An Energy Aware Genetic Algorithm Based Scheduling Model For Cloud Computing

[1] Waliullah,[2]Zahid Raza
[1][2]School of Computer & Systems Sciences.
[1][2]Jawaharlal Nehru University
New Delhi
[1] waliul80_scs@jnu.ac.in,[2]zahidraza@mail.jnu.ac.in

***Abstract*-- Cloud computing has emerged as a revolutionary technology in the recent past. It has gained huge popularity among organizations due to its business oriented nature. Being a business model, the service provider and the customer both are concerned about economies of the services. Scheduling is responsible for utilization of resources to a large extent. Good scheduling policies result in optimal usage of the resources. Energy is a significant resource in the field of computing which should be emphasized to be utilized optimally with the help of a good scheduling policy. Moreover energy conservation can a play major role in system reliability and environmental protection. Scheduling in cloud has been proven to be an NP hard problem due to the heterogeneity of the participating resources and huge search space. GA has gained popularity among the evolutionary techniques to deal with NP hard problems like scheduling. The proposed work presents a model with scheduling strategy to minimize the energy consumption of a job submitted to the cloud. It is done ensuring resource assignment according to the precedence and communication cost constraints. The work includes the analysis of the experimental results under various conditions**.

***Keywords*– cloud computing, Genetic Algorithm (GA), Scheduling, Directed Acyclic Graph (DAG), EnergyEfficient**.

I. INTRODUCTION

One of the most emerging technologies which has reshaped the IT industry in the recent past is Cloud computing. There are two entities involved in a cloud computing environment, the user and the service provider. User is the entity which is entitled to use the services of a cloud over the internet. Service provider is the entity which builds its own infrastructure to provide services. The user is charged for the services it uses on the “pay as you go” basis. There exists a service level agreement between the user and the service provider to ensure the Quality of Service (QoS) [13].

Data centers constitute the backend of cloud computing infrastructure. Worldwide 5 hundred thousand data centers have been deployed currently which require a huge amount of energy. Energy cost constitutes a large part of the total operational cost [4-5]. As estimated by the Gartner group the energy consumption amounts to 10% of the total operational cost of the data centers which is estimated to rise to 50% in the next few years [6]. The cost of energy for running servers may have already exceeded the cost of hardware itself. Data centers consumed about 1.5% of the total world electricity in 2010 and approx. 50 million metric tons of CO2 is emitted out annually [7-9]. Even the cost of cooling systems fall under the range of $2 to $5 million per annum for classical data centers. Most of the times cooling systems expenditure amounts more than core IT equipment [10, 12, 19]. Scheduling is at the core of the processes which are responsible for utilizing the deployed resources to a reasonable level. If the resources consuming energy are utilized properly, the energy consumption will also be minimized to a certain extent. In other words, energy consumption can be directly linked to the scheduling strategy of a cloud computing model. Scheduling in cloud computing has been proven to be an NP hard problem. NP hard is a class of problems where it is not possible to obtain the exact solution in polynomial time. To tackle such problems several heuristic techniques have been devised which instead of finding the exact solution find a solution which is very close to the exact one. They do so in polynomial time, Genetic algorithm, simulated annealing, Tabu search are some heuristic techniques to name. This work uses GA to minimize the energy consumption in cloud computing scheduling. Genetic Algorithm (GA) is a general purpose search algorithm based on the process of evolution observed in the nature. It is a heuristic search algorithm, a part of evolutionary computing and a booming area of Artificial Intelligence, inspired by the Darwinian theory of evolution “survival of the fittest” for getting optimum solution of a problem which is not solved by traditional methods. In other words, GA is an adaptive heuristic search algorithm which uses the idea of natural selection and genetics [21].

II. THE PROPOSED MODEL

Job scheduling in cloud may be multi objective or single objective depending upon the requirements and the constraints considered. In this paper we consider the problem for single objective optimization which is minimizing the energy consumption in cloud computing through scheduling using genetic algorithm. The job is submitted in the form of a directed acyclic graph (DAG). The nodes of the DAG represent the tasks in the job and the edges represent the communication between the tasks. The edge weights represent the communication cost between the tasks. Since the DAG is a directed graph, it represents a precedence constrained job. We assume that the resources in the cloud are grouped into clusters with the help of virtualization. The resources in a cluster have different specifications. The job is submitted to a particular cluster. Since the nodes (resources) in the cluster have different specifications each task will take different amount of time to run on each node. More is the time consumed by a task on anode more is the energy consumed as energy is directly proportional to time.

The total energy consumption of any heterogeneous computing system depends on the factors such as processors, networks, disks, memory, cooling system, fans and other various components. A lot of research work is being done on system energy consumption, only processor's energy is considered in this work since processors consume a major part of the total energy of system. The power consumption of a processor comprises of two parts namely static part and the dynamic part. The static part is the leakage power of a circuit and dynamic power is the switching power of a circuit [15-16].

The power consumption of a processor p under execution is given by [16]:

Ptotal = Pstatic + Pdynamic………………………………(1)

Where *Ptotal* is the total power of the processor, *Pstatic* is the static power consumption which is a constant, and *Pdynamic* is the dynamic power consumption. When processor *p* is in the idle mode *Ptotal* = *Pstatic*. When processor *p* is operated at a voltage level vand frequency levelf the dynamic power *Pdynamic* can be calculated using the equation

Pdynamic= c* v2 *f……………………………………(2)

Where ci is a constant known as physical capacitance of the processor pi..Power consumed over a period of time is known as energy which can be calculated as

Energy = Ptotal * TAT……………….……………… (3)

Where T is the time for which a task is executed on processor p.

A. Notation Used

The various notation used under the work are presented below:

a) ***TAT:*** Turnaround time
b) ***Pdynamic,i:*** Dynamic power of processor i
c) ***Pstatic,i:*** Static power of processor i
d) ***pi :*** Processor i
e) ***cij :*** Physical capacitance of processor i while executing task j
f) **vij :** Voltage of processor i while executing task j
g) ***fij :*** Frequency of processor i while executing task j
h) ***tij :*** Execution time of task j on processor i
i) ***tkj:*** Represents task j
j) ***Ci:*** Represents the communication cost of the tasks which are predecessors of the task tkj

B. Fitness Function

The model considers a task to be allocated on a processor which results in minimum energy consumption. Since we have calculated the time for each task on each processor, the tasks should be allocated in such a way that they are scheduled on a processor which takes least time to execute it. The dynamic power of a processor *pi* is given by equation (2). The static power of any processor is constant which need not be calculated. Considering these factors, the energy for a chromosome (schedule) can be calculated as:

Energy = Σ cijv2ijfij (tij + max(Ci)) + (Σ Pstatic,i)*TAT* ……………………………………………………………….. (4)

Here, the first summation in equation (4) calculates the dynamic energy and the second summation calculates the static energy. The dynamic power is multiplied by the time for which a task tkj executes on the processor pi which gives the energy and finding its summation gives the total dynamic energy for all the tasks. Static power is constant so its summation is calculated and is multiplied by TAT the turnaround time of the task. The above equation acts as the fitness function or the objective function as we need to minimize this energy using GA over generations. The energy is calculated for each chromosome in the population, the minimum energy is recorded and the schedule (chromosome) corresponding to that minimum energy is recorded, this is repeated for all the generations until the termination condition is met.

Fitness Function the model considers a task to be allocated on a processor which results in minimum energy consumption. Since we have calculated the time for each task on each processor, the tasks should be allocated in such a way that they are scheduled on a processor which takes least time to execute it. The dynamic power of a processor *pi* is given by equation (2). The static power of any processor is constant which need not be calculated.

Considering these factors, the energy for a chromosome (schedule) can be calculated as:

Energy = Σ *cijv2ijfij** (*tij* + *max*(*Ci*)) + (Σ *Pstatic,i*)**TAT* ……………………………………………………………(5)

Here, the first summation in equation (4) calculates the dynamic energy and the second summation calculates the static energy. The dynamic power is multiplied by the time for which a task tkj executes on the processor pi which gives the energy and finding its summation gives the total dynamic energy for all the tasks. Static power is constant so its summation is calculated and is multiplied by TAT the turnaround time of the task. The above equation acts as the fitness function or the objective function as we need to minimize this energy using GA over generations. The energy is calculated for each chromosome in the population, the minimum energy is recorded and the schedule (chromosome) corresponding to that minimum energy is recorded, this is repeated for all the generations until the termination condition is met.

2 4	1	2	2 3	1	3	4	1	

Figure 1- A Sample Chromosome Structure

C. The Proposed Algorithm

The proposed algorithm schedules the job in the cloud so that the energy consumption can be minimized. The proposed strategy GA (Genetic Algorithm) is based on using Roulette wheel selection for selecting the chromosomes followed by performing the operations like crossover and mutation, if needed, on the population of chromosomes generated randomly. After performing the above mentioned operations the chromosome with minimum Energy value is obtained, which offers the way, scheduling should be done to minimize the energy consumption during job execution. The model follows a multi cluster cloud model in which each cluster might have some specialization. Once a job is submitted appropriate clusters are evaluated matching the job's requirements. Each cluster is then evaluated for the energy consumed, if selected while executing the job with a final selection of that cluster which results in minimum energy consumption. Since GA is used for assessing the suitability of the scheduling scheme, initially a population comprising of chromosomes is created with each chromosome corresponding to a scheduling scheme and a certain energy consumption in the process. Accordingly we have a population of many chromosomes with different allocation hence different energy consumption profiles. The purpose of GA is then to find solution which minimizes this energy consumption over generations by generating various allocations for the job modules. The model uses roulette wheel selection for selection of chromosomes for mating using the single point crossover to produces two child chromosomes. The pool of the parent and child chromosomes are then combined and the best chromosomes are selected to restore the population while avoiding any repetitions. For every generation the best chromosome obtained is stored being replaced with another in case if it has a better fitness than the current one.

To avoid local optima Mutation is performed every 5th generation selecting 25% of the population mutating 10% genes of the chromosome. The above steps are repeated till the specified number of generations acting as the termination condition. The algorithm is presented in the box below.

```
Schedule (job)
{
Submit the job in the form of a DAG
Select the virtual cluster(s) for evaluation
For each cluster
  {
Calculate the execution time of each task on every processor
    // Generate ECT matrix
Generate a population of chromosomes randomly
    // No. of chromosomes = 50-100
Calculate energy for each chromosome in the population set
        // Using the fitness function using the equation (4)
Save the Best chromosome in the population
While (terminating condition is not met)
{
Perform selection
        // Roulette wheel selection
Perform crossover
        // Single point crossover
Perform mutation
        // Every 5th generation for 25% of the population
        // No. of genes mutated is equal to 10% of the size of the chromosomes
Calculate energy for each chromosome in the population set
        // Using the fitness function using the equation (4)
Compare the best chromosome of the current generation with previous best
        // Retain the best solution
}
Record the chromosome which corresponds to minimum energy
// Estimate the turnaround time for the chromosome offering minimum energy
}
Select the cluster with chromosome offering the minimum energy consumption
Schedule the job according to the chromosome recorded above
}
```

III. SIMULATION STUDY

Simulation experiments were conducted to observe the scheduling of the job on the selected cluster. The experiments was conducted on Intel Core-i3 @1.8 GHz using MATLAB 7.6.0 (R2008a). The data values taken in the experiment are generated dynamically during execution with the parameter ranges as listed in Table 1. The results obtained are presented as Figure 1-4.

Parameters Used			
S.No.	Parameter	Notation Used	Range
1	No. of Resources/Processor	no_processor	5-20
2	Clock frequency of processor	fi	10-20
3	No. of tasks in a job	no_task	5-50
4	No. of instructions in a module	no_ins	100-500
5	Inter-task communication	ITC	2-8
6	Population Size	no_pop	50-200
7	Size of chromosome	size_chrom	5-50
8	No. of generation	no_generation	100,200
9	Crossover considered during experiment	Single point	--
10	Mutation performed after no. of generations	Mu	5,15,20
11	% of Population selected for Mutation	--	25%
12	Rank selection method	Roulette wheel	--

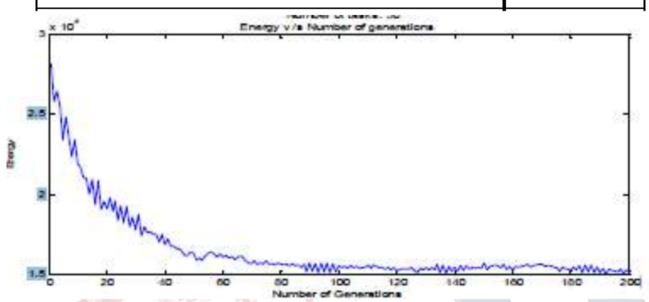

Figure 1: Energy Consumed v/s Number of Generations for 50 tasks

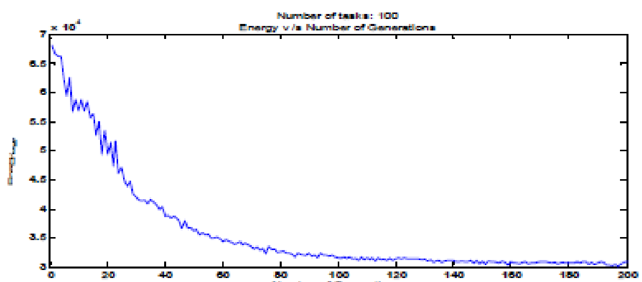

Figure 2: Energy Consumed v/s Number of Generations for 100 tasks

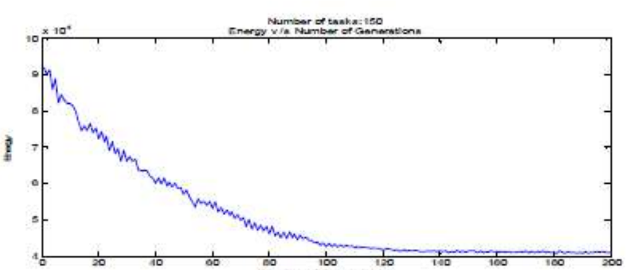

Figure 3: Energy Consumed v/s Number of Generations for 150 tasks

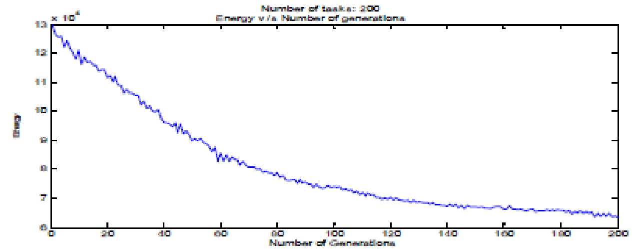

Figure 3.5: Energy Consumed v/s Number of Generations for 200 tasks

IV. OBSERVATIONS

After the experiments were carried out, the results were recorded and are plotted above. This section deals with the analysis and observations of the results. From Figure 3.2 which shows the energy graph for 50 tasks, it can be observed that the energy consumption as reported by the best chromosome in the population decrease quickly up to 50 generations and then gradually up to 100 generations. After 100 generations the energy value decreases at very slow rate. The rate of convergence in the beginning is highest which reduces in the middle and goes very low towards the end. The same pattern can be observed even from Figure 3.3 which shows the energy graph for 100 tasks, where the energy values can be seen decreasing quickly till 70 generations and then gradually up to 120 generations and becoming very slow later. Further, the rate of convergence for 100 tasks is seen lower as compared to that of 50 tasks. The same pattern is reported even in Figure 3 and Figure 4 for 150 and 200 tasks respectively with the rate of convergence of 200 tasks being the slowest among the set owing to the larger batch size. It can be concluded from the results that the model tries to converge to the allocation which results in a better energy consumption over the generations. Further, as the number of tasks increases the rate of convergence decrease. The results have been found to work satisfactorily converging within 200 generations.

V. CONCLUSIONS AND FUTURE SCOPE

The proposed model uses genetic algorithm (GA) for scheduling a job on the cloud system so that the energy of the jobs that need to be executed on the system can be minimized. Here, the study is based on roulette wheel selection method for a single point crossover and mutation implemented after certain number of generations for certain percentage of the population. From the obtained results it can be said that allocations of tasks is done in such a way that the energy consumption is minimized. Tasks are allotted to those processors frequently which have such specifications that result in minimal energy consumption.
Over the generations, energy value for the entire job reduces considerably. Further, simulation study reveals that the rate

of convergence of the model depends on the number of tasks submitted for execution. As the number of tasks in the job increases, the rate of convergence decrease. Since, scheduling is an NP-hard problem, the results obtained cannot be treated as the most optimized results. Accordingly, the proposed model can be tested for different selection, crossover and mutation schemes and compared with established peers for performance evaluation. Further, it also opens the possibility of the use of some other soft computing approaches like particle swarm optimization or ant colony optimization to be used and compared for performance evaluation. Multi objective functions or parameters can also be explored by optimizing more than one parameter.

REFERENCES

1. M. Sajid and Z. Raza, “Cloud Computing: Issues and Challenges”, International Conference on Cloud, Big Data and Trust, RGPV, pp. 35-41, Nov 13-15, 2013.

2. T. Dillon, C. Wu and E. Chang, “Cloud computing: Issues and challenges”, 24th IEEE International Conference on Advanced Information Networking and Applications, pp. 27- 33, 2010.

3. “State of the Data Center 2011,” Emerson Network Power, 2011.

4. M. A. Vouk, “Cloud computing – Issues, research and implementations”, Journal of Computing and Information Technology, Vol. 16, no. 4, pp. 235-246, June- 2008.

5. X. Fan, W.-D. Weber, and L. A. Barroso, “Power Provisioning for Warehouse-sized Computer,” Proceedings of the ACM International Symposium on Computer Architecture, San Diego, CA, June 2007.

6. R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, and X. Zhu, “No “Power” Struggles: Coordinated Multi-level Power Management for the Data Center,” APLOS 2008.

7. M. N. Huhns and M. P. Singh, "Service-Oriented Computing: Key Concepts and Principles", IEEE Internet Computing, Vol. 09, pp. 75 - 81, 2005.

8. Gartner Group, available at: http://www.gartner.com/

9. A. Vasan and A. Sivasubramaniam, “Worth their watts?- An Empirical Study of datacenter servers,” IEEE 16th International Symposium on High Performance Computer Architecture (HPCA), 2010.

10. IDC, “Worldwide Server Power and Cooling Expense 2006-2010,” Market Analysis, 2006.

11. J. G. Koomey, “Growth in Data center electricity use 2005 to 2010,” Analytics Press, 2011

12. “Reducing Data Center Cost with an Air Economizer,” Intel IT@Intel Brief, August 2008..

13. N. Rasmussen, “Calculating Total Cooling Requirements for Data Centers,”White Paper #25, American Power Conversion, 2007.

14. S. Sindhu and S. Mukherjee, “A Genetic Algorithm Based Scheduler For Cloud Environment”, 4th International Conference on Computer and Communication Technology (ICCCT), pp. 23-27, 2013.

15. K. Li, X. Tang, and K. Li, “Energy-Efficient Stochastic Task Scheduling on Heterogeneous Computing Systems”, IEEE Transactions On Parallel And Distributed Systems, Vol. 25, No. 11, pp. 2867-2876, November 2014.