New Heuristic Function in Ant Colony System for Job Scheduling in Grid Computing

KU RUHANA KU-MAHAMUD[1], MUSTAFA MUWAFAK ALOBAEDY[2]
[1,2]School of Computing
College of Arts and Sciences,
Universiti Utara Malaysia,
06010 Sintok, Kedah,
MALAYSIA
E-mail: [1]ruhana@uum.edu.my, [2]new.technology@hotmail.com

Abstract: - Job scheduling is one of the main factors affecting grid computing performance. Job scheduling problem classified as an NP-hard problem. Such a problem can be solved only by using approximate algorithms such as heuristic and meta-heuristic algorithms. Ant colony system algorithm is a meta-heuristic algorithm which has the ability to solve differenttypes of NP-hard problems.However, ant colony system algorithm has a deficiency in its heuristic function which affects the algorithm behavior in terms of finding the shortest connection between edges. Thispaper focuses on enhancing the heuristicfunction where information about recent ants' discoveries will be taken into account. Experiments were conducted using a simulator with dynamic environment features to mimicthe grid environment.Results show that the proposed enhanced algorithm produce better output in term of utilization and makespan.

Key-Words: -Ant colony optimization, ant colony system, heuristic function, job scheduling, grid computing.

1 Introduction

The concept of grid computing goes back to 1969 when Leonard Kleinrock wrote "We will probably see the spread of computer utilities, which, like present electric and telephone utilities, will service individual homes and offices across the country" [1]. From that time, many researchers presented many works and contributed in grid computing fields.Grid computing evolves from existing technology such as distributed computing, web service, and Internet [2]. Grid computing can be defined as "Geographically distributed computers, linked through the internet in a grid-like manner and are used to create virtual supercomputers of vast amount of computing capacity which are able to solve complex problem from e-Science in less time than known before" [3]. [4] Presented an extensive definition for grid as "A hardware and software infrastructurethatprovides transparent, dependable, pervasive and consistent access to large-scale distributed resources owned and shared by multiple administrative organizations in order to deliver support for a wide range of applications with the desired qualities of service. These applications canperformeither high throughput computing, on-demand computing, data intensive computing, or collaborative computing".

Grid computer could be distributed geographically through different organization using different platform[5]. Two services are offered by grid, computing-intensive services and data-intensive services [6]. In computing services, grid process tasks which are not possible or very difficult to process them in traditional computer resource, while data storage services provide a storage which is available through many mirrors and servers. Grid computing provides powerful computation resources for complex tasks such as scientific research, stock markets, and business requirements for organizations. However, grid computing is still in the development stage, and there are many challenges are to be addressed in the future. [3]

From previous definitions, it can be concluded that grid computing is a collection of geographically distributed and heterogeneous resources. The resources are connected like a grid using internet technology to form a virtual supercomputer that has the capacity to solve very complex problems. Grid can be used by different fields such as science, commerce and education.

In grid computing, job scheduling algorithm is a main issue for grid computing performance [7], [8], [9] Scheduling can be done in simple way by assigning the incoming task to available resource. Better computing performance can be obtained in

ISBN: 978-1-61804-148-7

grid if more advanced and sophisticated scheduler algorithm is applied. Scheduler should take into account many aspects such as dynamic tasks environment, joining and dropping of resources from grid and evaluating current load of resources. Scheduler can be in hierarchical or distributed organization to deal with large scale grid.

Scheduling problem is defined as a NP-Hard problem [10]. In grid environment, the scheduler is responsible to find suitable resource to process specific task. Job scheduling problem can be presented as a directed graph (digraph) with nodes and directed edges. Resources and jobs are represented as nodes and directed edges represent processing cost and pheromone. Fig 1 depicts the graph representation.

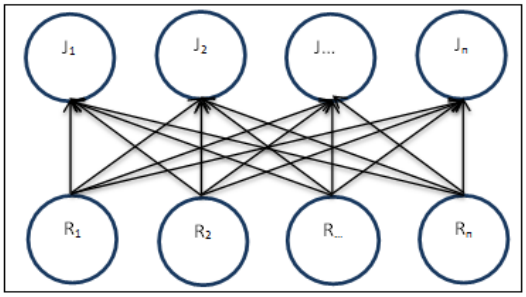

Figure 1: Directed Graph representation of jobs and resources.

The NP-Hard problem needs heuristic and meta-heuristic methods such as Ant Colony Optimization (ACO), Simulated Annealing (SA), Tabu Search (TS), Scatter Search (SS), and Genetic Algorithm (GA) to solve. ACO algorithms which are inspired by biological ants presented many solutions for different types of NP-Hard problems [11]. Ant Colony System (ACS) algorithm which is a member of ACO algorithms is one of the best algorithms for solving NP-Hard problem [12].

In grid computing systems, many criteria depend on scheduling algorithm efficiency such as grid performance, utilization, Quality of Service (QoS), and load balancing [13], [14].[3] Stated that "Rather than a problem, scheduling in grid systems is a family of problems. This is due to the many parameters that intervene scheduling as well as due to the different needs of grid-enabled applications". In scheduling process, there are many factors and parameters that should be taken into account such as job size, resource capacity, network speed, current load, and expected time to complete. In addition, the dynamic and heterogeneous nature of grid environment makes the scheduling process more critical such as joining and dropping resources to the gird [15]. If the scheduling algorithm is not efficient, grid system user will experience a delay in response time, especially when the number of tasks is increased [16]. Therefore, scheduling algorithm is an important part in grid computing systems and it needs to be improved to cater the dynamic requirements. Hence, a sufficient algorithm for dynamic grid scheduling problem is crucial to enhance the grid computing systems performance [16], [17].

2 Ant Colony System

Biological ants have the ability to discover the shortest route from the nest to the source of food [18]. The ants have the ability to communicate with the environment even though they do not have an advanced vision system [19]. Ants use a chemical substance called "pheromone" to communicate with the environment and between each other [20]. Pheromone substance has evaporation property which is a powerful mechanism to update the route information. Ant deposits pheromone along the pathwhile it moves looking for food. The following ant is more likely to select the route with richer pheromone. This mechanism will make the ant choose the shortest path. In artificial ants, [21] presented an algorithm called ant colony system that mimic the behavior of ant colony to solve optimization problems such as Traveling Salesman Problem (TSP) and network routing. In ACS algorithm, ants apply exploitation and exploration mechanisms when they select the next city to move to. In addition, ACS applies local pheromone update and global pheromone update to direct the search for next iteration. The global update is calculated based on the quality of the best tour so far while local update applies evaporation concept. In ACS algorithm, the updating functions focus on pheromone and neglect the heuristic value.
There are various enhanced ACS algorithms used for job scheduling problem [13], [22], [23].

The classical ACS is not efficient in large-scale computation problem due to the stagnation nature of pheromone in ACS [7]. ACS suffers from a deficiency in the heuristic update function.Pheromones updatefunction only focuses on local and global pheromone. ACS algorithm utilizes the value between the nodes to use it as a heuristic value to calculate the probability of choosing the next node. One part of the algorithm called heuristic function is not updated at any time throughout the process. Such behavior is a contradiction to the concept of heuristic. According to [24] the word "heuristic" comes from Greek and means "to know", "to find", "to discover" or "to guide an investigation". Therefore,update function

to the heuristic value is needed to reflect the new information discovered by the ants.

3 Enhanced ant colony system

Enhancement of ACS algorithm has been proposed by many researches to solve the optimization problem [25]. In their works, they proposed many ideas to increase the algorithm performance. However, all the studies have focused on local and global pheromone update functions. In general, problems are modeled as graph that consists of nodes and edges. ACS algorithm utilizes the values between the edges to use it as a heuristic value for the calculation of probability to choose the next node but the heuristic value is not updated at any time during execution.

The proposed new heuristic function will update the heuristic value every time the ants find a better solution in the iteration. This is done to reflect the new solution status. After an ant has constructed its solution, a global update process will be applied to update the best-so-far solution. This event will change the environment for the next iteration. A function will be triggered at this moment to reflect this change and thus a new heuristic value will be obtained. The new information will be applied to the best-so-far edge. The pseudo-code for the proposed Enhanced Heuristic Function in Ant Colony System (EHF_ACS) is shown in Fig2.

```
Step0: for each path in the best tour do step 1 to 2
    Step1: if path i (i = 1, 2, n)is not updated before
                do step 2
        Step2: η_i = η_i + (δ/ best-so-far tour)
        // δ is parameter from (0-10)
End
```

Figure 2: Pseudo-code for the proposed enhanced heuristic function

By applying this function, the heuristic values will change according to the quality of the best-so-far solution. Best solution will increase the heuristic value and vice versa. The parameter δ will determine how much the influence of the updating value should be applied to the heuristic value. If δ = 0 then no update will occur which reflects the heuristic value in the traditional ACS. The heuristic value on each edge will be updated only one time during the whole process if it is belongs to the best-so-far edge. This condition will eliminate the issue of stagnation that may occur if the heuristic value is updated more than one time.The generic EHF_ACS algorithm is as depicted in Fig3. The difference between this algorithm and the classical ACS is the application of the enhanced heuristic function after global pheromone update activity is performed.

```
Procedure EHF_ACS
    Initialize parameters
    While (termination condition not met) do
        Construct Ants Solutions
        Apply Local Pheromone Update
    End - While
    If (New AntSolution better than Global Best
        Solution)Global Best Solution = New Ant
        Solution
    Apply Global Pheromone Update
    Apply New Heuristic Function
End - Procedure
```

Figure 3: The EHF_ACS algorithm

In order to apply EHF_ACS to job scheduling problem, there are several initializations will have to be performed on the following parameters: i. List of jobs and resources. ii. Array for the pheromone. iii. The variables alpha (α), beta (β), delta (δ), *p*, *q*, number of ants and number of iteration.

Ants will be randomly distributed to resources after the initialization process. All ants will move concurrently and each ant will start building a solution which is a function of processing all jobs using all resources. Each time an ant moves from resource (node) to job, the pheromone on that connection (edge) will be evaporated according to local update function. The benefit of local update function is to reduce the probability of selecting the same resource (node) by the following ant. Local update also helps to reduce stagnation problem and increase exploration mechanism. After all ants have constructed their solutions, the best solution will be selected based on makespan and utilization criteria. The best solution will be saved as global best solution if it is better than the current global best solution. A global update function will be applied at this step using the global best solution. The benefit from global update function is to increase the probability of selecting this particular resource for the next iteration. The enhanced heuristic function will start immediately after the global update in order to update the heuristic values. Fig4shows the EHF_ACS algorithm for grid job scheduling.

ISBN: 978-1-61804-148-7

```
Procedure EHF_ACS
    Step 0: Set Parameters, Jobs array& Resource array;
    Step 1: Calculate Expected Completion Time (ECT);
    Step 2: Set Load array;
    Step 3: Set Heuristic array;  /// 1 / ECT
    Step 4: Set Pheromone Array; /// τ0
    Step 5: Initialize ants array;
    Step 6: While (termination condition not met)
    do steps 7-15
        Step 7: For each ant (ant[i], i = 0, …, m) do step 8
            Step 8: Create Thread;  /// one thread for each ant
End - For
Step 9: For each Thread (Threadi,i = 0,…, m) do
Steps 10-11;
            Step 10: Construct ant[i] Solution;
            Step 11: Apply Local Pheromone Update;
End - For
Step12:If(Best Anti Solution is better than Global
                Best Solution)Do Step 13
Step 13:Global Best Solution = Best Anti
                    Solution;
Step 14: Apply Global Update Pheromone;
Step 15:Apply New Heuristic Function;
End - While
End - Procedure
```

Figure 4: EHF_ACS algorithm for job scheduling.

The quality of each ant solution is measured using the makespan and resource utilization criteria. In this case, shorter makespan and higher utilization means better solution quality. After completing all iterations, the assignment process (i.e. distribution of jobs to resources based on the final solution from EHF_ACS) starts and load information is updated for each resource. The whole process will be repeated at specific interval to schedule new incoming jobs.

4 Grid Computing Simulator

A simulator has been developed using C# language for the purpose of evaluating the proposed algorithm in scheduling jobs in the grid environment. The simulator consists of three parts which are: 1-Job Generator Engine: This engine has the responsibility to generate jobs with specific configuration according to user requirements. 2- Resource Generator Engine: This engine is responsible to create different type of resources according to user requirements. 3- Enhanced Ant Colony System Engine with new heuristic function: This is the main part of the simulator. After creating jobs and resources, the user can enter all the algorithm parameters to the engine through graphical user interface and starts the simulation process.

5 Experiments and results

Experiments were performed to test the performance of the proposed algorithm using the developed simulator. Jobs were created with size varies from 500 – 1000 Million Instruction (MI) and number of jobs is set between 10 and 100. Seven resources were created for these experiments with capacity varying from 50 – 250 Million Instruction per Second (MIPS) with load from 0 – 9.In total, the number of experiments conducted is 100 for ACS and 100 for EHF_ACS. 10 instances were created based on incremental number of jobs from 10-100 with interval of 10 jobs. Each 10 experiments were conducted using the same instances to get the average and standard deviation of each 10 experiments' results. Other parameters settings for ACS and EHF_ACS algorithms are presented in Table 1.

Table 1: The algorithm's parameters

No. of Ants	7	α	1
Initial Pheromone (t_0)	0.0001	β	2
No. of Iteration	1000	p	0.5
δ	0.5	q	0.9

The performance metrics used to evaluate the proposed algorithm were makespan and utilization. Shorter makespan indicates faster performance in term of processing time while utilization criteria indicate the quality of jobs scheduling and load balancing policy. Contradiction between makespan and utilization may occur if there is limited resource. Therefore, load balancing provide fair distribution rather than equal distribution in order to get good performance. Table 2 presents the comparison between ACS and EHF_ACS in terms on makespan and utilization. The results are again presented in Fig 5 and Fig 6 which display the makespan, utilization, average utilization (AvgUT) and standard deviation (sd) of 7 resources with various tasks for ACS and EHF_ACS.

Table 2: ACS and EHF_ACS performances

	ACS				EHF_ACS			
Task	**Makespan**	**Utilization**	**Avg UT**	**sd**	**Makespan**	**Utilization**	**Avg UT**	**sd**
10	**12.52**	64.56	62.48	1.04	15.929	**77.715**	74.82	3.82
20	22.638	74.716	71.59	3.3	**20.541**	**90.14**	85.74	5.63
30	33.364	**92.396**	83.94	5.24	**29.395**	88.246	82.94	3.86
40	**35.566**	**88.658**	81.57	5.46	43.547	85.604	81.41	2.81
50	51.913	88.984	77.1	5.7	**42.575**	**92.96**	87.85	3.14
60	65.359	88.896	82.81	5.1	**50.327**	**94.369**	87.5	4.62
70	64.82	77.747	74.02	7.26	**59.836**	**89.643**	84.14	3.61
80	71.647	**96.154**	79.75	8.79	**65.747**	96.14	87.39	5.32
90	101.043	80.417	73.95	3.8	**73.134**	**94.106**	87.56	4.48
100	110.039	82.26	74.68	3.75	**80.284**	**95.472**	86.62	4.99

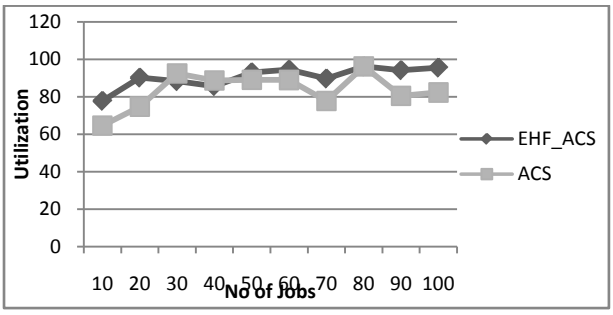

Figure 5: Utilization of 7 resources

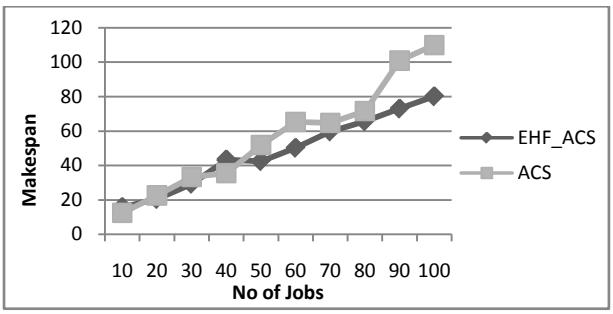

Figure 6: Makespan of 7 resources

The stability of resource utilization with various numbers of jobs is also tested. The unstable utilization pattern using classical ACS is depicted in Fig 7 whereas EHF_ACS algorithm produces a more stable utilization pattern as shown in Fig 8.

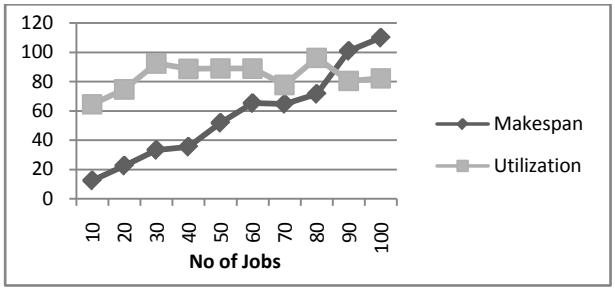

Figure 7: Utilization and makespan using ACS

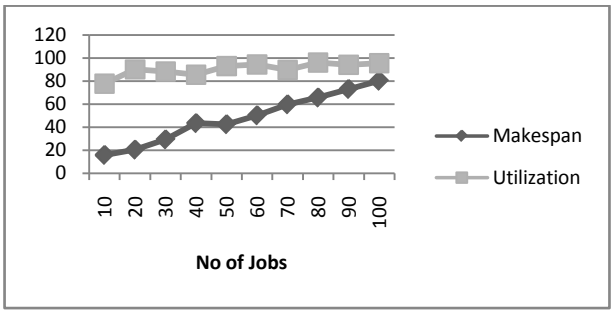

Figure 8: Utilization and makespan using EHF_ACS

6 Summary

The enhanced ACS algorithm that incorporates the new heuristic function was able to capture the new information that is used to update the heuristic value. The obtained information provides more experience to ants for their next iteration. No stagnation will occur when the updating activity is fixed to only one time for each edge in the whole process. Results show that EHF_ASC outperformed the traditional ACS in scheduling of jobs in the grid computational environment. EHF_ACS algorithm also shows high resource utilization rate for various number of jobs.

References:

[1] R. Wankar, Grid computing with Globus: An overview and research challenges,*International Journal Computer Science and Applications*, vol. 5(3), 2008,pp. 56-69.

[2] F. Magoules, J. Pan, K. A. Tan andA. Kumar, *Introduction to grid computing.* Boca Raton, FL: Taylor & Francis Group, 2009

[3] F. Xhafa andA. Abraham, Computational models and heuristic methods for grid scheduling problems,*Future Generation*

Computer Systems, vol. 26(4), 2010,pp. 608-621.

[4] F. Magoules, T. Nguyen andL. Yu, *Grid Resource Management Toward Virtual and Services Compliant Grid Computing*. Boca Raton, FL: CRC Press, Taylor & Francis Group, LLC, 2009.

[5] S. C. Lin, and E. Yen, *Grid Computing (International Symposium on Grid Computing*. New York, USA: Springer, 2009.

[6]F. Xhafa, and A. Abraham, *Metaheuristics for scheduling in distributed computing environments*. Heidelberg, Germany: Springer, 2010.

[7] P. Mathiyalagan, S. Suriya andS. Sivanandam, Modified Ant Colony Algorithm for Grid Scheduling,*International Journal on Computer Science and Engineering*, vol. 02(02), 2010, pp. 132-139.

[8] K. Kousalya andP. Balasubramanie, To Improve Ant Algorithm's Grid Scheduling Using Local Search,*International Journal of Intelligent Information Technology Application*, vol. 2(2), 2009, pp. 71-79.

[9] P. Visalakshi and S. N. Sivanandam, Dynamic Task Scheduling with Load Balancing using Hybrid Particle Swarm Optimization,*Int. J. Open Problems Compt. Math*, vol. 2(3), 2009, pp. 475-488.

[10] L. Wei, X. Zhang, Y. Li and Y. Li, An improved ant algorithm for grid task scheduling strategy,*Physics Procedia*, vol. 24(c), 2012, pp. 1974-1981.

[11] M. Dorigo, M.Birattari and T. Stutzle, Ant Colony Optimization, *Computational Intelligence Magazine*, vol. 1(4), 2006, pp. 28 – 39.

[12] M. Gendreau andJ. Potvin, *Handbook of Metaheuristics*. US: Sprink, 2010.

[13] L. M. Nithya and A. Shanmugam, Scheduling in Computational Grid with a New Hybrid Ant Colony Optimization Algorithm,*European Journal of Scientific Research*, vol. 62(1), 201, pp. 273-281.

[14] Y. Zhu and Q. Wei, An improved ant colony algorithm for independent tasks scheduling of grid,*Proceedings of the 2nd International Conference on Computer and Automation Engineering (ICCAE), China*, vol. 2, 2010, pp. 566 – 569.

[15] M. Malarvizhi and V.R Uthariaraj,A Minimum Time to Release Job Scheduling Algorithm in Computational Grid Environment,*Proceedings of the Fifth International Joint Conference on INC, IMS and IDC, Seoul, Korea*, 2009, pp. 13-18.

[16] K.R. Ku-Mahamud and H.J.A. Nasir, An enhanced ant colony algorithm for job scheduling in grid computing,*Proceeding of the Fourth Asia International Conference on Mathematical/Analytical Modeling and Computer Simulation, Kota Kinabalu, Malaysia*, 2010, pp. 40-45.

[17] D. Maruthanayagam and R. UmaRani,Enhanced ANT Colony algorithm for Grid Scheduling,*International Journal of Computer Technology and Applications*, vol. 1(1), 2010, pp. 43-53.

[18] M. Dorigo and T. Stutzle, *Ant colony optimization*. Cambridge, Massachusetts, London, England: MIT Press, 2004.

[19] S. Camazine, J. Deneubourg, N. Franks, J. Sneyd, G. Theraula, and E. Bonabeau, *Self-Organization in Biological Systems*. New Jersey, UK: Princeton University Press, 2003.

[20] M. Dorigo and L. M. Gambardella,Ant Colonies for the Traveling Salesman Problem,*Journal of BioSystem*, vol. 43(2), 1997, pp. 73-81.

[21] M. Dorigo and L.M. Gambardella,Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem,*IEEE Transactions on Evolutionary Computation*, vol. 1(1), 1997, pp. 53 – 66.

[22] S. Lorpunmanee, M. N. Sap, A.H. Abdullah and C. Chompoo-inwai, An ant colony optimization for dynamic job scheduling in grid environment,*International Journal of Computer and Information Science and Engineering*, vol. 1(4), 2007, pp. 207-214.

[23] L. Wei, X. Zhang, Y. Li, and Y. Li, An Improved Ant Algorithm for Grid Task Scheduling Strategy,*Physics Procedia*, vol.24(c), 2012, pp. 1974-1981.

[24] R. Sarker, H.A. Abbass and C. Newton, *Heuristic and Optimization for Knowledge Discovery*. USA: Idea Group Publishing, 2002.

[25] S. Ilie and C. Badica, Effectiveness of Solving Traveling Salesman Problem Using Ant Colony Optimization on Distributed Multi-Agent Middleware,*Proceedings of the 2010 International Multiconference on Computer Science and Information Technology (IMCSIT)*, Wisla, Poland, 2010, pp. 197 – 203.