A self-configuration model for execution management in Grid computing

Mohamad Fadaeian[1] and Mehdi N.Fesharaki[2]

[1] Computer Department, Science and Research Branch, Islamic Azad University, Tehran, Iran

[2] Computer Department, Science and Research Branch, Islamic Azad University, Tehran, Iran

Abstract

This paper presents a model for self-configuration of execution management in Grid environment. This model makes it unnecessary to use brokers. No broker in large systems like Grid has many advantages such as elimination of point of failures for the system, increasing accessibility, and reducing the complexity of brokers' communication. Networks formed logically between Grid's nodes have been used to present this model. Using this model, nodes able to implement submitted jobs to the system are found without using a central node autonomously by questioning neighbor nodes. This model also can improve function during runtime when submitted jobs to a node are almost similar to the former jobs. In the end of the paper, presented model is simulated by Netlogo multi-agent environment and using given scenario. Conclusions show function improvement during runtime.

Keywords: Self-configuration, Grid computing, OGSA Architecture, Execution management

1. Introduction

Grid has been introduced in 2011 according to resource sharing [1]. Main Grid services and their jobs are presented in OGSA architecture [2]. One of these services is job management which has many responsibilities like finding a resource able to do jobs submitted to the system. Brokers do this in OGSA architecture, but presenting architecture for autonomous Grid when using something similar to broker, not only becomes a point of failure in system planning but also makes the brokers' coordination highly difficult. A broker's role in a distributed system is resource information maintenance and searching for appropriate resource according to the information for submitted jobs. Using one broker in the network leads to a hot point if there were many resources. Every second lots of applications have been sending to the broker and it will make a specific decision for each of them due to its database which should be updated moment by moment using every accessible resources. Moreover, using a number of brokers makes their communication complex and affects on runtime. But in suggested method, resources autonomously search for an appropriate resource after getting the submitted job and allocate the job to that resource, so that the system can be scalable. Also the system do not need a point of failure which its unavailability can put the whole system into trouble. In this paper's presented model, an appropriate resource is found by resources themselves and in a peer to peer form. Conclusions of this simulation are shown in diverse states at the end.

2. Related Work

Grid computing system use brokers to find appropriate resource. In Globus as the implementation of OGSA architecture, GRAM component is to manage Grid resource allocation. GRAM server acts as a broker in order to job submission [3]. There are also three layers in Unicore which the highest layer has several Usites and each Usite has a number of Vsites. User must announce characteristics of the Vsite which the job is going to be done in that and as a broker; the Vsite server keeps necessary information in form of IDB (Incarnation Data Base) [4].

Self-configuration and self-adaptive systems are introduced by different researches. In [5] a self-adaptive model proposed based on biological systems. An architecture called Accord is presented for autonomous Grid computing systems in [6]. This architecture makes programs and systems' behavior and interactions determined dynamically and based on needs and states of runtime. Also two scenarios are presented to check self-management in this architecture. Parameters of these two scenarios are simulated and results are compared with the states of no self-management. A programming framework is introduced in [7] to provide autonomy in Grid computing level. This framework is expanded version of the noticed architecture in [6]. But the bug is that autonomy has been studied only in the level of function so that there is no solution providing autonomy in infrastructure level. An agent framework also is presented in [8] according to cooperation and combination of application programs in autonomous Grid computing. This framework makes agents able to discover, select and combine the elements. In addition there are some negotiation and interaction protocols for agents so they can dynamically interact with each other. But also this framework provides autonomy merely in function level.

There is an architecture introduced in [9] in order to have an adaptive scheduling on application programs. In [10], execution satisfaction degree is used for dynamic attribution of resources so that each program is constantly sending this degree to the firmware agent application. Due to users' behavior, a schedule based on adaptive anticipation is presented for tasks runtime in [11]. Former experiences of users' execution are used for this purpose. In this architecture, reconfigured architecture of resources is needed. in [12] for efficient and effective scheduling during runtime, an adaptive framework is suggested for information discovery because considering that information changes during runtime; then the information given statically by a resource at the beginning of execution is not suitable for an effective schedule during this time. So here an efficient schedule is to provide by updating resources information.

3. Self-configuration presented model for Grid computing

The purpose of Self-configuration is to define new neighbors according to execute main resource’s submitted jobs. In presented model each resource is to have a manager agent which has to manage the resource and communicate with the others. For a hypothetical Grid system' reconfiguration, if a resource was unable to do a submitted job, it has to search for another resource able to perform that job. This search is considered to be peer to peer and without central node. Different networks are defined in presented model. These networks are logical and in fact they are formed inside information and rules of the agents relevant to each resource.

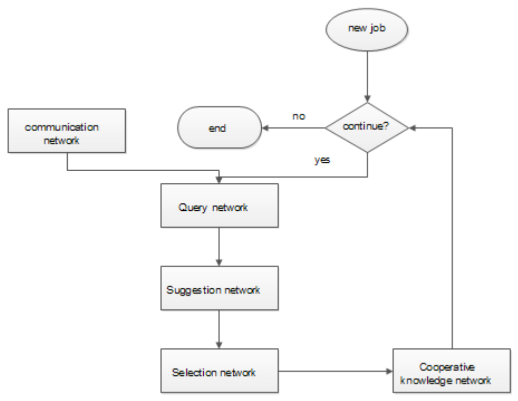


Figure 1. Self-configuration model for Grid

Figure 1 shows self-configuration model for Grid. There are 5 networks and not all of them are to be logical, but the communication network is defined as a physical one to one connection between resources. Different parts of the model are introduced bellow.

Communication network: makes a connected graph which demonstrates resources relationships in a physical form. This network does not change during configuration process but it effects on searching and selection networks and is assumed as an accidental network in the simulation.

Query network: is a directed graph which is to select neighbors who each node of them questions about answering ability to the submitted job. Selection is on the nodes so that each of them selects its query neighbors considering cooperation knowledge network and communication network. This comes out of considering physical distance between resources and also cooperation background with other resources. This background is getting formed in cooperation knowledge network. Each time of loop execution, Query network is to appear as a different network.

Suggestion network: makes a directed and weighted graph which forms based on the query network. In this network, each agent having an entry link to it checks if the resource under its management can answer submitted job of questioner agent or not. If the answer was yes and the resource under its management could answer, the agent makes a directed link with positive weight out of it to the questioner agent. If not, a directed link with negative weight is made in the same direction.

Selection network: here the wiring-network changes. In fact the goal of all components of the model is to update wiring-network which determines logical neighbors of a node. These logical neighbors have to do the job if a resource could not answer that. This network initially forms a regular graph. If a node and its logical neighbors could not answer a job, one neighbor having positive-weight and directed link out of it will be selected as a new logical neighbor from suggestion network and then makes a new. Simultaneously it chooses a logical neighbor having less cooperation and eliminates its wiring neighbor link with that. (So the number of wiring-network links stays constant.) But if no positive-weight link entered the resource agent, manager agents make resources which have one link with negative weight out of them find a resource able to answer the job.

Cooperation knowledge network: here each resource agent is neighbor with resource manager agents which have more cooperative background with them. This network emerges every time the loop of self-configuration model of Grid runs and forms a regular graph like wiring-network. Suggestion network get form stage by stage in running process of self-configuration model's loop because cooperation knowledge network shows resources having most cooperative background with noticed resource and therefore is appropriate to find a resource able to answer to another new job. But if only cooperation knowledge network was used, it is possible that wiring-network and also cooperation knowledge network divide in separate parts after some times of running mentioned loop and this would lead to many problems. One problem is that load traffic will not be balancing in different parts of the system. In addition, a new job might be submitted to a resource which is neither that resource nor its neighbors and also all resources in that separate could not execute when there is a resource able to answer in another part which cannot be found because of disconnection in destination resource system.

When a new job is added to the system, resource checks if it or its logical neighbors could do that or not. (This is done in "continuation" part of self-configuration model of Grid). If the answer was yes and that job could be done by that resource or its neighbors, it will not enter the model loop. But if they could not answer, manager agent will enter the loop to find a resource able to do that job. Then a number of resources are chosen to be asked if they can do the job or not. This is what all manager agents which they and their neighbors cannot answer do so that the query network forms. In the next stage, agents of questioned resources check if they are able to do the job or not. Through a positive/negative-weight link, the answer will be sent to the resource manager agent who the job is submitted to. In fact if they could answer, a neighborhood will be suggested to the responsible manager agent. After suggestion network forms, if a resource agent unable to do a submitted job would find an able resource (means suggested links with positive weight entered the resource manager agent), it makes a neighborhood with that. Otherwise, all questioned resource manager agents unable to execute a job should find a resource with this ability. After updating cooperation knowledge network, self-configuration model will be checked in "continuation" phase to see if there is any unresolved job or not and this loop is on until there is no submitted job unresolved.

4. Simulation

Simulation has been done in Netlogo software which is the product of NorthWestern University of America. Netlogo is designed for simulation of multi-agent environments [13]. There is a comparison between simulation environments in [14] defining Netlogo as one of the best simulation multi-agent environments.

In this simulation, one agent is considered for each resource and constant rules are defined for each agent. According to simulation scenario, there are some resources in Grid environment. Each resource is able to do determined number of various kinds of words. Some jobs have been made at the beginning of the program which each of them is allocated to one resource. Every job has a kind number and every resource accidentally can execute a number of jobs. If allocated jobs to a resource could not be done by it or its logical neighbors (means that the kind number of job was not the same with the numbers of jobs' execution ability in that resource and its logical neighbors), simulation enters the loop of model's self-configuration.

Results of simulation

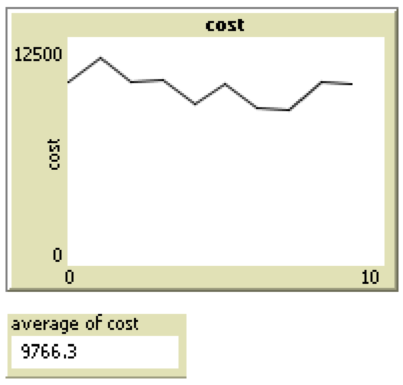
To obtain specific results out of simulation, different parameters and measures criteria should be studied. These criteria include:

- Maximum number of steps: maximum number of self-configuration model execution loop for Grid which all jobs are submitted to resources able to execute them, is called number of steps.
- Average number of steps: jobs are allocated to the appropriate resources in a number of self-configuration model execution loops for Grid. The average number of self-configuration model loop for all jobs is another measurement criterion.
- Cost: model execution cost is considered as the total number of queries in each time of running program. That is because the less the number of queries are, the less resources were involved to execute jobs.

Through results of simulation with fixed parameters

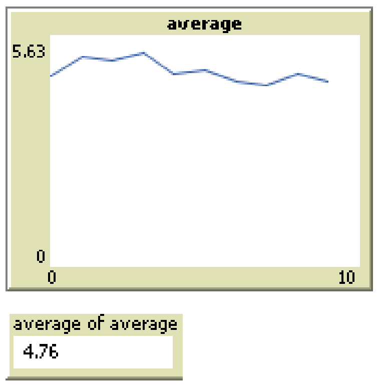
- Number of resources: 300
- Number of jobs: 200
- Number of job kinds: 30
- Ability of execution the number of job kinds: 1
- Number of selected neighbors for query: 2

Variable parameter β is given by averaging 10 times running the simulation. At first variable β is assigned zero and then 0.8.



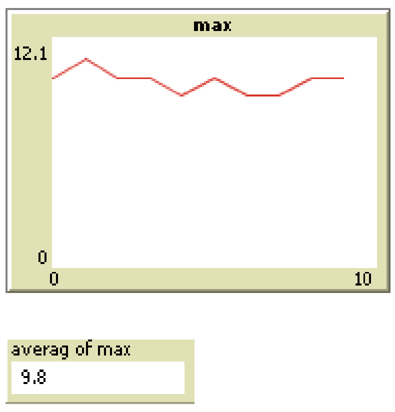
average of cost
9766.3

Fig2. A. Cost amount



average of average
4.76

Fig2. B. Average step amount



averag of max
9.8

Fig2. C. Maximum step amount
(β-0)

Part A in figure 2 shows the diagram of cost and average amount for 10 times running. The diagram of average number of steps and average amount in 10 times running is presented in part B and then, maximum number of steps and its average amount again in 10 times running are given in part C. Figure 3 is to compare measurement criteria under the effect of parameter β with the value of 0.8 and

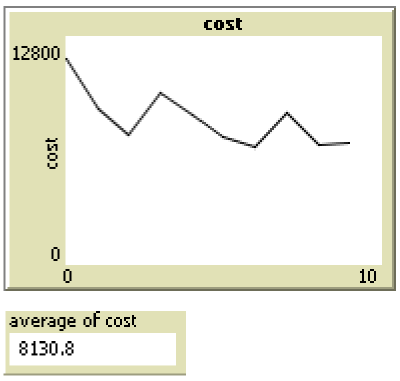


Fig3. A. Cost amount

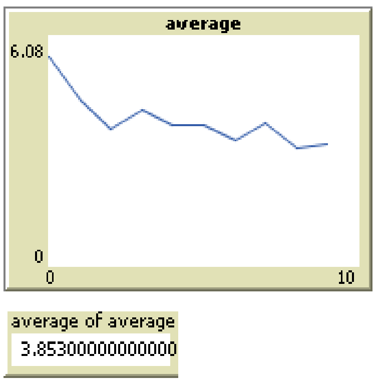


Fig3. B. Average step amount

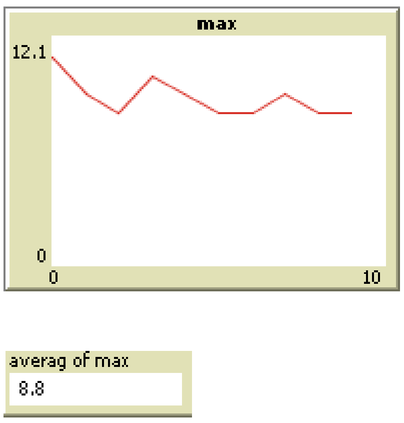


Fig3. C. Maximum step amount
(β-0.8)

in twice 10 times running the simulation program with 10 times running in former stage.

5. Conclusion

In this paper a model for Grid self configuration with the approach of OGSA execution management ability got presented. Because of complexity and costly Grid management due to high number of resources and also various firmwares for Grid, using autonomous computing in Grid management seems to be necessary. After that, a model was presented for Grid self-configuration which showed its ability to do jobs without using brokers and central models. No brokers in system, made model capable from many aspects including: scalability, cooperation between Grids and various firmwares, and also no need for point of failures in system design. Simulation by Netlgo showed that the more specific resources are (the more similar the number of submitted jobs to a node is) the higher efficiency will presented model have.

References

[1] Ian Foster, Carl Kesselman, Steven Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *International Journal of High Performance Computing Applications*, vol. 15, no. 3, pp. 200-222, august 2001.

[2] Ian Foster, H.kishimoto, A.savva, "http://forge.gridforum.org/projects/ogsa-wg," 24 july 2006. [Online].

[3] I. Foster, "Globus Toolkit Version 4: Software for Service-Oriented Systems," in *IFIP International Conference on Network and Parallel Computing*, verlag, 2006.

[4] Sergio Andreozzi, Laurence Field, Balazs Konya, "http://forge.ogf.org/sf/projects/glue-wg," 3 march 2009. [Online].

[5] Touraj Banirostam, Mehdi N. Fesharaki, "MMBSM: A Meta Model for Biological System Modeling," *IJCSI International Journal of Computer Science*, vol. 9, no. 5, pp. 140-145, 2012.

[6] Hua Liu, Viraj Bhat, Manish Parashar, and Scott Klasky, "An Autonomic Service Architecture for Self-Managing Grid Applications," in *The 6th IEEE/ACM International Workshop on Grid Computing*, Piscataway, NJ, USA, 2005.

[7] Hua Liu, Manish Parashar, "Accord: A Programming Framework for Autonomic Applications," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transaction on*, vol. 36, no. 3, pp. 341-352, 2006.

[8] Zhen Li, Manish Parashar, "Enabling Dynamic Composition and Coordination for Autonomic Grid Applications using the Rudder Agent Framework," *Journal of Thr Knowledge Engineering Review*, vol. 21, no. 3, 2006.

[9] Ahmed Mohamed A. Ghanem, Ahmed I. Saleh, Hesham Aarafat Ali, "high performance adaptive framework for scheduling grid workflow application," Hamburg, 2010.

[10] hao liu, soren aksel sorensen, "on-line feedback-based automatic resource configuration for distributed application," *Cluster Computing: The Journal of Networks, Software Tools, and Applications, Springer Science+Business Media B.V.*, vol. 13, p. 397–419, 2010.

[11] Christian Glasner, JensVolkert, "Adaps – A three-phase adaptive prediction system for the run-time of jobs based on user behaviour," *Journal of Computer and System Sciences,* vol. 7, p. 244–261, 2011.

[12] Guiyi Wei, Yun Ling , Athanasios V. Vasilakos , Bin Xiao , Yao Zheng, "PIVOT: An adaptive information discovery framework for computational grids," *Information Sciences,* vol. 180, p. 4543–4556, 2010.

[13] "http://ccl.northwestern.edu/netlogo/," Center for Connected Learning and Computer-Based Modeling, Northwestern University, 1999. [Online].

[14] CM Macal, MJ North, "Tutorial on agent-based modelling and simulation," *Journal of Simulation*, vol. 4, no. 3, pp. 151-162, 2010.