Empowering Grids with Flexibility to Cope with Uncertainties

Daniel M. Batista and Nelson L. S. da Fonseca
Institute of Computing
State University of Campinas
Avenida Albert Einstein, 1251 – 13084-971 – Campinas – SP – Brazil
Email: {batista, nfonseca}@ic.unicamp.br

***Abstract*— Grids are systems that involve coordinate resource sharing and problem solving in heterogeneous dynamic environments. In order to make grids effective it is necessary to provide them with the capability of dealing with uncertainties on resource availability and estimation of applications computational and communications demands. This paper presents a procedure for self-adjusting the resources allocated to an application and a scheduler which takes into consideration the uncertainties on the estimation of applications demands. Moreover, a brief survey of resource allocation schemes in different existing grids is provided.**

I. Introduction

Grids are typically composed of heterogeneous resources. The availability of these resources fluctuates during the execution of a grid application due to the lack of ownership of resources. Moreover, grid applications typically demand large amount of resources and they have diverse quality of service requirements. Moreover, computational and communication demands of an application are not always well known at the time of its submission. Therefore, there is a need to empower grids with the ability to cope with fluctuations of resource availability as well as with uncertainties on the requirements of the applications.

The need to consider the dynamics of grids has been recognized as one of major challenges in computer science as stated in [1]: "Manufacturers agree that the architecture of future supercomputers will be massively parallel, and as a consequence, they will need to be fault tolerant and well suited to dynamicity. So, a kind of auto-organization will also be needed, since efficient control of these very large systems will not necessarily be possible solely from the outside.".

This paper presents a procedure for self adjusting the resources allocated to a grid application and a scheduling approach to deal with uncertainties on the demands of the applications. Moreover, it provides a brief survey of the capability of existing systems to cope with these uncertainties. The aim of the present paper is to emphasize the need to adopt techniques to existing systems to empower them with the flexibility to deal with uncertainties in real grid networks.

The rest of the paper is organized as following. Section II describes a procedure for self-adjusting the resources allocated in a grid and a scheduler for dealing with uncertainties on applications requirements. Section III summarizes the resource allocation schemes in different grid systems and Section IV indicates how to integrate the self-adjustment procedure and the fuzzy scheduler into these systems. Section V draws conclusions and points out directions for future work.

II. Procedures to deal with uncertainties in grids

We have developed proactive and reactive procedures to cope with different types of uncertainties in grids. To deal with unknown demands of applications a scheduler based on fuzzy optimizations was designed and extensively tested. Moreover, a procedure for adjusting the resources allocated to an application when resource availability changes was also developed and tested.

In our previous work [2], we developed a scheduler called Integer Linear Programming with Discrete Time (ILPDT) to allocate grid resources to the tasks of an application. We then developed a novel scheduler, based on fuzzy optimization to account for the unknown demands of grid applications. These unknown demands originate from the lack of possibility to predict the amount of data generated by several e-Science applications. Moreover, imprecision on the estimation of computational demands can make ineffective schedules based on deterministic values. The fuzzy scheduler represents these uncertainties as fuzzy numbers with a triangular shape.

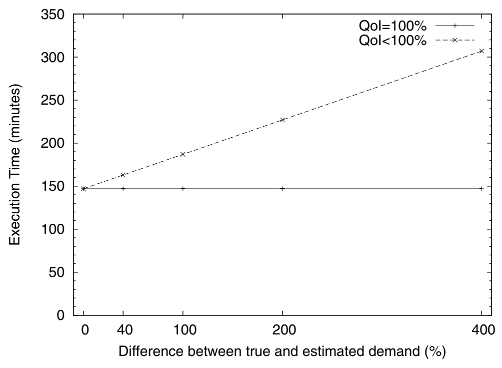

Fig. 1. Execution time when the requirement estimation is precise (QoI=100%) and when it is not (QoI$<$ 100%)

Figure 1 illustrates the execution time of an application when the estimation of the communication demand is exact (curve QoI=100%, QoI=Quality of Information) and when it is not (curve QoI$<$ 100%). Note the difference between the execution time when the estimation of communication demand is imprecise. It can be of 27% when the difference between true and estimated demand is 100% and up to 109% when the true value is 400% larger than the estimated one.

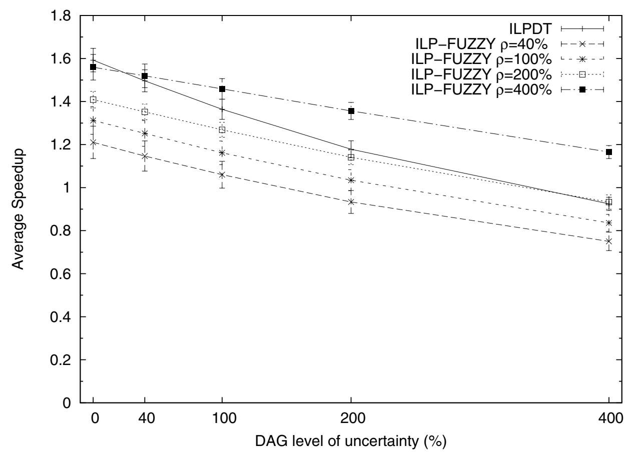

Fig. 2. Uncertainty Handling Results

Figure 2 illustrates the benefits of using a scheduler based on fuzzy optimization (ILP-FUZZY) designed for different uncertainty levels (ρ). Figure 2 plots the speedup (the ratio between the serial execution of the application and the time it takes to execute when in a parallel execution) as a function of the uncertainty on communication demands which is expressed in the application input Directed Acyclic Graph (DAG) representing the dependencies among the tasks composing the application. Figure 2 also shows the speedup of the deterministic scheduler (ILPDT) on which ILP-FUZZY is based. The ILPDT scheduler produces higher speedup values than those produced by the ILP-FUZZY schedulers designed to operate under uncertainty levels lower than 100%. ILPDT also performs better for low levels of uncertainty in the input DAG than the ILP-FUZZY scheduler designed for a 200% uncertainty level. However, when the ILP-FUZZY scheduler is designed for a high degree of uncertainty (400%), it produces higher speedup values than those produced by the ILPDT scheduler. For instance, the speedup produced by ILP-FUZZY for $\rho = 400\%$ can be $\approx 26\%$ higher than that given by the ILPDT scheduler. Moreover, the ILPDT speedup drops much faster than those given by the ILP-FUZZY scheduler as a function of the uncertainty on the DAGs weights. Extensive results reinforce the advantage of employing fuzzy schedulers when the uncertainties on the application demands are high, which is quite common in real grid networks.

To cope with fluctuations of resource availability, a procedure to re-allocate resources during a grid application execution was developed [3]. The procedure involves task migration, resource monitoring and performance prediction. Task migration transfers the code of a task and its computational context to a host where it will be executed. Monitoring keeps track of resources availability and forecasting tries to predict the fluctuation of resources availability. The following steps, shown in Figure 3, compose the proposed procedure:

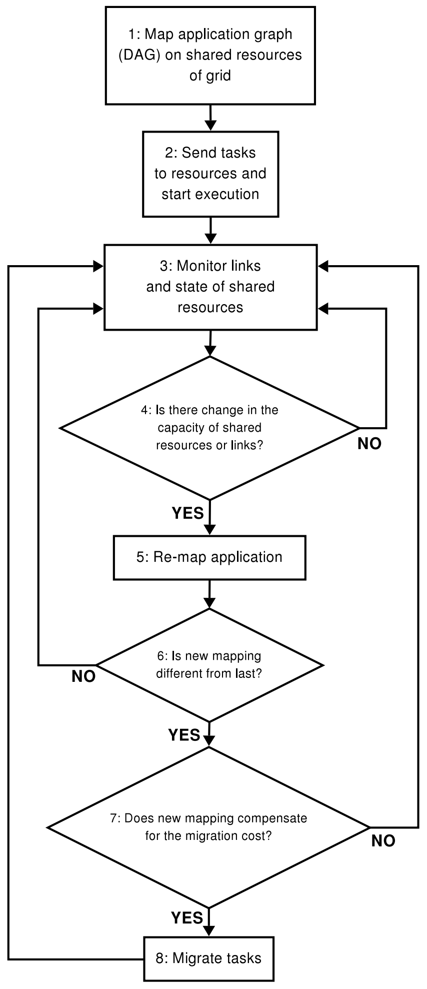
1: Map application graph (DAG) on shared resources of grid
2: Send tasks to resources and start execution
3: Monitor links and state of shared resources
4: Is there change in the capacity of shared resources or links?
NO
YES
5: Re-map application
6: Is new mapping different from last?
NO
YES
7: Does new mapping compensate for the migration cost?
NO
YES
8: Migrate tasks

Fig. 3. Procedure to Re-allocate Resources

Step 1-) Mapping the application description onto the graph describing the grid resources and production of a schedule for the beginning of task execution and data transfer; **Step 2-)** Transferring the codes and data to the hosts where the tasks will run. The execution of the tasks begins as soon as the transfer is completed; **Step 3-)** Monitoring the resources of the grid to detect any variation in the availability of hosts and links; **Step 4-)** Gathering of the data collected in Step 3 and comparison with the scenario used for previous task scheduling. If no change is detected, periodic monitoring of the grid (Step 3) is continued; **Step 5-)** Production of a new scheduling considering the current grid state. Only the unfinished tasks of the application must be scheduled; **Step 6-)** Verification of whether or not the schedule derived is the same as the current one; **Step 7-)** Comparison of the cost of the solution derived in Step 5 with the cost of the current solution. The cost of the solution derived in Step 5 should include the cost of migration of the tasks. If the predicted schedule length (application execution time) produced by the new schedule is greater than that obtained by the current schedule, monitoring the grid resources (Step 3) shall continue. The cost of migration of a task involves the time needed to complete the execution, as well as the time to transfer data. A task is only worth moving if a reduction in execution time of the entire application compensates for the cost; **Step 8-)** Migration of tasks to the designated hosts on the basis of the most recent schedule.

The proposed procedure was extensively validated using different grids and input DAGs. In all experiments the advantage of self-adjusting was clear. On the average, the execution time of the application was 25% lower when the self-adjusting procedure was employed than when it was not employed in experiments with reduction of resource availability. Moreover, when resources are added, the gain was on average 5%.

Results derived from extensive validation experiments, not described in this paper due to space limitation, encourage the adoption of both procedures in existing grids. In the next section, we briefly survey aspects of existing systems in order to identify how the procedures we proposed can be incorporated into these systems.

III. Existing Resource Allocation Techniques in Grids

This section provides a brief overview of nine existing grid systems and how resource allocation is carried out in each one of these systems.

A. NWS

The Network Weather Service (NWS) [4] is a distributed system used by several grid systems for producing short term performance predictions of computational and network resources. It involves monitoring and prediction but does not include (re)scheduling of tasks.

Current implementation of NWS collects measurements on the availability of CPU, TCP connection establishment time, end-to-end latency and available bandwidth. A set of different time series are applied to recent collected data and the one which produced the most accurate result is used to predict performance in the short term. The frequency at which probes are sent for measuring the grid resources is periodically adjusted in order to minimize the interference with the application. Moreover, it is sent with a frequency to produce representative measurement data sets. Sensors are organized in a hierarchical manner to optimize the generation of predicted values.

NWS works on small time scales and it does work well for long term predictions. Therefore, it is not proper to applications which takes hours to run. Besides that, uncertainties on applications demands are not accounted in prediction. Errors in estimating the execution time of the order of 25% were reported. Moreover, NWS produces graphics for bandwidth availability predictions in which the differentiation between predicted and measured values are not easy to evaluate.

B. GRACE

The GRid Architecture for Computational Economy (GRACE) [5] allocates resource on the basis of supply and demand dynamics. A resource broker deals with resource discovery and adaptation of resource allocation given changes in availability. It presents the grid to the users as a single computational system. The `Nimrod/G` resource broker is recommended and it was used in experiments conducted.

GRACE allows the monitoring of several parameters, including: CPU process power, memory, storage capacity and network bandwidth. Moreover, detailed measurements about software libraries access and memory pages can be generated.

Performance of GRACE was evaluated on the `EcoGrid`, a testbed which covers resources in four continents. CPU intensive applications were executed during one-hour period. A reduction of the order of 30% on the cost due to intelligent utilization of resources was observed. However, the major drawback of this proposal is the lack of flexibility to adjust the resource allocation given changes of resource availability.

C. Migration Framework for Grids

The migration framework for grids, presented in [6], adopts a policy which determines that migration should be pursued in case the gain in execution time exceeds 30% of the estimated one, which is derived by using a pre-defined model. Link bandwidth and processing power are the major metrics used in this estimation. The NWS system is employed in this framework. It was tested on GrADS testbed. Up to 70% gain in the execution time was obtained and similar gain values were found when the availability of resources increases. The major drawback of this proposal is the need of CPU time to produce estimations derived by using the adopted model.

D. GridWay

The GridWay system [7] utilizes the Globus middleware to build a system capable of adapting itself to environment changes, specially to CPU-intensive applications. Application requirements, bandwidth availability, migration overhead and processing power of potential new host are considered in the migration decision making process. Uncertainties on the application requirements and on estimations are not accounted in the decision process. Although not mandatory, the NWS is used but no forecast of resource availability is carried out. Experiments with CPU-intensive applications pointed out a gain in execution time of about 15%. Manual migration, contrary to non-migration decisions, led to performance degradation.

E. G-QoSM

The Grid-QoS Management (G-QoSM) framework [8] allocates resources based on the Service Level Agreements between users and service providers. The grid is capable of furnishing QoS and three classes similar to the classes of Internet Diffserv QoS framework: the QoS guaranteed class, the QoS controlled-load class and the best effort one. Both performance degradation and incoming new services are adopted in the reallocation of resources. The Network Resource Manager

(NRM) is employed to estimate the available bandwidth and the information gathering procedure of the Globus middleware is used to monitor the availability of processing power. Sampling of intra domain resources is more frequent than the sampling of inter domain resources. Although only link bandwidth and processing power are accounted for, G-QoSM is supposed to be able to monitor several QoS-related metrics. Uncertainty on resource availability as well as on application demands are not considered. Moreover, G-QoSM does not involve forecasting.

F. VNET and VTTIF

In the system presented in [9], the grid network is seen as an overlay network and the VNET and the Virtual Topology and Traffic Inference Framework (VTTIF) mechanisms are used for managing and for defining the grid topology, respectively. Adaptation to resource availability is carried out by dynamically changing the overlay network topology which has an initial configuration of a star. VTTIF monitors the traffic pattern passively and measurement results dictate topology changes. Only the communication pattern is considered in rearranging the overlay topology. In [9], no information is provided about initial scheduling and mechanisms to deal with application and resource availability uncertainties. A grid to evaluate the concept was built. It was observed that gains varying from 20 to 50% in execution time were achieved after the changes in topology. These changes took on the average about one minute to complete.

G. GHS

The Grid Harvest Service (GHS) [10], as the NWS system (Subsection III-A), focuses on monitoring and prediction of the grid state. The purpose of GHS is to achieve higher levels of scalability and precision of predictions than the ones obtained by the NWS system, specially for applications which run for long periods. Passive and active monitoring techniques are used to evaluate the end-to-end bandwidth availability. Neural networks are used to predict the available bandwidth and latency experienced. GHS re-schedules tasks to enhance the performance of applications. Two scheduling algorithms try to achieve such goal. One algorithm tries to minimize mean difference of execution time of tasks and the other one tries to maximize the number of tasks mapped onto a single resource. Experimental results point out the advantage of using GHS when compared to both the NWS system and to the AppLeS scheduler [13] in relation to the gain of execution time.

H. Workflow Based Approach (WBA)

The algorithm proposed in [11] is oriented to workflow based applications which are data intensive. Changes on resource availability trigger the re-scheduling of tasks but no migration of process context is carried out. The schedule produced by the Task Based Approach guides the scheduling of tasks but it does not consider dependencies in the workflow. Schedulers take into account existing processing power and bandwidth. Simulations using the NS-2 simulator indicate the need to adopt mechanisms for dealing with uncertainties on the estimation of resource availability. Execution time half of those produced when the grid does not employs WBA were found.

I. Dynamic Scheduler of Scientific Workflows

The dynamic scheduler presented in [12] is able to schedule tasks described by graph with cycles, by eliminating cycles first. The scheduler uses genetic optimizations and can be paralleled. Uncertainties on the applications demands are assumed when predicting the execution times. Migration decisions are taken in case the observed execution time exceeds the predicted value. No monitoring is employed. Experimental results indicate that a 25% reduction on the execution time is possible.

Table I displays the main aspects of resource allocation of the mentioned systems. Most of the proposals considers processing power and link bandwidth to (re)-schedule tasks. This is not sufficient for all types of grid applications, specially those requiring large amount of storage space and those requiring low end-to-end latencies such as interactive visualization.

Several proposals use the NWS system which was shown to be ineffective for applications requiring long execution times. Furthermore, existing systems are oriented to specific types of applications which implies on the lack of transparency to grid users. Moreover, only the G-QoSM system takes into account classes of services and QoS requirements in the resource allocation process. Besides that, uncertainties on applications demands are largely ignored in the proposals, which can make ineffective resource allocation/scheduling.

IV. Integration of Proposed Schemes and Existing Grids

In this section, we point out how the procedure for self-adjustment and fuzzy schedulers can be integrated with existing grid systems.

The NWS and the GHS systems can be used in the third step of the self-adjustment procedure for monitoring and prediction of resource availability. The decision of task migration based on potential performance enhancement in Step 7 can take advantage of the scheme presented in GRACE as well as that in the migration framework scheme. The dynamic scheduler presented in [12] can be integrated with the scheduling and rescheduling steps. Moreover, the NWS and the GHS schemes can be used to decrease the degree of uncertainty expressed in the ILP-FUZZY scheduler constraints, decreasing unnecessary task migration.

The procedure for self-adjustment can be used to improve the performance of existing systems. It can be combined with the procedure for adapting grid topology proposed in [9], making the topology more robust to fluctuations in resource availability.

The ILP-FUZZY scheduler can be integrated into the GRACE system so that users who define their requirements close to actual demands of the applications receive benefits in resource allocation.

TABLE I
COMPARISON OF MECHANISMS IN TERMS OF RESOURCE MONITORING, TASK MIGRATION, UNCERTAINTY HANDLING AND RESOURCE PREDICTION

Ref	Monitoring	Triggering	Reactions to change	Rescheduling	Treatment of uncertainty	Prediction
[4]	i) Active to the network ii) Active and passive to the hosts	Frequency of measurements i) adaptive to CPU ii) constant to network	–	–	Prediction of hosts state	i) Historical series ii) Applies the best series chosen with historical
[5]	Not specified (can use Nimrod/G	Rules based on the application performance	Not specified	Not specified	–	–
[6]	NWS	i) New better resources ii) Performance degradation	Migration if gain higher than 30%	Requirements matching	On applications	–
[7]	Not specified (can use NWS)	i) New better resources	Migration if gain $>$ threshold	Requirements matching	–	–
[8]	i) Hierarchical to the network ii) Globus Service to the hosts	i) Infeasibility of support QoS ii) Release of previously occupied resources iii) QoS degradation	Adjust of allocation (does not mention migration)	Requirements matching	–	–
[9]	i) Passive to the network - traffic matrix ii) No information to the hosts	Changes in traffic of the virtual topology	Topology adaptation	Not specified	Inference of virtual topology	–
[10]	i) No information on mechanisms to the network ii) Active and passive to the hosts	Rules based on link and host status	Migration to idle resources	To first host found which support the requirements	Prediction of hosts state	i) Statistics to the CPU ii) ANN to the network
[11]	Not specified	Not specified	Not specified	Workflow based heuristics	Mechanism not scalable	–
[12]	Not specified	i) Execution fault ii) Performance degradation	Migration to better resources	Cycle elimination and genetic algorithm	Ameliorate negative impact of wrong decisions employing specific models	–

Furthermore, the ILP-FUZZY scheduler can decrease the number of executions in the migration framework for grids system when the execution time does not scale linearly with the estimated input demand predictions. For such type of pattern, repeated executions do not help in understanding the application behaviour.

Moreover, the ILP-FUZZY scheduler can enhance the GridWay, the G-QoSM and the workflow based approaches when application demands are not fully known as well as when declared requirements are less demanding than the actual ones.

V. CONCLUSIONS AND FUTURE WORK

The emerging technologies of grid networks will allow diverse and highly demanding new applications which were not possible before. Resource allocation is the key to effective and efficient service provisioning. However, several challenges need to be overcome in order to make these systems flexible for service provisioning. One major challenge is to make grids general enough to efficiently accommodate a large spectrum of applications, releasing users from the need to understand the limits and capabilities of specific existing systems.

This paper presented mechanisms and procedures to deal with uncertainties on applications demands as well as resource availability. Resource allocation schemes in different existing grids were briefly surveyed and the lack of support to cope with uncertainties in theses systems were pointed out. Ongoing research investigates the introduction of the proposed mechanisms and procedures into existing grids.

REFERENCES

[1] D. E. Baz, "The rise of parallelism (and other computing challenges)," Dec 2007, International Science Grid This Week (iSGTW). Acessed at 16 Dec 2007. [Online]. Available: {http://www.isgtw.org/?pid=1000812}

[2] D. M. Batista, N. L. S. da Fonseca, and F. K. Miyazawa, "A Set of Schedulers for Grid Networks," in *SAC '07: Proceedings of the 2007 ACM Symposium on Applied Computing*. New York, NY, USA: ACM Press, 2007, pp. 209–213.

[3] D. M. Batista, N. L. S. da Fonseca, F. Granelli, and D. Kliazovich, "Self-Adjusting Grid Networks," in *Proceedings of the IEEE International Conference on Communications, 2007 – ICC'07*, Jun 2007, pp. 344–349.

[4] R. Wolski, N. T. Spring, and J. Hayes, "The Network Weather Service: a Distributed Resource Performance Forecasting Service for Metacomputing," *Future Generation Computer Systems*, vol. 15, no. 5–6, pp. 757–768, 1999.

[5] R. Buyya, D. Abramson, and J. Giddy, "A Case for Economy Grid Architecture for Service Oriented Grid Computing," in *Proceedings of the 15th International Parallel and Distributed Processing Symposium*, Abr 2001, pp. 776–790.

[6] S. S. Vadhiyar and J. J. Dongarra, "A Performance Oriented Migration Framework for the Grid," in *3rd IEEE/ACM International Symposium on Cluster Computing and the Grid(CCGRID'03)*, 2003, pp. 130–137.

[7] R. S. Montero, E. Huedo, and I. M. Llorente, "Grid Resource Selection for Opportunistic Job Migration," in *Proceedings of the 9th International Euro-Par Conference*. Springer Berlin / Heidelberg, 2003, pp. 366–373.

[8] R. Al-Ali, A. Hafid, O. Rana, and D. Walker, "QoS Adaptation in Service-Oriented Grids," in *Proceedings of the 1st International Workshop on Middleware for Grid Computing (MGC2003)*, 2003.

[9] A. I. Sundararaj, A. Gupta, and P. A. Dinda, "Dynamic Topology Adaptation of Virtual Networks of Virtual Machines," in *LCR '04: Proceedings of the 7th workshop on Workshop on languages, compilers, and runtime support for scalable systems*. New York, NY, USA: ACM Press, 2004, pp. 1–8.

[10] X. Sun and M. Wu, "GHS: A Performance System of Grid Computing," in *19th IEEE International Parallel and Distributed Processing Symposium*, 2005, http://doi.ieeecomputersociety.org/10.1109/IPDPS.2005.234. Accessed at 21/05/2006.

[11] J. Blythe, S. Jain, E. Deelman, Y. Gil, K. Vahi, A. Mandal, and K. Kennedy, "Task Scheduling Strategies for Workflow-based Applications in Grids," in *IEEE International Symposium on Cluster Computing and Grids (CCGRID'05)*, vol. 2, May 2005, pp. 759–767.

[12] R. Prodan and T. Fahringer, "Dynamic Scheduling of Scientific Workflow Application on the Grid: a Case Study," in *SAC'05: Proceedings of the 2005 ACM symposium on Applied computing*. New York, NY, USA: ACM Press, 2005, pp. 687–694.

[13] F. Berman, R. Wolski, H. Casanova, W. W. Cirne, H. H. Dail, M. Faerman, S. Figueira, J. Hayes, G. Obertelli, J. Schopf, G. Shao, S. Smallen, N. Spring, A. Su, and D. Zagorodnov, "Adaptive computing on the Grid using AppLeS," *IEEE Transactions on Parallel and Distributed Systems*, vol. 14, pp. 369–382, Apr 2003.