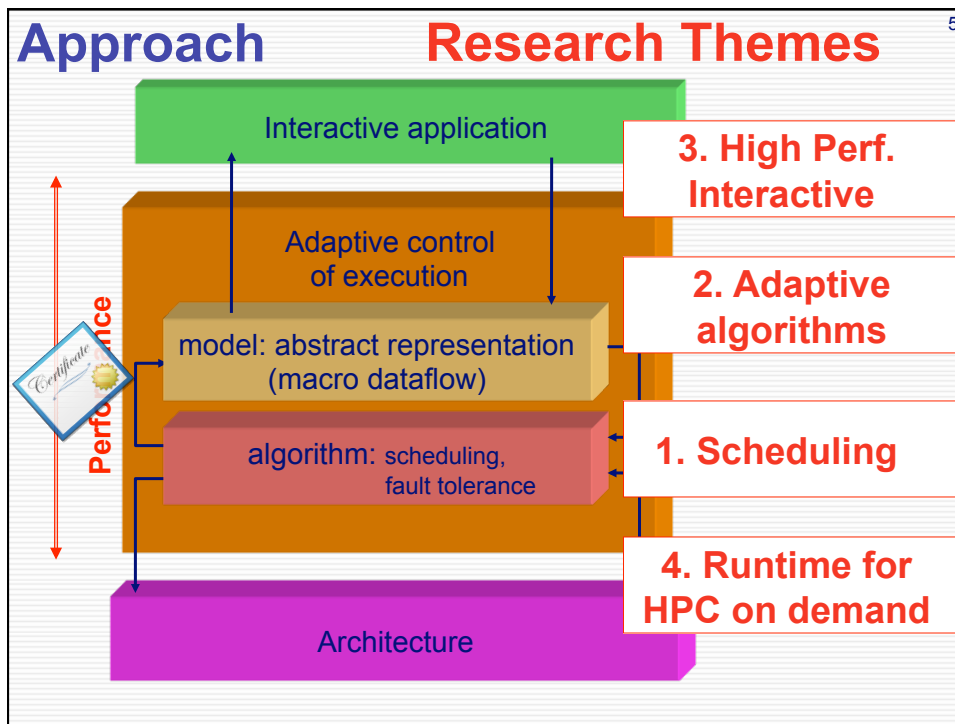
MOAIS objective

4

- To provide end-to-end parallel programming solutions for high-performance interactive computing with provable performances.



Performance is multi-objective



- ⁶
- ## Achievements for 2005-2009 and research directions
1. **Scheduling**
 2. Adaptive algorithms
 3. High-Performance Interactive Computation
 4. Runtime for HPC on demand

1. Scheduling

7

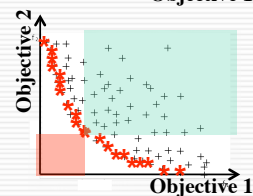
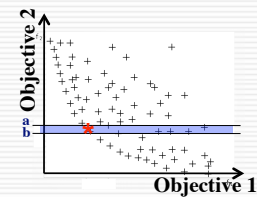
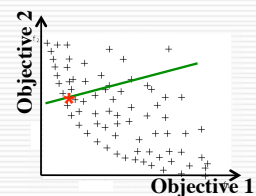
- **Formalization of the related problems:**
 - Modeling of an adaptive application
 - Formalization and optimization w.r.t. multi-objective
 - Design of scalable scheduling algorithms
- **Approach**
 - Classical combinatorial optimization
 - complexity and bounds, approximation
 - Non standard methods
 - game theory, distributed analysis

Multi-objective approximation

8

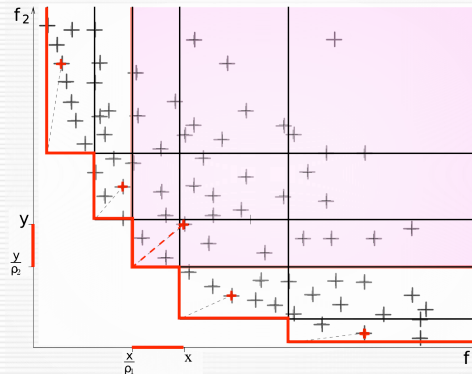
$\begin{bmatrix} \text{Min Objective1} \\ \text{Min Objective2} \end{bmatrix}$

- **Reduction to a single objective:**
 - **To aggregate all objectives into one.**
 - Linear or convex combination to optimize.
 - **To constrain all objectives except one.**
 - E.g. minimize Makespan using Memory < S.
- **Computation of the Pareto set.**
 - **Pareto** : any solution non dominated by another one.



Approximation of the Pareto set ⁹

- Pareto set can be large and not convex,
- But can be approximated with a polynomial number of near-optimal solutions [Papadimitriou 2000]



- Applications:
 - Makespan/Memory [IPDPS08]
 - Makespan/Reliability [JPDC09]

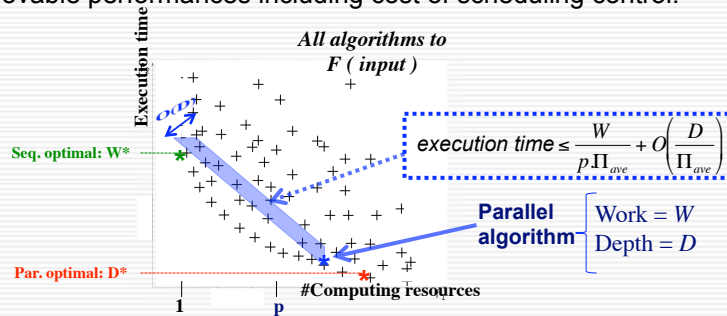
Research directions ¹⁰

- **Generic method for non-correlated objectives?**
 - Correlated: MinSum/Makespan : $\text{Min } \sum C_i / \text{Max } C_i$
 - Non correlated: Reliability/Makespan : $e^{-\lambda t} / \text{Max } C_i$
- **Dealing with many objectives [diversity]**
 - For k objectives (eg Makespan for each user):
constant approximation to the Pareto set [IPDPS09]
- **Towards a decentralized control,
but keeping in mind a global objective**
 - Instead of global performance, local point of view (Game theory)
 - Consider classical method, but to provide distributed algorithms

List scheduling vs Work-stealing

11

- **List scheduling:** “ Greedy”: no idle processors while ready tasks.
 - Many variants to cope with multi-objectives (eg Makespan + Comm. Amount)
- **Work-stealing:** [... Cilk90, Athapascan/Kaapi, ... CilkArts08,... X10, TBB]
 - Restricted distributed implementation
 - But with provable performances including cost of scheduling control.



- **Need of a new analysis** to cope with multi-objective and various implementation features [mixed push/pull...]

12

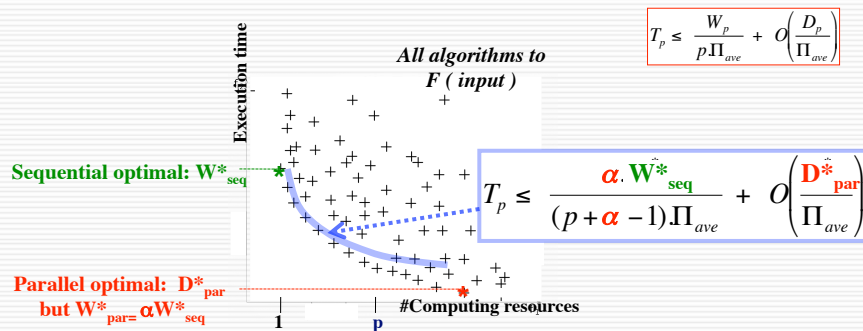
Achievements for 2005-2009 and research directions

1. Scheduling
2. **Adaptive algorithms**
3. High-Performance Interactive Computation
4. Runtime for HPC on demand

2. Adaptive algorithms

Objective: To design and analyze algorithms that may **obviously** adapt their execution under the control of the scheduling

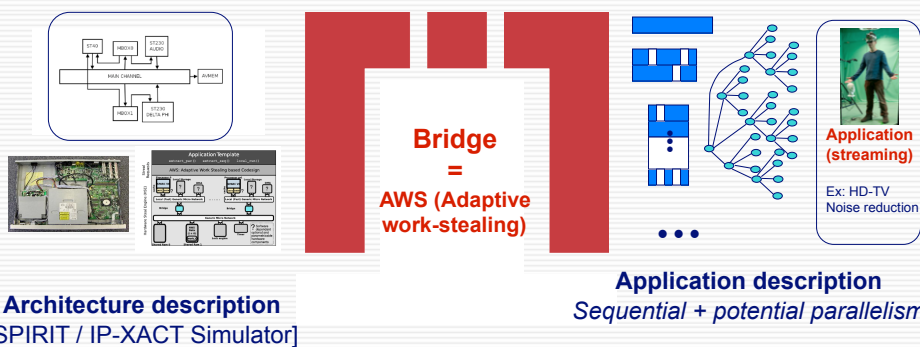
- **Heterogeneous resources, variable speeds: work-stealing based**



- **Multi-objective problem:** work W_p increases when depth D_p decreases
 - Adaptive recursive coupling of two algorithms Seq and Par [PASCO'07, Europar'08]
 - Provable performances, both theory and practice [demo Adaptive/Krash]

Adaptive streaming

- Use case: HDTV on MPSoCs [ST Microelectronics]
Minalogic SCEPTRE contract (ST, TIMA, MESCAL, ARENAIRE, COMPSYS, VERIMAG)

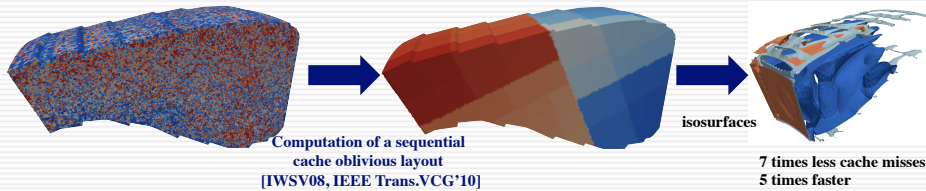


- Cache & processor oblivious stream computations [PDP08]
- Extensions to the programming of large MPSoC [Nano 2012 / ST, HiPeCoMP contract]

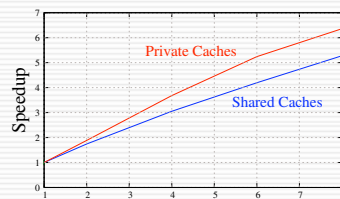
Mesh layouts for adaptive algos

15

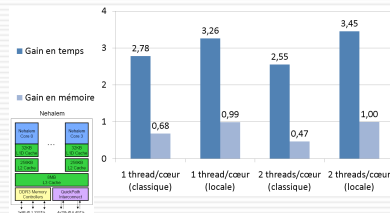
- Intensive computations on mesh (CEA-DIF contract)



Private caches (eg Opteron-875)
Parallel then sequential



Shared cache (eg Nehalem)
performances limited by cache size



Towards generic adaptation techniques cache/prcoessor ?

Research directions

16

- Heterogeneous and hierarchical architectures
 - Both processor and memory
- Impact of variable loads
- Adaptive resource allocation
 - Energy, time constraints
- Programming models for adaptive algorithms

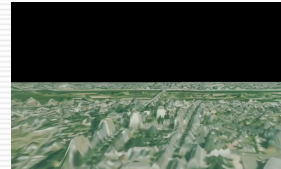
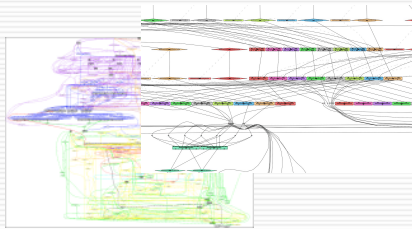
3. HPiC : Interactivity

19



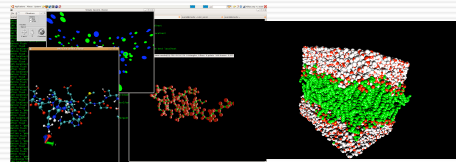
Middleware dedicated to interactive applications [Supercomputing08]

- Hierarchical components
- Data flow



Telepresence [ANR Dalia][VRST08]
3D reconstruction (Grenoble and Bordeaux)
Geographic information service (Orléans)

- Coarse grain mapping
- Online frequency calibration
 - multi-objective
 - Tuned sampling -> adaptive

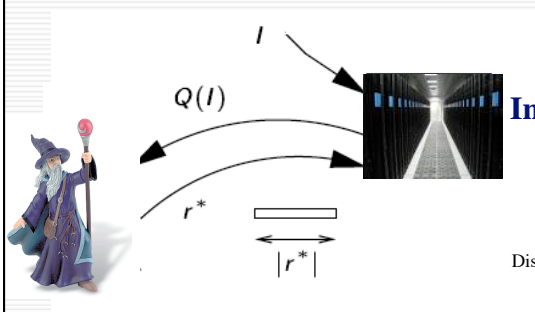


Molecular dynamic
[ANR FVNano]

3. HPiC: Towards interactive computing

20

- Using interaction with **expert** to improve performances
 - *“Human in the loop” => “Expert in the loop”*
- Theoretical approach: information (small) from an **Oracle** to improve complexity, performance ratio
 - Interactive complexity classes



Information (guesses) to improve approximation schemes

algorithm	approx. ratio	$ r^* $	complexity
MA^G	$(k - g)$	$g \log(m)$	$O(m^g * kn)$
MA^G_{β}	$(k - g)(1 - \beta)$	$\log(k) + g \log(m)$	$O(k * m^g * kn)$
MA^{G^2}	$\frac{k}{g+1}$	$g(\log(k) + \log(m))$	$O((km)^g * kn)$

Discrete Resource Sharing Scheduling Problem [IPDPS09]

4. Runtime for HPC “on demand”

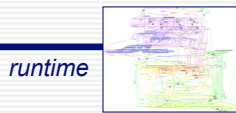
Objective: scheduling control at runtime based on efficient distributed macro dataflow representation.

➤ Kaapi middleware

```

1 struct sum {
2   void operator()(int& a, int& b, int& r) const {
3     r = a + b;
4   }
5 };
6
7
8 struct fib {
9   void operator()(int n, int& a, int& b, int& r) const {
10    if (n <= 2) r = a;
11    else
12      fib(n-1, a, b, r);
13    r = a + b;
14  }
15 };
16
17 struct sum_fib {
18   void operator()(int& a, int& b, int& r) const {
19     r = a + b;
20   }
21 };

```



Distributed nested macrodataflow graph



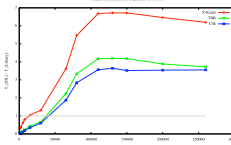
Distributed stack

▪ Key points for performance:

- Local serialization (“*work-first principle*”) [J. CLSS’07, ICCS07]
- **Scheduling** : Pre-partitioning + “work-stealing” (Lock-free / wait-free)
- Fault-tolerance protocols, from scheduling properties
 - *coordinated protocol* [ICTTA’06, TSI07, MCO08] + *TIC protocol* [EIT05, TDSC09]

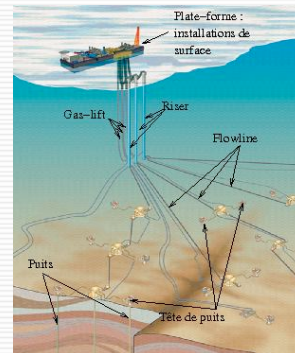
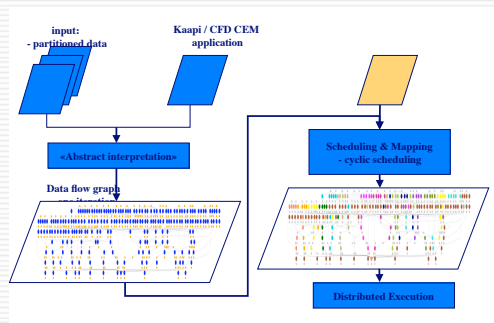
▪ Positioning:

- Work stealing: Intel TBB, Cilk++, X10
- Grid / global platforms: tolerate failures and falsification: Satin (FT)



Kaapi applications

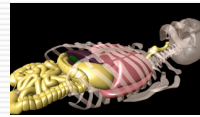
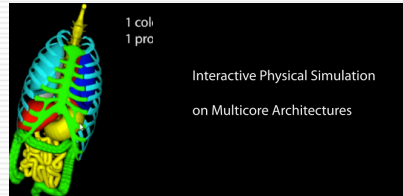
- *Process engineering - CAPE-OPEN standard [IFP]*
- Combinatorial Optimization (QAP, Q3AP: PRISM/Gilco) [ANR CHOC]
- Numerical Kernel [ANR DISCOGRID]
- Chemical combustion [NUMED]
- Computer algebra [...LinBox...]



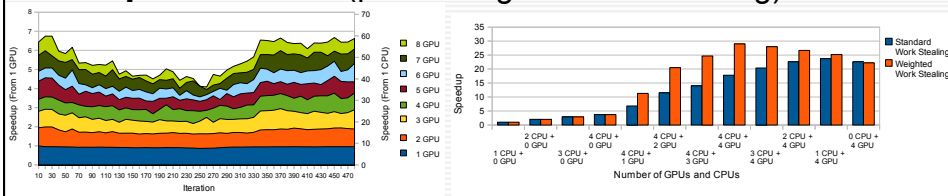
Scheduling: partitioning + work-stealing

Multi CPUs/GPUs Kaapi

23



- **Parallelization implicit to SOFA user** [Demo]
- **Mixed scheduling in Kaapi**
 - Initial distribution: graph partitioning CPU&GPU
 - Online work stealing CPUs & GPUs
- **+ Improvements** (partitioning and work stealing)

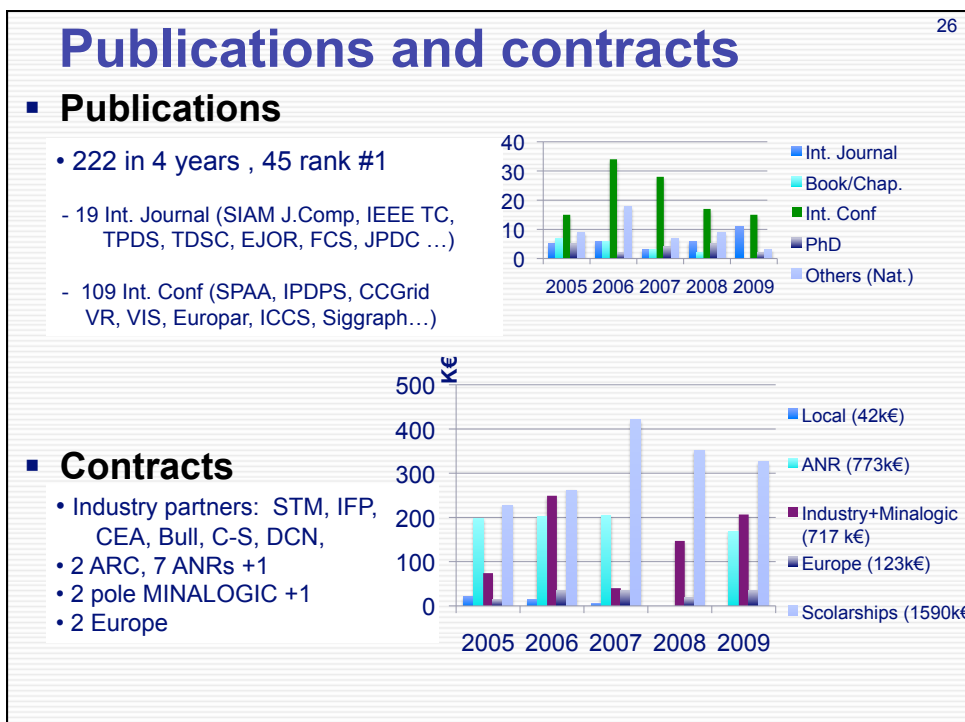
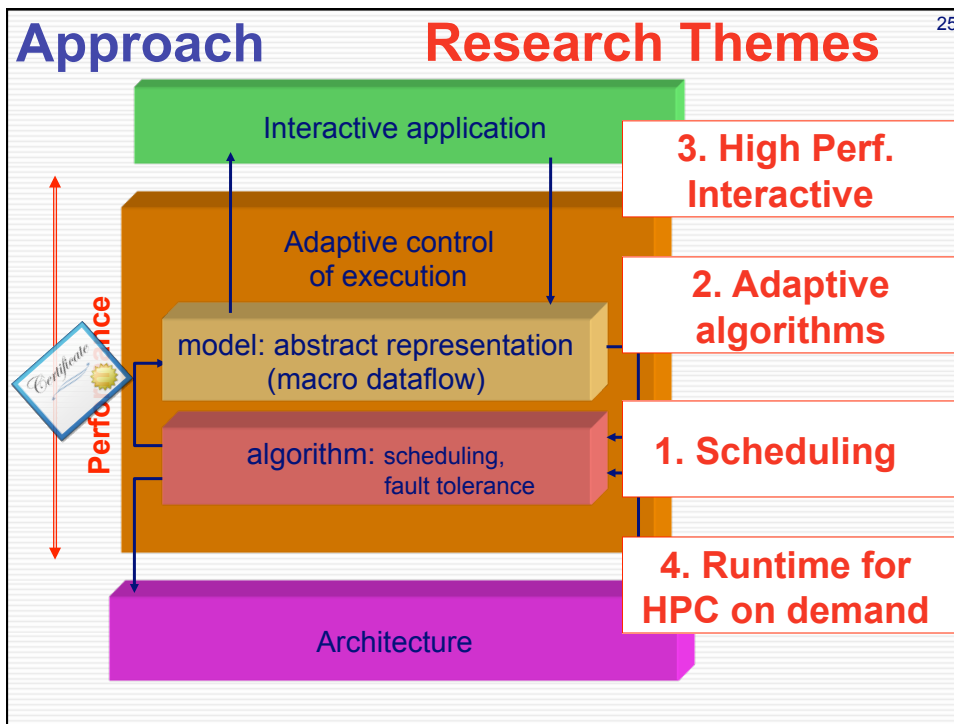


HPC on demand: research directions

24

- **Secure computing on exascale platforms**
- **Efficient parallel execution with tiny size grain**
 - Speedup on thousands of cores at fixed problem size
 - Adaptive algorithms [Prototype Xkaapi]
- **Hierarchical memory**
 - work-stealing provable performances?
- **“Oblivious tuning” of parallel applications**
 - Monitoring and retro-control
 - Deployment and shrinking





Highlights

- 1st prize Plugtest Nov. 2007 and Nov. 2008

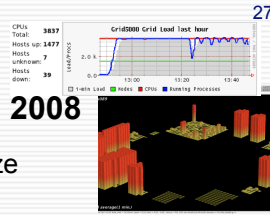


Dec. 2006: special Jury prize

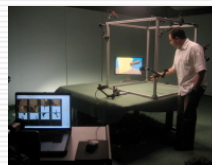
Nov. 2007: 1st prize

- Nqueens(23) in 2107s with 3654 cores

Nov. 2008: Pricing: 1st prize Grid5K + Intrigger (Japan)



- SIGGRAPH Aug. 2007/2009 Emerging Technologies



~4000 visitors

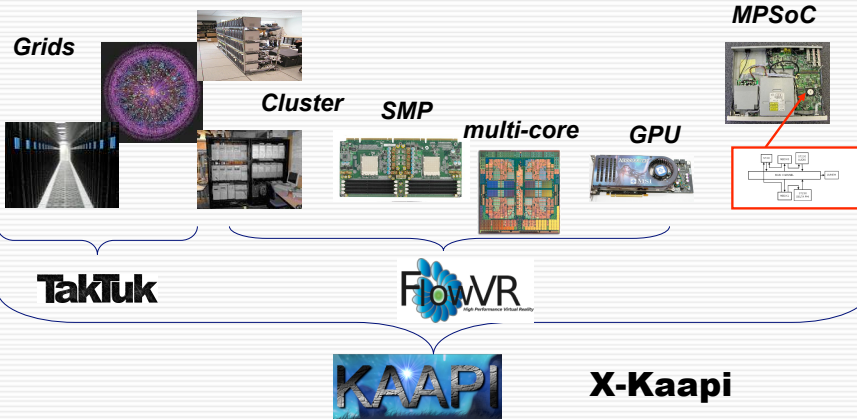
- Valorization: start-up



(Sep. 2007)

- co-founded by former PhD C. Menier [joined MOAIS / PERCEPTION]
- transfer: parallel 3D modeling

Softwares



+ Dmake, Krash, Triva...

Multi-objective

Adaptive

Performance

Perspectives summary 2010-2014

- **Manycore / exascale programming:
dealing with uncertainties**
 - Heterogeneous computing
 - GPUs, MPSoCs, manycore
 - Large scale autonomous computing
 - faults : Fail-stop and malicious
 - Oblivious algorithms with provable performances
 - Game theory

Summary

“To provide parallel programming schemes, interfaces and tools for high performance interactive computing that enable to achieve provable performances on distributed parallel architectures, from multi-processors system-on-chip to lightweight grids and global computing platforms.”

Submission Id: 0063

VIRTUALIZATION GATE

INRIA / Grenoble Universities
4D View Solutions

SIGGRAPH'09 [MOAIS - PERCEPTION - EVASION]

Moais Exhibitions

Grimage / FlowVR

SOFA / Kaapi

Adaptive Xkaapi / Krash



The virtual reality environment developed within the framework of the Moais exhibition is a virtual environment for the visualization of the results of the simulation of the flow of a fluid around a body. The user can interact with the environment in real time and see the results of the simulation as they change. The user can also interact with the environment in real time and see the results of the simulation as they change.

The user can interact with the environment in real time and see the results of the simulation as they change. The user can also interact with the environment in real time and see the results of the simulation as they change.

The user can interact with the environment in real time and see the results of the simulation as they change. The user can also interact with the environment in real time and see the results of the simulation as they change.

The user can interact with the environment in real time and see the results of the simulation as they change. The user can also interact with the environment in real time and see the results of the simulation as they change.



Using KAAPI to parallelize the SOFA framework

KAAPI (Kernel Adaptive Architecture for Parallel Processing) is a library that allows to parallelize the SOFA framework. It is designed to be used in a multi-processor environment and to be able to handle the complexity of the SOFA framework.

KAAPI (Kernel Adaptive Architecture for Parallel Processing) is a library that allows to parallelize the SOFA framework. It is designed to be used in a multi-processor environment and to be able to handle the complexity of the SOFA framework.

KAAPI (Kernel Adaptive Architecture for Parallel Processing) is a library that allows to parallelize the SOFA framework. It is designed to be used in a multi-processor environment and to be able to handle the complexity of the SOFA framework.

KAAPI (Kernel Adaptive Architecture for Parallel Processing) is a library that allows to parallelize the SOFA framework. It is designed to be used in a multi-processor environment and to be able to handle the complexity of the SOFA framework.

KAAPI (Kernel Adaptive Architecture for Parallel Processing) is a library that allows to parallelize the SOFA framework. It is designed to be used in a multi-processor environment and to be able to handle the complexity of the SOFA framework.

KAAPI (Kernel Adaptive Architecture for Parallel Processing) is a library that allows to parallelize the SOFA framework. It is designed to be used in a multi-processor environment and to be able to handle the complexity of the SOFA framework.



Executing Parallel Applications on Dynamic Systems

With the fast increase in hardware resources and the need for parallel computing, a large number of desktop applications would benefit from the power increase due to the parallelization of very low level hardware. However, such systems present some strong constraints due to the multiplicity of software running concurrently at a given time. As such, the load of these machines is heavily subjected to large variations.

The MOAIS team has been developing tools and scheduling techniques such as adaptive algorithms to achieve performance of parallel applications on dynamic resources. Adaptive algorithms reduce the work performed due to the parallelization by taking better decisions on the decomposition of the work into tasks. The introduction of our work in real production systems poses to be developing an efficient solution of the scheduling problem responsible and real-world applications.

In this document, we present a complete system process to be developing an efficient solution of the scheduling problem responsible and real-world applications. The process is a new scheduler for parallel computation: the REALITY framework, using a work-stealing scheduling engine. Adaptive algorithms are in use case developed on top of XKAAPI.

Different platforms from benchmarks to fully dynamic are proposed for test during the experiments. These experiments are needed by XKAAPI: General Inspection and Analysis of System Management (GIASM) as a tool developed by the MOAIS team to monitor heterogeneous environments on dedicated hardware systems. Such monitor is achieved by connecting with the Operating System and generating a real-time load on the machines.

To ensure a high quality of the resulting environment and reproduce experiments on top of it, XKAAPI generates a very low latency in the system.

Further information:

- EPFL: 2008 System Process and Offload Based: XKAAPI: Reproducible CPU Load Generation on User-Cost Machines, April 2008.
- In France: 2008 Embedded Thesis Team: Xkaapi: Real-time, Scalable, High Quality and Low Latency Load Generation: Design and Architectural Aspects, ITC, September, 2008, pp. 20-29.



MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.

MOAIS generating heterogeneity on a core system.



- ### Topics
- Design and analysis of parallel algorithms for computer algebra
 - Practical parallel implementation of symbolic or symbolic-numeric algorithms
 - High-performance software tools and libraries for computer algebra
 - Applications of high-performance computer algebra
 - Distributed data-structures for computer algebra
 - Hardware acceleration technologies (multi-cores, GPUs, FPGAs) applied to computer algebra
 - Cache complexity and cache-oblivious algorithms for computer algebra
 - Compile-time and run-time techniques for automating optimization and platform adaptation of computer algebra algorithms

- ### Important Dates
- Paper submission deadline: April 2, 2010
 Notification of acceptance: May 10, 2010
 Camera-ready version: May 28, 2010
 Tutorials and Workshop: July 21-23, 2010

- ### Program Committee
- Daniel Augot
 - Jean-Claude Bajard
 - Olivier Beaumont
 - Bruce Char
 - Gene Cooperman
 - Gabriel Dos-Reis
 - Jean-Christophe Dubacq
 - Jean-Guillaume Dumas
 - Jean-Charles Faugere
 - Matteo Frigo
 - Thierry Gauthier
 - Pascal Giorgi
 - Stef Grallat
 - Jeremy Johnson
 - Erich Kallofen
 - Herbert Kuchen
 - Philippe Langlais
 - Anton Leykin
 - Genadiy Malitschchok
 - Michael Monagan
 - Winfried Neun
 - Clement Perret
 - Nicolas Pinto
 - Manuel Prieto-Matias
 - Markus Pueschel
 - Nathalie Revol
 - David Saunders
 - Eric Schost
 - Wolfgang Schreiner
 - Arne Storjohann
 - Steven Toledo
 - Gilles Villard
 - Yuzhen Xie
 - Kazuhiko Yokoyama



The workshop was held in a beautiful location with a view of the mountains. The participants enjoyed the scenery and the company of their colleagues.

The workshop was held in a beautiful location with a view of the mountains. The participants enjoyed the scenery and the company of their colleagues.

The workshop was held in a beautiful location with a view of the mountains. The participants enjoyed the scenery and the company of their colleagues.

The workshop was held in a beautiful location with a view of the mountains. The participants enjoyed the scenery and the company of their colleagues.

The workshop was held in a beautiful location with a view of the mountains. The participants enjoyed the scenery and the company of their colleagues.

