Heuristic vs Meta-Heuristic Approaches for Load Balancing in Cloud Environment

[1]Pooja Mangla, [2]Dr. Sandip Kumar Goyal

[1]PhD Scholar, [2]Guide & Head of Computer Department
Computer Department MMU, Mullana, Ambala. Haryana, India

***Abstract*: Cloud computing is the new technology which is totally based on virtualization. Load balancing is one of the most important issues which play a great role in scheduling of resources, and even in task management. In this research paper, we are going to compare some of the heuristic and meta heuristic techniques.**

***Index Terms*: Cloud Computing, Load Balancing, Heuristic, Meta-Heuristic**

I. INTRODUCTION TO CLOUD COMPUTING:

Cloud computing is a new technology which uses virtual machine instead of physical machine to host, store and network the different components. Cloud Computing refers to manipulating, configuring, and accessing the applications online. It offers online data storage, infrastructure and application. A Cloud is a kind of parallel and distributed system possessing a group of inter-connected and virtualized computers that are dynamically scheduled and highlighted as one or more unified computing resources based on service-level agreements established through conciliation between the service provider and consumers [1].
Cloud Computing is a way of managing large number of highly virtualized resources such that, from a management perspective, they resemble a single large resource. Cloud computing is the provision of dynamically scalable and often virtualized resources as a service over the Internet (public cloud) or intranet (private cloud).

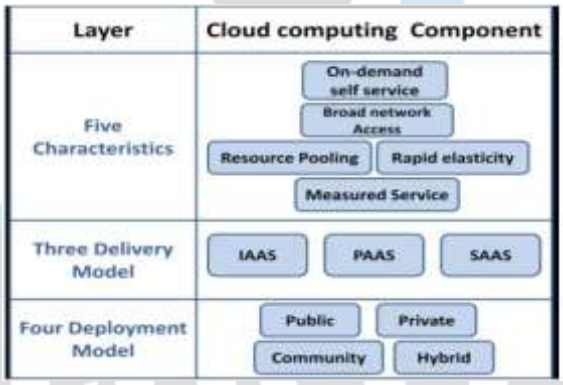

Figure 1.1: Model for Cloud Computing

II. LOAD BALANCING IN CLOUD COMPUTING:

Load balancing is a generic term used for distributing a larger processing load to smaller processing nodes for enhancing the overall performance of system. Load Balancing helps in improving the performance substantially, maintenance of system stability, cost effectiveness, etc. Load balancing is a process of reassigning the total load:

- to the **individual nodes** of the collective system
- to make **resource utilization** effective
- to improve the **response time** of the job
- simultaneously removing **a condition** in which some of the nodes are over loaded while some others are under loaded.

Load Balancing is a process of reassigning the total load to the individual nodes of the collective system to make more resource utilization effective and to improve the response time of the job, simultaneously removing a condition in which some of the nodes are overloaded while some others are under loaded. An ideal load balancing algorithm should avoid overloading or under loading of any specific node. But, in case of a cloud computing environment the selection of load balancing algorithm is not easy because it involves additional constraints like security, reliability, throughput etc. So, the main goal of a load balancing algorithm in a cloud computing environment is to improve the response time of job by distributing the total load of system. The algorithm must also ensure that it is not overloading any specific node.

III. RESEARCH ALGORITHMS:

In the current research, the performance of heuristic and meta-heuristic algorithms, which were derived in our previous research papers, are compared to each other and the algorithms are defined below:

3.1 PROPOSED HEURISTIC ALGORITHM:

To better use tremendous capabilities of this large scale distributed system, effective and efficient scheduling algorithms are needed. Many such algorithms have been designed and implemented. We introduce a new scheduling algorithm based on two conventional scheduling algorithms, Min-Min and Max-Min, to use their cons and at the same time, overcome their pros. This heuristic scheduling algorithm, called min-min min-max selective, is evaluated using a Cloud simulator called CloudSim by comparing to its performance against the two basic heuristics which it comes from.

```
for all tasks Ti in the Meta-set Mi on all the resources Rj
calculate the average task length avgTL
do until all tasks in Mi are mapped
for each task in Meta-Set Mi
if ti.getMI() <= avgTL
        create a sub-list of tasks tk
        for all resources rj
        find the task tk in the meta-set with earliest completion time and the machine that obtains it
update Cij for all other tasks in Meta-Set
else
        create a sub-list of tasks tk
        for all the resources rj
        find the minimum completion time of ith task in the Meta-Set
        find the task tk in the meta-set with earliest completion time and the machine that obtains it
        assign the task tk to the machine rj which gives maximum minimum completion time
        delete this task from the Meta-Set
        update Cij for all other tasks in Meta-Set
```

Reviewing Min-Min and Max-Min heuristics, it can be seen that depending on the length of unassigned tasks in MT (meta-task), one of these heuristics has better results than the other one [7]. For example, if there is only one long task and too many short tasks, Max-Min will execute long task first and allows short tasks to be executed concurrently with the long task, resulting better makespan and even better resource utilization rate and load balancing level, compared to Min-Min that executes all short tasks first and then executes the long task.

3.2 PROPOSED META-HEURISTIC ALGORITHM

In the proposed model, hybrid DE and GA are used [5]. In this, Differential Evolution starts upto the point where the trial vector is generated. If that vector satisfies the equation, then it is included in the population otherwise algorithm enters the Genetic algorithm phase and generates a new candidate solution.

The pseudo-code for efficient load balancing using DE & GA, shown above, defines the working of hybrid approach for load balancing in cloud environment. It creates new candidate solutions (called agents) by combining the parent individual and several other individuals of the same population. These agents are moved around in the search-space by using mathematical formulae to combine the positions of existing agents from the population. If the new position of an agent is improved than it is accepted and forms part of the population, otherwise the new position is easily throw away. The series of act ion is repetition until achieve the results and by doing so it is hoped, but not guaranteed, that a satisfactory solution will eventually be discovered. This is a greedy selection scheme that often outperforms traditional EAs (Evolutionary Algorithms).

1. Sampling the search space at multiple, randomly chosen initial points i.e. a population of individual vectors.
2. Differential evolution is a nature derivative-free continuous function optimizer, it encodes the parameters as a floating-point numbers and manipulates them with simple arithmetic operations. For this differential evolution it mutates a (parent) vector in the population with a scaled difference of the other randomly selected individual vectors.
3. The resultant mutation vector is a crossed over with corresponding parent vector to generate a trial or a offspring vector.
4. Then, finally it takes a decision in a one-to-one selection process of each pair of offspring and parent vectors.
5. If the best population is generated, it is taken into consideration; otherwise the population is generated by using Genetic algorithm.
6. In Genetic algorithm, crossover and mutation operations are applied on the candidates and new candidate generation is achieved.
7. The one with a better fitness value survives and enters the next generation.

Figure 2: Pseudo-code of DEGA model.

3.2 PERFORMANCE METRICS:

In this research work, makespan, average resource utilization and load balancing level are calculated in different scenarios where scenario 1 signifies the existence of heavy load, scenario2 signifies the existence of least load and scenario 3 signifies the medium load at each of the virtual machines available in the existing cloud.

- **Makespan:** Makespan is a measure of the throughput of the heterogeneous computing systems, such as cloud. It can be calculated as the following relation:

makespan = max(CTi) , where ti belongs to MT

The less the makespan of a scheduling algorithm, the better it works.

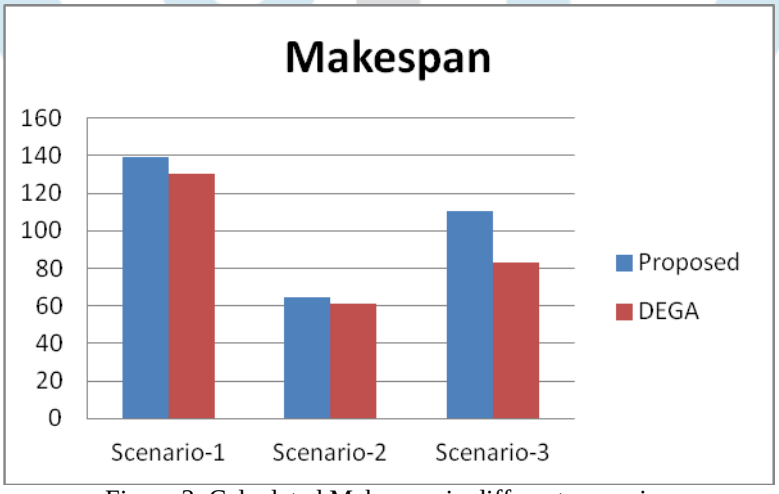

Figure 3: Calculated Makespan in different scenarios

- **Average resource utilization rate:** It is one of the metrics that is used in [14]. Average resource utilization of each resource can be calculated.

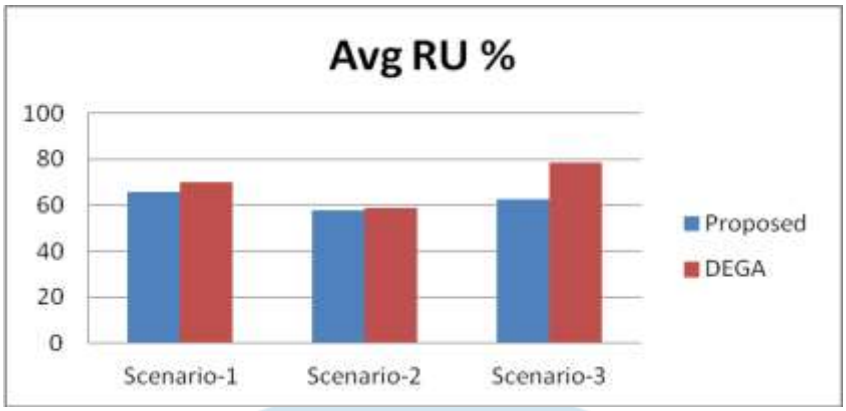

Figure 4: Calculated Average Resource Utilization in different scenarios

- **Load balancing level:** The mean square deviation of *ru is defined as :* The best and most efficient load balancing level is achieved if d equals zero and β equals 1. So, scheduling algorithm will have better performance if d is close to 0 and β is close to 1. It is the other metric that is used in [14].

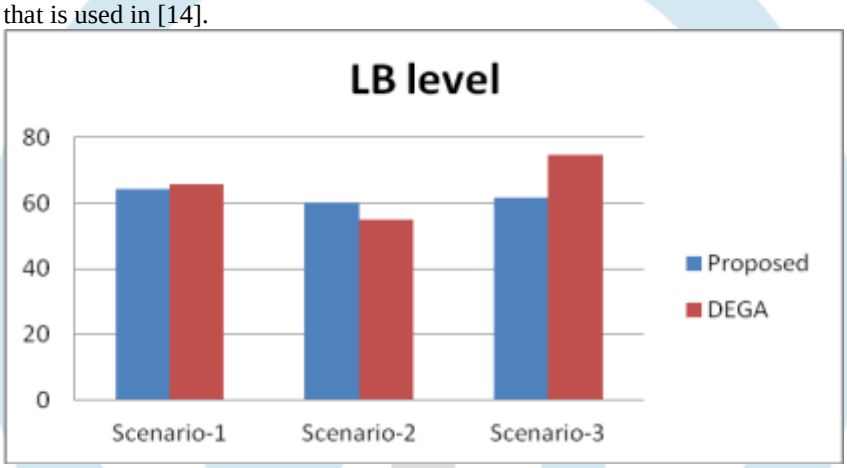

Figure 5: Calculated Load Balancing Level in different scenarios

IV. CONCLUSION:

In this research work, two different techniques for load balancing are being compared. In this method, three parameters i.e., makespan, average resource utilization and load balancing level, are calculated and the corresponding results are shown. It has been observed that the results generated from this comparison shows that meta-heuristic techniques are better than the heuristic techniques.

REFERENCES

[1]. S. S. Moharana, R. D. Ramesh & D. Powar , ―Analysis Of Load Balancers In Cloud Computing , International Journal Of Computer Science And Engineering (IJCSE) Vol.2(2), pp. 101-108, 2013.

[2]. Ts`epoMofolo, R Suchithra, ―Heuristic Based Resource Allocation Using Virtual Machine Migration: A Cloud Computing Perspective‖, International Refereed Journal of Engineering and Science (IRJES), ISSN (Online) 2319-183X, Volume 2, Issue 5(May 2013), PP.40-45

[3]. G.Ramya1, Dr P. Keerthika2, Dr P. Suresh3 and Ms M. Sivaranjani, ―OPTIMIZED SCHEDULING OF TASKS USING HEURISTIC APPROACH WITH COST-EFFICIENCY IN CLOUD DATA CENTERS‖, International Conference on Modern Engineering, Science & Technology– 2016 (IER-ICMEST'16).

[4]. J. Cao, D. P. Spooner, S. A. Jarvis, G. R. Nudd: Grid Load Balancing Using Intelligent Agents, Future Generation Computer Systems, Vol. 21, No. 1, pp. 135-149 (2005).

[5]. Cuckoo Genetic Optimization Algorithm for Efficient Job Scheduling with Load Balance in Grid computing, ―International Journal Computer Network and Information Security, Vol. 8, 59-66, 2016.

[6]. Marjan Kuchaki Rafsanjani and Amid Khatibi Bardsiri, ―A HYPER-HEURISTIC METHOD FOR SCHEDULING THEJOBS IN CLOUD ENVIRONMENT‖, , Informatics Engineering, an International Journal (IEIJ), Vol.4, No.1, March 2016

[7]. Ehab Nabiel Alkhanak, Sai Peck Lee, Teck Chaw Ling, ―A Hyper-Heuristic Approach using a Prioritized Selection Strategy for Workflow Scheduling in Cloud Computing‖, The 4th International Conference on Computer Science and Computational Mathematics (ICCSCM 2015)

[8]. Amirkhan Pinjari, B. R. Mandre , ―Cloud-Scheduling Algorithm using Hyper heuristic Approach: Study and Implementation‖ , International Journal of Computer Systems (ISSN: 2394-1065), Volume 02– Issue 06, June, 2015

[9]. A. T.Velte, T. J. Velte, R. Elsenpeter, ―Cloud Computing: A Practical Approach‖, TATA McGRAW-HILL Edition 2010.