A CUSTOMIZED TASK SCHEDULING IN CLOUD USING GENETIC ALGORITHM

M.Rathna Devi[1], A.Anju[2]

[1]A.P/Department of IT, KCG College of Technology, Chennai
[2]A.P/Department of IT, KCG College of Technology, Chennai

Abstract

Cloud computing is an emerging technology in distributed computing which provides pay per use according to user demand and requirement. The primary aim of the Cloud computing is to provide efficient access to distributed resources. Scheduling of task is a critical issue in cloud computing, because it serves many users. The An approach for categorizing the tasks as Hard Real-Time Tasks (critical tasks that need to be completed on time with high rates of confidentiality) and Soft Real-Time Tasks (tasks that can be completed with certain delay and still can be efficient in its own way) before they are scheduled is applied. From the results observed the efficient processor for a particular combination of the tasks is determined thus producing customized results for each of the tasks. Efficient task scheduling is of high criticality for obtaining high performance in heterogeneous multiprocessor systems. Since task scheduling is a NP-hard problem, The Genetic Algorithm, an Evolutionary Algorithm which make use of techniques inspired by evolutionary biology such as inheritance, mutation, selection and crossover that is capable of producing optimal solutions.

***Keywords**-Task Scheduling, NP-hard problem, Genetic Algorithm, Hard Real-Time Tasks, Soft Real –Time Tasks*

--***--

1. INTRODUCTION

Cloud computing is a recent and emerging technology which aims to share resources, data, storage, infrastructure and software through internet access based on user demand. Cloud provides three types of services which are Infrastructure as a Service (IaaS), which provides infrastructure such as storage and computation resources to cloud users. Platform as a Service (PaaS),in which customer can deploy their application in that platform. Software as a Service (PaaS), which provides the software to the user without installation on their own machines and they can use directly from the cloud.

Initially the cloud customer will come to know about the services provided by the cloud. According to the need customer will request the cloud provider. If the cloud provider agrees to provide service then an agreement will be signed between the client and server before the deployment of the task. There will be a resource monitor to monitor both the deployment tool and the resources. Client will access the resource within the contract period and it will be pay per use policy.

Cloud computing, now-a-days used by many IT companies. The facilities of the cloud are provided through Internet. So, the customers can have systems with less configuration with only internet facilities. They can hire a platform in cloud, make use of the software in the cloud, do their computations in the virtual machines and they can store their informations in the storage provided by the cloud.

The main issue in Cloud computing is scheduling of the task and resource management. Scheduling means allocating the task to the available resources. But in cloud computing the allocation is purely based on profit to the cloud provider. Since many customers can request the service of the cloud at the same time, the cloud scheduler has to use some intelligent algorithm to schedule the task by considering the profit to the provider. This problem can be solved by categorizing the task based upon its importance and allocating the processor type based upon its requirements. By categorizing the task we can utilize the resources efficiently.

A scheduling technique can utilize the resources efficiently. Genetic algorithm is one proven algorithm for scheduling the task. This paper proposes refining of the basic genetic algorithm.

The rest of the paper is composed as follows. Section II it describes the related work for scheduling in cloud computing. Section III describes about problem statement. Section IV discusses about the proposed work. Section V is discusses the results. Section VI tells about the conclusion of this paper.

2. RELATED WORK

The basic ideas about scheduling in Cloud Computing and scheduling techniques are discussed in [1] and [2]. Paper [3] compares different types of workflow scheduling algorithms. Paper [4] describes genetic simulated annealing algorithm for optimized task scheduling by considering multiple objectives. Paper [5] author presented an optimized algorithm for task scheduling based on Activity Based Costing (ABC). Paper [6] describes Ant colony optimization in which random optimization Search approach is used for allocating the incoming Jobs to the virtual machines. Paper [7], proposed Particle Swarm Optimization (PSO) algorithm for workflow scheduling. . Paper [8] proposed a genetic algorithm based scheduler which makes the scheduling decision by evaluating the entire group of tasks in a job queue and minimizes makespan and better load balancing.

3. PROBLEM STATEMENT

There are a variety of tasks that need to be executed in the cloud computing in the real world with efficient execution time and best utilization of the resources available in the cloud Environment. The customized approach is followed in this paper to achieve better off results to the original scheduling in cloud computing. A solution that can be best utilized for providing efficient make span rate along with best use of the cloud resources is bought about to light. Thus the solution determined should fulfill both the criterion.

4. PROPOSED SOLUTION

An approach for categorizing the tasks as Hard Real-TimeTasks (critical tasks that need to be completed on time with high rates of confidentiality) and Soft Real-Time Tasks (tasks that can be completed with certain delay and still can be efficient in its own way) before they are scheduled is applied. A combination of the tasks with their corresponding processor speed is tabulated. From the results observed the efficient processor for a particular combination of the task is determined thus producing customized results for each of the tasks. After categorizing the task and processor Genetic algorithm is applied to find out the optimal schedule, so that to minimize the completion time and flowtime. The results obtained from the existing and the proposed concepts are compared to determine the efficient scheduling of tasks. The Soft real time tasks can be scheduled to i3 and i5 processor (i.e) if a soft real time task is arriving and an i3 processor is free, that job is given to that processor or if no i3 processor is free it can be scheduled to i5 processor but never to i7 processor. The Hard real time tasks can be scheduled to i5 and i7 processor (i.e) if a hard real time task is arriving and i5 processor is free at that time it scheduled to that machine or if i5 processor is not free, it can be scheduled to i7 processor and never to i3 processor.

4.1 Categorization of Tasks

In order to achieve a customized approach in the tasks scheduling, the tasks are initially categorized as Hard Real-Time Tasks and Soft Real –Time Tasks based on their rate of confidentiality and the also the rate at which the tasks need to be completed.

There are various kinds and various streams of engineering disciplines for which the tasks need be scheduled with high efficiency. Some of the disciplines include,

1. Image Processing
2. Marketing
3. Biometrics applications
4. Robotics
5. Media Files

HARD REAL-TIME TASKS
BIOMETRICS
BANKING TRANSACTIONS
ROBOTICS
STOCK EXCHANGE
ASTRONOMICAL

SOFT REAL –TIME TASKS
NETWORK SECURITY
MEDIA FILES
MARKETING
DATA MINING
IMAGE PROCESSING

The Processors are then identified for the tasks that are to be scheduled using the customized approach in scheduling using GA.

Let us consider the most widely used processors from the Intel Family namely the I3, I5, I7 processors.

I3 Processor

- Clock rate of 2.0Ghz to 2.50GHz
- Slower compared to i5 and i7
- Capable of simple tasks
- Applied for SS combination
- Eg:-SS-Mediafile:Image Processing

I5 Processor

- Clock rate of 2.67GHz to 2.87GHz
- Intermediary speed between i3 and i7
- Capable of mediocre level applications
- Applied in both SH and HS combinations
- Eg: HS-Stock Exchange :Marketing SH-Image Processing :Security

I7 Processor

- Clock rate of 1.7GHz to 3.7GHz
- Comparatively faster to i5 and i3 processors
- Capable of executing high-end applications
- Applied to HH Combination of Tasks
- Eg: Data mining : Media Files

Once the Processors are identified then the scheduling is done using the Genetic Algorithm to find the optimal schedule.

4.2 Genetic Algorithm

Genetic algorithms (GAs) are computerized search and optimization algorithms based on the mechanics of natural genetics and natural selection and comes under the class of the non-traditional search and optimization techniques used for searching large solution spaces [9]. It is a heuristic based evolutionary algorithm. The template given in below has been used for this work.

```
Initialization: Create the initial population P of n individuals
Fitness: Calculate the fitness of each individual of the population P.

while (stopping criteria not met)
{
    Selection: Select a subset of individuals from P.
    Crossover: Perform cross-over on each of the selected individuals with probability $P_c$
    Mutation: With probability $P_m$, mutate each individual
    Fitness: Evaluate the fitness of each individual in the new population.
}
return Best found solution
```

1) Representation: The representation of individuals of the population is a key issue in evolutionary-like algorithms, individuals referring to a solution of the problem. Representation determines the type of operators that could be used to ensure the evolution of the individuals and also impact on the feasibility of individuals. Different types of representations are reported in the literature. A direct representation is used here the advantage being that it defines a feasible schedule in a straightforward way and is obtained as follows. Each individual is represented as a vector, of size equal to the number of tasks to be scheduled and entry in position i indicates the processor to which task i is assigned. The values in this vector are in the range (1, number of processors). It is to be noted that a processor number can appear more than once.

2) Initialization: During initialization, a set of individuals are generated randomly from a uniform distribution to form a solution space of the desired population size.

3) Fitness function: The problem is formulated as a multi-objective problem to achieve minimum makespan that is, the time when last task is finished and flowtime that includes minimizing the sum of finalization times of all the tasks. These two criteria are defined as follows:

$$\text{makespan:} \min_{S_i \in Sched} \{\max_{j \in Jobs} F_j\} \qquad (1)$$

$$\text{flowtime :} \min_{S_i \in Sched} \{\sum_{j \in Jobs} F_j\} \qquad (2)$$

where F_j denotes the time when task j finalizes, Sched is the set of all possible schedules and Jobs the set of all jobs to be scheduled.

A weighted sum function is used here since makespan and flowtime are measured in the same unit. Here, the value of mean flowtime, flowtime/number of machines, is used to evaluate flowtime. Both values are weighted in order to balance their importance. Fitness value is thus calculated as:

$$\text{fitness} = w_1 \text{makespan} + w_2 \,.\, \text{mean flowtime} \qquad (3)$$

Where weights w_1 and w_2 are chosen such that $w2=1-w_1$ and $w_1+w_2=1$¸ which can be fixed by roper tuning. As the aim is to minimize the combined objective function, the individual that has a lower fitness value will be the best one.

4) Selection: Selection operators are used to select good individuals and forms a mating pool to which the crossover operators will be applied. For this problem a Binary Tournament Selection is used. The Selection is done by making two randomly selected individuals to participate in the tournament and choose the best among them. This scheme probabilistically duplicates some chromosomes in the next generation. Research find out this is better than other selection methods suitable for the problem [10].

5) Crossover: The aim of any evolutionary algorithm is to obtain descendants of better quality that will feed the next generation and enable the search to explore new regions of solution space not explored yet. Crossover achieves this by selecting individuals from the parental generation and interchanging their genes, to obtain new individuals. Depending on the representation of individuals, a single point crossover and fitness based crossover is used in this problem.

In single point crossover, the crossover operation selects a random pair of individuals and chooses a random point in the first individual. For the sections of both individuals from that point to the end of each individual, crossover exchanges machine assignments between corresponding tasks. Every individual is considered for crossover with a certain probability.

In fitness based crossover, a crossover mask is computed using the fitness of the individual as follows. For all tasks *i = 1 to number of tasks*

individual[i] = individual₁ [i],

if individual$_1$ [*i*]= individual$_2$ [*i*] (4)

otherwise *individual[i]= individual$_1$ [i]*with probability p = f2/(f1 + f2);

individual$_2$ [i], with probability 1-p; (5)

where f1 = fitness(individual$_1$) and f2 = fitness(individual$_2$). If the two parent individuals have similar fitness then the genes of the new descendants are calculated with probability ~0.5 .When there is a large difference in the fitness of the two parent individuals schedules, then it is quite probable that a chromosome of new structure will be obtained.

6) Mutation: ***:***Mutation is a alters one or more gene values in an individual from its initial state resulting in new individuals helping to arrive at better solution than was previously possible. This also prevent the population from stagnating at any local optima. Mutation occurs based on a user-definable mutation probability usually set fairly low otherwise the search will turn into a primitive random search. In this paper, move mutation operator and based on the direct representation of the individual, taking into account the specific need of load balancing of resources a rebalancing mutation is used.

The swap mutation operator interchanges the allocation of two jobs since movement of jobs between machines are considered to be effective. This operator should be applied to two jobs assigned to two different machines.

In rebalancing mutation type the solution is improved by rebalancing the machine loads and then mutating it. First, a machine *m*, from most overloaded resources is chosen at random; further, we identify two tasks, t_1 and t_2 such that task t_1 is assigned to another machine m_2 whose *ETC* for the machine m_1 is less than or equal to the *ETC* of task t_2 assigned to m_1; tasks t_1 and t_2 are interchanged. Secondly, if rebalancing was not possible, then rebalancing is done by *move*. After this, a mutation is applied. This completes one generation of the GA and the best value is stored. All the individuals available at the end of the first iteration will be treated as parents for the second iteration. This procedure is repeated for the number of iterations as given by the user.

5. RESULTS

CloudSim is used for checking the performance of improved scheduling algorithm. CloudSim is an extensible simulation toolkit that enables modeling and simulation of Cloud computing systems and application provisioning environments. The CloudSim toolkit provides the model and features of Cloud system components such as data centers, virtual machines (VMs) and resource provisioning policies. We have fixed the number of virtual machines and varied the number of cloudlets. The implementation was done for with and without categorization of tasks. . We have considered Virtual Machines as resource and Cloudlets as tasks/jobs. The makespan and flowtime that the algorithms produce are shown in tables and the graphs corresponding to these tables have been shown. In first case, we have fixed the number of virtual machines as 20 and we are varying the number of cloudlets from 20 to 100 with a difference of 20. We have run each algorithm 5 times and the average of these 5 runs is noted down in Table II shown below:

Table 2: Makesans for VM and Varying Cloudlets

VM fixed:20	Cloudlets				
	20	40	60	80	100
With Categorization	11.7	32.6	47.3	87.5	103.6
Without Categorization	20.5	50.9	87.7	110.3	135.5

Table 2: Flow Time for VM And Varying Cloudlets

VM fixed:20	Cloudlets				
	20	40	60	80	100
With Categorization	9.6	26.4	35.2	71.0	89.7
Without Categorization	18.3	45.9	79.1	96.7	117.6

6. CONCLUSION

In this paper we have proposed a method for categorizing the tasks before they are scheduled using the Genetic Algorithm. This customized approach produces the optimal schedule with reduced makespan and flowtime. In future, the same categorization can be done with some other evolutionary algorithms.

REFERENCES

[1] Huang Q.Y., Huang T.L.,"An Opt imist ic Job Scheduling Strat egy based on QoS for Cloud Comput ing", IEEE Int ernat ional Conference on Intelligent Comput ing and Integrated Systems (ICISS), 2010, Guilin, pp. 673-675, 2010

[2] Hsu C.H. and Chen T.L., " Adapt ive Scheduling based on Quality of Service in Het erogeneous Environment s", IEEE 4th Internat ional Conference on Mult imedia and Ubiquitous Engineering (MUE), 2010, Cebu, pp. 1-6, 2010

[3] Sabyasachi Mukherjee and O. S. Khanna "Fairness Evaluation of a DSCP Based Scheduling Algorithm for Real-Time Traffic in Differentiated Service Networks" IJIEE Vol. 3, No. 4, July 2013.

[4] G. Guo-Ning and H. Ting-Lei, "*Genetic* Simulated Annealing Algorithm for Task Scheduling based on Cloud Computing Environment," In Proceedings of International Conference on Intelligent Computing and Integrated Systems, 2010, pp. 60-63

[5] Q. Cao, W. Gong and Z. Wei, "An Optimized Algorithm for Task Scheduling Based On Activity Based Costing in Cloud Computing," In Proceedings of Third International Conference on

[6] Bioinformatics and Biomedical Engineering, 2009, pp. 1-3

[7] Medhat A. Tawfeek, Ashraf El-Sisi, Arabi E. keshk, Fawzy A. Torkey "Cloud Task Scheduling Based on Ant Colony Optimization" In Proceeding of IEEE International Conference on Computer Engineering & Systems (ICCES), 2013

[8] L. Wu, S. Panday, R. Buyya, (2010) "A Particle Swarm Optimization-Based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments", 24th IEEE International Conference on Advanced Information Networking and Applications.

[9] J. H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, AnnArbor, MI, 1975.

[10] Javier Carretero, Fatos Xhafa and Ajith Abraham, "Genetic Algorithm Based Schedulers for Grid Computing Systems", International Journal of Innovative Computing, Information and Control 3, 6, 2007.