High Performance Applications on Hierarchical Shared Memory Multiprocessors

Christiane Pousa Ribeiro[1], **Márcio Castro**[2], **Fabrice Dupros**[3]
Alexandre Carissimi[4], **Luiz Gustavo Fernandes**[2], **Jean-François Méhaut**[1]

[1]INRIA Mescal Research Team,
LIG Laboratory, University of Grenoble, France

`{christiane.pousa,jean-francois.mehaut}@imag.fr`

[2]GMAP Research Team
PPGCC - PUCRS, Brazil

`{marcio.castro,luiz.fernandes}@pucrs.br`

[3]BRGM - Orléans Cedex 2, France

`f.dupros@brgm.fr`

[4]Universidade Federal do Rio Grande do Sul(UFRGS)
Porto Alegre - RS - Brazil

`asc@inf.ufrgs.br`

Abstract. *Cooperative researches between Brazilian and French universities are not recent. In the last years, these cooperations have allowed research groups to exchange knowledge and experiences in many different fields. High Performance Computing is one of these research areas in which research groups from both countries are collaborating in order to achieve meaningful results. In this scenario, cooperations between France and Brazil researchers are becoming more and more frequent. This work presents the results of some of these cooperations concerning the impact of hierarchical Shared Memory Multiprocessors on High Performance Applications.*

1. Introduction

In the last years, universities of Brazil and France have been doing cooperative researches on HPC (High Performance Computing). This cooperation has been allowing research groups to exchange knowledge and experiences. In the last two decades, as results of these cooperations, several Brazilians students have done doctoral studies in French universities. After the PhD., these students became professors and researches in Brazilian universities and have been in contact with their French university. Among these professors, we highlight Gustavo Fernandes [Fernandes 2002] and Alexandre Carissimi [Carissimi 1999] that are part of the work that we present in this paper. Nowadays, the exchange still occurs, allowing continuity of the cooperation between the two countries[1].

One of these collaborations between brazilian and french researches concerns the impact of hierarchical shared memory multiprocessors (NUMA platforms) on High Per-

[1]Christiane Pousa is under grant CAPES and Marcio Castro is under grant INRIA

formance Applications. Specifically, in these studies we have analysed the impact of different NUMA architectures on two geoscientific applications ([Dupros et al. 2009][2] and [Castro et al. 2009]). In both applications, the concept of memory affinity has been used in order to comprehend how they behave when different memory policies are applied. In this work, we present our solutions and the results we have obtained through this collaboration.

This paper is organized as follows: in section 2 we take a look at NUMA architectures and memory affinity. In Section 3 we discuss different ways to manage memory affinity on NUMA architectures. The results that were obtained with the two geoscientific applications are shown in Section 4. Finally we conclude with a short summary of our findings and highlight the future works in section 5.

2. NUMA Architectures and Memory Affinity

A NUMA (Non-Uniform Memory Access) platform is a multiprocessed system in which the processing elements are served by multiple memory levels, physically distributed through the platform. These multiple memory levels are seen by the developer as a single shared memory. NUMA platforms combine the efficiency and scalability of MPP (Massively Parallel Processing) with the programming facility of SMP (Symmetric Multiprocessor) machines. However, due to the fact that the memory is distributed between the machine nodes, the time spent to access data is conditioned by the distance between the processor and memory banks. A memory access by a given processor could be local (if data is in the processor local memory) or remote (it has to use the interconnection network to access the data) [Ribeiro and Méhaut 2008].

As these platforms are being widely used in HPC and memory access cost can be expensive, it is important to assure memory affinity on them. Memory affinity is the guarantee that processing units will always have their data close to them. Thus, to reach the maximum performance on NUMA architectures it is necessary to schedule processors and data so as the distance between them are the smallest possible [Carissimi et al. 2007]. To assure memory affinity in these platforms many different solutions were proposed by research groups. Such solutions are based in algorithms, mechanisms and tools that do memory pages allocation, migration and replication to guarantee memory affinity. However, they do not allow the use of different memory policies in the same application. Additionally, most of them must be explicitly integrated in the source code or operating system and it may be a complex task.

3. Analysis and Control of Memory Affinity

In this section we present the work that we have done to deal with memory affinity on NUMA platforms. These solutions were based on an extensive analysis of how memory affinity impacts on the performances of several applications. The main goals of ours solutions are: simplicity, portability and performance.

Before studying ways to manage memory affinity, we did several experiments considering different NUMA platforms and applications. Such experiments helped us to understand not only the importance of memory affinity management in these platforms but

[2]NUMASIS http://numasis.gforge.inria.fr/

also the relation between memory affinity techniques, NUMA platforms and applications characteristics (presented in [Ribeiro et al. 2008]). Thus, it was possible to identify what is important to consider while developing memory affinity solutions.

MAI (Memory Affinity Interface) is the first solution we have proposed to manage memory affinity on NUMAs [3]. MAI is an API (Application Programming Interface) that provides a simple way to control memory affinity on application over NUMA platforms. It simplifies memory affinity management issues, since it provides simple and high level functions that can be called in the application source code. MAI main characteristics are: (i) simplicity of use (less complex than other solutions: NUMA API, numactl etc) and fine control of memory (several variable based memory policies), (ii) portability (it works in different platforms and with different compilers) and (iii) performance (better performance than other standard solutions). A full description of MAI and some experiments made with it can be found in [Ribeiro and Méhaut 2008].

Our second solution is called MApp (Memory Affinity preprocessor). It is a preprocessor that enables an implicit control of memory affinity in NUMA platforms. MApp made optimizations in the application considering memory affinity aspects in the compile time. This preprocessor apply a memory policy on the application variables considering the NUMA platform characteristics. MApp main characteristics are: (i) simplicity of use (implicit memory affinity optimizations, no source code modifications) and (ii) portability (it works in different platforms and with different compilers).

4. Results

In this section we present an overview of the results obtained with MAI and MApp. The results were obtained from several experiments that we have done on NUMA platforms (AMD Opteron, Itanium 2 and SGI) using Geoscience applications (ICTM [Castro et al. 2009] and Ondes 3D [Dupros et al. 2009]).

ICTM (Interval Categorizer Tessellation Model) is a multi-layered tessellation model for the categorization of geographic regions considering several characteristics (relief, vegetation, climate, etc.), using information extracted from satellite images.

ONDES3D is an application for the simulation of seismic wave propagation in three dimensional geological media. It is developped by the French geological survey (BRGM [4]) and is mainly used for strong motion analysis and seismic risk assessment.

The experiments made with MAI have shown that the applications implemented with it have better performance than standard solutions. The results show average gains of 30% in the considered platforms. Additionally, similar gains have also been obtained in the platforms with different compilers (ICC, PGI and GCC). The use of MAI is less complex than others APIs and it has seven memory policies to manage memory affinity.

The first experiments with MApp have shown that the preprocessor works well in different platforms. The optimizations generated with MApp is done in compile time and it is transparent for developers. MApp have show capable to do memory affinity optimizations considering the architecture characteristics. Support for other compilers are been implemented in MApp (PGI and ICC).

[3]MAI can be download from http://mai.gforge.inria.fr/

[4]Bureau de recherches géologiques et minières - www.brgm.fr

5. Conclusions and Future Works

In this paper, we presented the research that have been done on memory affinity management by brazilian (PUCRS and UFRGS) and french (LIG and Grenoble University) researches. This cooperation lead to the developement of MAI and Mapp softwares to manage memory affinity on NUMAs. Furthermore, we have also presented some results obtained with such solutions using Geoscience applications over three NUMA platforms.

We have observed some good improvements in the applications when our solutions were applied (on average, performance gains were about 30%). Additionally, our solutions are simple to use, less intrusive and portable. This research generated as results five works that were submitted/accepted in events [Carissimi et al. 2007], [Ribeiro et al. 2008], [Ribeiro and Méhaut 2008], [Castro et al. 2009], [Pousa et al. 2009] and [Dupros et al. 2009].

The main contribution of this work has been promoting the iteration between the research groups in Brazil and France. The research theme is very important in the scope of HPC, as NUMA platforms are becoming widely used in parallel and distributed computing. As future works we highlight: improve MAI and MApp, submission of a paper concerning the MApp, and continue to work in cooperation in this research theme.

Referências

Carissimi, A. (1999). *Athapascan-0 : Exploitation de la multiprogrammation légere sur grappes de multiprocesseurs*. PhD thesis, Institut National Polytechnique de Grenoble.

Carissimi, A., Dupros, F., Mehaut, J.-F., and Polanczyk, R. V. (2007). Aspectos de Programação Paralela em arquiteturas NUMA. In *VIII Workshop em Sistemas Computacionais de Alto Desempenho*.

Castro, M., Fernandes, L. G., Pousa, C., Méhaut, J.-F., and de Aguiar, M. S. (2009). NUMA-ICTM: A Parallel Version of ICTM Exploiting Memory Placement Strategies for NUMA Machines. In *PDSEC '09: Proceedings of the 23rd IEEE International Parallel and Distributed Processing Symposium - IPDPS (to appear)*, Rome, Italy. IEEE Computer Society.

Dupros, F., Pousa, C., Carissimi, A., and Méhaut, J.-F. (2009). Parallel Simulations of Seismic Wave Propagation on NUMA Architectures. In *ParCo'09: International Conference on Parallel Computing (submitted)*, Lyon, France.

Fernandes, L. G. (2002). *Parallélisation d'un Algorithme d'Appariement d'Images Quasi-dense*. PhD thesis, Institut National Polytechnique de Grenoble.

Pousa, C., Castro, M., Fernandes, L. G., and Méhaut, J.-F. (2009). MAI: Memory Affinity Interface for NUMA Platforms. In *Euro-Par 2009 conference (submitted)*, Delft,Netherlands. Springer-Verlag-Lecture Notes in Computer Science.

Ribeiro, C., Dupros, F., Carissimi, A., Marangozova-Martin, V., Méhaut, J.-F., and de Aguiar, M. S. (2008). Explorando Afinidade de Memória em Arquiteturas NUMA. In *WSCAD '08: Proceedings of the 9th Workshop em Sistemas Computacionais de Alto Desempenho - SBAC-PAD*, Campo Grande, Brazil. SBC.

Ribeiro, C. P. and Méhaut, J.-F. (2008). Memory Affinity Interface. Technical Report RT-0359, INRIA.