Supporting Malleability in Parallel Architectures with Dynamic CPUSETS Mapping and Dynamic MPI

Márcia C. Cera¹, Yiannis Georgiou², Olivier Richard², Nicolas Maillard¹, and Philippe O.A. Navaux¹

¹ Universidade Federal do Rio Grande do Sul, Brazil
{marcia.cera,nicolas,navaux}@inf.ufrgs.br
² Laboratoire d’Informatique de Grenoble, France
{Yiannis.Georgiou,Olivier.Richard}@imag.fr

Abstract. Current parallel architectures take advantage of new hardware evolution, like the use of multicore machines in clusters and grids. The availability of such resources may also be dynamic. Therefore, some kind of adaptation is required by the applications and the resource manager to perform a good resource utilization. Malleable applications can provide a certain flexibility, adapting themselves on-the-fly, according to variations in the amount of available resources. However, to enable the execution of this kind of applications, some support from the resource manager is required, thus introducing important complexities like special allocation and scheduling policies. Under this context, we investigate some techniques to provide malleable behavior on MPI applications and the impact of this support upon a resource manager. Our study deals with two approaches to obtain malleability: dynamic CPUSETS mapping and dynamic MPI, using the OAR resource manager. The validation experiments were conducted upon Grid5000 platform. The testbed associates the charge of real workload traces and the execution of MPI benchmarks. Our results show that a dynamic approach using malleable jobs can lead to almost 25% of improvement in the resources utilization, when compared to a non-dynamic approach. Furthermore, the complexity of the malleability support, for the resource manager, seems to be overlapped by the improvement reached.

1 Introduction

Nowadays, a widely used trend for clusters of computers is to be composed by multicore machines. Furthermore, today’s parallel architectures eventually include some dynam- icity or flexibility in their resource availability. One example is the shared utilization of multicore machines. As applications have different execution times, the cores availability change dynamically. In such dynamic scenario, achieving a good resource utilization is a challenge. This paper studies alternatives to improve the resources utilization of current parallel architectures. Our motivations concern about Quality of Service advantages - better resources are used, faster work is executed and more users can be satisfied; idle cycles minimization or elimination - improvement of resources utilization can reduce idle cycles and conserve energy.

Improving resource utilization is directly related to the resource management and allocation. Resource management systems (RMS) are responsible to schedule jobs upon
the available resources and also to launch and monitor them. According to Feitelson and Rudolph [1], jobs can be: rigid, moldable, malleable or evolving. We are specially interested in malleable jobs because they can adapt themselves to resources with dynamic availability, and thus provide a better utilization of the current parallel architectures resources. However, malleable jobs demand a special treatment, with more complex allocation and scheduling policies. Thus, we investigate the complexity of treating malleable jobs and compare it with the gain/improvements of resources utilization.

The contribution of this paper is twofold: To study two different approaches that provide malleability to MPI jobs; To experiment those techniques upon a flexible resource management system in order to measure the advantages and the difficulties to support malleable jobs.

Our study uses two approaches to provide malleability: dynamic CPUSets mapping and dynamic MPI. The first one is well-adapted to multicore machines and allows any parallel job to exploit malleability, through a fine control and manipulation of the available cores. The second approach uses a dynamic MPI application, able to adapt itself to the available cluster nodes. Note that, dynamic MPI means that applications are developed to be malleable employing MPI-2 dynamic process creation and fault tolerance mechanisms.

The impact of the support of these two malleability approaches upon resource managers, is evaluated considering a specific flexible RMS called OAR [2]. The experiments were performed upon Grid5000 platform using real workload traces of a known supercomputer. Our goal is to make a close analysis of the malleable application behavior in a scenario near to the real one. The results obtained show that cluster utilization can be improved in almost 25% using both of malleability approaches, when compared with a non-malleable approach. Furthermore, the advantages obtained by a possible integration of malleable jobs upon a resource manager seem to outperform the complexities of their support.

The remainder of this paper is organized as follows. Section 2 introduces the definitions and related works about job malleability and its support in today’s resource managers. After that, Section 3 describes the two approaches to provide malleable behavior and the reasons of choosing them. Explanations about experimenting applications and their results, with CPUSets mapping and dynamic MPI, are shown in Section 4. Finally, Section 5 describes our concluding remarks and the future work perspectives.

## 2 Related Works and Definitions

In the context of resource managers or job schedulers, parallel applications are represented as jobs, which have a running time $t$ and a number of processors $p$. Feitelson and Rudolph [1] proposed four jobs classes based on the parameter $p$ and the resource manager support. **Rigid** jobs require a fixed $p$, specified by the users, for a certain $t$ time. In **moldable**, the resource manager choose $p$ at start time from a range of choices given by the users. The job adapts to $p$ that will not change at execution time. In **evolving**, $p$ may change on-the-fly as job demand, which must be satisfied to avoid crashes. In **malleable**, $p$ may also change on-the-fly, but the changes are required by the resource manager. In other words, malleable jobs are able to adapt themselves to changes in resources availability.