Development of parallel codes using PL-Grid infrastructure.

Rafał Kluszczyński¹, Marcin Stolarek¹, Grzegorz Marczak¹,², Łukasz Górski², Marek Nowicki², Piotr Bała¹

¹ ICM, University of Warsaw, Pawińskiego 5a, 02-106 Warszawa
² WMiI, Nicolaus Copernicus University, Chopina 12/18, 87-100 Toruń

email: bala@icm.edu.pl

Keywords: Grid, UNICORE, PL-Grid, parallel computing, Java, PCJ

Introduction

Parallel computing is now widely used for scientific simulations and data processing. The technical possibilities to increase performance of the single processor are stopped and software developers have to use other methods to speed up number crunching. The obvious way is use of multicore and multimode systems. Another possibility, however still no so widely used, is utilization of the power of the hardware accelerators such as GPU or FPGA cards. In this case programming is still low level and requires special programming skills and dedicated software environments.

The parallel systems are becoming widely spread but the largest ones offering thousands or even millions of cores are still expansive and require dedicated technical infrastructure such as power supplies and cooling. Therefore the largest systems are available in the dedicated computer centers equipped with the necessary facilities. Users have to access systems remotely dealing with the local setup, installed software and scheduling mechanism. The Grid, from its beginning, has been considered as possibility to solve this problem and to provide seamless access to the distributed computational resources. There has been number of different grid infrastructures used but UNICORE [1] is example of the most successful one.

The development of the parallel applications require frequent but relatively short runs necessary for the scalability and performance tests. Because such jobs have different characteristics they conflict with the long production runs focused on the generation of the results. On the other hand, on the fully loaded systems availability of the several nodes at the same time is difficult to achieve. Therefore development of the parallel codes requires special setup both in the scheduling system and grid middleware used to access resources. In this case short jobs have to be promoted and run with the reduced waiting time.

Description of solution

The PL-Grid infrastructure [2] is a good example of resources which can be used to store and process data. UNICORE is one of the middlewares which enable access to computing power and data resources in seamless and secure way. UNICORE infrastructure in PL-Grid is build based on the on European Middleware Initiative (EMI) releases, which are distributed as RPM packages. This allows for easy installation and configuration with only few additional dependencies.

PL-Grid users can access the resources using UNICORE client applications: UNICORE Rich Client (URC) and UNICORE Commandline Client (UCC). Rich Client provides users with the graphical interface for preparing tasks and managing files. It also supports graphical way of designing and running workflows.
UNICORE installation in PL-Grid allows for access to the number of multimode clusters providing necessary resources for parallel computations. Standard UNICORE installation shares job queues with the production jobs and is not optimized for short development and benchmarking parallel jobs.

Results

In this paper we present solution to the problem of execution of short computational jobs which require allocation of large number of nodes. The solution is based on the resource reservation mechanism available on the PL-Grid infrastructure. The UNICORE middleware allowed us to efficiently use reservation mechanism. The size of reservation can be adjusted, in the particular case it contains of 3 sets of 2 nodes of the same parameters, in total 6 nodes. This allows for fast access to the minimal configuration necessary for testing parallel applications. This result has been achieved by proper configuration of the middleware and infrastructure components.

In addition, the software installed on different target systems has been adjusted to allow user to run parallel jobs on different systems without modifications. The proper definition of the modules and their aliases was the key.

The solution has been validated with the development of the parallel applications created with the help of Parallel Computing in Java (PCJ) library [3].

Conclusions

Presented work clearly shows that grid technology became mature enough to offer reliable services designed to suit requirements of different scientific communities. In particular, with the use of dedicated partitions and reservations it speeds up development and testing of parallel applications.

References

1. UNICORE middleware website: http://unicore.eu
2. PL-Grid: http://www.plgrid.pl
3. PCJ library http://pcj.icm.edu.pl