On the Design of Job Scheduling Strategy Using Agent Replication for Computational Grids

T. Altameem,

Dept. of Computer Science, RCC, King Saud University,
P.O. Box: 28095 – 11437 Riyadh-Saudi Arabia.

Summary
This paper is to consider the scheduling of jobs to the resources in grid networks. In this paper, a scheduling system for computational grids with a mobile-agent based method is proposed. The main contribution of this work is to maximize throughput and minimize the entire response time of executing user applications. The proposed system is compared with a scheduling system that uses a recent scheduling algorithm in terms of response time and grid load. The simulation experiments show that the proposed system can lead to performance improvements.

Key words:
Agents, Mobile agents, Resources scheduling, Response time, Grid load, Speedup.

1. Introduction

Grid computing is applied to many fields, such as scientific computing, data management, system integration, etc. one of the most important problems in grid computing is how to assign grid resources to applications tasks reasonably. Effective resource scheduling is a challenging issue in any grid computing system. This is due to the dynamic, heterogeneous and autonomous behavior of the grid. The grid has no local resources and therefore does not have control over them. Also, the grid has no control over the set of tasks submitted to it [16].

Grid applications are becoming increasingly network dependent with more demanding requirements in area such as data access or interactivity. The specific kind of tasks that request computation are usually referred to as jobs and are dependent on storage, network capacity and computation. When a job is submitted to be executed in the grid, it is delivered to the Grid Scheduling System (GSS). The GSS has the responsibility to select a suitable resource and then control the job execution. The selection of that resource depends on a matchmaking process between jobs submitted and available resources in the grid [30].

In [26], it had stated that effective scheduling of jobs to resources of a grid is complicated by the geographical distance, the different administrative domains with different local policies, the size and heterogeneity of the grid and the grid dynamism. Hence, batch scheduling techniques that assume constant availability and map out the entire schedule before running jobs may not be successful. Scheduling algorithms have to work with the assumption that the information used for mapping jobs to resources will quickly be out of date. These complications lead to a need to an effective and efficient scheduling algorithm to be employed [26].

In this paper, a system for scheduling jobs in computational grids with an agent-based method is presented. The main idea of this method is dispatching multiple mobile agents to different domains at the same time to execute different jobs in a parallel fashion. The main objective of the proposed system is to minimize the response time.

The rest of this paper is structured as follows. Section 2 investigates currently agent-based scheduling systems. Section 3 provides a brief description of grid scheduling. Section 4 provides a brief description of agents and mobile agents. In Section 5, the structure and the operation of the proposed multiagent-based system is presented. Section 6 describes the simulation environment, augments some results and discusses the performance of the system. Section 7 concludes the paper.

2. Related work and scope

Firstly, we review the related work of job scheduling in grid computing environments. Then, we introduce our scope for solving the problem of job scheduling in grids.

2.1 Related work

There is a tremendous effort of research has been done to achieve algorithms to efficiently schedule resources in grids. R. S. Chang, J. S. Chang and P.S. Lin [32] presented a scheduling algorithm that simulates the behavior of ants called Balanced Ant Colony Optimization (BACO). They assumed each job is an ant and the algorithm sends the ants to search for resources. The main objective of their algorithm is to balance the entire system

Manuscript received March 5, 2011
Manuscript revised March 20, 2011

load while trying to minimize the response time of a given set of jobs.
In [31], G. Ali, N. A. Shaikh and Z. A. Shaikh presented a framework that integrates grid systems and agent systems to perform computations in the grid. Their framework is platform independent and can work over Linux, UNIX and Windows operating systems.
P. Huang and et al [16] presented a scheduling method that take into account historical grid trade and dynamic variation of the grid. The method is a combination of static scheduling and dynamic adaptation using reinforcement learning technique.
T. Altameem and M. Amoon [1] presented an agent-based approach for dynamic adjustment of scheduled jobs in grids. Their approach employs mobile agents for achieving the adjustment services.

2.2 Scope

The scope of this paper is to design a scheduling system for grids and to evaluate its performance. Within this scope, the main contribution is introducing a system with a scheduling strategy that depends on using mobile agents. The scheduling strategy depends on the response time when selecting a resource to execute a certain job. The proposed system is compared with the scheduling algorithm in [7].

3. Proposed System

Most of the grid scheduling systems contains the same set of modules. These modules include a Portal module, through which users can submit the jobs to the system; a Scheduler module, which assigns jobs received from users to grid resources; Information server module which collects information about the state of the grid resources. The information for each resource includes resource speed, resource type, bandwidth between the scheduler system and the resource, and current load of the resource. The scheduling decisions taken by the Scheduler module depends on the information collected by the Information Server module about the grid resources. The components of the proposed system are shown in Fig. 1.

4.1 The System's Operation

The main purpose of the proposed system is to transparently locate, monitor, and manage resources in the grid. Also, the system can assign jobs of user applications to the most suitable resources in the grid. The system mainly depends on a set of mobile agents. These agents are distributed into the sites of the grid carrying jobs to be executed. The system is characterized by the variables in Table 1.
Users can submit their jobs through Grid Portal. The scheduler receives user jobs along with its information from the Grid Portal. Job information includes job number, job type, job size and QoS requirements, such as the deadline to complete its execution, the number of required resources and the type of these resources. The scheduler assigns each job to the resource that can execute the job with the lowest response time.

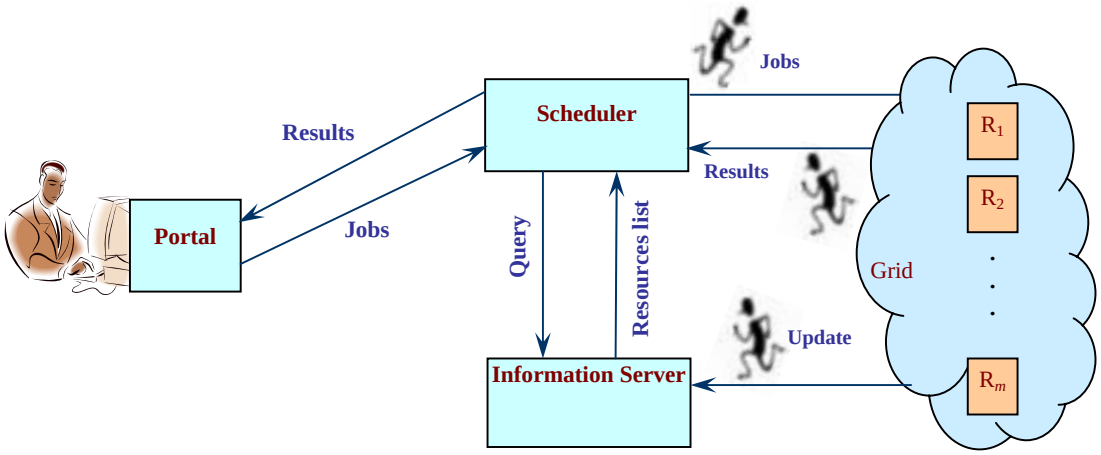

Fig. 1. Architecture of the Proposed System.

Table 1

Characteristics of the proposed system.

Variabl	Description
n	The number of jobs to be scheduled
m	The number of available resources
TR_{ij}	The response time of a resource j for a job i
TM_{ij}	The job transmission time between the GS to the resource j
TS_{ij}	The job service time on the resource j
TW_{ij}	The waiting time of the job i in the queue of the resource j
L_i	The size of a given job i
BW_{ij}	The bandwidth between the GS and the resource j on which the job i can be executed
SR_i	The service time needed for job i
S_j	The speed of the resource j

The scheduler asks the Information server module for a list of suitable resources for the job. After obtaining the list, it computes the response time of each resource in the list when executing the job.

The response time of a job includes transmission time of the job with its data, waiting time of the job in the resources queue and the job service time on the resource. The response time of a resource j for a job i is defined by:

$$TRij = TMij + TWij + TSij \qquad (1)$$

The TMij is the waiting time of the job i in the queue of the resource j. It can be estimated by the sum of all service times of jobs in the waiting queue of the resource j that have been arrived to before arriving of job i. It can be defined by:

$$TMij = \frac{L_i}{BW_{ij}} \qquad (2)$$

The TWij can be defined by:

$$TWij = \sum_{x=1}^{QueueLength} TS_{xj} \qquad (3)$$

The TSij can be defined by:

$$TSij = \frac{SR_i}{S_j} \qquad (4)$$

After that, the scheduler sorts the list of the suitable resources for each job in an ascending order according to the response time of each resource. The scheduler assigns the list of resources of each job to a mobile agent. For example, the list that will be assigned of job j1 will be J = {jx | x = 1, 2, 3,…, m} with response times TR = {TR1x | x = 1, 2, 3,…, m }. :

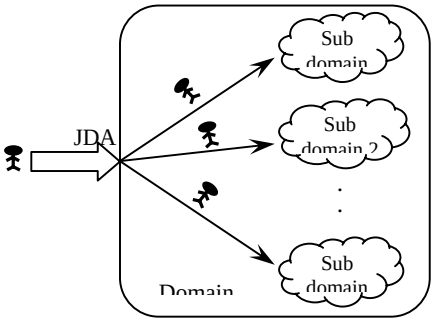

Fig. 2. JDA replication.

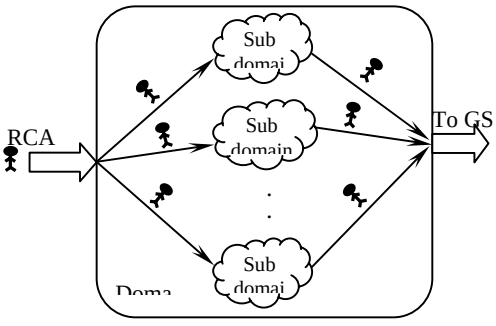

Fig. 3. RCA replication.

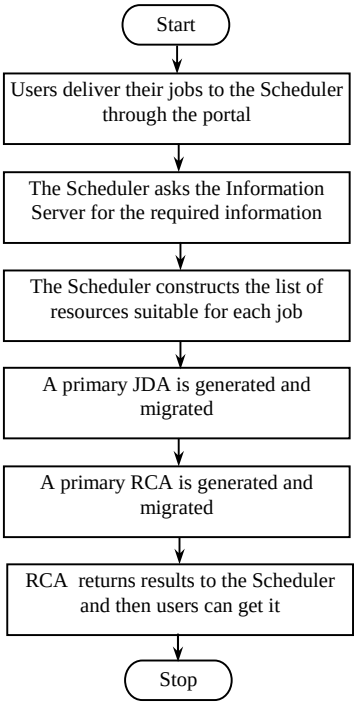

Fig. 4. The operation of the proposed system.

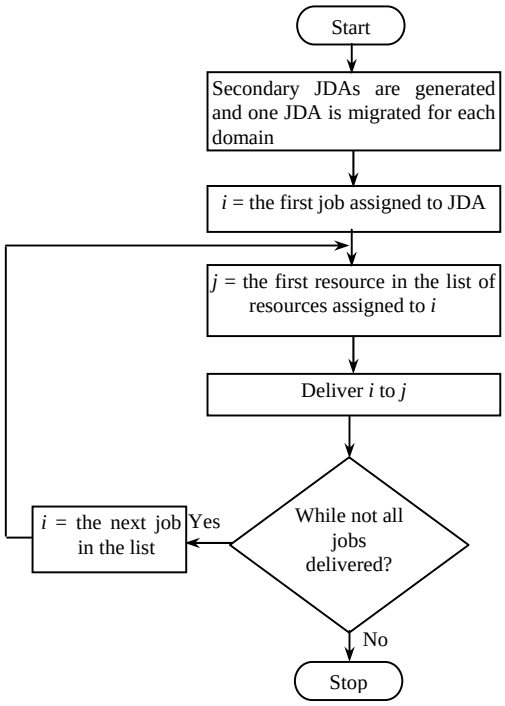

Fig. 5. JDA logic.

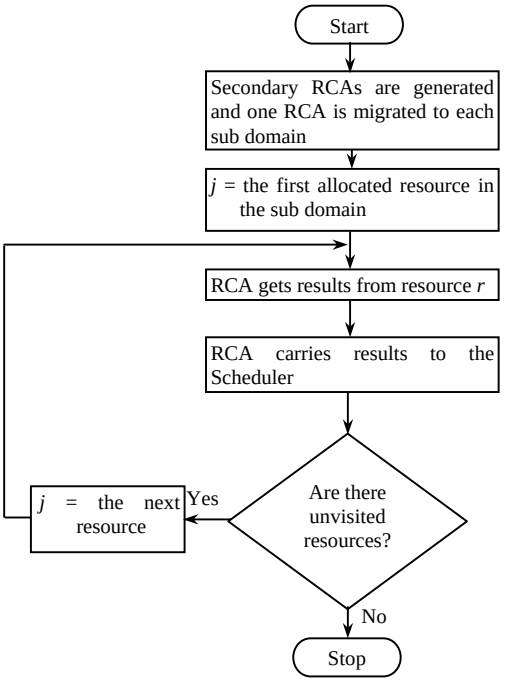

Fig. 6. RCA logic.

Two types of mobile agents can be generated by the scheduler. The first one is called the Job Distribution Agent (JDA) which is responsible of delivering jobs to their assigned grid resources. The second agent is called Result Collector Agent (RCA) which is used to collect results of executed jobs from resources.
The JDAs are sent to the assigned grid resources carrying jobs to be executed. Along with the job, each agent must carry the list of resources constructed by the scheduler. Each agent can carry all the jobs assigned to the resources located in the same domain. So, at the first, there is only one assigned agent for each domain. This agent is replicated to multiple agents when reaching at the domain as shown in Fig. 2. Each replicated agent will be assigned to a certain sub domain. So, it can deliver jobs to the resources of that sub domain in parallel with the other replicated agents.
Each agent will be dispatched to the first resource in the list of resources in its assigned sub domain. Then, the agent will deliver the job to that resource. After that, the agent will continue its work and will consider the next job and assigns it to a resource using the same rules.
The RCA will start collecting results from the resources in the domain. This agent will be launched after a certain time period of launching the job agent. The time period is estimated as the average response time required for executing a job. As the JDA agent, the result agent will also be replicated. As shown in Fig. 3, each agent of the replicated agents will be assigned to a certain sub domain for collecting results in a parallel fashion. So, it can collect results from the resources of that sub domain. The replicated agents will only pass on the allocated resources to collect results. After finishing the collection of results, the agent will return back to the scheduler. The scheduler module can now deliver results to user.
The Information Server module receives requests from the scheduler module about resources status. Then, it prepares the required information and sends it back to the scheduler. The algorithm in Fig. 4 describes the main operation of the system. Fig. 5 and Fig. 6 depict the flowchart for the JDA and RCA internal logics, respectively.

6. Simulation Experiments

In this section, the performance of the proposed system is investigated. Section 6.1 describes the general simulation setup including the simulated grid environment. Section 6.2 shows the experiment results and gives a brief analysis.

6.1 Simulation Grid Environment

We have implemented our proposed system using the GridSim toolkit. The GridSim [3] toolkit is a Java-based discrete event grid simulation toolkit that provides features for application composition, information services for resource discovery, and interfaces for assigning application tasks to resources and managing their execution.
GridSim is designed as a multi-layer architecture for extensibility. This allows new components or layers to be added and integrated into GridSim easily. In addition, the layered GridSim architecture captures the model of the Grid computing environment.
The GridSim toolkit has been applied successfully to simulate a Nimrod-G [4] like grid resource broker and to evaluate the performance of deadline and budget constrained cost- and time- optimization scheduling algorithms [3]. This toolkit is used to simulate the proposed system and to obtain results.
A grid is simulated with a number of resources up to 10000 resources. These resources are distributed among different domains in the grid. The number of domains (agents) and the number of jobs are determined in each simulation experiment. The maximum number of jobs in user applications is 10000 jobs. The size of each job ranges from 1 Kbytes to 10 Mbytes. The initial size of the job agent is 20 Kbytes.

6.2 Results and analysis

In this section, we compare our proposed system with the agent based scheduling system called Sapra. The details of Sapra algorithm can be found in [7]. The comparison is done in terms of throughput, Response time and speedup.

6.2.1 Throughput

Fig. 7 and Fig. 8 show the effect of the number of agents on the throughput of the grid. The proposed system is compared with the Spara algorithm. In this experiment, the numbers of jobs used are 5000 and 10000 jobs and the number of agents (domains) ranges from 1 to 100 agents. It is shown that, as the number of agents increases the grid load increases in both systems. The rate of this increment is higher in case of the proposed system than in case of the other system. This is due that each mobile agent in the proposed system can carry many jobs to different resources in the same domain.

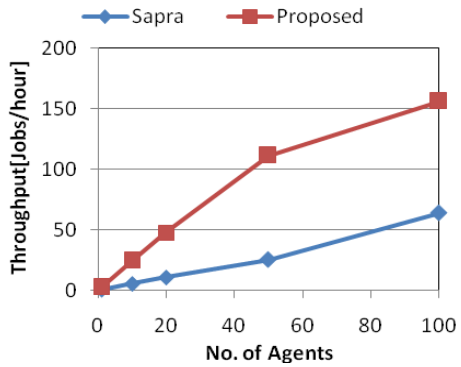

Fig. 7. No. of jobs = 5000.

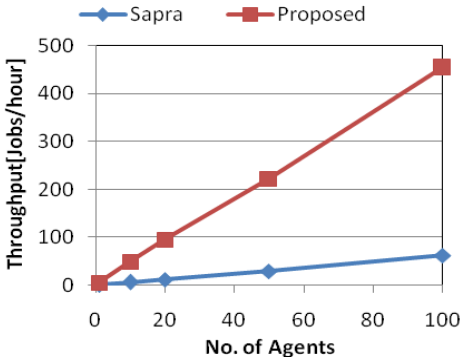

Fig. 8. No. of jobs = 10000.

6.2.2 Response Time

Fig. 9 and Fig. 10 depict a response time comparison between the proposed scheduling system and the Spara scheduling systems. In this experiment, the number of agents (which is the same as the number of domains) ranges from 1 up to 100 agents and the numbers of jobs are 5000 and 10000 jobs. The figures show that, the proposed system provides response time better than that obtained by the Spara system.

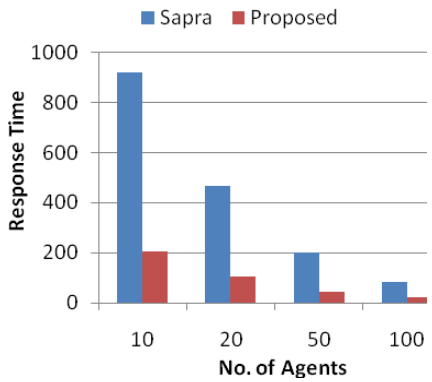

Fig. 9. No. of jobs = 5000.

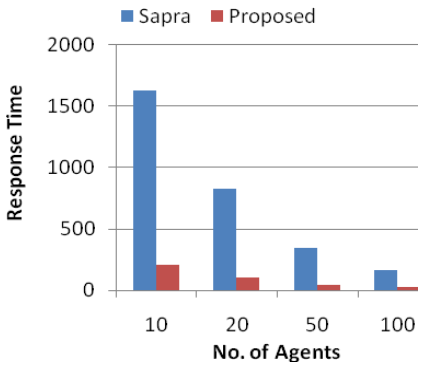

Fig. 10. No. of jobs = 10000.

This is due to the time exhausted by mobile agents for waiting results at each resource in the Spara system. In the proposed system, the mobile agent will not wait for results. The agent has another trip (return trip) to collect results. In both systems, the response time decreases as the number of mobile agents increases.

6.2.3 Speedup

Speedup is a good criterion that can be used for comparing two applications. It relates the response time of an application to the response time of another application when executing the both applications on the same grid. Assume T1 is the time required for executing the first program on the grid and T2 is the time taken for executing the second program on the same grid. Thus the speedup can be estimated as

$$S = \frac{T_1}{T_2}.$$

In this experiment, the number of agents ranges from 1 up to 100 agents and the numbers of jobs are 5000 and 10000 jobs.

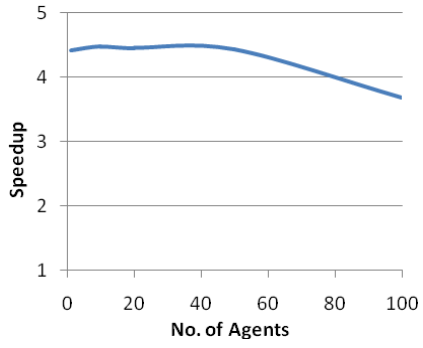

Fig. 7. No. of jobs = 5000

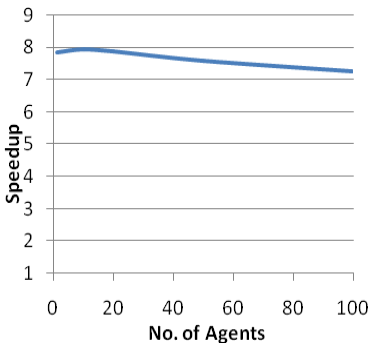

Fig. 8. No. of jobs = 10000.

Fig. 11 and Fig. 12 show a speedup obtained by of using multiple mobile agents in the proposed system over the agents used in the Spara single agent-based scheduling system. It is shown that there is a gain in speedup obtained when using the proposed system. This is due to using multiple parallel agents for delivering jobs and collecting results in each domain instead of using only one agent per domain. Also, it is shown that as the number of agents increases the speedup tends to decrease with a low rate. This decrement in the speedup is due to the increase in the grid load caused by the replicated agents.

7. Conclusions

In this paper, we presented a proposed system for scheduling jobs into computational grids. The system depends on using multiple mobile agents for submitting jobs to different domains in computational grid. Through simulation, we have evaluated the performance of the proposed system.

The proposed system is compared with the Spara scheduling system in terms of throughput, response time and speedup. The experimental results show that the proposed system effectively schedule jobs in the grid and it is proven that the proposed system provides better throughput and response time. Also, the proposed system provides a speedup over the Spara.

References

[1] M. Amoon and T. Altameem, "An Agent-Based approach for dynamic adjustment of scheduled Jobs in Computational Grids," Journal of Computer and Systems Sciences International, vol. 49, no. 5, pp. 765–772, Oct. 2010.

[2] F. Bellifemine, A. Poggi and G. Rimassa, "Developing ti agent systems with a FIPA-compliant agent framework," Software - Practice Experience, Vol. 31, pp. 103-128, John Wiley Sons, 2001.

[3] R. Buyya and M. Murshed, "GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing," The Journal of Concurrency and Computation: Practice and Experience, Vol. 14, Issue 13–1, Wiley Press (2002).

[4] R. Buyya, D. Abramson and J. Giddy, "Nimrod-G: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid", in Proc. of 4th International Conference and Exhibition on High Performance Computing in Asia-Pacific Region (HPC Asia 2000), IEEE Computer Society Press, Beijing, China (May 14–17, 2000).

[5] J. Cao, S.A. Jarvis, S. Saini, D.J. Kerbyson and G.R. Nudd, "ARMS: An Agent-based Resource Management System for Grid Computing," Scientific Programming, Special Issue on Grid Computing, Vol. 10, No. 2, IOS Press, September 2002, pp. 135-148.

[6] H. Casanova, "Distributed computing research issues in grid computing", ACM SIGACT News Journal, Vol. 33, Issue 3, Sep. 2002, pp. 50- 70.

[7] R. S. Chang, J. S. Chang and P. S. Lin, "Multi-Agent Systems for Adaptive and Efficient Job Scheduling Service in Grids," Int. Journal of Computer Applications, Vol 1, No. 9, 2009, pp. 26-30.

[8] D. Chess, B. Grosof, C. Harrison, D. Levine, C. Parris and G. Tsudik, "Itinerant agents for mobile computing," Technical report, IBM Corporation, New York, March 1995.

[9] S. Choi, M. Baik, C. Hwang, J. Gil and H. Yu , "Mobile agent based adaptive scheduling mechanism in peer to peer grid computing," in Proc. of International Conference on Computational Science and its Applications (ICCSA 2005), LNCS 3483, May 2005, pp. 936-947.

[10] S. Choi, M. Baik, J. Gil, S. Jung, C. Hwang, "Adaptive Group Scheduling Mechanism using Mobile Agents in Peer-to-Peer Grid Computing Environment," Applied Intelligence, Special Issue on Agent-based Grid Computing , Volume 25, Number 2, pp. 199-221, October 2006.

[11] L. Chunlin, Z. J. Xiu and L. Layuan, "Resource Scheduling with Conflicting Objectives in Grid Environments: Model and Evaluation," Journal of Network and Computer Applications, Vol. 32, Issue 3, 2009, pp. 760-769.

[12] B. Fechner, U. H¨onig, J. Keller, and W. Schiffmann "Fault-Tolerant Static Scheduling for Grids," in Proc. of IPDPS conference, Miami, Florida USA, April 14-18, 2008.

[13] S. Franklin and A. Graesser, "Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents," in Proc. of the Third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag, 1996.

[14] A4 Project. http://www.dcs.warwick.ac.uk/research/hpsg.

[15] A. Gounaris, R. Sakellariou, N. W. Paton and A. A. A. Fernandes, "Resource Scheduling for Parallel Query Processing on Computational Grids," in Proc. of International Conference on Grid Computing, Pittsburgh, USA, Nov. 2004, pp. 396-401.

[16] P. Huang, H. Peng, P. Lin and X. Li, "Static Strategy and Dynamic Adjustment: An Effective Method for Grid Task Scheduling," Journal of Future Generation Computer Systems, Vol. 25, Issue 8, pp. 884-392, 2009.

[17] C. Huang and C. Pattinson, "Using Mobile Agent Techniques for Distributed Manufacturing Network Management," in Proc. of PGNet 2nd Annual conf., Liverpool, 2001.

[18] H. A. James, Scheduling in Metacomputing Systems, Ph.D. Thesis, Department of Computer Science, University of Adelaide, Australia, 1999.
[19] K. Krauter, R. Buyya and M. Maheswaran, "A Taxonomy and Survey of Grid Resource Management Systems," Software: Practice and Experience, 32(2):135-164, John Wiley & Sons, Inc, NJ, USA, February 2002.
[20] L. Lu and S. Yang, "DIRSS-G: An Intelligent Resource Scheduling System for Grid Environment Based on Dynamic Pricing," International Journal of Information Technology, Vol. 12, No. 4, pp. 120-127, 2006.
[21] S. S. Manvi and M. N. Birje, "An Agent-based Resource Allocation Model for Grid Computing," in Proc. of the 2005 IEEE International Conference on Services Computing (SCC'05), Orlando, Florida, USA, July 11-15, 2005.
[22] S. Mary Saira Bhanu and N. P. Gopalan , "A Hyper-Heuristic approach for Efficient Resource Scheduling in Grid," IJCCC International Journal of Computers, Communications and Control, Vol. III (2008), No. 3, pp. 249-258.
[23] D.G.A. Mobach, B.J. Overeinder, N.J.E. Wijngaards, and F.M.T Brazier "Managing Agent Life Cycles in Open Distributed Systems," in Proc. of the 18th ACM Symposium on Applied Computing, pp. 61- 65, 2003.
[24] R. Moreno and A. B. Conde, "Job Scheduling and Resource Management Techniques in Economic Grid Environments," Lecture Notes in Computer Science, Springer-Verlag, Vol. 2970(2004), pp. 25-32.
[25] R. Pfeifer and C. Scheier, Understanding Intelligence, MIT Press, 1999.
[26] K. Ranganathan and I. Foster, "Simulation Studies of Computation and Data Scheduling Algorithms for Data Grids," Journal of Grid Computing, Vol. 1, 2003, pp. 53-62.
[27] G. Weiss, Multiagent Systems- A Modern Approach to Distributed Artificial Intelligence, MIT Press, 1999.
[28] N. J. E. Wijngaards, B.J. Overeinder, M. van Steen and F.M.T. Brazier, "Supporting internet-scale multi-agent systems," Data Knowledge Engineering, 41(2-3), 2002, pp. 229–245.
[29] M. Wooldridge, An Introduction to Multiagent Systems, John Wiley & Sons, 2002.
[30] R. McClatchey et al., "Data Intensive and Network Aware Grid Scheduling," Journal of Grid Computing, Vol. 5, 2007, pp. 43-64.
[31] G. Ali, N. Shaikh and Z. Shaikh, " Integration of Grid and Agent Systems to Perform Parallel Computations in a Heterogeneous and Distributed Environment," Australian Journal of Basic and Applied Sciences, Vol. 3, No. 4, 2006, pp. 3857-3863.
[32] R. S. Chang, J. S. Chang and P. S. Lin, "An Ant Algorithm for Balanced Job Scheduling in Grids", Journal of Future Generation Computer Systems, vol. 25, pp. 20-27, 2009.