Chapter 5
Speculative Scheduling of Parameter Sweep Applications Using Job Behaviour Descriptions

Attila Ulbert
Eötvös Loránd University, Hungary

László Csaba Lőrincz
Eötvös Loránd University, Hungary

Tamás Kozsik
Eötvös Loránd University, Hungary

Zoltán Horváth
Eötvös Loránd University, Hungary

ABSTRACT

The execution of data intensive Grid applications raises several questions regarding job scheduling, data migration, and replication. This paper presents new scheduling algorithms using more sophisticated job behaviour descriptions that allow estimating job completion times more precisely thus improving scheduling decisions. Three approaches of providing input to the decision procedure are discussed: a) single job description, b) multiple job descriptions, and c) multiple job descriptions with mutation. The proposed Grid middleware components (1) monitor the execution of jobs and gather resource access information, (2) analyse the compiled information and generate a description of the behaviour of the job, (3) refine the already existing job description, and (4) use the refined behaviour description to schedule the submitted jobs.

DOI: 10.4018/978-1-60960-603-9.ch005
INTRODUCTION

Resource management is one of the major tasks of Grid middleware. Resources include available computing power (i.e. CPUs), memory and secondary storage. The strategies implemented by the middleware fundamentally determine how early a job can finish its execution and provide the desired computing results. For data intensive parameter sweep applications the placement of data onto Storage Elements (SEs) and the selection of Computing Elements (CEs) have substantial impact on their completion time, therefore the combined efficiency of resource management and scheduling strategies significantly determine the performance of the Grid.

The resource management and scheduling algorithms may take into account the current state of the Grid, or statistics collected on the performance of the Grid components and applications. Some of the resource management strategies make use of sophisticated economy-based decision algorithms (Bell, Cameron, Carvajal-Schiaffino, Millar, Stockinger, & Zini, 2003), others focus chiefly on data replication, and present replica management Grid middleware (Laure, Stockinger, & Stockinger, 2005). Scheduling algorithms may apply statistical prediction methods (Gao, Rong, & Huang, 2005) (Nabrizyski, Schopf, & Weglarz, 2003), which can be used to rank the CEs by the estimated job completion time and select the optimal target CE.

Our resource management and scheduling approach is based on the realization that the completion time of a job on a CE can be determined exactly only after the given job has terminated. Furthermore, we could make perfect scheduling decisions if we were able to run the job on all possible CEs of the Grid one by one within the same circumstances, register the finishing times and run the job on the “best” CE. Obviously, such perfect decisions are not possible to be made, and we can only mimic the process of the selection of the best CE (Lőrincz, Kozsik, Ulbert, & Horváth, 2005).

In order to predict the completion time of the job the proposed scheduling strategies need to know the state of the Grid, the characteristics of the CEs and the expected resource access patterns of the job. For each job, the proposed Grid middleware services will (1) monitor the execution of the job and gather resource access information, (2) generate a compact description of the behaviour of the job, (3) use the job behaviour description to calculate the expected completion time of the job and schedule the job accordingly, and (4) refine the already existing behaviour description using the behaviour description reflecting its latest execution.

Our proposed scheduling strategies also take into consideration the effects of data replication and provide replication commands harmonising with the actual scheduling decision. For example, if the job accesses large chunks of data, it is most likely a good idea to schedule it to the Computing Element (or to a location in its neighbourhood) where the input files are available. However, if the job had to wait too long before it could be started on the chosen Computing Element, it would be worth copying the input files to another Grid component where the job can be executed earlier. In the case of jobs that are less data intensive (use less and smaller input files), the nearness of the files is not so important since the cost of the replication is very low. Furthermore, knowing the resource access patterns of the job the files can be replicated parallel to the execution of the job by fetching the necessary file fragments “just-in-time”.

RELATED WORK

Our approach focuses on the resource access of jobs; the scheduling decisions are made based on the finishing time estimations exploiting the knowledge of the behaviour of jobs.

Nabrizyski et al. (Nabrizyski, Schopf, & Weglarz, 2003) gives an excellent overview of Grid resource management. Besides presenting