Advanced Strategies for Efficient Workflow Management with Grid*W*ay[*]

J.L. Vázquez-Poletti, E. Huedo, R.S. Montero, and I.M. Llorente

Universidad Complutense de Madrid, 28040 Madrid, Spain,
WWW home page: `http://asds.dacya.ucm.es/`

Abstract. Bioinformatics is demanding the computational power offered by Grid Computing. On the other hand, a workflow management system is needed for the complex applications pertaining to this research area. In a precedent contribution, a Bioinformatics application which perforns protein clustering was ported to the Grid using the Grid*W*ay metascheduler on the EGEE production infrastructure. The reason for the Grid approach is that a single machine is subject to memory restrictions and the input database size grows everyday. Nevertheless, execution times obtained on the Grid were higher due to the overheads imposed by its highly dynamic, heterogeneous and faulty environment. In this paper we introduce some optimization strategies that will shorten execution times in the considered workflow.

1 Introduction

Bioinformatics stands among the different research areas which are profusely using Grid Computing. The complexity of the proposed problems involves the management of workflows. The workflow concept appears to satisfy the need to automate procedures in data transfer and job execution. So, a workflow management system is the one which defines, manages and executes workflows over a Grid [1].

In a previous contribution [2], a protein clustering application called *cd-hit* [3], used by the Spanish National Oncology Research Center (Centro Nacional de Investigaciones Oncológicas - CNIO)[1], was introduced for its porting to the Grid. Several workflow management systems were considered for this

[*] This research was supported by Consejería de Educación of Comunidad de Madrid, Fondo Europeo de Desarrollo Regional (FEDER) and Fondo Social Europeo (FSE), through BioGridNet Research Program S-0505/TIC/000101, and by Ministerio de Educación y Ciencia, through research grant TIN2006-02806. This work makes use of results produced by the Enabling Grids for E-sciencE project, a project co-funded by the European Commission (under contract number INFSO-RI-031688) through the Sixth Framework Programme. EGEE brings together 91 partners in 32 countries to provide a seamless Grid infrastructure available to the European research community 24 hours a day. Full information is available at http://www.eu-egee.org/.

[1] http://www.cnio.es/

task: The Directed Acyclic Graph Manager (DAGMan) [4] provided by Condor, Pegasus [5], Triana [6] standing on the Grid Application Toolkit (GAT) from GridLab [7], ICENI [8], GridAnt [9], Gridbus [10] and Grid*W*ay [11]. From these, Grid*W*ay was chosen because of its workload management and fault tolerance capabilities. On the other hand, the EGEE infrastructure was employed due to its vastness of resources as it is a production Grid.

Different results were obtained depending on the number of partitions performed to a mid-sized protein database, and a performance study was accomplished. Nevertheless, some lacks were found in the model (how the algorithm was ported onto the Grid), as it is described in Section 2. In this work we present new improvement strategies in order to minimize the execution time (see Section 3). Finally, the conclusions and the indication of future work are presented in Section 4.

2 Previous Results and Model Limitations

The Bioinformatics application called *cd-hit* performs protein clustering, in order to eliminate redundancies in a protein database. Its algorithm is represented in Figure 1 and works as described below. In the beginning, a tool called *cd-hit-div* performs a protein database division. The first division, which is the representative one, is processed by the *cd-hit* tool in order to perform a comparison with itself (represented as the A task in Figure 1). The output is then compared to te rest of partitions through the *cd-hit-2d* tool (represented as the B tasks). The first partition which results from the last operation is at this time the representative one (the A' task), and the process starts over until there aren't any more partitions. Finally, when the last comparison is performed, all the outputs of the *cd-hit* tool are merged with the *clstr_merge.pl* tool.

In the previous work, a mid-sized database with 504,876 proteins (435MB) was processed. This input database pertained to the RefSeq [2] database, provided by the National Center for Biotechnology Information (NCBI). The infrastructure employed for running the experiments belongs to the Enabling Grids for E-sciencE (Table 1).

With the purpose of porting the application to the Grid, tasks are divided in two types. In the first type, the job executes both *cd-hit-2d* and *cd-hit* over the given database division. In the second type, the job executes just *cd-hit-2d*. Both the database division and final merging are locally performed. As it can be understood from Figure 1, tasks from a certain level cannot start until the first type job from the previous level is not finished, so task dependencies must be managed in this workflow. In a workflow, the node path translated into a sequence of tasks, to which the completion time is subject, is called *critical path* [12]. In this case, the *critical path* is composed by the execution of the first type tasks (the shaded nodes in the Figure).

Considering the jobs submitted and in particular their input, both the file size and the number of the resulting tasks depend on the number of database

[2] http://www.ncbi.nlm.nih.gov/RefSeq/

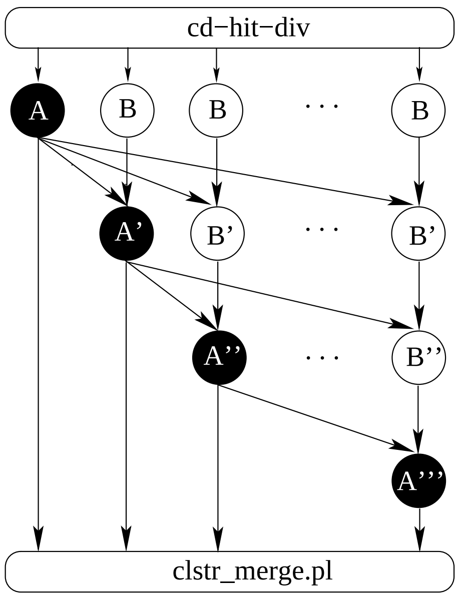

Fig. 1. The *cd-hit* algorithm. White tasks execute *cd-hit-2d* and shaded tasks execute both *cd-hit-2d* and *cd-hit*.

Table 1. Testbed resources. All DRMS are PBS.

Site		Processor	Nodes	Speed
BIFI	ES	Intel P.IV	56	3.2GHz
CESGA	ES	Intel P.III	16	500MHz
CGG	FR	Intel P.III	58	1.2GHz
CIEMAT	ES	Intel Xeon	226	3.2GHz
GRIF	FR	Intel P.IV	14	2.8GHz
JINR	RU	Intel P.D	30	2.8GHz
L.-HEP	UK	Intel P.IV	374	3GHz
PNPI	RU	Intel P.IV	60	3GHz
RAL	UK	Intel P.IV	62	2.8GHz
RALPP	UK	Intel P.III	1064	1GHz
ScotGRID	UK	Intel Xeon	6	2.8GHz
SINP	RU	Intel Xeon	94	2.8GHz

Table 2. Input file sizes and number of tasks for each database division.

DB Div.	Mean Size	Tasks
10	44MB	45
12	36.5MB	66
14	31.5MB	91
16	27.5MB	120
18	24.5MB	153
20	22MB	190
22	20MB	231

divisions (Table 2). However, the *cd-hit* and *cd-hit-2d* executable file sizes are both 1.1MB.

As can be seen in Figure 1, increasing the number of database divisions favors the parallelism level of the application, although the computation to file transfer ratio worsens. Moreover, the time a task waits in the remote queuing system is not related to the number of divisions as it only depends on the remote resource load status, which is high because the EGEE infrastructure is at production level. Therefore, the queuing time is the most significant part of the walltime, not considering transfer times, of each task in all the experiments.

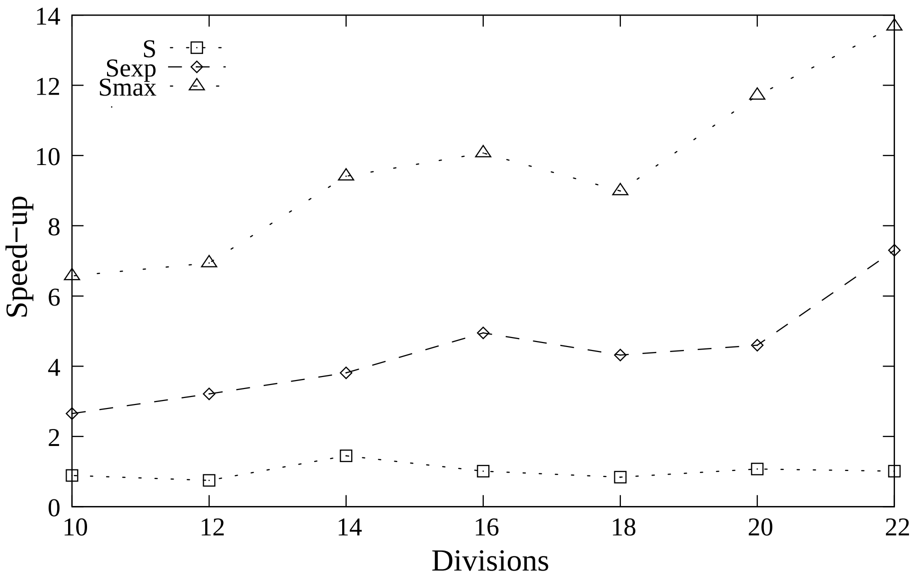

Fig. 2. Speed-up of the workflow execution for different number of database divisions.

Furthermore, we analyzed the potential speed-up that could be obtained for this kind of applications, where the level of parallelism is not constant. To this end, we consider: the speed-up (S) which is the speed gain of the workflow through the Grid approach, compared to the execution on a single machine; the speed-up calculated without considering either queue wait times or job failures (S_{exp}); and an upper bound limit (S_{max}) taking into account just the *critical path*. These three values are represented in Figure 2. The obtained speed-up is very limited, principally due to the file transfer times and the job reschedules. Job reschedules, represented in Figure 3, are due to execution errors (i.e. middleware failures) or suspension timeouts (a job waits in the remote queue more than 5 minutes).

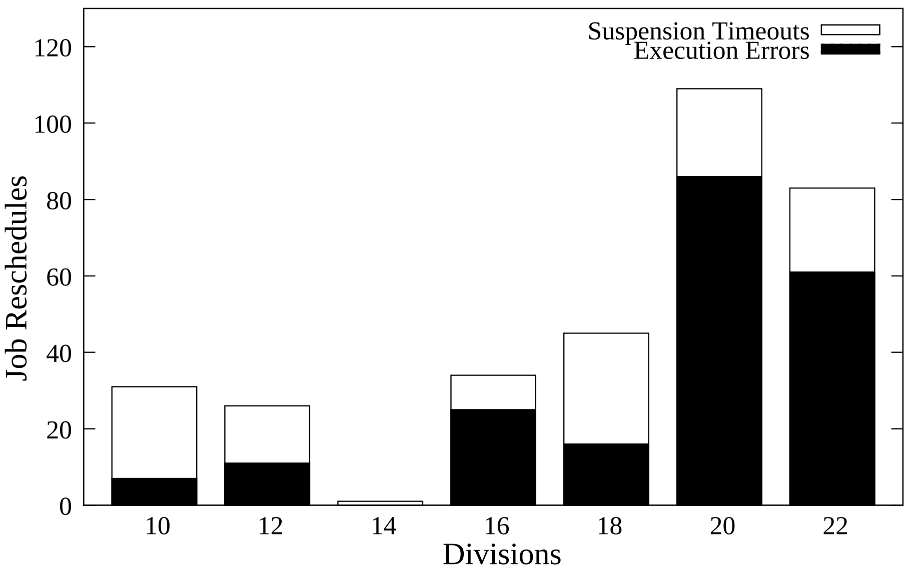

Fig. 3. Number of jobs rescheduled in each experiment.

Summarizing, these results correspond to the analysis of a Bioinformatics workflow porting to a production Grid environment. Due to memory restrictions, this kind of workflow computations cannot be accomplished in a single computer so the Grid approach is still valid, even wih these initial performance results. Anyway, Grid*W*ay is proven to be a robust and reliable workflow engine and its use is encouraged for the next steps that must be performed.

3 Proposed Strategies

The approach with the RefSeq protein database portion resulted in the acquisition of a simple model. Nevertheless, CNIO proposed the analysis of various meta-genomes, starting with the first published one from Sargasso Sea. This time, the input database contains up to 4,186,284 proteins (1.7GB). Even with a smaller database, the model needs to be revisited and optimized.

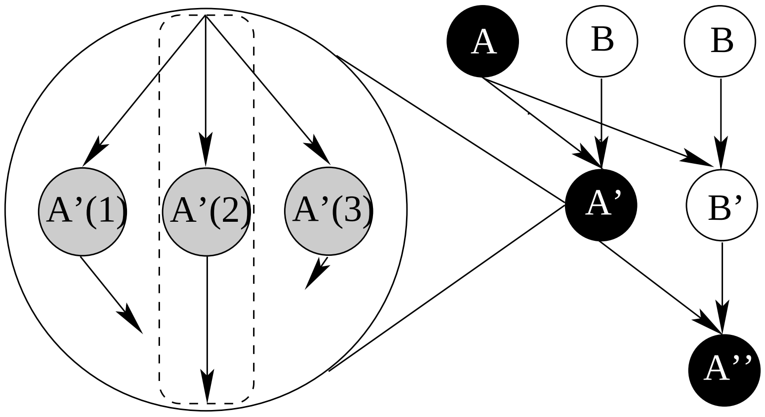

Fig. 4. The *redundancy* strategy.

We divide the optimization strategies in two main types: *redundancy* and *agglomeration*. With the first strategy described in Figure 4, *redundant* tasks are being created for specific nodes of the workflow. When one of these *redundant* task ends, the node is considered executed and the rest of copies are killed. In this way, the more submitted *redundant* tasks, the higher is the possibility for a node to end shortly as not all of its *redundant* tasks would be executed in the same resource. Variants of this strategy include the replication of all the nodes, only those from the *critical path*, and finally, tasks above a defined *blocking* threshold. The *blocking* threshold is the number of tasks that depend on a single one above which, this thask should be replicated. The number of tasks *blocked* by a single node of the workflow is calculated by:

$$b_{i,j} = \begin{cases} ST(N-i) & \text{if } i = j \\ (i-1) + ST(N-i) & \text{if } i \neq j \end{cases} \tag{1}$$

being i the column and j the level where the node is located in the workflow, represented in the example at Figure 5. N is the number of workflow levels. Finally, $ST(n)$ is the *blocking subtree* generated by a node integrating the *critical path* and can be estimated as:

$$ST(n) = \frac{n(1+n)}{2} \tag{2}$$

which is the sum of terms belonging to an arithmetic progression.

Additionaly, we propose the *agglomeration* strategy described in Figure 6, where all the tasks beyond a specific level aren't executed on the Grid but locally.

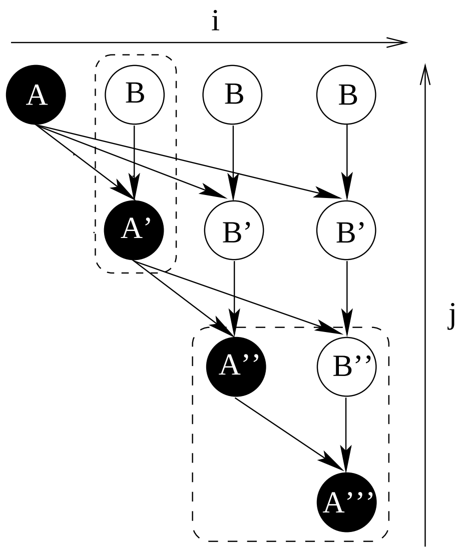

Fig. 5. In this example, B blocks A'. Then A'', A''' and B'' belong to the *blocking subtree* generated by A'.

This decision is taken when the overhead imposed by the Grid is higher than the local execution of the some tasks. This overhead can be estimated by taking the times either from the execution of similar workflows or even from the already finished tasks. Finally, the use of *redundancy* or *agglomeration* shouldn't be exclusive. A combination of both can throw interesting results as the employment of one strategy doesn't restrict the other and may provide cumulative benefits.

4 Conclusions and Future Work

In this contribution we have described a Bioinformatics application performing protein clustering that was successfully ported to the Grid. The nodes of the implemented workflow are distributed and executed among the different Grid resources in order to bypass the memory restrictions inherent to a single machine. Even if times obtained are not low enough compared to local executions, the restrictions mentioned before and the growing input size of the protein databases needed for production purposes, make Grid Computing still stand as a good solution.

At this point, the model, understood as the algorithm's implementation, needs to be optimized. Two strategies with variants were introduced in this contribution. It is our idea to implement all of them so a further study can be performed, not only with every single solution, but also combining them.

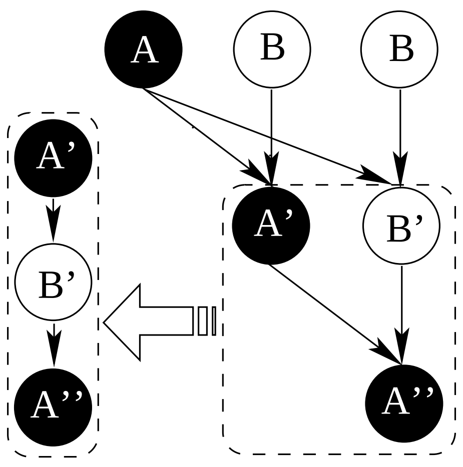

Fig. 6. The *agglomeration* strategy.

References

1. Yu, J., Buyya, R.: A Taxonomy of Workflow Management Systems for Grid Computing. Grid Computing **3** (2005) 171–200
2. Vázquez-Poletti, J.L., Huedo, E., Montero, R.S., Llorente, I.M.: Workflow Management in a Protein Clustering Application. In: Proc. 5th International Workshop on Biomedical Computations on the Grid (BioGrid'07) on the 7th IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2007), IEEE CS (2007) to appear
3. Li, W., Godzik, A.: CD-HIT: a Fast Program for Clustering and Comparing Large Sets of Protein or Nucleotide Sequences. Bioinformatics **22** (2006) 1658–1659
4. Thain, D., Tannenbaum, T., Livny, M.: Condor and the Grid. In: Grid Computing. John Wiley & Sons, Inc. (2003) 299–335
5. Deelman, E., Blythe, J., Gil, Y., Kesselman, C., Mehta, G., Patil, S., Su, M.H., Vahi, K., Livny, M.: Pegasus: Mapping Scientific Workflows onto the Grid. In: Proc. Second European AcrossGrids Conference (AxGrids 2004). Volume 3165 of Lecture Notes in Computer Science. (2004) 11–20
6. Taylor, I., Shields, M., Wang, I.: Resource Management for the Triana Peer-to-Peer Services. In Nabrzyski, J., Schopf, J.M., Węglarz, J., eds.: Grid Resource Management. Kluwer Academic Publishers (2004) 451–462
7. Kurowski, K., Ludwiczak, B., Nabrzyski, J., Oleksiak, A., Pukacki, J.: Dynamic Grid Scheduling with Job Migration and Rescheduling in the GridLab Resource Management System. Scientific Programming (AxGrids 2004 Special Issue) **12** (2004) 263–273
8. McGough, S., Young, L., Afzal, A., Newhouse, S., Darlington, J.: Workflow Enactment in ICENI. In: Proc. UK e-Science All Hands Meeting. (2004) 894–900
9. Laszewski, G.V., Amin, K., Hategan, M., Zaluzec, N., Hampton, S., Rossi, A.: Gridant: A Client-Controllable Grid Workflow System. In: Proc. 37th Annual Hawaii International Conference on System Sciences (HICSS'04). (2004)

10. Yu, J., Buyya, B.: A Novel Architecture for Realizing Grid Workflow using Tuple Spaces. In: Proc. 5th IEEE/ACM International Workshop on Grid Computing (Grid 2004). (2004)
11. Huedo, E., Montero, R.S., Llorente, I.M.: A Framework for Adaptive Execution on Grids. Software – Practice and Experience (SPE) **34** (2004) 631–651
12. Pinedo, M.: Scheduling: Theory, Algorithms, and Systems. Second edn. Prentice Hall, New Jersey, NJ (2002)