INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team Runtime

Efficient Runtime Systems for Parallel Architectures

Bordeaux - Sud-Ouest

Theme : Distributed and High Performance Computing

Table of contents

RUNTIME is an INRIA Project-Team joint with University of Bordeaux (UB1) and CNRS (LaBRI, UMR 5800)

1. Team

Research Scientist

Olivier Aumage [INRIA, Research Associate (CR)]
Alexandre Denis [INRIA, Research Associate (CR)]
Brice Goglin [INRIA, Research Associate (CR)]
Emmanuel Jeannot [INRIA, Research Associate (CR), from Sep. 2009, HdR]

Faculty Member

Raymond Namyst [University Bordeaux 1, Team Leader, Professor (Pr), HdR]
Denis Barthou [ENSEIRB, Professor (Pr), from Sep. 2009, HdR]
Marie-Christine Counilh [University Bordeaux 1, Assistant Professor (MdC)]
Guillaume Mercier [ENSEIRB, Assistant Professor (MdC)]
Samuel Thibault [University Bordeaux 1, Assistant Professor (MdC)]
Pierre-André Wacrenier [University Bordeaux 1, Assistant Professor (MdC)]

Technical Staff

Ludovic Courtès [INRIA, Research Engineer (IR), until Apr. 2009]
Nathalie Furmento [CNRS, Research Engineer (IR)]
Yannick Martin [INRIA, Associate Engineer, from Dec. 2009]
Cécile Romo-Glinos [INRIA, Associate Engineer, until Aug. 2009]
Ludovic Stordeur [INRIA, Associate Engineer, from Oct. 2009]

PhD Student

Cédric Augonnet [University Bordeaux 1, École Normale Supérieure de Lyon grant, since Sep. 2008]
François Broquedis [University Bordeaux 1, MESR grant, since Sep. 2007]
Louis-Claude Canon [University Bordeaux 1, MESR grant, from Sep. 2007]
Jérôme Clet-Ortega [University Bordeaux 1, MESR grant, since Sep. 2007]
François Diakhaté [CEA, since Sep. 2007]
Mathieu Faverge [INRIA, ANR grant, since Nov. 2006]
Sylvain Henry [University Bordeaux 1, MESR grant, from Nov. 2009]
Julien Jaeger [University Versailles, ANR grant, from Oct. 2007]
Stéphanie Moreaud [INRIA, ANR grant, since Sep. 2007]
François Trahay [University Bordeaux 1, MENRT grant, since Apr. 2006]

Visiting Scientist

Akihiro Nomura [University of Tokyo, Mar. 2009]
Tomoya Adachi [University of Tokyo, Mar. 2009]
Vasco Pedro [Universidade de Evora, Universidade Nova de Lisboa, Apr. 2009]
Salvator Abreu [Universidade de Evora, Universidade Nova de Lisboa, Apr. 2009]
Kazuki Ota [University of Tokyo, Dec. 2009]
Yukata Ishikawa [University of Tokyo, Dec. 2009]

Administrative Assistant

Sylvie Embolla [INRIA]

2. Overall Objectives

2.1. Designing Efficient Runtime Systems

The RUNTIME research project takes place within the context of high-performance computing. It seeks to explore the design, the implementation and the evaluation of novel mechanisms needed by **runtime systems** for parallel computers. *Runtime systems* are intermediate software layers providing parallel programming environments with specific functionalities left unaddressed by the underlying operating system. Runtime systems can thus be seen as functional extensions of operating systems, but the boundary between them is rather fuzzy since runtime systems may actually contain specific extensions/enhancements to the underlying operating system (e.g. extensions to the OS thread scheduler). The increasing complexity of modern parallel hardware, making it more and more necessary to postpone essential decisions and actions (scheduling, optimizations) at run time, emphasizes the role of runtime systems.

One of the main challenges encountered when designing modern runtime systems is to provide powerful abstractions, both at the programming interface level and at the implementation level, to deal with the increasing complexity of upcoming hardware architectures. While it is essential to understand – and somehow anticipate – the evolutions of hardware technologies (e.g. programmable network interface cards, multicore architectures, hardware accelerators), the most delicate task is to extract models and abstractions that will fit most of upcoming hardware features.

The originality of the runtime group lies in the fact that we address all these issues following a global approach, so as to propose complementary solutions to problems which may not seem to be linked at first sight. We actually realized, for instance, that we could greatly improve our communication optimization techniques by increasing the functionalities of the underlying core thread scheduler. This illustrates why most of our research efforts have consisted in cross-studying different topics, and have led to co-designing many software.

Our research project centers on three main directions:

Mastering large, hierarchical multiprocessor machines

- Thread scheduling over multicore machines
- Data management over NUMA architectures
- Task scheduling over GPU heterogeneous machines

Optimizing communication over high performance clusters

- Scheduling data packets over high speed networks
- New MPI implementations for Petascale computers
- Optimized intra-node communication

Integrating Communications and Multithreading

- Parallel, event-driven communication libraries
- Communication and I/O within large multicore nodes

Beside those main research topics, we obviously intend to work in collaboration with other research teams in order to *validate* our achievements by integrating our results into larger software environments (MPI, OpenMP) and to *join* our efforts to solve complex problems.

Among the target environments, we intend to carry on developing the successor to the PM^2 software suite, which would be a kind of technological showcase to validate our new concepts on real applications through both academic and industrial collaborations (CEA/DAM, Bull, IFP, Total). We also plan to port standard environments and libraries (which might be a slightly sub-optimal way of using our platform) by proposing extensions (as we already did for MPI and Pthreads) in order to ensure a much wider spreading of our work and thus to get more important feedback.

Finally, as most of our work proposed is intended to be used as a foundation for environments and programming tools exploiting large scale, high performance computing platforms, we definitely need to address the numerous scalability issues related to the huge number of cores and the deep hierarchy of memory, I/O and communication links.

3. Scientific Foundations

3.1. Runtime Systems Evolution

This research project takes place within the context of high-performance computing. It seeks to contribute to the design and implementation of parallel runtime systems that shall serve as a basis for the implementation of high-level parallel middleware. Today, the implementation of such software (programming environments, numerical libraries, parallel language compilers, parallel virtual machines, etc.) has become so complex that the use of portable, low-level runtime systems is unavoidable.

Our research project centers on three main directions:

Mastering large, hierarchical multiprocessor machines With the beginning of the new century, computer makers have initiated a long term move of integrating more and more processing units, as an answer to the frequency wall hit by the technology. This integration cannot be made in a basic, planar scheme beyond a couple of processing units for scalability reasons. Instead, vendors have to resort to organize those processing units following some hierarchical structure scheme. A level in the hierarchy is then materialized by small groups of units sharing some common local cache or memory bank. Memory accesses outside the locality of the group are still possible thanks to bus-level consistency mechanisms but are significantly more expensive than local accesses, which, by definition, characterizes NUMA architectures.

Thus, the task scheduler must feed an increasing number of processing units with work to execute and data to process while keeping the rate of penalized memory accesses as low as possible. False sharing, ping-pong effects, data vs task locality mismatches, and even task vs task locality mismatches between tightly synchronizing activities are examples of the numerous sources of overhead that may arise if threads and data are not distributed properly by the scheduler. To avoid these pitfalls, the scheduler therefore needs accurate information both about the computing platform layout it is running on and about the structure and activities relationships of the application it is scheduling.

As quoted by Gao *et al.* [51], we believe it is important to expose domain-specific knowledge semantics to the various software components in order to organize computation according to the application and architecture. Indeed, the whole software stack, from the application to the scheduler, should be involved in the parallelizing, scheduling and locality adaptation decisions by providing useful information to the other components. Unfortunately, most operating systems only provide a poor scheduling API that does not allow applications to transmit valuable *hints* to the system.

This is why we investigate new approaches in the design of thread schedulers, focusing on high-level abstractions to both model hierarchical architectures and describe the structure of applications' parallelism. In particular, we have introduced the *bubble* scheduling concept [12] that helps to structure relations between threads in a way that can be efficiently exploited by the underlying thread scheduler. *Bubbles* express the inherent parallel structure of multithreaded applications: they are abstractions for grouping threads which "work together" in a recursive way. We are exploring how to dynamically schedule these irregular nested sets of threads on hierarchical machines [24], the key challenge being to schedule related threads as closely as possible in order to benefit from cache effects and avoid NUMA penalties. We are also exploring how to improve the transfer of scheduling hints from the programming environment to the runtime system, to achieve better computation efficiency.

Aside from greedily invading all these new cores, demanding HPC applications now throw excited glances at the appealing computing power left unharvested inside the graphical processing units (GPUs). A strong demand is arising from the application programmers to be given means to access this power without bearing an unaffordable burden on the portability side. Efforts have already been made by the community in this respect but the tools provided still are rather close to the hardware, if not to the metal. Hence, we decided to launch some investigations on addressing this issue. In particular, we have designed a programming environment named STARPU that enables the programmer to offload tasks onto such heterogeneous processing units and gives that programmer tools to fit tasks to processing units capability, tools to efficiently manage data moves to and from the offloading hardware and handles the scheduling of such tasks all in an abstracted, portable manner. The challenge here is to take into account the intricacies of all computation unit: not only the computation power is heterogeneous among the machine, but data transfers themselves have various behavior depending on the machine architecture and GPUs capabilities, and thus have to be taken into account to get the best performance from the underlying machine. As a consequence, STARPU not only pays attention to fully exploit each of the different computational resources at the same time by properly mapping tasks in a dynamic manner according to their computation power and task behavior by the means of scheduling policies, but it also provides a distributed shared-memory library that makes it possible to manipulate data across heterogeneous multicore architectures in a high-level fashion while being optimized according to the machine possibilities.

Optimizing communications over high performance clusters and grids Using a large panel of mechanisms such as user-mode communications, zero-copy transactions and communication operation offload, the critical path in sending and receiving a packet over high speed networks has been drastically reduced over the years. Recent implementations of the MPI standard, which have been carefully designed to directly map *basic* point-to-point requests onto the underlying low-level interfaces, almost reach the same level of performance for very basic point-to-point messaging requests. However more complex requests such as non-contiguous messages are left mostly unattended, and even more so are the irregular and multiflow communication schemes. The intent of the work on our NEWMADELEINE communication engine, for instance, is to address this situation thoroughly. The NEWMADELEINE optimization layer delivers much better performance on *complex* communication schemes with negligible overhead on basic single packet point-to-point requests. Through Mad-MPI, our proof-of-concept implementation of a subset of the MPI API, we intend to show that MPI applications can also benefit from the NEWMADELEINE communication engine.

The increasing number of cores in cluster nodes also raises the importance of intra-node communication. Our KNEM software module aims at offering optimized communication strategies for this special case and let the above MPI implementations benefit from dedicated models depending on process placement and hardware characteristics.

Moreover, the convergence between specialized high-speed networks and traditional ETHERNET networks leads to the need to adapt former software and hardware innovations to new message-passing stacks. Our work on the OPEN-MX software is carried out in this context.

Regarding larger scale configurations (clusters of clusters, grids), we intend to propose new models, principles and mechanisms that should allow to combine communication handling, threads scheduling and I/O event monitoring on such architectures, both in a portable and efficient way. We particularly intend to study the introduction of new runtime system functionalities to ease the development of code-coupling distributed applications, while minimizing their unavoidable negative impact on the application performance.

Integrating Communications and Multithreading Asynchronism is becoming ubiquitous in modern communication runtimes. Complex optimizations based on online analysis of the communication schemes and on the de-coupling of the request submission vs processing. Flow multiplexing or transparent heterogeneous networking also imply an active role of the runtime system request submit and process. And communication overlap as well as reactiveness are critical. Since network

request cost is in the order of magnitude of several thousands CPU cycles at least, independent computations should not get blocked by an ongoing network transaction. This is even more true with the increasingly dense SMP, multicore, SMT architectures where many computing units share a few NICs. Since portability is one of the most important requirements for communication runtime systems, the usual approach to implement asynchronous processing is to use threads (such as Posix threads). Popular communication runtimes indeed are starting to make use of threads internally and also allow applications to also be multithreaded. Low level communication libraries also make use of multithreading. Such an introduction of threads inside communication subsystems is not going without troubles however. The fact that multithreading is still usually optional with these runtimes is symptomatic of the difficulty to get the benefits of multithreading in the context of networking without suffering from the potential drawbacks. We advocate the importance of the cooperation between the asynchronous event management code and the thread scheduling code in order to avoid such disadvantages. We intend to propose a framework for symbiotically combining both approaches inside a new generic I/O event manager.

4. Application Domains

4.1. Panorama

The RUNTIME group is working on the design of efficient runtime systems for parallel architectures. We are currently focusing our efforts on High Performance Computing applications that merely implement numerical simulations in the field of Seismology, Weather Forecasting, Energy, Mechanics or Molecular Dynamics. These time-consuming applications need so much computing power that they need to run over parallel machines composed of several thousands of processors.

Because the lifetime of HPC applications often spreads over several years and because they are developed by many people, they have strong portability constraints. Thus, these applications are mostly developed on top of standard APIs (e.g. MPI for communications over distributed machines, OpenMP for shared-memory programming). That explains why we have long standing collaborations with research groups developing parallel language compilers, parallel programming environments, numerical libraries or communication software. Actually, all these “clients” are our primary target.

Although we are currently mainly working on HPC applications, many other fields may benefit from the techniques developed by our group. Since a large part of our efforts is devoted to exploiting multicore machines and GPU accelerators, many desktop applications could be parallelized using our runtime systems (e.g. 3D rendering, etc.).

5. Software

5.1. ForestGOMP

Participants: Olivier Aumage, François Broquedis, Ludovic Courtès, Pierre-André Wacrenier.

FORESTGOMP is an OPENMP environment based on both the GNU OPENMP run-time[1] support and the MARCEL thread library (described in Section 5.4). It is designed to schedule efficiently nested sets of threads (derived from nested parallel regions) over hierarchical architectures. Indeed, approaching the theoretical performance of hierarchical multicore machines requires a very careful distribution of threads and data over the underlying architecture so as to minimize cache misses and NUMA penalties.

[1] GNU OPENMP (GOMP) is distributed as part of the GNU Compiler Collection (GCC). See http://gcc.gnu.org/projects/gomp/ for more information.

Language extensions such as OPENMP provide opportunities to improve thread scheduling over hierarchical architectures in a portable way. High-level parallel constructs such as OPENMP's provide useful hints about a program's parallel structure. These hints can be leveraged by the underlying scheduler and run-time system to improve thread scheduling, e.g., by taking into account cache affinity among threads, or affinity between threads and data.

The FORESTGOMP runtime generates nested MARCEL bubbles each time an OPENMP parallel region is encountered, thereby grouping threads sharing common data. Topology-aware scheduling policies implemented by BUBBLESCHED can then be used. Such policies dynamically map bubbles onto the various levels of the underlying hierarchical architecture. FORESTGOMP allowed us to validate the BUBBLESCHED approach with highly irregular, fine grain, divide-and-conquer parallel applications. This approach has proved to yield noticeably better performance for a variety of parallel applications.

The experiments we conducted validate this approach in terms of ease of development for the programmer, flexibility, portability and performance. It is a way for experts to build and combine complex scheduling strategies that take characteristics of the application into account. Application programmers get a greater control on scheduling of their OPENMP programs while completely avoiding intrusive and non-portable changes to their applications.

François BROQUEDIS and Ludovic COURTÈS are the maintainers to this piece of software. It is freely available under the terms of the GNU General Public License version 2 at the following URL: http://runtime.bordeaux.inria.fr/forestgomp/

5.2. Hardware Locality

Participants: Jérôme Clet-Ortega, Brice Goglin, Samuel Thibault.

Hardware Locality (HWLOC) is a library and set of tools aiming at discovering and exposing the topology of machines, including processors, cores, threads, shared caches and NUMA memory nodes. It builds a widely-portable abstraction of these resources and exposes it to the application so as to help them adapt their behavior to the hardware characteristics. It targets many types of high-performance computing applications [25], from thread scheduling (it is already used by the MARCEL multithreading library and FORESTGOMP OPENMP runtime system) to placement of MPI processes (the OPEN MPI and MPICH2 libraries are also being ported on top of it).

HWLOC was initially developed inside the MARCEL library as a way to discover the hardware processor topology and place threads according to their affinities. The democratization of multicore processors in HPC raised the need to offer such hardware discovery and placing abilities to more than just the MARCEL library. The code was thus extracted out of MARCEL and made self-consistent as the `libtopology` project. Due to similar goals, it was then merged with the *Portable Linux Processor Affinity* library (PLPA) as HWLOC. PLPA brought a lot of users thanks to its integration in OPEN MPI [2] while `libtopology` brought a much more flexible interface, an advanced knowledge of the hardware, and more portability.

HWLOC is now developed in collaboration with OPEN MPI which now hosts the project. The core development is still mostly performed by Brice GOGLIN and Samuel THIBAULT from the RUNTIME team-project, but many outside contributors are joining the effort, especially from the OPEN MPI and MPICH2 communities. HWLOC is composed of 11 000 lines of C. It is distributed under the terms of the BSD License at the following URL: http://www.open-mpi.org/projects/hwloc/

5.3. KNem

Participants: Brice Goglin, Stéphanie Moreaud.

KNEM (*Kernel Nemesis*) is a Linux kernel module that offers high-performance data transfer between user-space processes. Based on OPEN-MX offering efficient intra-node communication (see 5.8 and [33]), it was decided to make this ability available to any MPI implementation without requiring the use of OPEN-MX and ETHERNET networks.

[2] Before being merged with `libtopology` as HWLOC, PLPA was distributed as an OPEN MPI sub-project at http://www.openmpi.org.

KNEM was developed in collaboration with the MPICH2 team at the Argonne National Laboratory. It offers a very simple message passing interface that MPICH2 uses when transferring very large messages between processes on the same node. Thanks to its kernel-based design, it is able to transfer messages through a single memory copy, much faster than the usual user-space two-copy model. KNEM also offers the optional ability to offload memory copies on INTEL I/O AT hardware which improves throughput and reduces CPU consumption and cache pollution. MPICH2 is responsible for deciding when to use KNEM or not, and when to enable I/O AT copy offload [27].

KNEM is now developed in collaboration the MPICH2 team[3] and with the OPEN MPI project[4]. Its main contributor is Brice GOGLIN. KNEM is composed of 4800 lines of C and it is distributed under the terms of the CeCILL-B License at the following URL: http://runtime.bordeaux.inria.fr/knem/

5.4. Marcel

Participants: Olivier Aumage, Ludovic Courtès, Nathalie Furmento, Samuel Thibault.

MARCEL is the thread library of the PM^2 software suite. MARCEL features a two-level thread scheduler (also called N:M scheduler) that schedules user-level threads on top of a set of kernel threads (usually one kernel thread per logical processor). Such a model achieves the performance of a user-level thread package while being able to exploit multiprocessor machines. The architecture of MARCEL was carefully designed to support a large number of threads and to efficiently exploit hierarchical architectures (e.g. multicore chips, NUMA machines).

At the core of the MARCEL architecture, we find the BUBBLESCHED scheduling framework (http://runtime.bordeaux.inria.fr/bubblesched/). Computing platforms are becoming increasingly hierarchical. As an answer to this trend, BUBBLESCHED provides the *application* programmer with high level constructs, called *bubbles*, to let him express the affinity between the various activities of his application: the application describes affinities between the threads it launches by encapsulating them into nested bubbles (threads which work on the same data for instance), which thus form a tree of the hierarchical activity structure of the application. Thanks to the HWLOC hardware discovery library (see Section 5.2), BUBBLESCHED then provides the *scheduler* programmer with a *hierarchical runqueues tree* that represents the detected hierarchical platform and a *toolkit* of basic operations to dynamically map the hierarchical activity structure (that is, the threads) of the application (as modeled by the bubbles) onto the hierarchical platform in a suitably tailored scheduling [12]. That permits to benefit from cache effects as much as possible, or favor bandwidth, or favor load balancing, or whatever fits the application best. Various scheduling algorithms are provided and can be combined to meet various application needs (*e.g.* cache affinity vs memory affinity [26])

Marcel has a dedicated module to handle memory called MAMI. It allows developers to manage memory with regard to NUMA nodes. Aside from usual memory allocation policies such as binding or interleaving, it also offers two memory migration strategies. The first method is synchronous and allows to move data on a given node on application's demand. The second method is based on a *Next-Touch* policy. MAMI also provides the application with hints about the actual cost of reading, writing, or migrating distant memory buffers. Moreover, MAMI gathers statistics on how much memory is available and left on the different nodes.

A trace of the scheduling events can be recorded and used after execution for generating an animated movie showing a replay of the execution: how bubbles and threads were created, how they got distributed over the machine, how they eventually got scheduled on processors, etc. End users may hence easily try and tune various bubble schedulers for their applications, and select the most suited one.

At very fine grains of parallelism, the inherent cost of parallelism management per grain is not negligible anymore when compared to the grains themselves. MARCEL thus provides a *seed* construct which can be seen as a precursor of thread. Creating a thread seed does not reserve any resource except from the information about the task to be run. It is only when the time comes to actually run the seed that MARCEL attempts to

[3] The MPICH2 project includes KNEM support since release 1.1.1. It is distributed from http://www.mcs.anl.gov/research/projects/mpich2/.

[4] The OPEN MPI project will include KNEM support starting with release 1.5. It is distributed from http://www.openmpi.org.

reuse the resources and the context of another, dying thread, significantly saving management costs when succeeding while keeping the penalty low otherwise.

In addition to a set of original extensions, MARCEL provides a POSIX-compliant interface which thus permits to take advantage of it by just recompiling unmodified applications or parallel programming environments (API compatibility), or even by running already-compiled binaries with the Linux NPTL ABI compatibility layer.

MARCEL consists in 83 000 lines of code. This library is developed and maintained by Samuel THIBAULT and Olivier AUMAGE. The software is freely available under the terms of the GNU General Public License version 2 at the following URL: http://runtime.bordeaux.inria.fr/marcel/.

5.5. NewMadeleine

Participants: Alexandre Denis, François Trahay.

The MADELEINE library which had been the communication subsystem of the PM^2 software suite for almost ten years has now been replaced in production by the NEWMADELEINE library. NEWMADELEINE is primarily dedicated to the exploitation of clusters interconnected with (possibly multiple) high-speed networks, potentially of different natures. NEWMADELEINE is a complete redesign and rewrite of MADELEINE. The new architecture is entirely modular, based on software components.

The NEWMADELEINE optimizing scheduler aims at enabling the use of a much wider range of communication flow optimization techniques such as packet reordering or cross-flow packet aggregation.

NEWMADELEINE targets applications with irregular, multiflow communication schemes such as found in the increasingly common application conglomerates made of multiple programming environments and coupled pieces of code, for instance. It is designed to be programmable through the concepts of optimization *strategies* (*what* to optimize for, what the optimization goal is) expressed in terms of *tactics* (*how* to optimize to reach the optimization goal), allowing experimentations with multiple approaches or on multiple issues with regard to processing communication flows. Tactics themselves are made of basic communication flows operations such as packet merging or reordering.

Special purpose strategies have also been developed. For example, a strategy is dedicated to heterogeneous multirail support. Another QoS-based strategy is responsible for differentiated service support: it allows to use distinct optimizations and priorities for distinct communication flows.

NEWMADELEINE also provides an interface to the MPI standard called Mad-MPI. This simple, straightforward proof-of-concept implementation provides a subset of the MPI API, to allow MPI applications to benefit from the NEWMADELEINE communication engine. Mad-MPI is based on the point-to-point nonblocking posting (`isend`, `irecv`) and completion (`wait`, `test`) operations of MPI, these four operations being directly mapped to the equivalent operations of NEWMADELEINE.

NEWMADELEINE has strong relationships established with other software projects in the RUNTIME project-team, each of whose having been the subject of dedicated work during the last year. Indeed, NEWMADELEINE is the direct core communication library of the Mad-MPI [3] and MPICH-Madeleine modules and has been ported as a communication subsystem target for the MPICH2-Nemesis software from Argonne National Laboratory [36]. It is built upon the PadicoTM software component model, and is now the default communication stack for clusters in PadicoTM. It fundamentally relies on the new PIOMAN [14] module and the MARCEL module for parallel processing of communication flows and progression. It now works together with the Fast User Trace module to provide post-mortem communication schemes analysis. And finally it directly depends on the recent work on *Non-Uniform Input-Output Access* (NUIOA)[9] when run on non-uniform hierarchical architectures.

The reference software development branch of the NEWMADELEINE software consists in 49 000 lines of code. NEWMADELEINE is available on various networking technologies: Myrinet, Infiniband, Quadrics and ETHERNET. This library, distributed as part of the PM^2 software is developed and maintained by Alexandre DENIS, François TRAHAY and Raymond NAMYST. The software is freely available under the

terms of the GNU General Public License version 2 at the following URL: http://runtime.bordeaux.inria.fr/newmadeleine/.

5.6. PIOMan

Participants: Alexandre Denis, François Trahay.

PIOMAN [46] is the event detector server used by the PM^2 software suite. It aims at providing the other software components with a service that can guarantee a predefined level of "reactivity" to I/O events. It is typically used by NEWMADELEINE and PadicoTM to quickly react to network events, such as the arrival of a new packet.

PIOMAN is able to isolate blocking system calls on dedicated threads so that the whole process is not suspended. It is actually a portable alternative to the Scheduler Activations model proposed by Anderson [48] and implemented in the LinuxActivations library [7]. PIOMAN is also able to handle non-blocking detection methods. It thus choose the more suitable method to use depending on the processors' load and the communication library's preferences. This way, the application is reactive whatever the context is.

The level of reactivity provided by PIOMAN allows NEWMADELEINE to make communications progress in the background (by making the *rendezvous* handshake progress for instance) and thus to fully overlap computation and communication [13].

Thanks to a scalable task onloading mechanism, PIOMAN is able to balance the processing of communication requests across the whole machine, message submission to the networks can be offloaded on idle core in order to overlap communication and computation even for eager message that require CPU-intensive memory copies [40].

MADELEINE, NEWMADELEINE and PadicoTM have been ported over PIOMAN. Within our collaboration with the Argonne National Laboratory, MPICH2-Nemesis is being ported over PIOMAN [36]. We also plan to use PIOMAN in MPI implementations such as YAMPII (within the collaboration with the University of Tokyo).

This library, distributed as part of the PM^2 software is developed and maintained by François TRAHAY. The software is freely available under the terms of the GNU General Public License version 2 at the following URL: http://runtime.bordeaux.inria.fr/pioman/.

5.7. MPICH2-NewMadeleine

Participants: Guillaume Mercier, François Trahay.

MPICH2-NEWMADELEINE is the successor to the former MPICH-Madeleine implementation which derived from MPICH 1.2.7. MPICH2-NEWMADELEINE is based on the most recent MPICH2 software (1.2.1) and utilizes the NEWMADELEINE communication library for all network communication. As far as intranode communication are concerned, the Nemesis communication subsystem is employed [6]. Nemesis is a generic communication subsystem which goal is to address the communication needs of a wide range of programming tools and environments for clusters and parallel architectures. It has been designed to yield very low latency and high bandwidth, especially for intranode communication.

The resulting MPI implementation exhibits excellent performance, especially in the shared-memory case, which is crucial in the case of NUMA clusters. The level of performance is indeed very good and MPICH2-Nemesis compares favorably with other next-generation MPI implementations such as Open MPI or GridMPI. The latencies achieved by MPICH2-NEWMADELEINE in shared-memory are currently the best among generic MPI implementations and are extremely close to that of highly-tuned vendor-specific ports.

The NEWMADELEINE communication library has been integrated as a Nemesis network module but some architectural changes have been made to the upper layers, in particular at the ADI3 level (ch3 has thus been modified). Those changes allow the MPICH2 software stack to efficiently take advantage of tag-matching capable interfaces such as NEWMADELEINE or Myricom MX. Also, all optimization mechanisms implemented in NEWMADELEINE are made available at the MPI level. For instance, MPICH2, with its NEWMADELEINE Nemesis module can use natively multirail configurations.

This work takes advantage of the Associate Team program in order to make frequent visits in order to coordinate the work.

MPICH2-NEWMADELEINE , a joint development between the ANL and the RUNTIME project-team will soon be available on the RUNTIME website and is developed and maintained by Darius BUNTINAS, Guillaume MERCIER, François TRAHAY and David GOODELL.

5.8. Open-MX

Participants: Nathalie Furmento, Brice Goglin, Ludovic Stordeur.

The OPEN-MX software stack is a high-performance message passing implementation for any generic ETHERNET interface. It was developed within our collaboration with Myricom, Inc. as a part of the move towards the convergence between high-speed interconnects and generic networks. OPEN-MX exposes the raw ETHERNET performance at the application level through a pure message passing protocol.

While the goal is similar to the old GAMMA stack [50] or the recent iWarp [49] implementations, OPEN-MX relies on generic hardware and drivers and has been designed for message passing. OPEN-MX is also wire-compatible with Myricom MX protocol and interface so that any application built for MX may run on any machine without Myricom hardware and talk other nodes running with or without the native MX stack. OPEN-MX is under experimentation at the Argonne National Laboratory as a networking layer for the PVFS2 parallel file-system on the upcoming BlueGene/P machine in the Argonne laboratory [5]. It will connect BlueGene specific nodes with generic 10 gigabit/s ETHERNET boards to generic I/O nodes with Myri-10G running in native MX mode.

OPEN-MX offers efficient data movement abilities thanks to copy offload abilities of modern hardware [8]. It also implements advanced memory management mechanisms to hide the usual overhead of memory pinning for both intra-node and inter-node communications [33], [32]. OPEN-MX is also an interesting framework for studying next-generation hardware features that could help ETHERNET hardware become legacy in the context of high-performance computing. Some innovative message-passing-aware stateless abilities, such as multiqueue binding and interrupt coalescing, were designed and evaluated thanks to OPEN-MX [34], [20], [30].

Brice GOGLIN, Nathalie FURMENTO and Ludovic STORDEUR are the main contributors to OPEN-MX. The software is already composed of more than 37 000 lines of code in the Linux kernel and in user-space. It is freely available under the terms of the GNU General Public License version 2 at the following URL: http://open-mx.org/.

5.9. PadicoTM

Participant: Alexandre Denis.

PadicoTM is a high-performance communication framework for grids. It is designed to enable various middleware systems (such as CORBA, MPI, SOAP, JVM, DSM, etc.) to utilize the networking technologies found on grids. PadicoTM aims at decoupling middleware systems from the various networking resources to reach transparent portability and flexibility: the various middleware systems use PadicoTM through a seamless virtualization of networking resources; only PadicoTM itself uses directly the networks.

PadicoTM architecture is based on software components. Puk (the PadicoTM micro-kernel) implements a light-weight high-performance component model that is used to build communication stacks. Typical communications stacks built in PadicoTM follow a three-layer approach. The lowest layer, called the *arbitration layer*, aims at making the access to the resources cooperative rather than competitive. It enables the use of multiple middleware systems atop a single network, as needed by code coupling programming models such as parallel objects or parallel components. This layer is based on PIOMAN to ensure high performance and good interactions between threads and networking. The middle layer, called the *abstraction layer*, decouples the paradigm of the programming interface from the paradigm of the network; for example, it can do dynamic

[5] ANL's BlueGene/P system is ranked #7 in the June 2009 Top500, with 450 Tflop/s

client/server connections over static SPMD networks. The highest level layer, called the *personality layer*, gives several API called "personalities" over the abstractions. It aims at providing the middleware systems with the API they expect. It enables PadicoTM to seamlessly integrate *unmodified* middleware systems.

PadicoTM currently supports most high performance networks (Infiniband, Myrinet, SCI, Quadrics, etc.), communication methods for grids (plain TCP, splicing to cross firewalls, routing, tunneling). Various middleware systems are supported over PadicoTM: various CORBA implementations (omniORB, Mico), popular MPI implementations (MPICH from Argonne – actually, MPICH/PadicoTM is derived from MPICH-Madeleine —, YAMPI from the University of Tokyo, our own Mad-MPI), Apache Portable Runtime, JXTA from Sun (in collaboration with the PARIS project), gSOAP, Mome (DSM developed in the PARIS project), Kaffe (Java virtual machine), and Certi (HLA implementation from the ONERA).

PadicoTM was started in the PARIS project (Rennes) in 2001, in collaboration with Christian PÉREZ and migrated in RUNTIME in October 2004 together with Alexandre DENIS. The current main contributors to PadicoTM are Alexandre DENIS and François TRAHAY (RUNTIME) with some occasional contributions from Christian PÉREZ (PARIS).

PadicoTM is composed of roughly 50 000 lines of C. It is free software distributed under the terms of the GNU General Public License, and is available for download at: http://runtime.bordeaux.inria.fr/PadicoTM/ . It is has been hosted on InriaGForge since mid-2005. PadicoTM is registered at the APP under number IDDN.FR.001.260013.000.S.P.2002.000.10000.

PadicoTM component model is now used in NEWMADELEINE. It is the cornerstone for networking integration in the projects "LEGO" and "COOP" from the ANR. Previously, it has been used by several projects: ACI GRID "RMI", ACI GRID HydroGrid, ACI GRID EPSN, Inria ARC RedGrid, and the European FET project POP.

5.10. StarPU

Participants: Cédric Augonnet, Samuel Thibault.

STARPU is a unified runtime system that offers a support for heterogeneous multicore architectures (multicore, GPGPUs, Cell ...). Its goal is to help higher level softwares such as compiler environment or specific high performance libraries to harness the power of those emerging architectures in a high level and portable way.

STARPU differs from most similar projects as it considers all computing resources simultaneously, instead of merely offloading computation onto one or several accelerators. It is organized around three main components : a distributed shared memory that offers a high level interface to manipulate data in a transparent and efficient manner, a unified execution model called "codelets" which model tasks that can be executed on different architectures, and a scheduling engine that makes possible to design scheduling policies which assign tasks to the different computing resources as efficiently as possible. Performance models of task duration and data transfers can be provided or auto-tuned to implement various scheduling heuristics.

STARPU has not only demonstrated that it is possible to distribute computations efficiently over a machine's heterogeneous computing resources, but it has also shown that it is possible to take advantage of the actual heterogeneity of the machine, since parts of applications are better suited to running on multicores while other parts benefit from running on a GPGPU.

Cédric AUGONNET is the main contributor to STARPU which is currently composed of more than 38 000 lines of code. It is freely available under the terms of the GNU Lesser General Public License (LGPL) version 2.1 at the following URL: http://runtime.bordeaux.inria.fr/StarPU/.

6. New Results

6.1. Efficient scheduling of OpenMP threads on NUMA machines

Participants: Olivier Aumage, François Broquedis, Raymond Namyst, Pierre-André Wacrenier.

To express parallelism, scientific programmers are used to program with OPENMP, a high level parallel language, that relies on a set of annotations (including scheduling directives). While OPENMP-parallelized applications suit well SMP computers, their execution on NUMA architectures are far from being optimal, particularly when considering irregular applications. This is due to the difficulty to combine load balancing and thread/memory affinity relations. Indeed, nowadays OPENMP runtimes do not map the application parallel structure to the underlying architecture considering threads and data relations.

To solve this problem, we designed "FORESTGOMP", an extension to the GNU OPENMP (GOMP) runtime support that relies on the MARCEL/BUBBLESCHED thread scheduling package already described in Section 5.1. This structured approach extends the scope of OPENMP to NUMA architectures and nested parallelism. Indeed, while the raw performance of FORESTGOMP on flat parallelism is similar to GOMP and ICC, FORESTGOMP nested parallelism outperforms them on irregular applications.

FORESTGOMP is also now able to take thread/memory affinities into account while distributing the load on hierarchical architectures. It relies on the MAMI memory manager to allocate, bind or migrate memory buffers [26], [42]. Moreover FORESTGOMP adopts a two-ways mechanism [24] to decide how often the distribution needs to be updated. First, every time the application programmer updates the memory affinities, the bubble scheduler is called to check the current distribution. This approach may not be sufficient for irregular applications, so FORESTGOMP also provides a more dynamic mechanism based on hardware counters inspecting. The runtime checks the counters on a regular basis and infers the amount of remote memory accesses initiated from the current processor while defining a threshold from which FORESTGOMP will call the scheduler for checking the current distribution. These two approaches are complementary. Indeed, in some cases updates from the application programmer will not need the scheduler to rethink the current distribution. In other cases the programmer is able to roughly define which part of his application will work on which data, but cannot tell precisely when and how. Hardware counters can help reacting at the right time for these situations.

6.2. High-performance memory migration in Linux

Participants: François Broquedis, Nathalie Furmento, Brice Goglin, Pierre-André Wacrenier.

We diagnosed a dramatic performance problem in the memory migration subsystem of the Linux kernel caused by the quadratic complexity of the model. We reworked the core implementation to restore a linear behavior and thus achieve a constant asymptotic throughput, as well as dramatically reduce its temporary memory consumption. This low-level improvement was integrated in the standard Linux kernel as of 2.6.29 and further optimized in 2.6.31. It enables high-performance static memory migration within multithreaded applications on Linux for the first time [29].

To further help dynamic management of threads and data within applications, we proposed a dynamic migration model called *Next-Touch*. It lets data buffers automatically follow their accessing threads when the workload balance changes in the machine. This work has been presented to the Linux kernel community [31] as an improved way to efficiently support irregular and dynamic parallelism within high-performance computing applications.

This low-level work is offered to user-space applications through the memory manager, MAMI, that we added to our thread library MARCEL. MAMI allows users to allocate memory by specifying a NUMA node, or based on the *First-Touch* policy, or the *Next-Touch* policy. It also supports memory migration between nodes, migration cost prediction, as well as remote read/write access overhead prediction.

MAMI is used by FORESTGOMP to place threads and memory in a combined manner so as to take memory affinity into account [26], [24].

6.3. Communication Optimization over High Speed Networks

Participants: Alexandre Denis, Raymond Namyst, François Trahay.

The NEWMADELEINE communication subsystem introduces fundamental changes in communication request handling and optimizations. Traditionally, communication libraries, being synchronous, tightly link the communication requests to the application workflow, and therefore transmit incoming packets immediately to the lower network layer without any accumulation. On the contrary, NEWMADELEINE keeps accumulating packets in its optimization window while the NICs are busy. As soon as a NIC becomes idle, the optimization window is analyzed so as to generate a new ready-to-send request to be transfered through the card: NICs are exploited at their maximum (they are not overloaded when there is a high demand of transfers and under exploited when there is not) and the communication optimizations are made *just-in-time* so they closely fit the ongoing communication scheme at any given time.

When at least one of the multiplexing units becomes idle, an *optimization function* is called to elect the next request to be submitted to each idle unit. In doing so, it may select a packet to be sent from the optimization window, or for instance, synthesize a request out of several packets from that window. A wide panel of arguments may be used as an input to the optimizing function. The optimization function is to be selected among an extensible and programmable set of *strategies*. Each strategy aims at some particular optimizing goal. A strategy is itself made of one or more tactics that apply some elementary optimizing operations selected from the panel of usual operations.

By design, NEWMADELEINE is able to be used by multi-threaded applications, and utilizes itself multi-threading. We have measure [39] the impact of various levels of multi-threading support on performance.

Finally, we have integrated NEWMADELEINE into MPICH [36] and evaluated its performance and the efficiency of our optimization mechanisms when using MPI.

6.4. Multithreaded Communication Engine

Participants: Alexandre Denis, Raymond Namyst, François Trahay.

The increase of the number of cores per node in clusters requires changes in the exploitation of high performance networks. The classical MPI approach that consists in running one MPI process per core do not scale because of memory limitations. The use of an hybrid approach mixing MPI and threads seems to be an efficient way to exploit nowadays clusters. Thus, communication libraries have to take into account that applications may be multithreaded.

We designed an efficient thread-safe communication library that allows threads to communicate concurrently. The use of PIOMAN as well as fine-grain locking permits to achieve good performance even on multithreaded environment. We analyzed [39] the impact of various way of ensuring thread-safety on performance.

We showed [46] that, by using a centralized I/O event manager, a high level of "reactivity" can be guaranteed in a portable way even during heavy computation phases. We have designed a scalable architecture for this I/O server named PIOMAN. By interacting with the thread scheduler, PIOMAN detects the completion of the communication queries (either by polling the network or waiting for interrupts) and triggers the appropriate callback as soon as possible. Communication processing can thus be parallelized and the progression of asynchronous communications in the background can be performed efficiently, allowing a full overlap of communication and computation [40].

PadicoTM, MADELEINE and NEWMADELEINE are already using PIOMAN, making them reactive even when many computing threads are running. We are currently developing an enhanced version of NEWMADELEINE that optimizes the communications more efficiently thanks to the multithreading support provided by PIOMAN. By delegating message submission to idle cores, NEWMADELEINE is able to send messages in parallel over several (potentially different) networks interfaces, reducing the transmission duration [15].

6.5. High-performance message passing over generic Ethernet hardware

Participants: Nathalie Furmento, Brice Goglin.

The OPEN-MX message passing stack (described in Section 5.8) offers a native message passing layer on any ETHERNET hardware. The API compatibility with the native Myrinet Express stack already enables existing parallel application to use OPEN-MX. Indeed, several legacy high-performance layers such MPICH2 or Open MPI run works transparently on top of OPEN-MX with satisfying performance thanks to advanced data movement techniques [8].

We showed that OPEN-MX opens a large room for innovative memory management optimizations. Indeed, thanks to OPEN-MX not requiring complex synchronization between the application, the driver and the NIC, we were able to implement an overlapped memory pinning model, causing the expensive pinning overhead to be hidden behind the actual communication time [32]. Moreover, by combining this idea with I/O AT copy offload, we implemented in OPEN-MX a dramatically improved intra-node communication stack. As soon as large messages are involved or processes are not sharing a cache, OPEN-MX now outperforms most existing MPI layers as soon as messages are large [33]. This work raised the awareness of inefficient large message intra-node communications in MPICH2 and OPEN MPI, leading to the development of our KNEM driver (see Section 6.8).

Finally, OPEN-MX is also an interesting framework for studying next-generation hardware features that could help ETHERNET hardware becoming legacy in the context of high-performance computing. We exhibited some cache-inefficiency problems in the OPEN-MX receive stack that are inherited from the ETHERNET model. By adding OPEN-MX-aware packet filtering capabilities in the *Multiqueue* firmware of Myri-10G boards, we are able to control the location of the processing of the incoming OPEN-MX traffic. We extended this model by providing an automatic binding facility for user-space applications. This model enables the whole processing of each incoming OPEN-MX packet on the core that runs its target application, causing the overall cache efficiency to improve dramatically [34], [20].

Another example of stateless offload ability that can easily be added to ETHERNET NICs so as to bring message passing performance improvements is a clever interrupt coalescing. Indeed, we showed that the usual coalescing of interrupts that was designed for TCP/IP only favors large messages while it dramatically increases small message latency. We designed a dedicated coalescing mechanism and showed that its implementation in Myri-10G NICs improves OPEN-MX performance with regards to both these important HPC metrics [30].

6.6. Scheduling over Heterogeneous Multicore Architectures

Participants: Cédric Augonnet, Raymond Namyst, Samuel Thibault.

In almost every computer nowadays lies a Graphical Processing Unit (GPU) and in that GPU lies a nominal computing power that makes the power of even the most recent multicore central processing units looks anemic in comparison. Innovative processing architectures such as the Cell Broadband Engine from IBM found in the Sony's Playstation 3 also come full of promises. Thus, it did not take long in these days of ever growing computing needs for people to explore these new lands.

The power of these new heterogeneous computing architectures does not come for free, however. Cell's multiple synergistic processing units (SPUs) are equipped with a very small amount of memory. GPUs put drastic constraints on the data access pattern and require highly regular computations to actually deliver their full power. While scheduling over those architectures, the problem of mapping tasks onto available units is not the only one anymore. One also has to provide constructs and mechanisms to tailor those tasks to the characteristics of a given processing unit — refinement/filtering mechanisms — as well as to make sure that such tasks have the suitable data at hand when needed — memory/caching management and consistency mechanisms.

We thus designed a unified scheduling engine that makes it possible to easily implement task scheduling policies on top of heterogeneous multicore architectures. Combined with the use of performance models, which can be obtained through auto-tuning mechanisms [21], [17], [16], we have shown that substantial performance improvements result from the use of such scheduling policies. Not only does STARPU allow to make use of all computing resources at the same time (regular CPUs as well as a mixture of heterogeneous GPUs), but its scheduling engine even enables to *benefit* from the actual heterogeneity of the machines [23], [41].

The memory management library of the StarPU runtime system (described in Section 5.10) has been designed in order to leverage the inner complexity of accelerator programming by automating data coherency management, and can even *prefetch* data ahead of the actual computation to increase yet more the efficiency of the computation. Our approach thus makes it possible to efficiently handle arbitrarily large datasets instead of being limited by the size of accelerators' embedded memory. Therefore, we used StarPU to implement dense linear algebra parallel kernels that run simultaneously on multicore processors and GPGPUs. We also demonstrated the flexibility of our approach by adding support for the Cell/BE processor with very little efforts [22]. We showed that in addition to a unified execution model, it is important to exhibit an expressive interface to let the programmer (or the upper library) express his/her knowledge at the algorithmic level in order to give hint to the runtime system.

6.7. Optimization of the Sparse Direct Linear Solver PaStiX

Participant: Mathieu Faverge.

In the context of distributed NUMA architectures, a work has recently begun, in collaboration with the INRIA ScalApplix team-project, on studying optimization strategies and improving the scheduling of communications, threads and I/O on the sparse direct linear solver PaStiX. The solver provides the NewMadeleine and Marcel libraries with an advanced application to validate those strategies. Mathieu Faverge studies these aspects in the context of the NUMASIS ANR project. It has been proved that NUMA allocation can significantly improve the efficiency. In the *out-of-core* context, new problems related to the scheduling and the management of the computational tasks may arise (processors may be slowed down by I/O operations). Thus, specific algorithms for this particular context have to be designed and implemented.

6.8. High-Performance Intra-node Communications

Participants: Brice Goglin, Guillaume Mercier, Stéphanie Moreaud.

We showed in [33] that the major MPI implementations had severe performance problems for large-message intra-node communication. Using the operating system assistance and the copy offload abilities of modern hardware gave a significant performance improvement that was unfortunately only available through the Open-MX stack. We thus extracted the optimized intra-node communication model out of Open-MX and created the KNem driver so as to offer the same abilities to any existing MPI stack (see Section 5.3).

We showed that KNem indeed improves intra-node communication performance significantly. We proposed an automatic way to determine when memory copies should be offloaded on dedicated hardware based on hardware cache sharing between the processing cores [27]. We also showed that collective MPI operations were significantly improved thanks to our model, but also required specific careful tuning due to their intensive memory requirements [45], and we described how each communication strategy performance depends on process placement [37].

This work was initiated in the context of our collaboration with the MPICH2 team (see Section 8.6) and is now also pursued with the Open MPI project (see 8.7).

6.9. Topology-aware High-Performance Computing

Participants: François Broquedis, Jérôme Clet-Ortega, Brice Goglin, Emmanuel Jeannot, Guillaume Mercier, Stéphanie Moreaud, Samuel Thibault.

The democratization of multicore processors and NUMA machines spreads complex and hierarchical architectures to the whole world of high-performance computing and even more. So far, the need to master the internal hardware topology was critical only to large shared-memory machines but now comes to smaller nodes and clusters as well.

We showed that a proper MPI processes binding policy within NUMA nodes induces significant impact for parallel application performance [35], [43]. We proposed an automatic placement scheme that gathers information about the application communication patterns during a preliminary run so as to place processes according to their communication affinities and to the hardware characteristics such as shared caches or NUMA nodes. We developed a specific algorithm (called TREEMATCH) for matching the processes to the resources in order to reduce the communication cost of the application. However, in order to be able to place the MPI processes onto the various computing cores, we need to acquire the most encompassing vision of the architecture.

The HWLOC software (see Section 5.2) answers this problem by offering a detailed knowledge of the hardware in a portable and abstracted manner. We showed that HWLOC can help popular high-performance OPENMP or MPI software [25]. Indeed, scheduling OPENMP threads according to their affinities or placing MPI processes according to their communication patterns shows interesting performance improvement thanks to HWLOC. An optimized MPI communication strategy may also be dynamically chosen according to the location of the communicating processes in the machine and its hardware characteristics.

7. Contracts and Grants with Industry

7.1. PhD thesis co-supervised with CEA/DAM

Participants: François Diakhaté, Raymond Namyst.

3 years, 2007-2010

We did set up a collaboration with the CEA/DAM (French Atomic Energy Commission, Marc PÉRACHE, Bruyère le Chatel) on the support of nuclear simulation programs (adaptive mesh) on large clusters of SMP (thousands of processors) and on Itanium2-based NUMA machines. In October 2007, François DIAKHATÉ started a PhD thesis granted by the CEA under the co-supervision of Marc PÉRACHE and Raymond NAMYST about the use of virtualization within parallel applications over clusters of multiprocessor machines.

7.2. Contract between INRIA and Myricom

Participant: Brice Goglin.

The OPEN-MX software stack was developed within an industrial contract with Myricom, Inc. (US Company building high-speed interconnect hardware and software). Myricom, Inc. provides us with high-performance networking hardware while we develop the OPEN-MX software stack to expose a high-performance MPI layer on top of any generic generic ETHERNET interfaces (see Section 6.5). The contract officially ended in 2008 and has been pursued as an informal collaboration since then.

7.3. Nvidia Professor Partnership

Participants: Cédric Augonnet, Raymond Namyst.

We established a Professor Partnership between Raymond NAMYST and Nvidia in the context of our work on scheduling on heterogeneous multicore platforms (see Section 6.6). Nvidia provided us with multiple high-end GPUs so as to support our research.

8. Other Grants and Activities

8.1. ANR projects

The National Agency for Research (ANR) launched in 2008 a program called COSINUS about design and numerical simulation for scientific research, industry and services. We wrote a research proposal called ProHMPT (*Programming Heterogeneous Multicore Processor Technologues*) which has been selected by the national committee. It was granted a three-years funding starting in 2009.

As an answer to the CONTINT 2009 ANR call for projects about Content and Interaction, we participate to a research proposal called MEDIAGPU (*Massive multimedia GPU-Based Processing*) which was granted a three-years funding starting starting in 2010. It aims at designing, modeling, and implementing new mathematical and algorithmic models to deal with large multimedia contents with a particular focus on executing them on GPUs. It involves 3D and video industrial specialists, Multimedia academic specialists, and HPC and hybrid industrial and academic specialists.

We participate to a research proposal to the ANR COSINUS program called COOP which was granted a three-year funding starting in 2010. It aims at establishing generic cooperation mechanisms between resource management, runtime systems, and application programming frameworks to simplify programming models, and improve performance through adaptation to the resources. It involves academic partners and EDF R&D.

Additionally, three projects were granted funding by the ANR program CIGC (*Calcul Intensif et Grilles de Calcul*, about the development of High Performance computing and Grids) from 2006 to 2009:

LEGO Proposing and to implementing a multiparadigm programming model (component, shared data, master-slave, workflow) comprising state of the art grid programming.

NUMASIS Designing new methods and mechanisms for an efficient scheduling of processes and a clever data distribution on NUMA platforms in the context of seismology applications.

PARA Studying and developing new optimization methods for an optimal use of the different parallelism levels for both new generation of generic processors and more specialized systems (GPU, Cell processor, APE).

8.2. NEGST (NExt Grid Systems and Techniques)

Participants: Alexandre Denis, Raymond Namyst.

3 years, january 2006 – march 2009.

This project is funded by the CNRS and Japan Science and Technology Agency and is led by Serge PETITON (INRIA Grand-Large) and Ken MIURA (National Institute of Informatics Center for Grid Research and Development).

It aims at promoting collaborations between Japan and France on grid computing technology. Following successful France-Japan workshops hosted by CNRS in Paris and NEREGI/NII in Tokyo, three important novel research issues have been identified: 1) Instant Grid and virtualization of grid computing resources, 2) Grid Metrics and 3) Grid Interoperability and Applications. The objective is to accelerate the intensive works of several research teams in these subjects in both countries. An international testbed including the French Grid'5000 project and its Japanese counterpart NEREGI is used to demonstrate and validate systems, software and applications.

A new proposal "Framework and Programming for Post-Petascale Computing (FP3C)" for the ANR-JST program is currently beeing submitted.

8.3. COST Action IC0805 ComplexHPC (Open European Network for High-Performance Computing in Complex Environments)

Participant: Emmanuel Jeannot.

4 years, may 2009– may 2013.

The goal of the Action is to establish a European research network focused on high performance heterogeneous computing in order to address the whole range of challenges posed by these new platforms including models, algorithms, programming tools and applications. The network will aim at contributing to exchange information, identify synergies and pursue common research activities, therefore reinforcing the strength of European research groups and the leadership of Europe in this field. This Action gathers more than 20 countries and 30 partners in Europe.

Emmanuel JEANNOT is the chair of this action. And we actively participate in the different working groups such as "*Efficient use of complex systems with an emphasis of computational library and communication library*"; "*Algorithms and tools for mapping and executing applications onto distributed and heterogeneous systems*" or "*Applications of hierarchical-heterogeneous systems.*"

8.4. PHC Pessoa MAE Grant

Participants: Cédric Augonnet, Olivier Aumage, Jérôme Clet-Ortega.

2 years, 2008 – 2010.

We set up a collaboration with the team of Associate Professor Salvador Abreu at University of Lisbon and University of Évora, Portugal, about the design of a constraint solving engine (such as GNU Prolog) for parallel architectures. We are working on porting the engine on top of the runtime systems provided in our team. We also plan to study the exploitation of new processing hardware such as the heterogeneous multicore Cell processor which shows some interesting potential for this specific use.

8.5. PEPPHER

3 years, 2010-2012

This project is funded under the Seventh Framework Program of the European Commission in the ICT-2009.3.6 Computing Systems theme.

It aims at providing a unified framework for programming architecturally diverse, heterogeneous many-core processors to ensure performance portability. PEPPHER will advance state-of-the-art in its five technical work areas: 1) Methods and tools for component based software 2) Portable compilation techniques 3) Data structures and adaptive, autotuned algorithms 4) Efficient, flexible run-time systems 5) Hardware support for autotuning, synchronization and scheduling.

The PEPPHER consortium consists of european research centres and universities (INRIA, Chalmers, LIU, KIT and UNIVIE), a major company (Intel) and European multi-core SMEs (Codeplay and Movidia).

8.6. Associate Team between Runtime and MPICH2 team

Participants: Brice Goglin, Guillaume Mercier, Stéphanie Moreaud, François Trahay.

Our proposal for an associate team between our group and the MPICH2 development team was accepted in December 2007 and renewed in 2008 and 2009. Thanks to this Associate Team program, visits from both sides have been scheduled throughout this year. NEWMADELEINE has been integrated as a network module in the Nemesis communication channel (which is now the default communication channel in MPICH2 since release 1.1). A paper ([36]) describing this work has been published the to the IPDPS conference. This work has triggered some reflections about the support of tag-matching capable interfaces and networks in MPICH2. This has lead to the development of a new version of the MX module that implements the support for Myrinet networks in MPICH2. Also some work regarding the integration of some mechanisms of the Open-MX software into the Nemesis channel has started during this year and the results are very promising [27]. We also integrated our HWLOC (see Section 5.2) into MPICH2 allowing its new process manager (Hydra) to take advantage of the knowledge of the underlying architecture [25].

8.7. Collaboration with the Open MPI project

Participants: Brice Goglin, Samuel Thibault.

We established a collaboration with the OPEN MPI project in the context of development of the HWLOC software (see Section 5.2) since our former `libtopology` software was merged with the OPEN MPI PLPA sub-project.

This collaboration was also informally extended to the development of high-performance intra-node communication with OPEN MPI over our KNEM driver (see Section 5.3).

8.8. PHC Sakura MAE Grant

Participants: Alexandre Denis, Guillaume Mercier, Raymond Namyst, François Trahay.

During his stay at University of Tokyo (Japan) in 2008, Raymond NAMYST has worked with Prof. Yutaka ISHIKAWA on the design of a multicore-enabled framework for communication and I/O in next generation clusters.

By comparing our respective approaches, we have realized that most of our optimization techniques are very similar in the sense that requests are gathered and arranged based on some application-specific optimization policy. The challenge is now to design a unified multicore-enabled framework to integrate and optimize both communication and remote file IO. Such a framework will form a great research vehicle to develop new optimization policies for the future multicore technologies. This research proposal has two main goals: providing a unified framework for dealing with IO and communication together with a common database of optimization strategies, and addressing a large spectrum of programming paradigms, such as multithreading, pure message-passing or hybrid programming.

Our proposal for a collaboration between our group and the group of Prof. Yutaka ISHIKAWA was accepted and has started at the beginning of 2009. Thanks to this program, several visits from both sides have been scheduled throughout this year. A preliminary integration of the YAMPII 3 MPI implementation over NEWMADELEINE was developed. Some work regarding the integration of NEWMADELEINE and PIOMAN in PGAS has also started during this year and we expect promising results.

8.9. INRIA – EDF R&D

A contract ("*accord cadre*") between INRIA and EDF R&D was signed. Preliminary meetings have taken place in 2008 to draw the main interesting points of collaboration between INRIA and EDF R&D. The Runtime project-team is part of this process and has proposed several research directions in the "Computer science for intensive simulation" topic, coordinated by Franck Cappello. A project proposal between Runtime and EDF R&D is expected to be submitted in January 2010.

8.10. INRIA-UIUC joint laboratory for Petascale Computing

The Runtime project is part of the joint laboratory that was setup between INRIA and University of Illinois Urbana-Champaign (UIUC) about Petascale Computing (http://jointlab.ncsa.illinois.edu/). Several presentations were made by Runtime members at the starter meeting in Paris in June 2009, and at the H3 meeting (Hybrid, Hierarchical, Heterogeneous) in Illinois in November 2009.

8.11. Expertise

Olivier AUMAGE was involved as an expert for the ANR institution's "international white program" (Programme Blanc International).

Raymond NAMYST serves as an expert for the following institutions:

- CEA/DAM ("scientific advisor" for the 2008-2010 period) ;
- DGRI (French Ministry of Research since 2009) ;
- GENCI (http://www.genci.fr/?lang=en, since 2009) ;
- ANR (Member of the "COSINUS" scientific committee since 2008) ;
- CEA-EDF-INRIA School technical committee (2009).

Raymond NAMYST was member of the following PhD committees: Wilfried JOUVE, Camille COTI (reviewer), Brice VIDEAU (reviewer) and Ludovic HABLOT.

Emmanuel JEANNOT was member of the thesis committee of Hala Sabbah (University of Franche-Comté) and Adrien Bellanger (INPL, Nancy).

Alexandre DENIS was member of the thesis committee of Hazem Fkaier (University Paris 13).

8.12. Committees

Raymond NAMYST was a member of the following program committees: *EuroPVMMPI 2009*, *ICPADS 2009*, *CCGRID'09*, *HPCVirt 2009* and *Parco 2009*.

Guillaume MERCIER is member of the "International Workshop on Parallel Programming Models and Systems Software for High-end Computing" *P2S2* program committee.

Alexandre DENIS is member of the *RenPar 2009* and *CFSE 2009* program committees.

Emmanuel JEANNOT is member of the program committee of the 2010 International Symposium on Cluster Computing and the Grid (CCGrid 2010). He is also member of the steering committee of the IEEE conference on cluster computing (Cluster).

Emmanuel JEANNOT is member of the steering committee and the direction committee of the ADT Aladdin-G5K and served as head of the Bordeaux site since October 2009.

8.13. Invitations

Raymond NAMYST has spent two weeks (November 2009) at the University of Illinois at Urbana-Champaign (USA). He has worked with the team of Pr Wen Mei Hwu on the optimization of parallel applications for heterogeneous architectures equipped with GPU accelerators. This collaboration is supported by the INRIA-UIUC joint laboratory (8.10).

GuillaumeMERCIER visited the Argonne National Laboratory (Illinois) for two weeks in 2009. This visit took place in the framework of the "Associate Team" MPI-Runtime between RUNTIME and the Radix Lab, responsible for the development of the MPICH2 software.

Alexandre DENIS, Guillaume MERCIER, Raymond NAMYST and François TRAHAY visited the University of Tokyo, Japan, for an aggregate duration of 6 weeks. These visits took place in the framework of the PHC Sakura between RUNTIME and the Parallel and Distributed System Software Laboratory. Several talks were given regarding the NEWMADELEINE communication library and the PIOMAN progression engine.

9. Dissemination

9.1. Reviews

Olivier AUMAGE was involved in the paper reviewing processes of the CCGrid, Cluster, EuroPVM/MPI, HPCC2009, ICPADS and RenPar conferences, and of the IEEE Parallel Computing journal.

Alexandre DENIS was involved in the paper reviewing process of the International Conference on Cluster Computing (Cluster 2009), the EuroPVM/MPI 2009 Users' Group Meeting, the IEEE International Symposium on Cluster, Cloud, and Grid Computing (CCGrid 2009), the International Conference on Parallel and Distributed Systems (ICPADS'09), the Rencontres Francophones du Parallélisme (RenPar 2009), and the Conférence Française sur les Systèmes d'Exploitation (CFSE 2009).

Brice GOGLIN was involved in the paper reviewing process of the workshop on High-Performance Interconnects for Distributed Computing (HPIDC 2009) and of the International Symposium on Parallel and Distributed Processing (IPDPS 2010).

Guillaume MERCIER was involved in the paper reviewing process of the workshop on High-Performance Interconnects for Distributed Computing (HPIDC 2009), the 2009 International Symposium on Cluster Computing and the Grid (CCGrid 2010), the Journal of Supercomputing, the International Journal of High Performance Computing Applications and the IEEE Computer (special issue on Multicore systems).

Samuel THIBAULT was involved in the paper reviewing process of the Workshop on System-Level Virtualization for High Performance Computing (HPCVirt 2009).

9.2. Seminars

Cédric AUGONNET gave seminars about STARPU at the Workshop on Massively Multiprocessor and Multicore Computers organized by INRIA (Rocquencourt, Feb. 09), at the ANR PROHMPT meeting (Bordeaux), at the UVSQ (Versailles, Jun. 09), at the "GTGPU" meeting in Bordeaux University (Bordeaux, Oct. 09) and at CEA/DAM (Bruyères-le-Chatel, Nov. 09).

Olivier AUMAGE gave a seminar at the ANR PROHMPT meeting (Bordeaux, Mar. 09) about fine grain multithreading on modern architectures.

Alexandre DENIS gave a seminar about "Managing communications in a multi-threaded context" at the NEGST workshop in Versailles (March 2009), a seminar about "High-performance runtime systems for contemporary parallel architectures" at the PAAP workshop in Kyoto (April 2009), a seminar about "Optimizing communications on clusters of multicores" at the first workshop of the INRIA–University of Illinois joint lab (Paris, June 2009).

Brice GOGLIN was invited at the STFC Daresbury Laboratory (UK) to give a seminar about high-performance message over ETHERNET with OPEN-MX (Jan. 2009). He also gave a seminar about OPEN-MX on the CISCO booth at the SuperComputing conference (Portland, Oregon, Nov. 2009).

Samuel THIBAULT gave several seminars about STARPU at the COST Complex HPC meeting in Lisbonne (Oct. 2009) and at the "CIGIL" meeting in Lille (Dec. 2009).

François TRAHAY gave two seminars about NEWMADELEINE and PIOMAN at the University of Tokyo (Japan) in Apr. 2009 and Dec. 2009.

Stéphanie MOREAUD gave a seminar about high-performance communication between MPI processes on multicore architectures at the Junior Researchers Meeting on Multicores and Multiprocessors (Paris, June 09).

9.3. Teaching

Olivier AUMAGE taught a course on "System and Middleware for Parallel and Distributed Computing" in the Master of Science at the University Bordeaux 1, in cooperation with Alexandre DENIS. He gave a course about "High-Performance Communication Supports" and a course on "Programming Languages for Parallelism" at the ENSEIRB engineering school.

Alexandre DENIS taught a course on "System and Middleware for Parallel and Distributed Computing" in the Master of Science at the University of Bordeaux 1.

Brice GOGLIN taught courses on "Operating System" and on "Parallel and Distributed Systems" at the ENSEIRB engineering school.

10. Bibliography

Major publications by the team in recent years

[1] G. ANTONIU, L. BOUGÉ, P. HATCHER, M. MACBETH, K. MCGUIGAN, R. NAMYST. *The Hyperion system: Compiling multithreaded Java bytecode for distributed execution*, in "Parallel Computing", vol. 27, October 2001, p. 1279–1297.

[2] O. AUMAGE, L. BOUGÉ, A. DENIS, L. EYRAUD, J.-F. MÉHAUT, G. MERCIER, R. NAMYST, L. PRYLLI. *A Portable and Efficient Communication Library for High-Performance Cluster Computing (extended version)*, in "Cluster Computing", vol. 5, n^{o} 1, January 2002, p. 43-54.

[3] O. AUMAGE, E. BRUNET, N. FURMENTO, R. NAMYST. *NewMadeleine: a Fast Communication Scheduling Engine for High Performance Networks*, in "CAC 2007: Workshop on Communication Architecture for Clusters, held in conjunction with IPDPS 2007, Long Beach, California, USA", March 2007, http://hal.inria.fr/inria-00127356, Also available as LaBRI Report 1421-07 and INRIA RR-6085.

[4] O. AUMAGE, G. MERCIER. *MPICH/MadIII: a Cluster of Clusters Enabled MPI Implementation*, in "Proc. 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2003), Tokyo", IEEE, May 2003, p. 26–35.

[5] O. AUMAGE, G. MERCIER, R. NAMYST. *MPICH/Madeleine: a True Multi-Protocol MPI for High-Performance Networks*, in "Proc. 15th International Parallel and Distributed Processing Symposium (IPDPS 2001), San Francisco", IEEE, April 2001, 51, Extended proceedings in electronic form only..

[6] D. BUNTINAS, G. MERCIER, W. GROPP. *Implementation and Shared-Memory Evaluation of MPICH2 over the Nemesis Communication Subsystem*, in "Recent Advances in Parallel Virtual Machine and Message Passing Interface: Proc. 13th European PVM/MPI Users Group Meeting, Bonn, Germany", September 2006.

[7] V. DANJEAN, R. NAMYST, R. RUSSELL. *Linux Kernel Activations to Support Multithreading*, in "Proc. 18th IASTED International Conference on Applied Informatics (AI 2000), Innsbruck, Austria", IASTED, February 2000, p. 718-723.

[8] B. GOGLIN. *Improving Message Passing over Ethernet with I/OAT Copy Offload in Open-MX*, in "Proceedings of the IEEE International Conference on Cluster Computing, Tsukuba, Japan", IEEE Computer Society Press, September 2008, p. 223–231, http://hal.inria.fr/inria-00288757.

[9] S. MOREAUD, B. GOGLIN. *Impact of NUMA Effects on High-Speed Networking with Multi-Opteron Machines*, in "The 19th IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS 2007), Cambridge, Massachussetts", November 2007, http://hal.inria.fr/inria-00175747.

[10] R. NAMYST. *Contribution à la conception de supports exécutifs multithreads performants*, Université Claude Bernard de Lyon, pour des travaux effectués à l'école normale supérieure de Lyon, December 2001, Habilitation à diriger des recherches.

[11] S. THIBAULT, F. BROQUEDIS, B. GOGLIN, R. NAMYST, P.-A. WACRENIER. *An Efficient OpenMP Runtime System for Hierarchical Architectures*, in "International Workshop on OpenMP (IWOMP), Beijing,China", 6 2007, p. 148–159, http://hal.inria.fr/inria-00154502.

[12] S. THIBAULT, R. NAMYST, P.-A. WACRENIER. *Building Portable Thread Schedulers for Hierarchical Multiprocessors: the BubbleSched Framework*, in "EuroPar, Rennes,France", ACM, 8 2007, http://hal.inria.fr/inria-00154506.

[13] F. TRAHAY, E. BRUNET, A. DENIS, R. NAMYST. *A multithreaded communication engine for multicore architectures*, in "CAC 2008: Workshop on Communication Architecture for Clusters, held in conjunction with IPDPS 2008, Miami, FL", IEEE Computer Society Press, April 2008, http://hal.inria.fr/inria-00224999.

[14] F. TRAHAY, A. DENIS, O. AUMAGE, R. NAMYST. *Improving Reactivity and Communication Overlap in MPI using a Generic I/O Manager*, in "EuroPVM/MPI", F. CAPPELLO, T. HERAULT, J. DONGARRA (editors), Lecture Notes in Computer Science, vol. Recent Advances in Parallel Virtual Machine and Message Passing Interface, n^{o} 4757, Springer, 2007, p. 170-177, http://hal.inria.fr/inria-00177167.

Year Publications

Doctoral Dissertations and Habilitation Theses

[15] F. TRAHAY. *De l'interaction des communications et de l'ordonnancement de threads au sein des grappes de machines multi-cœurs*, Université Bordeaux 1, 351 cours de la Libération — 33405 TALENCE cedex, November 2009, Ph. D. Thesis.

Articles in International Peer-Reviewed Journal

[16] C. AUGONNET, S. THIBAULT, R. NAMYST. *StarPU: a Runtime System for Scheduling Tasks over Accelerator-Based Multicore Machines*, in "IEEE Transactions on Parallel and Distributed Systems", 2010, Submitted.

[17] C. AUGONNET, S. THIBAULT, R. NAMYST, P.-A. WACRENIER. *StarPU: A Unified Platform for Task Scheduling on Heterogeneous Multicore Architectures*, in "Concurrency and Computation: Practice and Experience - Euro-Par 2009 Best Papers issue", 2010, Submitted.

[18] L.-C. CANON, O. DUBUISSON, J. GUSTEDT, E. JEANNOT. *Defining and Controlling the Heterogeneity of a Cluster: the Wrekavoc Tool*, in "The Journal of Systems & Software", 2010.

[19] M. FAVERGE, P. RAMET. *A NUMA Aware Scheduler for a Parallel Sparse Direct Solver*, in "Parallel Computing", 2010, Submitted.

[20] B. GOGLIN. *NIC-assisted Cache-Efficient Receive Stack for Message Passing over Ethernet*, in "Concurrency and Computation: Practice and Experience - Euro-Par 2009 Best Papers issue", 2010, Submitted.

International Peer-Reviewed Conference/Proceedings

[21] C. AUGONNET, S. THIBAULT, R. NAMYST. *Automatic Calibration of Performance Models on Heterogeneous Multicore Architectures*, in "Proceedings of the International Euro-Par Workshops 2009, HPPC'09, Delft, The Netherlands", Lecture Notes in Computer Science, Springer, August 2009, http://hal.inria.fr/inria-00421333.

[22] C. AUGONNET, S. THIBAULT, R. NAMYST, M. NIJHUIS. *Exploiting the Cell/BE architecture with the StarPU unified runtime system*, in "SAMOS Workshop - International Workshop on Systems, Architectures, Modeling, and Simulation, Samos, Greece", Lecture Notes in Computer Science, July 2009, http://hal.inria.fr/inria-00378705.

[23] C. AUGONNET, S. THIBAULT, R. NAMYST, P.-A. WACRENIER. *StarPU: A Unified Platform for Task Scheduling on Heterogeneous Multicore Architectures*, in "Proceedings of the 15th International Euro-Par Conference, Lecture Notes in Computer Science, Delft, The Netherlands", Lecture Notes in Computer Science, vol. 5704, Springer, August 2009, p. 863–874, http://hal.inria.fr/inria-00384363.

[24] F. BROQUEDIS, O. AUMAGE, B. GOGLIN, S. THIBAULT, P.-A. WACRENIER, R. NAMYST. *Structuring the execution of OpenMP applications for multicore architectures*, in "Proceedings of 24th IEEE International Parallel and Distributed Processing Symposium (IPDPS'10), Atlanta, GA", IEEE Computer Society Press, April 2010, http://hal.inria.fr/inria-00441472.

[25] F. BROQUEDIS, J. CLET-ORTEGA, S. MOREAUD, N. FURMENTO, B. GOGLIN, G. MERCIER, S. THIBAULT, R. NAMYST. *hwloc: a Generic Framework for Managing Hardware Affinities in HPC Applications*, in

"Proceedings of the 18th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP2010), Pisa, Italia", IEEE Computer Society Press, February 2010, http://hal.inria.fr/inria-00429889.

[26] F. BROQUEDIS, N. FURMENTO, B. GOGLIN, R. NAMYST, P.-A. WACRENIER. *Dynamic Task and Data Placement over NUMA Architectures: an OpenMP Runtime Perspective*, in "Evolving OpenMP in an Age of Extreme Parallelism, 5th International Workshop on OpenMP, IWOMP 2009, Dresden, Germany", Lecture Notes in Computer Science, vol. 5568, Springer, June 2009, p. 79–92, http://hal.inria.fr/inria-00367570.

[27] D. BUNTINAS, B. GOGLIN, D. GOODELL, G. MERCIER, S. MOREAUD. *Cache-Efficient, Intranode Large-Message MPI Communication with MPICH2-Nemesis*, in "Proceedings of the 38th International Conference on Parallel Processing (ICPP-2009), Vienna, Austria", IEEE Computer Society Press, September 2009, http://hal.inria.fr/inria-00390064USA.

[28] L.-C. CANON, E. JEANNOT, J. WEISSMAN. *A Dynamic Approach for Characterizing Collusion in Desktop Grids*, in "Proceedings of 24rd IEEE International Parallel and Distributed Processing Symposium (IPDPS'10), Atlanta, GA", IEEE Computer Society Press, April 2010, http://hal.inria.fr/inria-00441256/.

[29] B. GOGLIN, N. FURMENTO. *Enabling High-Performance Memory-Migration in Linux for Multithreaded Applications*, in "MTAAP'09: Workshop on Multithreaded Architectures and Applications, held in conjunction with IPDPS 2009, Rome, Italy", IEEE Computer Society Press, May 2009, http://hal.inria.fr/inria-00358172.

[30] B. GOGLIN, N. FURMENTO. *Finding a Tradeoff between Host Interrupt Load and MPI Latency over Ethernet*, in "Proceedings of the IEEE International Conference on Cluster Computing, New Orleans, LA", IEEE Computer Society Press, September 2009, http://hal.inria.fr/inria-00397328.

[31] B. GOGLIN, N. FURMENTO. *Memory Migration on Next-Touch*, in "Proceedings of the Linux Symposium, Montreal, Canada", July 2009, p. 101–110, http://hal.inria.fr/inria-00378580.

[32] B. GOGLIN. *Decoupling Memory Pinning from the Application with Overlapped on-Demand Pinning and MMU Notifiers*, in "CAC 2009: The 9th Workshop on Communication Architecture for Clusters, held in conjunction with IPDPS 2009, Rome, Italy", IEEE Computer Society Press, May 2009, http://hal.inria.fr/inria-00356236.

[33] B. GOGLIN. *High Throughput Intra-Node MPI Communication with Open-MX*, in "Proceedings of the 17th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP2009), Weimar, Germany", IEEE Computer Society Press, February 2009, http://hal.inria.fr/inria-00331209.

[34] B. GOGLIN. *NIC-assisted Cache-Efficient Receive Stack for Message Passing over Ethernet*, in "Proceedings of the 15th International Euro-Par Conference, Lecture Notes in Computer Science, Delft, The Netherlands", Lecture Notes in Computer Science, vol. 5704, Springer, August 2009, p. 1065–1077, http://hal.inria.fr/inria-00379168.

[35] G. MERCIER, J. CLET-ORTEGA. *Towards an Efficient Process Placement Policy for MPI Applications in Multicore Environments*, in "EuroPVM/MPI, Espoo, Finland", Lecture Notes in Computer Science, vol. 5759, Springer, September 2009, p. 104-115, http://hal.inria.fr/inria-00392581.

[36] G. MERCIER, F. TRAHAY, D. BUNTINAS, E. BRUNET. *NewMadeleine: An Efficient Support for High-Performance Networks in MPICH2*, in "Proceedings of 23rd IEEE International Parallel and Distributed

Processing Symposium (IPDPS'09), Rome, Italy", IEEE Computer Society Press, May 2009, http://hal.archives-ouvertes.fr/hal-00360275USA.

[37] S. MOREAUD, B. GOGLIN, D. GOODELL, R. NAMYST. *Optimizing MPI Communication within large Multicore nodes with Kernel assistance*, in "CAC 2010: The 10th Workshop on Communication Architecture for Clusters, held in conjunction with IPDPS 2010, Atlanta, GA", IEEE Computer Society Press, April 2010, Submitted.

[38] M. NIJHUIS, H. BOS, H. BAL, C. AUGONNET. *Mapping and synchronizing streaming applications on Cell processors*, in "International Conference on High Performance Embedded Architectures & Compilers, Paphos, Cyprus", January 2009.

[39] F. TRAHAY, E. BRUNET, A. DENIS. *Analysis of the impact of multi-threading on communication performance*, in "CAC 2009: The 9th Workshop on Communication Architecture for Clusters, held in conjunction with IPDPS 2009, Rome, Italy", IEEE Computer Society Press, May 2009, http://hal.inria.fr/inria-00381670.

[40] F. TRAHAY, A. DENIS. *A scalable and generic task scheduling system for communication libraries*, in "Proceedings of the IEEE International Conference on Cluster Computing, New Orleans, LA", IEEE Computer Society Press, September 2009, http://hal.inria.fr/inria-00408521.

National Peer-Reviewed Conference/Proceedings

[41] C. AUGONNET. *StarPU: un support exécutif unifié pour les architectures multicœurs hétérogènes*, in "19ème Rencontres Francophones du Parallélisme, Toulouse / France", September 2009, http://hal.inria.fr/inria-00411581.

[42] F. BROQUEDIS. *Ordonnancement de threads OpenMP et placement de données coordonnés sur architectures hiérarchiques*, in "19ème Rencontres Francophones du Parallélisme, Toulouse / France", September 2009, http://hal.inria.fr/inria-00422213.

[43] J. CLET-ORTEGA. *Une stratégie efficace pour le placement de processus en environnement multicœur*, in "19ème Rencontres Francophones du Parallélisme, Toulouse / France", September 2009, http://hal.archives-ouvertes.fr/inria-00410756.

[44] M. FAVERGE. *Vers un solveur de systèmes linéaires creux adapté aux machines NUMA*, in "19ème Rencontres Francophones du Parallélisme, Toulouse / France", September 2009, http://hal.inria.fr/inria-00416496.

[45] S. MOREAUD. *Adaptation des communications MPI intra-nœud aux architectures multicœurs modernes*, in "19ème Rencontres Francophones du Parallélisme, Toulouse / France", September 2009, http://hal.inria.fr/inria-00430021.

[46] F. TRAHAY. *Bibliothèque de communication multi-threadée pour architectures multi-cœurs*, in "19ème Rencontres Francophones du Parallélisme, Toulouse / France", September 2009, http://hal.inria.fr/inria-00410355.

Workshops without Proceedings

[47] M. FAVERGE. *A NUMA Aware Scheduler for a Parallel Sparse Direct Solver*, in "Journées Informatique Massivement Multiprocesseur et Multicoeur, Rocquencourt, France", February 2009, http://www.labri.fr/~ramet/restricted/i3m.pdf.gz.

References in notes

[48] T. ANDERSON, B. BERSHAD, E. LAZOWSKA, H. LEVY. *Scheduler Activations: Effective Kernel Support for the User-Level Management of Parallelism*, in "ACM Transactions on Computer Systems", vol. 10, n^{o} 1, February 1992, p. 53-79.

[49] P. BALAJI, H.-W. JIN, K. VAIDYANATHAN, D. K. PANDA. *Supporting iWARP Compatibility and Features for Regular Network Adapters*, in "Proceedings of the Workshop on Remote Direct Memory Access (RDMA): Applications, Implementations, and Technologies (RAIT); held in conjunction with the IEEE International Confer ence on Cluster Computing, Boston, MA", September 2005.

[50] G. CIACCIO, G. CHIOLA. *GAMMA and MPI/GAMMA on GigabitEthernet*, in "Proceedings of 7th EuroPVM-MPI conference, Balatonfured, Hongrie", Lecture Notes in Computer Science, vol. 1908, Springer Verlag, Septembre 2000.

[51] G. R. GAO, T. STERLING, R. STEVENS, M. HERELD, W. ZHU. *Hierarchical multithreading: programming model and system software*, in "20th International Parallel and Distributed Processing Symposium (IPDPS)", April 2006.