International Journal of Computer Applications Technology and Research
Volume 5–Issue 10, 648-653, 2016, ISSN:-2319–8656

Proposing a Scheduling Algorithm to Balance the Time and Energy Using an Imperialist Competition Algorithm

Arman Alizadeh
Department of Computer Science and Research Branch
Islamic Azad University, Kish, Iran

Ali Harounabadi
Department of Computer
Islamic Azad University
Central Tehran Branch, Iran

Mehdi Sadeghzadeh
Department of Computer
Islamic Azad University
Mahshahr,Iran

Abstract: Computational grids have become an appealing research area as they solve compute-intensive problems within the scientific community and in industry. A grid computational power is aggregated from a huge set of distributed heterogeneous workers; hence, it is becoming a mainstream technology for large-scale distributed resource sharing and system integration. Unfortunately, current grid schedulers suffer from the haste problem, which is the schedule inability to successfully allocate all input tasks. Accordingly, some tasks fail to complete execution as they are allocated to unsuitable workers. Others may not start execution as suitable workers are previously allocated to other peers. This paper presents an imperialist competition algorithm (ICA) method to solve the grid scheduling problems. The objective is to minimize the makespan and energy of the grid. Simulation results show that the grid scheduling problem can be solved efficiently by the proposed method.

Keywords: Grid computing, scheduling, imperialist competition algorithm (ICA), Task Assignment, Meta-heuristic, independent task scheduling.

1. INTRODUCTION

Many applications involve the concepts of scheduling, such as communications, routing, production planning and task assignment in multi-processor system. Most problems in these applications are categorized into the class of NP-complete or combinatorial problems. This means that it would take amount of computation time to obtain an optimal solution, especially for a large-scale scheduling problem. A large variety of approaches have been applied to scheduling problems, such as genetic algorithms (GAs) [1], simulated annealing (SA)[2], particle swarm optimization (PSO) [3], ant colony optimization (ACO) [4], Queen-Bee Algorithm [5], tabu search (TS) [6], and various combinations of these [7, 8, 9, 10, 11] to produce better results in reasonable time.. This paper applies the imperialist competition algorithm to job scheduling problems in grid computing. Grid is a service for sharing computer power and data storage capacity over the Internet. The grid systems do better than simple communication between computers and aims ultimately to turn the global network of computers into one vast computational resource. Grid computing can be adopted in many applications, such as high performance applications, large-output applications, data-intensive applications and community-centric applications. These applications major concern to efficiently schedule tasks over the available processor environment provided by the grid. The efficiency and effectiveness of grid resource management greatly depend on the scheduling algorithm [12]. Generally, in the grid environment, these resources are different over time, and such changes will affect the performance of the tasks running on the grid. In grid computing, tasks are assigned among grid system [13]. The heuristic ICA [14], proposed in 2007, was inspired by the sociopolitical evolution of imperial phenomena and has been used for solving many optimization problems in continuous space. This paper proposes a discrete version of ICA for solving the independent task scheduling problem in grid computing systems. The present paper converts ICA from a continuous state algorithm to a discrete state algorithm by changing the assimilation stage. The resulting algorithm is compared with SA and other heuristic algorithms and is shown to produce better results than these. This algorithm simultaneously considers makespan and completion time by using appropriate weights in the mean total cost function. Furthermore, the dynamic situations are not considered in this paper.

2. BACKGROUND

2.1 Scheduling Method

The scheduling task issue is considered as a tough challenge which is composed of n tasks and m resources must be processed by a machine and does not top until the end of the performance. We used ETC matrix model described in [15]. The system assumes that the expected execution time for each task i, on every resource j is predetermined and is located in the matrix ETC, ETC [i, j]. Here, makespan is regarded as the maximum completion time in CompleteT [i, j], calculated in the following equation (1) [16]:

$$\text{Makespan} = \text{Max}\ (\text{completeT}\ [i,j])\ 1 \leq I \leq N,\ 1 \leq j \leq M \qquad (1)$$

In the above equation, CompleteT[i, j] is equal to the time when the task i on the source j is completed and it is calculated in equation (2):

$$\text{completeT}\ [i,j] = \text{Ready}[M] + \text{ETC}[i,j] \qquad (2)$$

2.2 Imperialist Competition Algorithm

The imperialist competition algorithm (ICA) proposed by (Atashpaz -Gargari and Lucas 2007) is a mathematical modeling of imperialist competitions. Similar to other evolutionary algorithms, the ICA algorithm begins with a number of initial random populations. Each random population is called a country. The possible solutions for ICA are called countries. A number of the best elements of the population are selected as imperialists, others which are governed by Imperialists are called colonies. By applying an assimilation policy in the direction of various optimization axes, imperialists gain the favor of their colonies. The total power of each empire is modeled as the sum of the imperialist power and a percentage of the mean power of its colonies. After the initial formation of empires, imperialistic competition starts among them. Any empire that has no success in the imperialistic competition with nothing to add to its power is eliminated from the competition. So the survival of an empire depends on its power to assimilate competitor's colonies. As a result, the power of greater empires is gradually increased in imperialistic competitions and weaker empires will be eliminated. Empires have to make improvements in their colonies in order to increase their power. For this reason, colonies will eventually become like empires from the point of view of power, and we will see a kind of convergence. The stopping condition of the algorithm is having a single empire in the world.

3. PREVIOUS RESEARCH

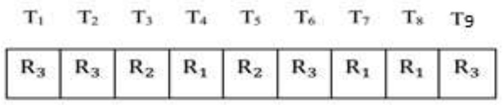
A large number of heuristic algorithms have been proposed for grid scheduling. Most of them try to minimize the maximum completion time of tasks, or makespan. Each task has its own deadline, and we try to decrease the makespan in order to prevent tasks from failing to execute because of their deadlines. That is, decreasing the makespan results in the ability to execute more tasks in the network. It also helps to provide efficient resource allocation and energy utilization.

The hierarchic genetic strategy (HGS) algorithm was proposed in [22] for scheduling independent tasks in a grid system and is implemented in dynamic grid environments in batch mode. This algorithm simultaneously considers optimization of flow time and makespan. The authors generate root nodes based on two other algorithms: longest job to fastest resource and shortest job to fastest resource (LJFR-SJFR) [23] and minimum completion time (MCT) [24], and they generate the rest of the population stochastically. In LJFR-SJFR, initially, the highest workload tasks are assigned to machines that are available. Then the remaining unassigned tasks are assigned to the fastest available machines. In MCT [24], tasks are assigned to machines that will yield the earliest completion time.

Balanced job assignment based on ant algorithm for computing grids called BACO was proposed by [20]. The research aims to minimize the computation time of job executing in Taiwan UniGrid environment which focused on load balancing factors of each resource. By considering the resource status and the size of the given job, BACO algorithm chooses optimal resources to process the submitted jobs by applying the local and global pheromone update technique to balance the system load. Local pheromone update function updates the status of the selected resource after job has been assigned and the job scheduler depends on the newest information of the selected resource for the next job submission. Global pheromone update function updates the status of each resource for all jobs after the completion of the jobs. By using these two update techniques, the job scheduler will get the newest information of all resources for the next job submission. From the experimental result, BACO is capable of balancing the entire system load regardless of the size of the jobs. However, BACO was only tested in Taiwan UniGrid environment.

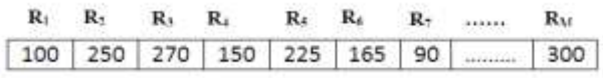
4. THE PROPOSED ALGORITHM

The proposed algorithm for task scheduling is a combination of ICA and GELS which is used for scheduling of independent tasks in computational grid environment. According to ICA's high performance in scheduling problems, a simple method for country representation is taken into consideration firstly. Natural numbers are used to encrypt countries. The value in each country is resource number ranging from 1 to M, where M is the number of all resources. Fig.1 illustrates an example of countries representation where 9 tasks are assigned to 3 resources. As Figure depicts, for example task 4 (T_4) is running on resource 1(R_1).



T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
R_3	R_3	R_2	R_1	R_2	R_3	R_1	R_1	R_3

Fig. 1. Country representation in ICA-GELS Algorithm.



R_1	R_2	R_3	R_4	R_5	R_6	R_7	......	R_M
100	250	270	150	225	165	90	....	300

Fig. 2. The amount of consumed energy for each task time unit.

$P_{j,i}$ is considered as a 1*M array where M is the number of resources. The value in each entry represents the amount of consumed energy for executing a task. Fig 2 shows a sample matrix where the consumed energy per each task time unit in second resource (R_2) is 250 J. The initial population of countries in the ICA is generated randomly. A random number between 1 and M is generated which is the resource number and the intended task will be execute on it. The independent task scheduling problem includes M tasks and M machines. Each task should be processed by one of the machines in way that makespan get minimum. Proposed algorithm takes two QoS parameters such as energy and time limitation into consideration. Each task can be execute on one resource and will not stop until execution is completed. Our algorithm applies ETC matrix model. Considering that proposed scheduling algorithm is static, it is assumed that expected execution time is determined for each task on each resource at prior and it is stored in ETC [i, j]. Also, Ready time [M] determines the time that machine M completes the previous assigned task. Makespan is considered as maximum completion time ($\mathrm{completeTime}\ (i,j)$), which is computed using equation (3).

$$\mathrm{Makespan} = \mathrm{Max}\ (\mathrm{completeT}\ [i,j]) \qquad (3)$$

$\mathrm{completion\ \_time}\ [i,j]$ is the time when task i completes on resource j and it is computed using equation (4) and TransferC and wait (i, j) are respectively the data transfer time and waiting execution time.

$$\mathrm{completeT}\ (i,j) = \mathrm{ETC}\ [i,j] + \mathrm{TransferC} + \mathrm{wait}\ (i,j) \qquad (4)$$

The objective of scheduling in the proposed algorithm is to map each task to each resource in way that makespan and total loosed tasks get minimum. For example, ETC matrix (table (1)) presents 9 tasks and 3 resources.

Table 1. ETC matrix

Task/Resource	P1	P2	P3
T1	2	3	1
T2	2	5	3
T3	1	3	4
T4	4	5	6
T5	8	5	6
T6	3	5	4
T7	4	2	4
T8	4	6	7
T9	2	3	1

4.1 The objective and fitness function in proposed algorithm

The main idea behind task scheduling is to minimize the makespan, which is the total execution time for all tasks. To solve task scheduling by ICA, a country is more suitable that minimize makespan and total consumed energy for all tasks. Equation (5) describes the first fitness function for each country. Also, equation (6) and (7) compute the energy consumption for each solution.

$$\text{Fitness}_1\,(C_i) = \alpha \times \frac{1}{\text{FTime}\,(C_i)} + (1-\alpha) \times \frac{1}{E_{(C_i)}} \qquad (5)$$

$$E_{(C_i)} = \sum_{j=1}^{M} \sum_{i=1}^{N} E_{i,j} \qquad (6)$$

$$E_{i,j} = (P_{j,i} - I_j)\text{ETC}\,[i\,,j] \qquad (7)$$

Where E is the total consumed energy to execute all tasks in country i and $P_{j,i}$ is the consumed energy for executing task i on resource j. Also I_j is the consumed energy of resource i when the resource is ideal. It can be concluded from equation (5) that when the consumed enregy and makespan are minimzied, the fitness function is maxmized and shows the more suitable solution to the problem. Also the coeffiction α determines the impact of each parameter on fitness value. For example Fig. 3 illustrates a solution to the scheduling problem. Considering the ETC matrix in table (1), the makespan equals to 12. Fig. 3 shows the total scheduling length for countries.

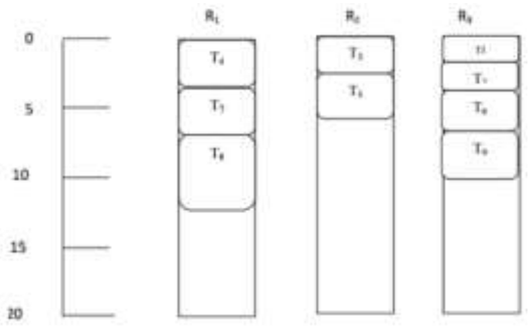


Fig. 3. Total scheduling length for countries

4.2 Second fitness function

As stated before, the main objective of task scheduling is to minimize the makespan. It is possible to have several countries with similar total scheduling time but with different workload on their resources. Therefore, second fitness function takes this factor into consideration. Such that, after achieving solutions with minimum makespan, second fitness function will be applied to them aiming at acquiring balanced solution in terms of workload. To gain maximum balanced workload, the fitness value should be computed for resources. To achieve the value of balanced workload we are have to compute sum of standard deviation. It is clear that we should compute the fitness for resources firstly. Equation (8) shows the second fitness function for balanced workload computing.

$$\text{Fitness}_2(\text{Countery}_i) = \frac{1}{\text{RU}_i} \qquad (8)$$

At fires maximum exaction time is specified using equation (9).

$$E = \text{Max}\,\{T_{ij} + \tau_{ij}\} \qquad (9)$$

To compute E's value in highest path, we use T_{ij} and τ_{ij} which are equal to data transmission time RMS and processing time, respectively. If we use u_i to represent fitness value of resource i for execting task j, we have:

$$u_i = \frac{T_{ij}+\tau_{ij}}{E} \qquad (10)$$

To compute fitness value for each resource we use the equation process which is used for computing E value. In other words, the value of E is specified firstly, then using equation (10) the fitness value for each resource is computed. To compute standard deviation, we are have to compute total fitness value at first. Consider the two solutions for task scheduling in Fig. 2 and Fig. 3, the scheduling total time for this solutions is equal to 12.

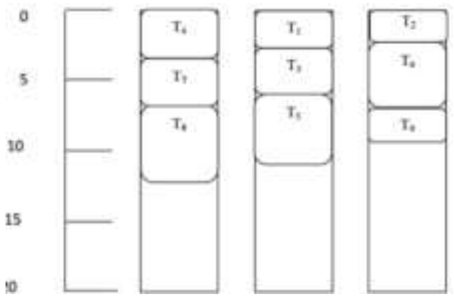
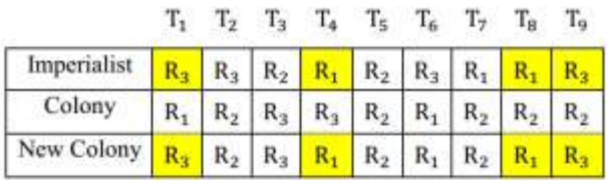


Fig. 4. Total scheduling time with respect to load balancing.

4.3 The assimilation and revolution operations in our proposed algorithm (ICA)

4.3.1 Assimilation operator

Historically, the assimilation policy was developed with the purpose of moving the culture and social structure of colonies toward the culture of the central government. In the proposed algorithm some cells approximately 40% (Duki, et al. 2010) of the Imperialist array are randomly selected (cells 1, 4, 8, and 9 in the Fig. 5 are imperialist and others are colony).

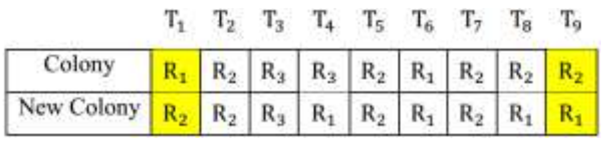


	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
Imperialist	R_3	R_3	R_2	R_1	R_2	R_3	R_1	R_1	R_3
Colony	R_1	R_2	R_3	R_3	R_2	R_1	R_2	R_2	R_2
New Colony	R_3	R_2	R_3	R_1	R_2	R_1	R_2	R_1	R_3

Fig. 5. Assimilation operator representation

4.3.2 Revolution operator

To perform this operation, two cells are first selected in the colony in a random manner and their values are exchanged. This stages (random exchange or revolution operation) is repeated based on the percentage of the total number of tasks. If the new colony is better than the old one, it replaces the old colony; otherwise, this procedure is repeated. This operation is illustrated in Fig. 6.



	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
Colony	R_1	R_2	R_3	R_3	R_2	R_1	R_2	R_2	R_2
New Colony	R_2	R_2	R_3	R_1	R_2	R_1	R_2	R_1	R_1

Fig. 6. Revolution operator representation

The Pseudo-code of the proposed algorithm is shown below.

```
Step 1 : Initialization
Step 2 : 2.1. Generate the k number of random country with
          length n
          2.2.Setting the Maximum: Maximum-Time (Mt)
               and Maximum-energy  (Me) according to the
user's requirement.
Step 3 : 3.1. Create initial empires.
Steo 4 : 4.1. Assimilation & Revolution
         4.2. Assimilation: Colonies move towards
              imperialist.
         4.3. Revolution: Random change occur in the
              charcteristics of some countries.
Step 5 : Position exchange between a colony and imperialist
Step 6 : Compute Makespan and energy for all country
Step 7 : 7.1. Imperialistic Competition
         7.2. Eliminate the powerless empires. Weak
              empires lose their power their power gradually
              and they will finally be eliminated.
Step 8 : Check stop condition satisfied.
Step 9 : The best country from the imperialist competitive
         algorithm as a best solution select.
```

Figure 7. Pseudo-code of the proposed algorithm.

5. DISCUSSION

In this section, aiming at evaluating performance of the proposed work, the simulation results will be presented. Our simulations are conducted on OPNET Modeler simulator (Modeler 2009). We have conducted several simulations to verify the effectiveness of our proposed algorithms. Our experiments are conducted in a system equipped with 4 GB of RAM and 1600 MHz CPU which is run Windows Xp. The performance of the proposed algorithm is compared with several task scheduling algorithms considering the simulation parameters demonstrated in table (2). The iteration parameter shows that to achieving the application execution time using existing algorithms, 100 iteration are performed and the average of the values are computed. Also the initial value for parameters which are used in ICA and GA algorithms are shown in table (2), respectively.

Table (2). Initial values for parameters for algorithms

ICA	Assimilation rate	2
	Revolution rate	0.1
	α rate	0.8
	Neighborhood radius (R)	0.1
	Maximum velocity	Sized of the input tasks
	Initial velocity	Between 1 and Max Velocity
	Constant Gravitation (G)	6.672
GA	P-Crossover	0.85
	P-Mutation	0.02

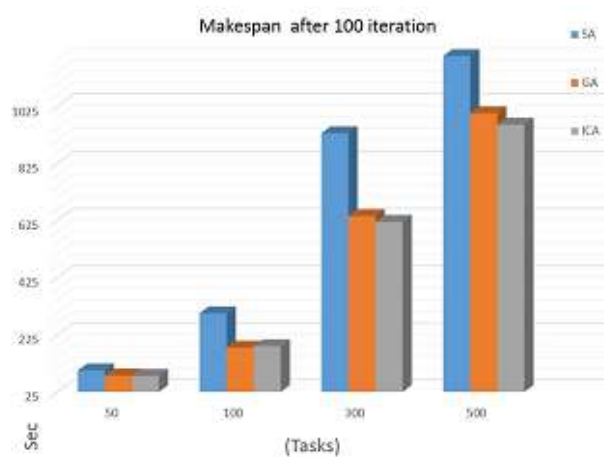


Figure 8. Comparison of makespan of algorithms

Fig.9 compares the algorithms in terms of consumed energy. As results show the consumed energy increases as time passes. The consumed energy for our algorithm is lower than other algorithms.

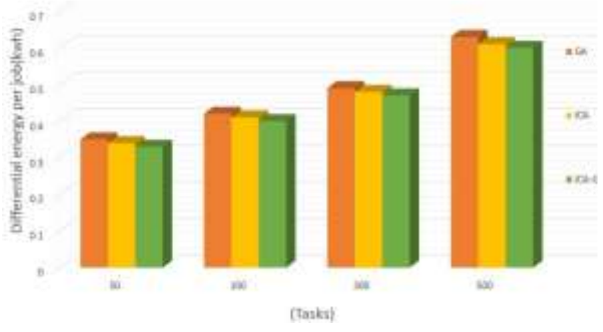


Fig. 9. Comparing consumed energy for task execution

6. CONCLUSIONS

In the proposed method, the researchers imperialist competitive algorithm which has been used for scheduling in grid environment. The proposed algorithm, in which a weighted objective function is used considering the degree of importance of time and energy of user's projects, gives more freedom for specifying the time and energy of users' projects. In the use of this algorithm, similar to the objective function, the time and the energy, along with their weight, are considered based on the user's perspective. The purposes of the proposed scheduling are to minimize completion time and energy of implementation of tasks for different tasks of users simultaneously.

The proposed scheduling is compared with two algorithms of GA and SA with regard to the parameters of time and energy. The results are investigated based on cost and energy requests in different charts. They show that if we have limited number of resources as well as high number of duties, the proposed scheduler has the best performance compared to the other three schedulers in reducing the overall time and energy of scheduling. The results of experiments show that the imperialist competitive algorithm can reach a high performance regarding creation of a balance between energy and tasks implementation scheduling. Further research can be conducted with regard to the practice of resource allocation to works in the proposed algorithm through using an approach based on fuzzy logic and using of fuzzy inference system.

7. REFERENCES

[1] J. Kołodziej and F. Xhafa, Integration of task abortion and security requirements in GA-based meta-heuristics for inde pendent batch grid scheduling, Computers &Mathematics withApplications 63(2) (2012), 350–364.

[2]A.Kazem, A.M. Rahmani and H.H. Aghdam, A modified simulated annealing algorithm for static scheduling in grid computing, Proceedings of the 8th International Conference on Computer Science and Information Technology (2008),623-627.

[3]Garcia-Galan, R.P. Prado and J.E.M. Exposito, Fuzzy scheduling with swarm intelligence-based knowledge acquisition for grid computing, Engineering Applications of Artificial Intelligence 25(2) (2012), 359–375.

[4] L. Wei, X. Zhang, Y. Li and Yu Li, An improved ant algorithm for grid task scheduling strategy, Physics Procedia, Elsevier 24 (2012), 1974–1981.

[5] Z. Pooranian, M. Shojafar, J.H. Abawajy and A. Abraham,An efficient meta-heuristic algorithm for grid computing,Journal of Combinatorial Optimization (JOCO) (2013),doi:10.1007/s10878-013-9644-6.

[6] Z. Pooranian, A. Harounabadi, M. Shojafar and J. Mirabedini, Hybrid PSO for independent task scheduling in grid computing to decrease makespan, International Conference on Future Information Technology IPCSIT 13,Singapore(2011), 435–439.

[7] S. Benedict and V. Vasudevan, Improving scheduling of scientific workflows using Tabu search for computational grids, Information Technology Journal 7(1) (2008), 91–97.

[8] R. Chen, D. Shiau and S. Lo, Combined discrete particle swarm optimization and simulated annealing for grid computing scheduling problem, Lecture Notes in Computer Science, Springer 57 (2009), 242–251.

[9] M. Cruz-Chavez, A. Rodriguez-Leon, E. Avila-Melgar, F. Juarez-Perez, M. Cruz-Rosales and R. Rivera-Lopez, Genetic-annealing algorithm in grid environment for scheduling problems, Security-Enriched Urban Computing and Smart Grid Communications in Computer and Information Science, Springer 78 (2010), 1–9.

[10] Z. Pooranian, A. Harounabadi, M. Shojafar and N. Hedayat, New hybrid algorithm for task scheduling in grid computing to decrease missed task, world academy of science, Engineering and Technology 79 (2011), 924–928.

[11] F. Xhafa, J. Gonzalez, K. Dahal and A. Abraham, A GA(TS) hybrid algorithm for scheduling in computational grids, Hybrid Artificial Intelligence Systems Lecture Notes in Computer Sci- ence, Springer 5572 (2009), 285–292.

[12]Lee, L.T., Tao, D.F., Tsao, C.: An Adaptive Scheme for Predicting the Usage of Grid Resources. Comput.Electr. Eng.33(1), 1–11 (2007)

[13]Salman, A., Ahmad, I., Al-Madani, S.: Particle Swarm Optimization for Task Assignment Problem. Microprocessors and Microsystems 26, 363–371 (2002)

[14] E. Atashpaz-Gargari and C. Lucas, Imperialist competitive algorithm: An algorithm for optimization inspired by imperialist competitive, IEEE Congress on Evolutionary computation, Singapore (2007), 4661–4667.

[15] Braun, T. D., Siegel, H. J , A taxonomy for describing matching and scheduling heuristics for mixed machine heterogeneous computing systems, Proceedings of the 17th IEEE Symposium on Reliable Distributed Systems, pp. 330335, 1998.

[16] Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S.,"BGSA: binary gravitational search algorithm". Nat. Comput. 9(3), 727–745, 2010.

[17] J.Kolodziej and F.Xhafa, Enhancing the genetic-based scheduling in computational grids by a structured hierarchical population, Future Generation Computer Systems,Elsevier 27(8)(2011),1035-1046

[18] A.Abraham,R.Buyya and B.Nath,Nature's heurustics for scheduling jobs on computational grids, Proceeding of the 8th IEEE International Conference on Advanced Computing and Communications, India(2008), 1-8.

[19] C.Weng and X.Lu, Heuristic scheduling for bag-of-tasks applications in combination with QOS in the computational grid, Future Generation Computer Systems 21(2)(2005),271-280.

[20] Chang, R. S., Chang, J. S., Lin, P. S., "Balanced Job Assignment Based on Ant Algorithm for Computing Grids", The 2nd IEEE Asia-Pacific Service Computing Conference, pp. 291295,2007.