International Journal of Computational Intelligence and Informatics, Vol. 4: No. 1, April - June 2014

Comparative Study of Job Scheduling In Grid Environment Based on BEE`S Algorithm

M Abdullah
Department of Computer Science
Jamal Mohamed College, Trichy-20.

P Selvi
Department of Computer Science
Jamal Mohamed College, Trichy-20.

***Abstract*- Grid computing is a new computing model; it's performing bulk operations among the connected network. Since it is possible to run multiple applications simultaneously may require multiple resources but often do not have the resources that time scheduling system is essential for allocating resources. Scheduling algorithms must be designed for Maximize utilization of the resources and minimize the execution time. In this Paper, comparing the make span and execution time of the bee's algorithm such as Artificial Bee Colony Algorithm, GA based ABC and an Efficient Modified Artificial Bee Colony Algorithm. The result shows that An Efficient Modified Artificial Bee Colony Algorithm will give better result compared to other algorithms.**

Keywords- Grid Computing, Task Scheduling, Artificial Bee Colony Algorithm, GA Base ABC, An Efficient Modified Artificial Bee Colony Algorithm

I. INTRODUCTION

Grid Computing is the combination of computer resources from multiple administrative domains to reach common goal. Grid computing provides a high performance computing platform to solve large scale applications by coordinating and sharing computational power, data storage and network resources across dynamic and geographically dispersed organizations. Grid Computing is an infrastructure that enables the integrated collaborative use of high-end computers, networks, databases and scientific instruments owned and managed by multiple virtual organizations [1]. Grid applications often involve large amount of data and/or computing and often require secure resources sharing across organizational boundaries. Grid computing is distributed that over a network to solve complex, massive, computation-intensive problems. The size of a grid may vary from small confined such as a network of computer workstations within a corporation to large such as public collaborations across many companies and networks [2]. The goal of grid computing is high performance, achieved by using spare computer power at little or no cost.

II. RELATED WORK

Schedulers are types of applications responsible for the management of jobs such as allocating resources needed for any specific job, partitioning of jobs to schedule parallel execution of task, data management, event correlations, and serial-level management capabilities. These schedulers then form a hierarchical structure, with meta-schedulers that form the root and lower level schedulers. The local scheduler is used for execute the specific job and meta-schedulers or a cluster scheduler for parallel executions [3]. The schedulers submit the user job to respective available resources based on the service level requirements. This will involve complex workflow management and data movement activities to occur on a regular basis. There are schedulers that must provide capabilities for areas such as

1. Advanced resource reservation
2. Serial-level argument validation and enforcement.
3. Job and resource policy management and enforcement for best turnaround times within the allowable budget constraints.
4. Monitoring job execution and status.
5. Rescheduling and corrective actions of partial fail over situations.

Job scheduling is known to be NP-complete, therefore the use of heuristics is the de facto approach in order to cope in practice with its difficulty. Thus, the Meta heuristics computing research community has already started to examine this problem. Ritchie and Levine used Single heuristic approaches for the problem in Local Search. QoS parameters such as make span, resource utilization, and security are found to be crucial for the performance of grids. Besides make span and cost, some recent researches also take security and reliability into account. Literature presents an iterative scheduling algorithm for a set of independent tasks in computational grids.

Yarkhanan and Dongarra used simulated annealing for grid job scheduling. The Simulated Annealing scheduler is compared to an Ad Hoc Greedy scheduler used in earlier experiments. The Simulated Annealing scheduler exposes some assumptions built into the Ad Hoc scheduler and some problems with the Performance Model being used [4]. Tabu Search and Simulated Annealing for scheduling jobs on computational grids. Some hybrid heuristic approaches have also been reported for the problem. The great performance of ACO algorithms for scheduling problems has been demonstrated in and ant colony system (ACS) algorithm, which is one of the ACO algorithms. D.Jeya Mala, V. Mohan used Artificial Bee Colony Algorithm for Software Testing Approach. Adil Baykasolu, Lale Ozbakir and Pinar Tapkan used Artificial Bee Colony Algorithm [5] and its application to Generalized Assignment Problem for grid scheduling. This paper takes the advantage of the ABC for the grid workflow scheduling problem.

Genetic Algorithm for grid job scheduling is addressed in several works. The benefits of the usage of the Genetic Algorithms to improve the quality of the scheduling are discussed. The result of this paper suggests the usage of local search strategy to improve the convergence when the number of jobs to be considered is big as in real world operation. In this paper, proposed an evolutionary approach to solve Grid job scheduling problem with time uncertainties. The aim of the optimization problem was to find the best Correct Fitness value at fastest convergent speed without losing stability.

III. ARTIFICIAL BEE COLONY ALGORITHM

Artificial Bee Colony (ABC) algorithm is an optimization algorithm based on the intelligent foraging behavior of honey bee swarm. In common ABC algorithm is used for optimizing multivariable functions. In ABC model, the colony consists of three groups of bees: employed bees, onlookers and scouts. It is assumed that there is only one artificial employed bee for each food source. That is, the number of employed bees in the colony is equal to the number of food sources around the hive. Employed bees go to their food source and come back to hive and dance on this area. The employed bee whose food source has been discarded becomes a scout and starts to search for finding a new food source. Onlookers watch the dances of employed bees and choose food sources depending on dances. The main steps of the algorithm are given below:

- Initialize
- Repeat
    1. Place the food sources on the employed bee's memory; Determines neighbor source, then evaluates its nectar amount and dances in the hive;
    2. Each onlooker watches the dance of employed bees and chooses one of their sources depending on the dances, and then goes to that source; After choosing a neighbor around that, evaluates its nectar amount; Place the onlooker bees on the food sources in the memory;
    3. Send the scouts to the search area for discovering new food sources; The best food source found so far is registered.
- Until (requirements are met)

In ABC, which is a population based algorithm, the position of a food source represents a possible solution to the optimization problem and the nectar amount of a food source corresponds to the quality of the associated solution. Providing that its nectar is higher than that of the previous one, the bee memorizes the new position and forgets the old one. The sources abandoned are determined and new sources are randomly produced to be replaced with the abandoned ones by artificial scouts. Several applications of ABC algorithm includes Biological simulation, Genetic Algorithm Improvement, Continuous Optimization, Travelling Salesman Problem (TSP), Ride- Matching Problem, Dynamic Allocation of Internet Service, Telecommunication Network Routing, Large Scale Precise, Job Shop Scheduling.

IV. GA BASED ABC ALGORITHM

This algorithm is the implementation of ABC algorithm with the key concept of GA technique. This algorithm provides various allocations of tasks to the available resources. The resource matrix is given as input to the algorithm. The probabilities of execution time calculated for each and every machine. Then allocate the task to the resource which executes the task in least time. Likewise all tasks allocated to the resources. After the allocation of tasks to the best available resources the makespan will be calculated. The Single Shift Neighborhood (SSN), Double Shift Neighborhood (DSN) and Ejection Chain Neiborhood (ECN) techniques will be applied on that scheduling and the respective makespans will be calculated. Then the makespans are compared for choosing the best scheduling. On the selected allocation the GA technique will be applied for better performance of the algorithm [6].

This algorithm implements various allocation techniques of tasks to the available resources. The resource matrix is given as input to the algorithm. Based on the size of the matrix the random process time assigned to the available resources. The probabilities of execution time calculated for each and every machine.

The algorithm can be viewed in the following procedure:

1) **Initialization of algorithm:** The maximum value of task's process time and resource's capacity are initialized at the beginning of the algorithm.
2) **Assignment of Resources:** Calculate the process time of each and every job in all available resources. Assign the resource which executes the job at minimum time.
3) **Makespan:** Calculate the makespan and resource utilization for the above assignment.
4) **Single Shift Neighborhood:** Apply Single Shift Neighborhood for each task to increase the resource utilization and calculate makespan.
5) **Double Shift Neighborhood:** Apply Double Shift Neighborhood for each task based on the unused resources.
6) **Ejection chain Neighborhood: Apply Ejection Chain Neighborhood and calculate makespan.**
7) **GA Technique:** Among the best schedule get so far applied to GA process to reduce makespan and to increase resource utilization.
8) **Terminal test:** Find the best optimal solution from the solution construction.

V. AN EFFICIENT MODIFIED ARTIFICIAL BEE COLONY ALGORITHM

In this proposed method, one additional step is added to standard Artificial Bee Colony Optimization is of mutation operator. Mutation operator is added after the employed bee phase of artificial bee colony algorithm. ABC algorithm has four phases initialization phase, employed bees phase, onlooker bees phase and scout bees phase, we add mutation phase after the employed bee phase. Employed bee phase do the local search and mutation after the employed bee phase explore the search space and do the searching new area of solution space. Through mutation, on the one side, there is a chance of changing the local best position, and the algorithm may not be trapped into local optimum [7]. On the other side, individual can make use of the others' advantage by sharing information mechanism. In this method, the mutation step is carried out on the probabilistic way in each food searching operation for each iteration during the life cycle of ABC optimization technique. Selection of food source is done in a random manner. Food Source is selected arbitrarily from the food size and mutation is performed. In mutation, generated offspring's replaces the older offspring's. The mutation operator used in this paper is uniform mutation. When performing mutation, we randomly select one food source x_{ij} and replace its one of the dimension value by random number generated in between lower and upper bound value of the food source.

Below the proposed ABC with mutation after employed bee phase is shown. First phase is initialization phase where individuals are randomly selected from the search space.

Algorithm 1: ABC with Mutation after Employed Bee Phase

[Initialization Phase]

```
for i=0 to max number of Food source NF do
        for d=0 to dimension size do
                Randomly initialize food source
        positions X_ij
        end for d
        Compute fitness of each food source
end for i
Repeat
```

[Employed Bee Phase]

```
for i=0 to max no of employed bee do
        for d= 0 to dimension do
                produce new candidate solution
        end for d
```

Compute fitness of individual

if fitness of new candidate solution is better than the existing solution replace the older solution.

end for i

for i = 0 to max number of food source NF do Calculate the probability for each food source.

end for i

[Crossover phase]

If crossover criteria satisfies

For i=0 to maximum no. of food source

Select two random individuals from the current population for crossover operation.

Apply crossover operation

New offspring generated from parents as a result of crossover. Replace the worst parent with the best new offspring if it is better.

End of i

[**Onlooker Bee Phase]**

for i=0 to max no of onlooker bee do

choose food source on the basis of probability P_i

For d= 0 to dimension do

Produce new candidate solution for food source position X_{ij}

End for d

Compute fitness of individual food source

If fitness of new candidate solution is better than

the existing solution replace the older solution.

End for i

[Mutation Phase]

If mutation criteria is met then

Select random particle from current population for mutation operator.

Apply mutation operation to generate new individuals new offspring generated from the result of mutation.

New set of sequence is generated for offspring

Compute the cost for that offspring

Compute the fitness of updated individual

[**Scout Bee Phase]**

If any food source exhausted than

Replace it by new random position generated by scout bee.

Memorize the best food source so far

Until (Stopping criteria met).

Next is employed bee phase where local search is performed by employed bees. Next we have added additional operator to the ABC algorithm to check whether it improves the performance of algorithm. We have added a mutation operator after employed bee. Mutation is performed on the individual food source or individual if mutation probability satisfied. Onlooker phase and scout bee phase performed on current population or food sources after the mutation phase. At last best food source or individual is considered as global best solution.

VI. PERFORMANCE EVALUATION

The GA based algorithm is developed and the result is compared with ABC. In ABC, we use different techniques like Single Shift Neighborhood (SSN), Double Shift Neighborhood (DSN) and Ejection Chain Neighborhood (ECN) for resource allocation. Then one of the Genetic Algorithm (GA) techniques implemented. Among the above calculated make spans the best schedule is given to the Grid scheduler who has minimum make span and the maximum resource utilization. First ABC with minimum make span value among SSN, DSN, and ECN techniques selected is compared with GA. ABC with SSN shows better result as compared with other techniques. Then the ABC Algorithm use the Mutation and Crossover Operator techniques are the better result compared to other algorithm.

EXPERIMENT 1:

TABLE I. COMPARISON OF MAKESPAN PRODUCED BY DIFFERENT ALGORITHMS

Iteration	**(Task, Resource)**	**ABC Algorithm**	**ABC with GA Algorithm**	**An Efficient ABC Algorithm**
100	(50,10)	133.23	98.34	87.56
	(50,20)	128.34	95.20	83.67
	(100,10)	98.354	89.87	77.34
	(300,10)	123.78	97.356	88.78
	(300,20)	235.56	117.54	99.67
300	(50,10)	167.23	95.34	84.56
	(50,20)	1338.34	187.20	73.67
	(100,10)	928.354	829.87	97.34
	(300,10)	1123.78	397.356	78.78
	(300,20)	95.56	77.54	59.67
Iteration	**(Task, Resource)**	**ABC Algorithm**	**ABC with GA Algorithm**	**An Efficient ABC Algorithm**
100	(50,10)	3.23	2	2
	(50,20)	8.34	5.20	3.6
	(100,10)	3.4	2.87	1
	(300,10)	2	1.35	0.78
	(300,20)	6	4	2
300	(50,10)	16.23	9.34	6.7
	(50,20)	8.34	7.20	5
	(100,10)	6	5	4
	(300,10)	3.78	3	2
	(300,20)	6.2	5.4	4

.

Above Table is comparing the Runtime of different Algorithm.

VII. CONCLUSION

Grid computing presents a new trend in distributed systems and Internet computing for coordinating large scale heterogeneous resources sharing and problem solving in dynamic organizations. Grid computing allow the challenges such as security, computational economy, uniform access, system management, resource discovery, resource allocation and scheduling, data locality, and network management. Grid scheduling is the process of scheduling applications over Grid resources. Grid Scheduling in the heterogeneous and dynamic nature of grid resources continues to be a tedious task. This paper is proposed with the packages of task scheduling, scheduling algorithms and a hybrid task scheduling algorithms with various factors. In this paper, a heuristic Algorithm is developed based on the QoS make span and resources utilization. Results show that GA based ABC technique is better than ABC with respect to the QoS constraints such as make span and resource utilization. An efficient modified ABC algorithm shows better result for high resource matrix in most of the cases. Future work will focus on inclusion of Deadline and Load balancing the resources and to implement and evaluate different QoS and different Job error ratio.

REFERENCES

[1] Foster, C. Kesselman, and S. Tuecke. "The anatomy of the Grid: Enabling Scalable Virtual Organizations," International Journal of Supercomputing Applications, pp.200-222, Fall.2001.

[2] Foster and C. Kesselman, editors, "The Grid: Blueprint for a New Computing Infrastructure," Morgan Kaufmann Publishers, 1999.

[3] O. H. Ibarra and C. E. Kim, "Heuristic algorithms for scheduling independent tasks on non-identical processors," J. Assoc. Compute, pp.280-289, 1977.

[4] A. Yarkhan, J. Dongarra, "Experiments with scheduling using simulated annealing in a grid environment," in: 3rd International Workshop on Grid Computing, 2002.

[5] Abraham, R. Buyya, and B. Nath. "Nature's heuristics for scheduling jobs on computational Grids," In The 8th IEEE International Conference on Advanced Computing and Communications (ADCOM 2000), India, 2000.

[6] A. Wright, "Genetic Algorithms for Real Parameter Optimization, Foundations of Genetic Algorithms," G. Rswlins(Ed.), Morgen Kaufmann publishers, CA, pp.205-218, 1991.

[7] Dervis Karaboga, Bahriye Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," J Glob Optim, pp.459-471, 2007.