A Multi-agent Task Delivery System for Balancing the Load in Collaborative Grid Environment

Mauricio Paletta[1], and Pilar Herrero[2]

[1] Departamento de Ciencia y Tecnología. Universidad Nacional Experimental de Guayana. Av. Atlántico. Ciudad Guayana. 8050. Venezuela.

mpaletta@uneg.edu.ve

[2] Facultad de Informática. Universidad Politécnica de Madrid. Campus de Montegancedo S/N. 28.660 Boadilla del Monte. Madrid. Spain.

pherrero@fi.upm.es

Abstract This paper focuses on improving load balancing algorithms in grid environments by means of multi-agent systems. The goal is endowing the environment with an efficient scheduling, taking into account not only the computational capabilities of resources but also the task requirements and resource configurations in a given moment. In fact, task delivery makes use of a Collaborative/Cooperative Awareness Management Model (CAM) which provides information of the environment. Next, a Simulated Annealing based method (SAGE) which optimizes the process assignment. Finally, a historic database which stores information about previous cooperation/collaborations in the environment aiming to learn from experience and infer to obtain more suitable future cooperation/collaboration. The integration of these three subjects allows agents define a system to cover all the aspects related with load-balancing problem in collaborations grid environment.

1 Motivation and Related Work

High-performance scheduling is critical to the achievement of application performance on the computational grid [2, 5]. One of its main phases is related to the load balancing problem. An efficient load balancing strategy avoids the situation where some resources are idle while others have multiple jobs queued up. Intensive research has been done in this area and many results have been widely accepted. An interesting reading on the state of the art about this area is given by Fangpeng et al in [4]. Some intelligent agents based approaches for this problem can be found in [3, 7].

However, there are other problems to deal with: 1) the complexity of scheduling problem increases with the size of the grid; 2) the dynamic nature of the grid requires effective and efficient dynamic allocation of available resources tech-

niques [10]; 3) to manage the situation in overloaded conditions. Therefore, a cooperative and dynamic load-balancing strategy becomes highly difficult to solve effectively by taking in consideration these aspects below.

This paper presents a new MAS-based approach to deal with the necessity previously mentioned. The proposal is defined by using some components: 1) awareness management concepts defined in the CAM (Collaborative/ Cooperative Awareness Management) [8] model; 2) a heuristic technique used to optimize the resources-processes assignments needed, named SAGE (Simulated Annealing to cover dynamic load balancing in Grid Environment) [12]; 3) a Radial Based Function Network (RBFN) based learning strategy [11] used to obtain more suitable future collaboration/cooperation based on the experience; and 4) a SOA-based framework to implement Intelligent Agents (IAs) for the grid environments called SOFIA [13].

These components complement each other because CAM manages resources interaction by having information of the environment, SAGE delivers the load dynamically in the environment, SOFIA and the learning strategy will allow the system to have a more suitable cooperation.

The rest of the paper is organized as follows. Section 2 reviewed the technical backgrounds of previous researches. The MAS-based system proposed in this paper is presented in section 3. Section 4 presents some implementation and evaluation aspects. Finally, section 5 exposes the paper conclusions as well as the future work related with this research.

2 Theoretical Background

Details of CAM model and its conceptualization about awareness management can be reviewed in [7, 8]. Given a distributed environment E containing a set of resources R_i ($1 \leq i \leq N$), and a task T which needs to be solved in this environment, CAM objective is to solve T in a collaborative way. T is a set of P tuples (p_j, rq_j) ($1 \leq j \leq P$), where the p_j are the processes needed to solve the task in the system, and rq_j are requirements needed to solve each of these p_j processes. We have:

1) *Focus*(R_i): It can be interpreted as the subset of the space in which the user has focused his attention aiming of interacting with it.

2) *NimbusState*(R_i): The state of R_i in a given time. It could have three possible values: *Null*, *Medium* or *Maximum*.

3) *NimbusSpace*(R_i): The subset of the space where R_i is present. It will determine those machines that could be taken into account in the collaborative process.

4) *TaskResolution*(R_i, T) = {(p_1, s_1), …, (p_P, s_P)}: Determines if there is a service in the resource R_i, being *NimbusState*(R_i) ≠ *Null*, such that could be useful to execute T (or at least a part of it). s_j ($1 \leq j \leq P$) is the "score" to carry out p_j in the resource R_i.

5) *AwareInt*(R_i, R_j): Quantifies the degree, nature or quantity of asynchronous bidirectional interaction between R_i and R_j. It could be Full, Peripheral or Null.

6) *InteractivePool*(R_i): this function returns the set of resources interacting with the resource R_i in a given moment.

On the other hand, SOFIA focuses on the design of a common framework for IAs with the following characteristics: 1) it merges interdisciplinary theories, methods and approaches, 2) it is extensible and open as to be completed with new requirements and necessities, and 3) it highlights the agent´s learning process within the environment. The SOFIA general architecture contains four main components (Fig. 3.1-a):

1) The Embodied Agent (IA-EA) or the "body": It is a FIPA based structure [6] as it has a Service Directory element which provides a location where specific and correspondent services descriptions can be registered. It encloses the set of services related to the abilities of sensing stimuli from the environment and interacting with it.

2) The Rational Agent (IA-RA) or the "brain": This component represents the agent's intelligent part and therefore, it encloses the set of services used for the agent to implement the process associated with these abilities.

3) The Integrative/Facilitator Agent (IA-FA) or the "facilitator": It plays the role of simplifying the inclusion of new services into the system as well as the execution of each of them when it is needed. The basic function of the IA-FA is to coordinate the integration between the IA-SV and the rest of the IA components. This integration is needed when a new service is integrated with the IA and therefore registered into the corresponding Service Directory or even when an existing service is executed.

4) The IA Services or "abilities" (IA-SV): It is a collection of individuals and independent software components integrated to the system (the IA) which implements any specific ability either to the IA-EA or the IA-RA.

Related with SAGE, it is a simulated annealing based method designed to solve the dynamic load-balancing problem in grid environments. The cost function (energy) that measures the quality of each solution, the mechanism of transition of the space of solutions from t to t+1 (dynamic of the model), as well as the parameters that control the rate of cooling, can be reviewed in detail in [12].

Learning cooperation/collaborations in this context means to learn the association between the situation grid environment is in a given moment ($E + T$), and the response given to that specific situation (*TaskResolution*(R_i, T) for each resource R_i). Strategy to achieve this goal, named CoB-ForeSeer (Cooperation Behavior Foreseer), is based on Radial Based Function Network (RBFN). Previous results of this approach can be reviewed in [11]. Basic idea of CoB-ForeSeer is, one, to keep a historic data with all those collaborations that were carried out in the environment. And second, to use this data to train the RBFN model to foresee next collaborations needed. Therefore, the RBFN topology has to be defined so as to receive $E + T$ in the input layer and to produce the corresponding *TaskResolution*(R_i, T).

Next section presents the way in which CAM, SOFIA, SAGE, and CoB-ForeSeer are put together aiming to define a multi-agent task delivery system for balancing the load in collaborative grid environment.

3 Collaborative Dynamic Load Balancing

Our approach integrates SOFIA's framework with the CAM model by adapting the CAM key concepts to the objectives to be achieved as following (see Fig. 3.1-b). In this approach, the IA-SV agent manages *Focus* and *Nimbus* of each resource (as "abilities"). The IA-EA agent ("body") manages the *InteractivePool* of the collaborative grid environment. The load-balancing process is performed by the IA-RA agent ("brain") by using the SAGE method as well as the CoB-ForeSeer strategy (see details below).

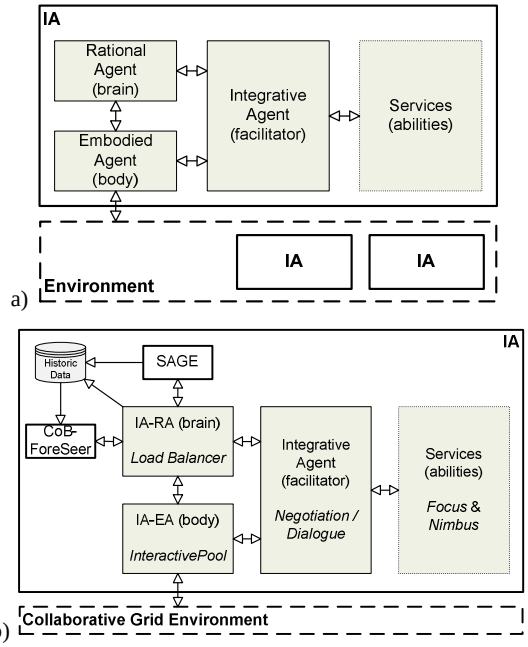

Fig. 3.1. a) The SOFIA general architecture; b) SOFIA-based system for balancing the load in collaborative grid environments.

On the other hand, in overloaded conditions (for example when *AwareInt* is *Peripheral* or *Null*) it is necessary to try to extend the *Focus* or *Nimbus* of one of the nodes so that the *AwareInt* could change to *Full*. This role is also performed by IA-RA through a negotiation process or dialogue that takes place between the IA-RA and the IA-SV. In this regard, the IA-FA agent ("facilitator") is responsible to manage this negotiation process (see details below).

The first thing IA-RA does when a load-balancing process is required in the grid environment is to obtain the corresponding *TaskResolution* "scores" s_j of each p_j by using CoB-ForeSeer. Once the answer is obtained, and depending on how it is, IA-RA takes one of three possible decisions, by using some rules as well as the current information associated with IA-EA (*InteractivePool*) and IA-SV (*Focus* and *Nimbus*):

1) Accept the processes-resources distribution given by CoB-ForeSeer.

2) Decline the answer because it is not keeping with the current situation. One of the reasons why this may happen is because the RBFN in CoB-ForeSeer is not sufficiently trained. In this case, IA-RA uses SAGE to find a better answer.

3) Decline the answer because the grid environment current conditions are overloaded. In this regard IA-RA initiates the negotiation process aiming to change the environment current conditions to obtain a new acceptable answer.

The negotiation process is performed by using a protocol defined as part of this proposal. This protocol is used by IA-RA, IA-FA and IA-SV and consists of the following dialogue:

- REQUEST (IA-RA → IA-FA): The load balancer in IA-RA is aware of an overload in any of the processes-resources assignments and it decides, with this message, to negotiate with any node an option relief.
- REQUEST (IA-FA → IA-SV): Once IA-FA receives the request from IA-RA, and as IA-FA knows what the current "abilities" (*Focus* and *Nimbus*) are, it asks for help aiming to find some node (IA-SV) that could change its abilities.
- CONFIRM (IA-SV → IA-FA): A node is confirming that it has changed its abilities and it is informing its new *Focus* and *Nimbus*.
- DISCONFIRM (IA-SV → IA-FA): A node is confirming that it cannot or it is not interested in changing its abilities.
- INFORM (IA-FA → IA-RA): Once the IA-FA receives the confirmations/ disconfirmations from the nodes, and it upgrades all the information related with the *Nimbus* and *Focus* of these nodes, IA-FA sends to IA-RA this updated information.

Next section presents some aspects related with the implementation and evaluation of the model proposed in this paper and previously defined.

4 Implementation and Evaluation

SOFIA was implemented using JADE [1]. JADE behaviour model associated to IA-RA, IA-EA, IA-FA, and IA-SV agents were implemented. Protocol for the negotiation process was implemented using the JADE ACL message class.

The evaluation of the model was carried out by generating different random scenarios in a simulated grid environment and under overload conditions. The tests were mainly focused in the negotiation process aiming to quantify the ability of the model to resolve all these specific situations. In addition, it used different configurations in the grid environment, from 5 nodes to 25 nodes varying in 5 for each test block aiming to evaluate the model capability for managing growth in the grid environment conditions.

The results indicate that the proposed model has 84% success in negotiating the way to resolve overload conditions. In those cases (16%) where the negotiation was not successful any of the nodes (IA-SV) would or could modify its *Fo-*

cus/*Nimbus* (abilities) based on the current grid environment configurations. Depending on the number of nodes in the system, the negotiation process requires different time periods for execution (from less than 1 second to about 82 seconds), some of which may not be acceptable because the dynamic of the load-balancing process.

5 Conclusions and Ongoing Work

In this paper we present a multi-agent system method to cover dynamic load-balancing problem in collaborative grid environments. This method is defined by using some researches previously developed. The method has been designed aiming to cover the requirements, a cooperative and dynamic load-balancing problem in collaborative grid environments has: effective and efficient dynamic allocation of available resources whatever the size of the grid and the current grid conditions are. The proposal model has the capability to learn from previous collaborations that were carried out in the environment to foresee future scenarios. The model has also the capability to negotiate a new grid configuration in overloaded conditions.

We are working on integrating this method in simulated grid modeling to obtain more accurate results in experiments. We are also working in reducing the time needed for the negotiation process by properly selecting the nodes with which to establish the dialogue and thus reducing the number of nodes to negotiate.

References

1. Bellifemine F., Poggi A., Rimassa G. (1999) JADE – A FIPA-compliant agent framework, Telecom Italia internal technical report, in Proc. International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAM'99), 97–108.
2. Berman F (1999) High-performance schedulers, in The Grid: Blueprint for a New Computing Infrastructure, Ian Foster and Carl Kesselman, (Eds.), Morgan Kaufmann, San Francisco, CA, 279–309.
3. Cao J, Spooner DP, Jarvis SA, Nudd GR (2005) Grid load balancing using intelligent agents, Future Generation Computer Systems, Vol. 21, No. 1, 135–149.
4. Fangpeng D, Selim GA (2006) Scheduling Algorithms for Grid Computing: State of the Art and Open Problems" Technical Report No. 2006-504, Queen's University, Canada, 55 pages, http://www.cs.queensu.ca/TechReports/Reports/2006-504.pdf.
5. Fidanova S, Durchova M (2006) Ant Algorithm for Grid Scheduling Problem, Lecture Notes in Computer Science, VIII Distributed Numerical Methods and Algorithms for Grid Computing, 10.1007/11666806, ISSN: 0302-9743 , ISBN: 978-3-540-31994-8 , Vol. 3743, 405–412.
6. Foundation for Intelligent Physical Agents (2002) FIPA Abstract Architecture Specification, SC00001, Geneva, Switzerland. http://www.fipa.org/specs/fipa00001/index.html.
7. Herrero P, Bosque JL, Pérez MS (2007) An Agents-Based Cooperative Awareness Model to Cover Load Balancing Delivery in Grid Environments, Lecture notes in computer science

2536, On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops, ISSN: 0302-9743, ISBN: 978-3-540-76887-6, Springer Verlag, Vol. 4805, 64–74.
8. Herrero P, Bosque J L, Pérez MS (2007) Managing Dynamic Virtual Organizations to get Effective Cooperation in Collaborative Grid Environments, Lecture notes in computer science 2536, On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops, ISBN: 978-3-540-76835-7, Springer Verlag, Vol. 4804, 1435–1452.
9. Jin R., Chen W., Simpson T.W. (2001) Comparative Studies of Metamodelling Techniques under Multiple Modeling Criteria, Struct Multidiscip Optim, Vol. 23, 1–13.
10. McMullan P, McCollum B (2007) Dynamic Job Scheduling on the Grid Environment Using the Great Deluge Algorithm, Lecture Notes in Computer Science, ISSN: 0302-9743, ISBN: 978-3-540-73939-5, Vol. 4671, 10.1007/978-3-540-73940-1, 283–292.
11. Paletta M., Herrero P. (2008) Learning Cooperation in Collaborative Grid Environments to Improve Cover Load Balancing Delivery, in Proc. IEEE/WIC/ACM Joint Conferences on Web Intelligence and Intelligent Agent Technology, IEEE Computer Society E3496, ISBN: 978-0-7695-3496-1, 399–402.
12. Paletta M., Herrero P. (2008) Simulated Annealing Method to Cover Dynamic Load Balancing in Grid Environment, in Proc. International Symposium on Distributed Computing and Artificial Intelligence 2008 (DCAI 08), Advances in Soft Computing, J.M. Corchado et al. (Eds.), Vol. 50/2009, Springer, ISBN: 978-3-540-85862-1, 1–10.
13. Paletta M., Herrero P. (2008) Towards Fraud Detection Support using Grid Technology, accepted for publication in a Special Issue at Multiagent and Grid Systems - An International Journal. (To be published).