AHSWDG:An Ant Based Heuristic Approach to Scheduling & Workload Distribution in Computational Grids

Rohit Saxena
M.Tech Scholar, Dept. of CSE, SRMSCET
Bareilly, India
saxena.rohit83@gmail.com

Ankur Kumar
M.Tech Scholar, Dept of CSE, SRMSCET
Bareilly, India
saxenaanksrms@gmail.com

Anuj Kumar
Research Scholar, Dept. of CSE, Gurukul Kangri
Haridwar, India
anurom_2001@rediffmail.com

Shailesh Saxena
Asst. Prof., Dept. of CSE, SRMSWCET
Bareilly, India
shaileshgla@gmail.com

***Abstract*—Due to the advent of technologies and large resource intensive applications, a large scale distributed and heterogeneous system like grids have emerged as popular platforms. Grid Computing is a kind of distributed computing that involves the integrated and collaborative use of geographically-dispersed resources. Hence, reliable resource sharing is required to process the huge amount of computational jobs across system. So, effective approaches are required for scheduling the jobs and balance the load distribution among the available resources. In this paper, a heuristic approach using Ant Colony Optimization for balanced workload distribution is proposed. In this, ants represent the submitted jobs while the ant's pheromone trail represents the computational capacity of the grid resources. The computational capacity of the resource is updated whenever the job is allocated to or released from it. In nutshell, the overall objective of the proposed Ant Based Heuristic Approach to scheduling & workload distribution (AHSWDG) is to distribute workload equally among the available resources. This research compares the proposed AHSWDG approach with the Random approach on the basis of finish time of the jobs and the utilization of grid resources in the system.**

Keywords—Computational Grids, Workload, Grid Resources, Grid Users, Grid Resource Broker, Grid Information Service, Ant Colony Optimization, Pheromone, AHSWDG.

I. INTRODUCTION

Scalable access to widely distributed resources can be done efficiently and effectively by grid computing. As the sharing, selection and aggregation of geographically-dispersed computing resources is possible through these computational grids which in turn present them varied as a single resource for solving large scale computing applications. This gave rise to the necessity of scheduling & workload distribution approach which takes into account the various requirements of grid environment [1]. The sole aim of job scheduling & workload distribution is to balance the entire load of the system and processing the submitted jobs at hand as soon as possible and return the output to the ever-demanding users. A scheduling & workload distribution strategy should be adaptive so as to adapt itself according to the varying situations of the entire environment and types of jobs. Therefore, there is a need for dynamic approach for job scheduling and workload distribution such as Ant Colony Optimization (ACO) which is appropriate for computational grids [2].

ACO is a heuristic technique for optimization introduced in early 1990's. The inspiring source of ACO is the foraging behavior of real ant colonies. Now, this behavior in relation to discrete optimization problems, continuous optimization problems and the problems in telecommunications such as routing & load balancing is exploited in artificial ant colonies for the search of approximate solutions [3]. ACO simulates the behavior of real ant colonies in nature in search of food and follow the pheromone trail laid by fellow ants on path from nests to food.

The rest of the paper is organized as follows. Section II introduces the related work about the job scheduling & balanced workload distribution in computational grids. Section III describes the system model using schematic diagram. Section IV details the proposed Ant-Based Heuristic Approach to Scheduling & Workload Distribution in Computational Grids (AHSWDG). Section V concludes the research with comparative analysis of AHSWDG with the Non-AHSWDG (Random) approach on the basis of finish time of the submitted jobs.

II. RELATED WORK

Distribution of workload of the system among the available resources has been a key area for research since decades. Researches have been performed to distribute the workload dynamically and statically. In [4], load balancing is performed at two levels: Intra-cluster (Inter-worker nodes) and Inter-clusters (Intra-Grid). In [6], centralized load balancing is performed in which the central node collects the load information from other computing nodes in the system and updates the same to other nodes to enable them to take load balancing decisions. In [2], balanced ant colony optimization algorithm has been proposed that inherits the basic ideas of ACO to decrease the computational time of jobs by local and global pheromone updates. In [7], enhanced ant colony optimization has been proposed that focus on the current status of the resources when new jobs are scheduled. The approach proposed in [8] is based on the general ant adaptive heuristics and an added load balancing guide component. The multiple ant colonies algorithm proposed in [9] improves the

978-1-4799-6023-1/15 $31.00 © 2015 IEEE
DOI 10.1109/CICT.2015.90

performance by considering the positive and negative feedbacks in search solutions and sharing search information. In [10], the proposed Ant Based Load Balancing Algorithm (ALBA) determines the best resource to be allocated to a job based on job characteristics and resource capacity.

The proposed approach AHSWDG find the most optimal grid resource to which the submitted job will be allocated for execution on the basis of its current computational capacity and while doing so it distributes the workload to the most optimal idle grid resource. In case of failure of allocated job on a grid resource, the failed job is moved to the Failed Job Queue and then finds the most optimal grid resource that is idle at that timestamp and allocates the failed job to that grid resource.

III. Heuristic Model of Computational Grids

The Computational Grids consists of the Grid Resources (GRs) that are registered with the Grid Information Service (GIS). A GR has heterogeneous or homogeneous set of machines that in turn have a cluster of processing elements. The registered GR execute the jobs submitted by the Grid Users (GUs). The jobs are not directly allocated to the GR rather the Grid Resource Broker (GRB) discovers the GR for the jobs submitted by the GUs by consulting the GIS. The GRB gets the list of GRs along with their respective characteristics (such as number of machines, MIPS Rating, operating system etc). It is the responsibility of the GRB to allocate the submitted job to the appropriate GR. In this heuristic model of computational grids, the GRB employs proposed ant-based approach for the scheduling & workload distribution of the submitted jobs. On allocation of job, the respective GRs break it into sub-jobs and allocate them among the set of machines depending on their respective computational capacities (MIPS Rating). The respective machines further break the sub-job equally and allocate them to processing elements. Processing elements process the allocated parts of job and returns output to the respective machines where it is combined into single sub-job and returned to the GR which sends the processed result to the GRB and hence to the GUs. The heuristic model for workload distribution is as shown in Fig. 1. In Fig. 1, it is assumed that there are *'r'* grid resources. Each grid resource has *'m'* machines. Each machine has *'x'* processing elements. GR_r denotes r^{th} grid resource while M_{rm} denotes m^{th} machine of r^{th} grid resource. Similarly, PE_{rmx} denotes x^{th} processing element of m^{th} machine of r^{th} grid resource.

IV. Proposed Ant Based Heuristic Approach to Scheduling & Workload Distributation in Computational Grids (AHSWDG)

In the proposed approach, we assume that the number of jobs (ants) in the Actual Job Queue *'J'* be *'n'*. Also, suppose the number of registered grid resources (*GRs)* in the system be *'r'*. For each resource, GR_i *(1 ≤ i ≤ r),* suppose the number of machines be *'m'*. For each machine *k (1 ≤ k ≤ m),* suppose the number of processing elements be *'x'*. The detailed control flow of the proposed AHSWDG is shown in Fig. 2.

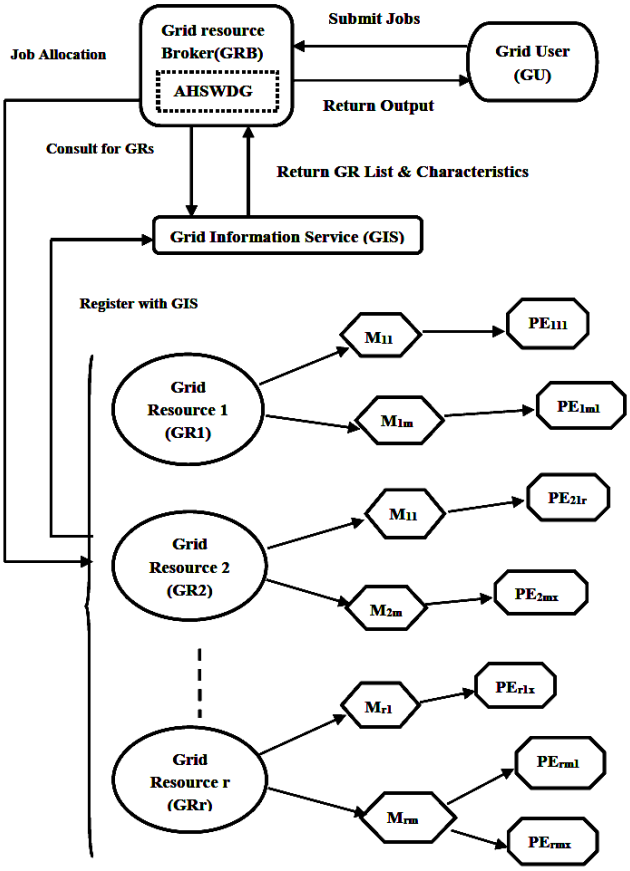

Fig. 1 Hueristic Model for Computational Grids

In AHSWDG, we associate the pheromone trail with the computational capacity (i.e. MIPS Rating) of the grid resource, GR_i. Hence, the initial pheromone trail represents the initial (i.e. the instant at which the grid resource is under-loaded or doing little work) computational capacity of the grid resource, GR_i. For machine *k*, all the processing elements has same MIPS (Million Instructions per Second) rating.

Hence, the initial computational capacity$(\tau_k(0))$ of machine *k* is

$$\tau_k(0) = x * (\text{MIPS Rating of One Processing Element}), \text{ for every } k, \text{ where } 1 \le k \le m \quad (1)$$

For grid resource, GR_i, the initial computational capacity$(\tau_i(0))$ is the sum of the initial computational capacity of the machines $(\tau_k(0))$ comprised by it. Hence,

$$\tau_i(0) = \sum_r \tau_k(0), \text{ for every } i, \text{ where } 1 \le i \le r \quad (2)$$

Allocation of job *'j'* to the grid resource, GR_i is done on the basis of the transition probability $'p_i(t)'$ for every *i,* $1 \le i \le r$. It is computed as:

$$p_i(t) = \frac{[\tau_i(t)]^{\alpha} * [\tau_i(0)]^{\beta}}{\sum_r [\tau_i(t)]^{\alpha} * [\tau_i(0)]^{\beta}} \quad (3)$$

where

α is the relative performance of the grid resource, GR_i, while it is executing the job.

β is the relative importance of grid resources while it idle.

The grid resource, GR_i , having maximum value of $p_i(t)$, *i.e.* $\boldsymbol{max\{p_i(t)\}}$, is considered eligible for executing the job.

At any instant *t*, i.e. when the grid resource, GR_i, is executing the allocated job *j, $(1 \le j \le n)$,* the job may successfully execute to completion or may fail to execute.

If the job *j, $(1 \le j \le n)$,* successfully executes to completion on the grid resource, GR_i, to which it was allocated, the computational capacity (or pheromone value) of that GR_i is updated as :

$$\tau_i(t) = \rho\tau_i(0) - \Delta \ , \quad (4)$$

$$\Delta = \quad * -C \ , \quad (5)$$

where

$\tau_i(t)$ *is updated computational capacity.*

$\tau_i(0)$ *is initial computational capacity computed in (2).*

ρ *is the pheromone decay parameter, i.e. the parameter that specifies the decay in computational capacity of grid resource, GR_i while the job is being executed and its value lies between 0 and 1.*

Δ *is pheromone variance, i.e. the parameter that specifies change in the computational capacity of grid resource, GR_i on job allocation.*

is Index of Overload.

C is computational complexity of allocated job.

Also, the Actual Job Queue, *'J'* is updated by removing job *j, $(1 \le j \le n)$,* from it, i.e.,

$$J = J - \{1\} \quad (5)$$

While the job *'j' $(1 \le j \le n)$* is under execution on the grid resource, GR_i *$(1 \le i \le r)$,* then the following steps are performed:

a. most optimal (if any) grid resource, OGR_i is obtained. Most optimal OGR_i is the one that has finished the execution of the job allocated to it and is idle,

b. job '*j'* i.e. currently executing job is paused at the current grid resource, GR_i, and resumed at the most optimal grid resource, OGR_i thus obtained in step (a).

c. Steps (a) & (b) are performed recursively until the job executes to completion.

d. Also, the respective computational capacities of the above said grid resources are updated.

If the job fails to executes on the grid resource, GR_i, to which it was allocated, the computational capacity (or pheromone value) of that GR_i is updated as :

$$\tau_i(t) = \rho\tau_i(0) + \Delta \ , \quad (6)$$

$$\Delta = \theta * -C, \quad (7)$$

where

$\tau_i(t)$ *is updated computational capacity.*

$\tau_i(0)$ *is initial computational capacity computed in (2).*

ρ *is the pheromone decay parameter, i.e. the parameter that specifies the decay in computational capacity of grid resource, GR_i while the job is being executed and its value lies between 0 and 1.*

Δ *is pheromone variance, i.e. the parameter that specifies change in the computational capacity of grid resource, GR_i on job allocation.*

θ *is Index of Underload.*

C *is computational complexity of allocated job.*

When the job *j, $(1 \le j \le n)$,* fails to execute on the particular grid resource, GR_i *$(1 \le i \le r)$,* then the following steps are performed:

a. failed job j_f, *$(1 \le j_f \le n)$,* is moved to Failed Job Queue, *FJ* .

b. computational capacity of the grid resource, GR_i, (which failed to execute the job) is updated by the (6).

c. most optimal (if any) grid resource, OGR_i is obtained and failed job j_f, *$(1 \le j_f \le n)$,* is allocated to it.

d. if OGR_i is not found then the failed job j_f, *$(1 \le j_f \le n)$,* is moved from Failed Job Queue, *FJ* to Actual Job Queue, *J.*

V. RESULTS & DISCUSSIONS

The proposed approach was simulated on the GridSim-5.2 Toolkit[11] with the help of MyEclipse 8.5 IDE using Java Programming Language. The simulation results of AHSWDG were obtained and were compared with the Non-AHSWDG, a built-in example of GridSim-5.2 Toolkit that uses Random (i.e. jobs are allocated to the grid resource whose resource id is computed randomly) mechanism. Following are the results:

Scenario 1. Comparison of Finish Time of proposed approach, AHSWDG with the Non-AHSWDG(Random) approach:

Simulations were performed by varying the number of jobs while keeping the number of resources 5 & 10 respectively.

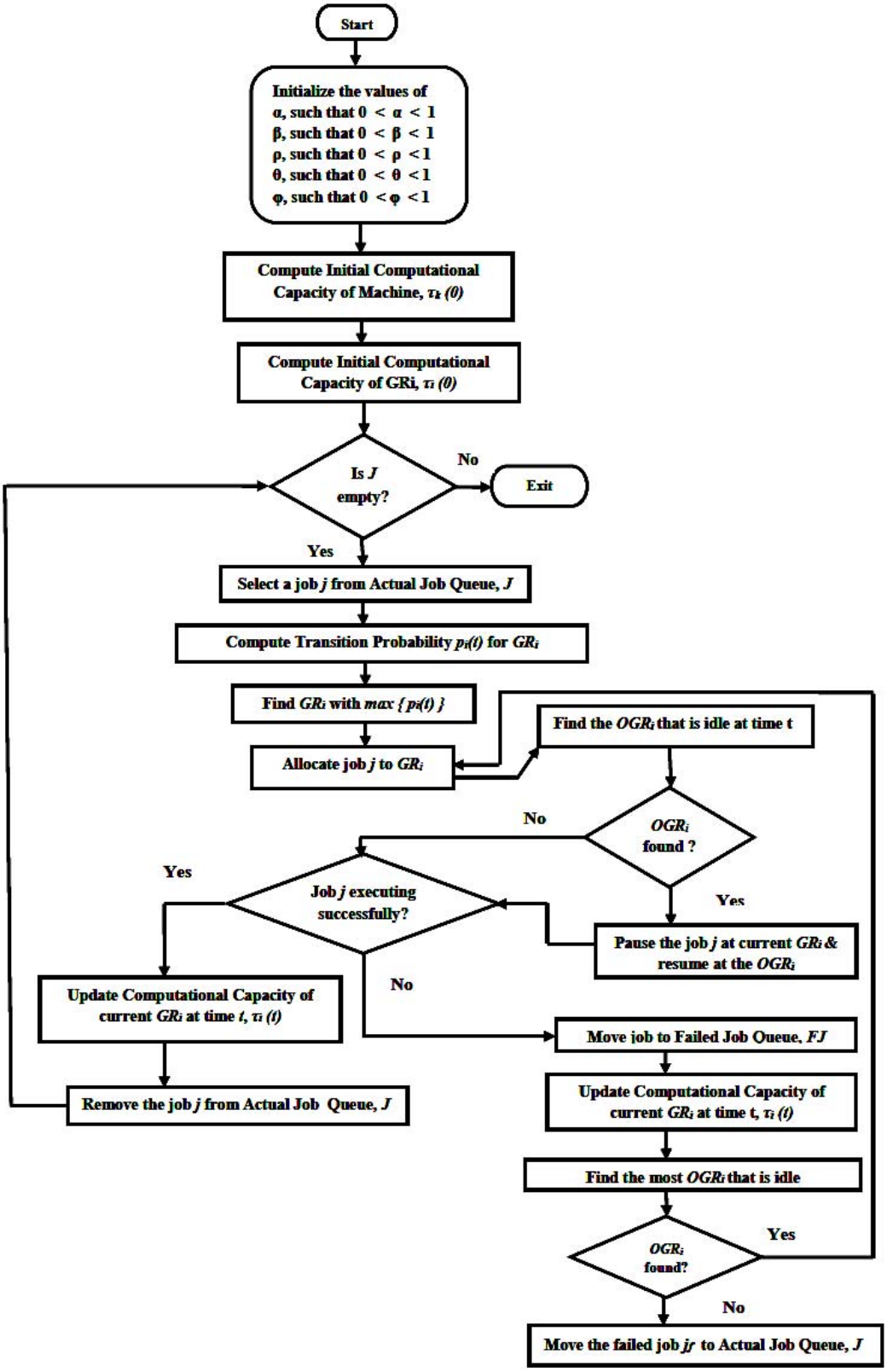

Fig. 2. Control Flow of AHSWDG

Table 1. Finish Time(secs) of Jobs when No. of Resources are 5 & 10 respectively

No. of Jobs	No. of Resources = 5		No. of Resources = 10	
	AHSWDG (Seconds)	Non-AHSWDG (Seconds)	AHSWDG (Seconds)	Non-AHSWDG (Seconds)
10	375.32	628.467	433.12	687.267
15	550.32	896.93	608.12	941.949
20	719.479	1020.59	720.27	1079.39
25	875.11	1263.36	932.92	1307.216
30	1036.279	1417.270	1094.08	1476.07
35	1150.66	1650.53	1261.16	1688.829
40	1376.28	1791.099	1380.08	1849.899
45	1535.68	2046.26	1593.48	2078.832
50	1615.48	2176.66	1753.28	2234.46
55	1864.4	2437.8	1890.2	2437.8043
60	2032.12	2576.068	2091.76	2632.86

The plots based on the above data are shown below:

Fig. 3. Comparison of Finish Time(secs) of AHSWDG with Non-AHSWDG(Random) respectively when Number of Grid Resources are 5.

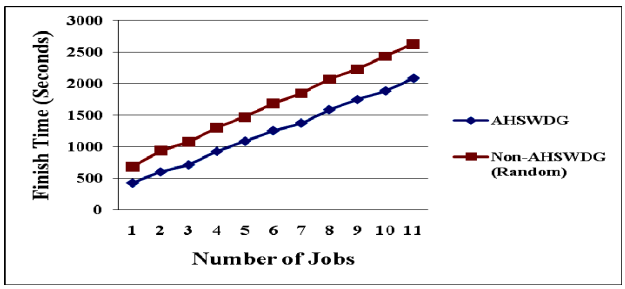

Fig. 4.Comparison of Finish Time(secs) of AHSWDG with Non-AHSWDG(Random) respectively when Number of Grid Resources are 10.

Scenario 2. Comparison of Resource Utilization of proposed approach, AHSWDG with Non-AHSWDG(Random) approach:

Simulations were performed by varying the number of jobs from 10 to 60 while keeping the number of resources constant, i.e. 10. Following are the results:

Table 2. Resource Utilization(%) : No. of Jobs are 10,15,20 & 25 respectively

	#JOBS=10		#JOBS=15		#JOBS=20		#JOBS=25	
	AHSWDG	Non-AHSWDG	AHSWDG	Non-AHSWDG	AHSWDG	Non-AHSWDG	AHSWDG	Non-AHSWDG
R1	8.22	0	5.86	5.86	8.64	7.66	8.64	10.6
R2	8.69	23.8	6.18	12.48	9.90	19.0	9.90	23.1
R3	28.7	7.95	25.6	40.80	20.9	9.17	20.9	21.2
R4	8.33	8.33	11.7	0	9.21	0	9.21	10.3
R5	7.02	44.5	5.00	5.18	8.95	3.61	8.95	4.03
R6	6.49	0	10.9	6.28	7.53	20.6	7.53	10.3
R7	9.18	8.81	6.54	6.28	9.32	9.82	9.32	3.12
R8	8.81	0	12.7	12.30	9.16	20.4	9.16	3.09
R9	7.95	6.49	5.63	0	8.54	0	8.54	11.0
R10	6.47	0	9.61	10.8	7.77	9.60	7.77	3.05

The plots based on the above data are shown below:

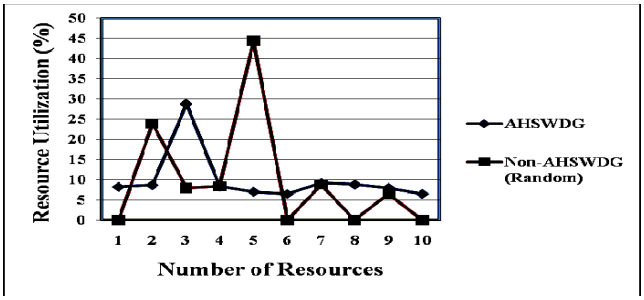

Fig. 5. Comparison of Resource Utilization(%) of AHSWDG with Non-AHSWDG respectively when Number of Jobs are 10.

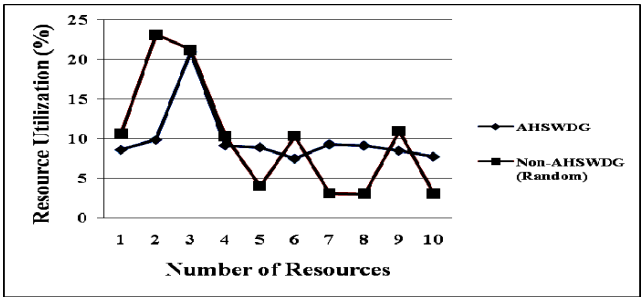

Fig. 6. Comparison of Resource Utilization(%) of AHSWDG with Non-AHSWDG respectively when Number of Jobs are 25.

VI. Conclusions & Future Work

As evident from the plots(Fig. 3 & Fig. 4) in scenario 1(Comparison of Finish Time of proposed approach, AHSWDG with the Non-AHSWDG(Random) approach), the proposed approach, AHSWDG, outperforms the Non-ASHWDG(Random) approach used in the example taken from GridSim-5.2 Toolkit. Further, the plot shows that with the

increase in number of resources, the proposed approach, AHSWDG, performs even better than the Non-AHSWDG(Random) approach used in example taken from GridSim-5.2 Toolkit.

Thus, we conclude that our proposed approach is better than the Non-AHSWDG(Random) approach we considered to compare. This point can be strengthened further by considering the scenario 2(Comparison of Resource Utilization of proposed approach, AHSWDG with Non-AHSWDG(Random) approach) with 10 resources & varying the number of jobs between 10 to 25. Table 2 represents this scenario and plots(Fig. 5 & Fig. 6) are depicting the variation.

It can easily be seen that the utilization of resources in our proposed approach is getting better with increase in no. of jobs i.e. all the grid resources are nearly equally utilized when no. of jobs are 25. While in the Non-AHSWDG, resources are unevenly utilized ie. percentage utilization of some resources is very less compared to very high utilization of other resources.

Therefore, on basis of the studies done through simulations in the previous section, it can be concluded that irrespective of the number of grid resources, the proposed AHSWDG outperforms the Non-AHSWDG(Random) approach used in the example of GridSim-5.2 toolkit.

Further, with the increase in the number of resources, the degree of improvement proves to be even remarkable. The reasons for this improvement can be listed as follows:

a. The proposed approach allocates the job to the most optimal grid resource, i.e., the grid resource that has maximum computational capacity (in Million Instructions Per Second) while the Non-AHSWDG(Random) mechanism in built-in example allocates jobs to grid resources in Non-AHSWDG fashion irrespective of the computational capacities of the grid resources.

b. The proposed approach employs the mechanism of migration of jobs to the most optimal and idle grid resource thus the execution of job takes lesser time duration than it would have taken usually.

c. The proposed approach also employs the mechanism for dealing with the jobs that fail to execute at the grid resource by moving the job into the Failed Job Queue(to keep track of failed job until it is executed) and then to allocate it to the most optimal idle grid resource rather than sending it back to the Actual Job Queue and adding it to the queue.

As the results present the comparison of the two aforesaid approaches on the basis the finish time of jobs and utilization of grid resources, efforts can be made to compare the aforesaid approaches on the basis of the other performance metrics such as throughput, average wait time and reliability metrics such as execution costs & communication costs.

VII. REFERENCES

[1] P. Keerthika and N. Kasthuri, "A Hybrid Scheduling Algorithm with Load Balancing for Computational Grids," Interational Journal of Science & Technology, vol. 58, pp. 13-28, 2013.

[2] Ruay-Shiung Chang*, Jih-Sheng Chang, Po-Sheng Lin, "An Ant Algorithm for Balanced Job Scheduling in Grids," Future Generation Computer Systems, 2009 , pp.20-27.

[3] Christian Blum, "Ant Colony Optimization: Introduction and Recent Trends," Physics of Life, Science Direct, pp. 271-350.

[4] Belabbas Yagoubi and Meriem Meddeber, "Distributed Load Balancing Model for Grid Computing," Revue ARIMA Journal, vol. 12 (2010), pp. 43-60

[5] R. Nicole, "Tree Based Approach For Load Balancing In Grid Environment," International Journal of Engineering Research & Technology, Vol. 1 Issue 9, November- 2012, ISSN: 2278-0181.

[6] Priyanka G. Gonnade(Kohale), and Sonali Bodkhe, "Genetic Algorithm for Task Scheduling in Distributed Heterogeneous System," International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 10, October 2012, ISSN: 2277 128X.

[7] Ku Ruhana Ku-Mahamud, Husna Jamal Abdul Nasir and Aniza Mohamed Din, "Grid Load Balancing Using Enhanced Ant Colony Optimization," Proceedings of the 3rd International Conference on Computing and Informatics, ICOCI, 2011,8-9 June, 2011 Bandung, Indonesia, pp. 37-42.

[8] Sowmya Suryadevera, Jaishri Chourasia, Sonam Rathore Abdul and Jhummarwala, "Load Balancing in Computational Grids Using Ant Colony Optimization Algorithm," International Journal of Computer & Communication Technology (IJCCT) ISSN (ONLINE): 2231 - 0371 ISSN (PRINT): 0975 –7449 Vol-3, Iss-3, 2012.

[9] S.Umarani, L.M.Nithya and A.Shanmugam, "Efficient Multiple Ant Colony Algorithm for Job Scheduling In Grid Environment," International Journal of Computer Science and Information Technologies, Vol. 3 (2) , 2012,3388-3393, ISSN: 0975-9646

[10] Sandip Kumar Goyal and Manpreet Singh, "Adaptive and Dynamic Load Balancing in Grid Using Ant Colony Optimization," International Journal of Engineering and Technology, Vol. 4 No 4 Aug-Sep 2012,pp. 167-174, ISSN:0975-4024

[11] Rajkumar Buyya, Manzur Murshed, "GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing".

[12] Y. Li, "A Bio-inspired adaptive job scheduling mechanism on a computational grid," International Journal of Computer Science and Network Security (IJCSNS), vol. 6, no. 3, pp. 1-7, 2006.

[13] Liu., A. Wang, " Grid task scheduling based on adaptive ant colony algorithm," in Proc. Inter. Conf. Management of e-commerce & e-Government, 2008, pp. 415-418.

[14] A. Ali, M. A. Belal, and M. B. Al-Zoubi, "Load balancing of distributed system based on multiple ant colonies optimization," Journal of Applied Sciences, vol. 7, no. 3, pp. 433-438, 2010.

[15] H. Yan, X. Shen, X. Li, and M. Wu, "An improved ant algorithm for job scheduling in grid computing," in Proc. 4th Inter. Conf. Machine Learning and Cybernetics, 2005, pp. 2957-2961.