VOL. 13, NO. 3, FEBRUARY 2018 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

MARKOV CHAIN MODEL AND PSO TECHNIQUE FOR DYNAMIC HEURISTIC RESOURCE SCHEDULING FOR SYSTEM LEVEL OPTIMIZATION OF CLOUD RESOURCES

Shrabanee Swagatika[1], Amiya Kumar Rath[2] and Prasant Kumar Pattnaik[3]
[1]Department of Computer Science and Engineering, SOA University Bhubaneswar, India
[2]Department of Computer Science and Engineering and IT, VSS University of Technology, India
[3]School of Computer Engineering, KIIT University, Bhubaneswar, India
E-Mail: shrabanee.swagatika@gmail.com

ABSTRACT

Cloud computing options a versatile computing infrastructure for large scale processing needs effective load equalization and migration of the cloud resources for rising the energy potency and resource utilization. The instructed resource allocation technique considers numerous parameters like migration time, waiting time, QoS, resource utilization etc. for effective and economical virtualized resource management and allocation. This paper aims at dispatching the computation load to any or all process nodes within the cloud computing atmosphere by considering the physical employment on every node therefore on stop bias during arranging computation resources and therefore improve the general computing performance in a heterogeneous cloud atmosphere.

Keywords: cloud computing, large-scale data processing, PSO, markov chain model, meta-heuristics.

1. INTRODUCTION

There are various models of cloud computing for intelligently utilizing the resources. For managing dedicated on-demand computing resources for many users, a huge amount of cost incurs in the form of power consumption. As a result, clients have to pay heavy amount for renting computational services. Moreover, cloud resources are limited and network traffic could easily clog the services. Cloud computing also adds excessive carbon footprints to the environment. In this report, we will review how researchers have sorted out the problem of optimizing resources' utilization and energy consumption in cloud computing.

Job scheduling is a very important task in cloud surroundings. Job scheduling is employed to apportion sure jobs to explicit resources especially time. In cloud computing, job-scheduling downside could be a biggest and difficult issue. The most aim of job programming formula is to boost the performance and quality of service and at identical time maintaining the potency and fairness among the roles and scale back the execution price. Associate degree economical job programming strategy should aim to yield less latency so the execution of submitted jobs takes place among a doable minimum time. The paper here to present our vision, discuss the comparative study of different dynamic resource scheduling algorithm for optimized energy-aware resource management, Associate in Nursing projected an economical algorithmic rule for virtualized information centers in order that Cloud resources is a property in thought technology to drive advancements for future generations. Scheduling is a vital issue of current implementation of cloud computing. The demand for planning is to realize superior computing. Typically, it's tough to search out Associate in Nursing best resource allocation for a particular job that minimizes the schedule length of jobs. There are 3 main phases of planning.

- Phase one is discovering a resource, that generates a listing of potential resources.
- Phase 2 involves gathering info concerning those resources and selecting the simplest set to match the applying needs. The selection of the simplest pairs of jobs and resources during this section may be a NP-complete downside.
- In the third section, the work is executed.

This paper aims to describe various methods used for task Scheduling problem for Cloud resources. Then we will describes background study of different scheduling techniques for Cloud resources and compare them with dynamic heuristic resource scheduling inspired from Particle Swarm Optimization .And we will show a comparative analysis of task Scheduling for them with a single dataset.

2. RELATED WORK

The resource management is essential to maintain the trade-off between the energy and performance across the dynamically arriving user's tasks in are source-abundant ubiquitous cloud environment. Several existing research systems address an emerging research constraint of Energy efficiency in heterogeneous cloud computing environment [10, 11]. The workload management is considered as the main technique, which has been used to minimize the energy consumption. Dynamic load balancing algorithms distribute the workload across the physical nodes during execution by making the distribution decisions based on the current load information [12]. An exponential smoothing forecasting mechanism based dynamic load balancing strategy [13] effectively reduces the server load and improves the

service quality by exploiting the Weighted Least Connection (WLC) algorithm [14], which is referred as Exponential Smooth forecast based on WLC (ESWLC) approach[26]. The WLC algorithm based load balancing based on the number of connections existing on each node to allocate the task to the node which has the least number of connections [26]. However, the WLC method lacks in analyzing the capabilities of each resource or node including storage capacity, processing speed, and bandwidth. Dual Direction FTP (DDFTP) algorithm [15] provides an efficient load balancing mechanism to utilize the server and network resources precisely in a dynamic environment. The mechanism of this approach can also be used for load balancing in a cloud environment. Load Balancing Min-Min (LBMM) algorithm [16] employs the Opportunistic Load Balancing (OLB) algorithm in three-level load balancing framework to distribute the workload efficiently. However, it leads the delayed processing time, since it lacks in considering the execution time of each resource. To achieve effective load balancing, the cloud data centers migrate and deploy the VM resources dynamically to provide the service, without violating the SLAs [17, 27].

The high energy consumption of the cloud data center is not only caused by the quantity of the computing resources, but also by the inefficient utilization of the cloud resources [18, 19, 24, and 32]. A scalable and virtual autonomic resources management framework [19] exploits an Ant Colony Optimization (ACO) based VM consolidation algorithm to balance the workload of the hosts by consolidating the VMs. ACO based scheduling approach [20] optimally allocates the input job requests to the corresponding virtual resources based on the demand of the users and the availability of the data center resources. It minimizes the Makespan of a given set of tasks by randomly searching the appropriate resources among the available resources. With the target of reducing the required number of hosts, EnaCloud [21] aggregates the workloads of VMs into a single host node based on the resource characteristics of the virtual resources. It facilitates the system in minimizing the energy consumption of the data center. VM selection and placement method [22] plays a significant role in the process of VM migration in the context of energy-efficient load balancing. VM migration algorithm [23] employs a fuzzy decision-making method to dynamically utilizes the energy-efficient cloud resources by Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) approach [31]. TOPSIS approach identifies the maximally loaded physical servers as the overloaded hosts and takes a migration decision. An energy-aware VM placement algorithm [24] focuses on reducing the energy cost and increasing the resource utilization during migration in the cloud data center. An energy-efficient approach [25] considers the performance degradation during the migration process and presents a dynamic consolidation framework to improve energy efficiency while ensuring the certain SLA levels. Markov chain model based dynamic consolidation approach [26] optimally manages the host overload detection in the cloud data center under the specific QoS goal. It makes the energy-efficient cloud data center by dynamically consolidating the VM resources. The dynamic placement of the workload relies on the current workload of the cloud resource and optimization technique of ACO meta-heuristics to accomplish the energy-efficient cloud data center [27]. An approach [28] enhances the standard Particle Swarm Optimization (PSO) algorithm by introducing the mutation mechanism and a self-adapting inertia weight methodology. It optimizes the task execution time in the sense of running time and the resource utilization by dynamically balancing the workload of the resources.

These conventional works describe that dynamic consideration of the cloud data center resources plays a crucial role in providing the effective cloud service. However, the cloud system still requires an optimal trade-off between the energy and performance instead of focusing on the dynamic resource allocation. Thus, the proposed approach dealing with the energy-performance trade-off by periodically monitoring the load of the resources, predicting the future utilization, and takes the migrating decision.

3. ARCHITECTURAL FRAMEWORK FOR SCHEDULING

3.1. Task scheduling procedure

Clouds aim to drive the planning of consequent generation knowledge centers by architecting them as networks of virtual services (hardware, database, computer program, application logic) in order that users will access and deploy applications from anyplace within the world on demand at competitive prices counting on their QoS needs. During a cloud setting, task programming is needed to distribute the dynamic work equally among all the nodes or resources. Load equalization or task programming helps in higher allocation of the resource. It helps in implementing fail-over and avoiding bottlenecks. With the assistance of Task programming techniques, knowledge are often sent and received with none major delays. In cloud computing setting, several algorithms area unit gift that helps in correct task programming between all out there servers. They will be applied within the cloud system with appropriate verifications.

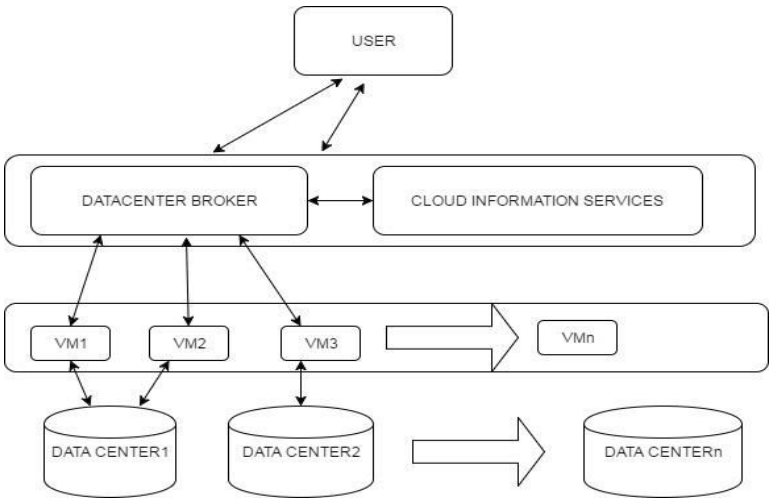


Figure-1. Resource allocation for cloud computing services.

Task programming algorithms area unit divided into 2 sorts in an exceedingly cloud environment: Batch mode heuristic programming algorithms (BMHA) and on-line mode heuristic algorithms. Several algorithms are provided by BMHA technique. From them, initial return initial Served algorithmic program (FCFS), spherical Robin programming algorithmic program (RR), Min-Min algorithmic program and Max-Min algorithmic program area unit main algorithms that provided by BMHA technique. In on-line mode heuristic programming algorithmic program, it's vital to estimate the correct load, it must do a comparison of all load during this technique. CPU load, the number of memory needed combining along to calculate the load of a machine. Users might expertise several major issues while not load leveling like delays, timeouts and long system responses.

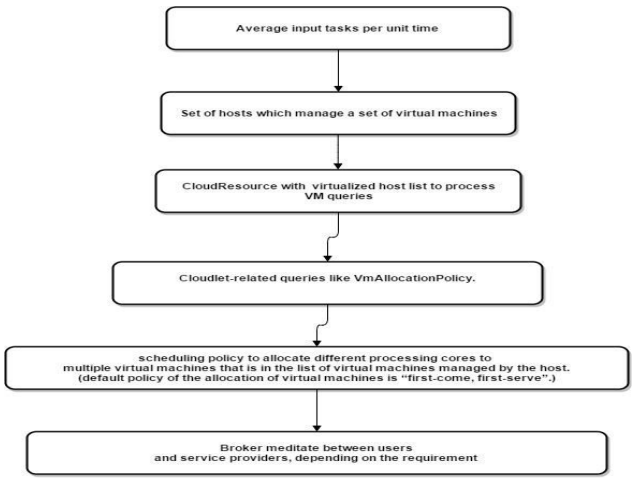


Figure-2. Control flow of resource allocation for cloud computing services.

3.2 Proposed methodology

3.2.1 The paper aims and objective

- To develop algorithms to predict future resource utilization and host overloading condition during resource allocation in dynamic cloud environment
- To perform energy-efficient VM migration to accomplish load balancing and improve the resource utilization of the cloud providers while consistently meeting the negotiated SLAs.

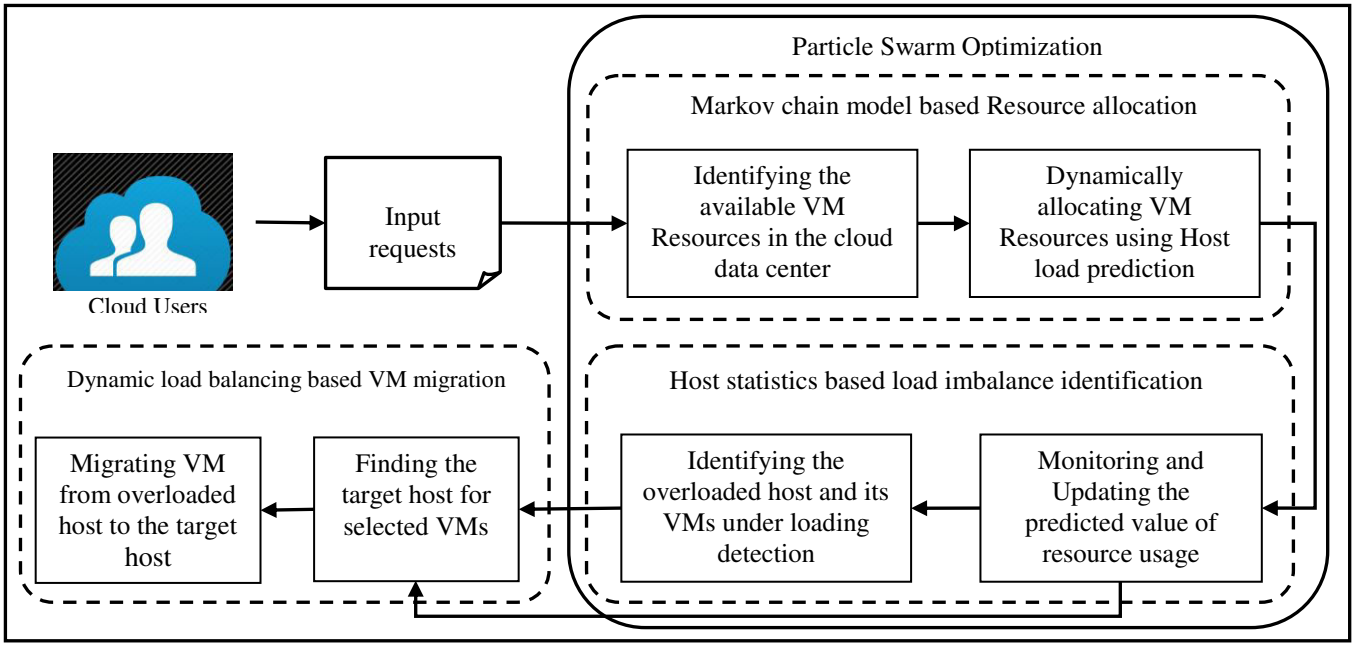


Figure-3. Proposed model for resource allocation for cloud computing services.

The proposed methodology targets to provide a novel resource allocation framework to support dynamic system level optimization of cloud computing services. It employs the Markov chain model, and PSO technique to enhance the dynamic resource allocation. The proposed system incorporates three phases of optimal resource allocation such as Markov chain model based resource allocation, Host statistics based load imbalance identification, and Dynamic load balancing based VM migration, as shown in Figure-3.

3.2.2. Markov chain model based resource allocation

The proposed system measures the availability regarding a load of each host according to the request characteristics of the cloud users. A Markov process may be a stochastic process (random process) within which the chance distribution of this state is not absolutely freelance of the trail of past states, a characteristic known as the Andrei Markov property.. Markov chain may be a discrete-time stochastic process with the Markov property. We will use a VM finding example to show Markov chains, since it is a simple and well-studied case. It is a technique in which the frequency of the VM sequence is higher than other regions. We want to develop a probabilistic model for VM selection, such that every task sequence is generated by the model for appropriate machine. Since selection is important, we want a model that generates sequences in which the probability of a selection depends on the previous selection. It is normally wont to predict the distant way forward for dependent events. This ideal may be applied to a simplified Datacenter prediction, though the accuracy is low in predicting immediate one.

A basic example of a Markov chain consists of a current state, usually compiled into a state vector c, and a random matrix P, AN nxn assortment of chance vectors supported the amount of states (n). This state is employed for the premise of the long run states. The ensuing Markov process is so shaped wherever c1=Pc0, c2=Pc1,... c(k+1)=Pck, wherever k is AN integer [29]. Power consumption of server may be diagrammatic as follows: P = zero.2782 *Util + fifty one.2765. Wherever P is that the power consumption of our physical server, and Util is that thecentral processor utilization. [30].

Algorithm 1: Modified Markov Chain Model for arranging VMs in decreasing order of Power utilization (MCM)

Input: VMList, No Of Tasks, Allocated Resource, State Set (Sx)

Output: VM allocation, New Coordinates

1. $VMp = \frac{P}{VMList(P)} -$ misc
2. x = initial state
3. while (VMp[T] >VM[Tassigned])(T for Theshold)
   {
4. repeat n times {
   Choose random neighbor y ∈ Sx
5. if (E(x)≤ E(y)) then
6. choose x
7. else
8. choose random z ∈(0,1)
9. if z< $\frac{e^{-E(y)kT/Nx}}{e^{-E(x)kT/Ny}}$ then x=y (by Boltzmann distribution)
10. }
11. T=T *0.9
12. }
13. return x

Where VMp is the power consumption by a VM, kT (a constant of the distribution) the merchandise of Boltzmann's constant and thermo

dynamical temperature. The Markov chain model assisted with PSO technique dynamically allocates the VM resources to the appropriate input requests by exploiting the knowledge of future prediction value of host load. The host load prediction refers the expected load of each host after allocating a specific request on the selected VM resource. The PSO technique optimally selects the feasible VM resource for each request from the available resources by focusing on the load balancing factor.

3.2.3 Host statistics based load imbalance identification

The proposed approach dynamically monitors and updates the statistics of the allocated hosts based on the resource usage predicted by Markov chain model. It facilitates the system to determine the overloaded hosts and select the VMs to be migrated based on the VM execution time, which leads the significant improvement on energy optimization in the cloud services.

A Hidden Markov Model (HMM) could be a framework with 2 levels of hierarchy. It is wont to model rather more complicated random processes as compared to a conventional Markov model. In an exceedingly specific state, associate observation is generated per associate associated likelihood distribution. It's solely the observation associated not the state that's visible to an external observer [33].

An HMM is characterized by the following:

A. N is that the range of states within the model. Si, i= 1, 2...n is a private state denote the set of states S. The state at time instant t is denoted by qt.
B. M is that the range of distinct observation symbols per state Vi wherever i = 1, 2 ... m is an individual image denote the set of symbols V.
C. The state transition probability matrix A = [aij], where

Here aij > 0 for all i, j. Also,

$$\sum_{j=1}^{N} aij = 1, 1 \le i \le N \quad (1)$$

D. The remark symbol probability matrix B = {bj (k)}, where

$$b_j(k) = p(V_k|S_j), 1 \le j \le N, 1 \le k \le M \quad (2)$$

$$\sum_{k=1}^{M} b_j(k) = 1, 1 \le j \le N \quad (3)$$

E. The initial state probability vector $= \pi_i$,

where

$$\pi_i = P(q_1 = S_i), 1 \le i \le N \quad (4)$$

Such that

$$\sum_{i=1}^{N} \pi_i = 1 \quad (5)$$

F. The remark sequence O = O1, O2, O3 ...Or, wherever every remark Ot is one amongst the symbols from V, and R is that the range of remarks within the sequence.

It manifest that an entire specification of associate HMM desires the approximation of 2 model parameters, N and M, and three risk distributions A, B, and π. We use the notation λ = (A, B, π) to specify the whole set of parameters of the model, where A, B implicitly contain N and M.

As mentioned on top of, associate observation sequence O may be generated by several potential state sequences. Take into account one such explicit sequence alphabetic character = q1, q2 ... qr. wherever q1 is that the initial state. The chance that O is generated from this state sequence is given by

$$P(O|Q, \lambda) = \prod_{t=1}^{R} P(O_t|Q_t, \lambda) \quad (6)$$

The statistical independence of observations of above equation can be assumed and expanded as

$$P(O|Q, \lambda) = b_{q1}(O_1) b_{q2}(O_2) b_{q3}(O_3) \dots \dots b_{qR}(O_R) \quad (7)$$

The state sequence probability P (Q | λ) is given as

$$P(O|Q) = \pi_{q1} a_{q1q2} a_{q2q3} \dots \dots a_{qR-1qR} \quad (8)$$

Thus, the generation of the observation sequence probability P (O | λ) by the HMM can be written as follows:

$$P(O \mid \lambda) = \sum_{all\ Q} P(O|Q, \lambda) P(Q|\lambda) \quad (9)$$

Deriving the worth of likelihood of the observation sequence P (O | λ) exploitation the direct definition of is computationally intensive. Hence, a procedure to calculate the worth named as a Forward-Backward procedure is employed to compute (O | λ). The likelihood of generation of the observation sequence determined by the HMM is given exploitation formula [33].

$$P(O \mid \lambda) = \sum_{all\ Q} P(O|Q, \lambda) P(Q|\lambda) \quad (10)$$

Here, the next load data is generated by proposed algorithm and it helps to estimate the upcoming load values approximation using probability of states and provide the new load pattern that arises in at least five steps ahead.

3.2.3 Dynamic load balancing based VM migration

Applying the Markov chain model and PSO technique on the cloud resources identify the target hosts for the selected VMs. The identification of target host ensures the load balancing and improves the resource utilization while migrating the VMs from the overloaded host to the target host. The dynamic load balancing based VM migration provides the seamless execution to the end

user, which increases the profit by reducing the energy consumption, and migration cost.

This work demonstrates the proposed approach using the CloudSim toolkit which is a generalized simulation framework. The CloudSim toolkit supports the modeling of simulation based trials to percept the actual behavior of the proposed algorithm. This simulation framework enables the modeling, simulation, and experimenting the application services and the cloud infrastructure.

4. SCHEDULING OF DATA CENTER RESOURCES

In this section, we have used Modified Load Balancing using PSO (MLPSO) algorithm because of its command over meta-heuristic methods and implemented in CloudSim simulator to schedule the tasks and discussed the results. Our main aim is to schedule maximum number of tasks in minimum time.

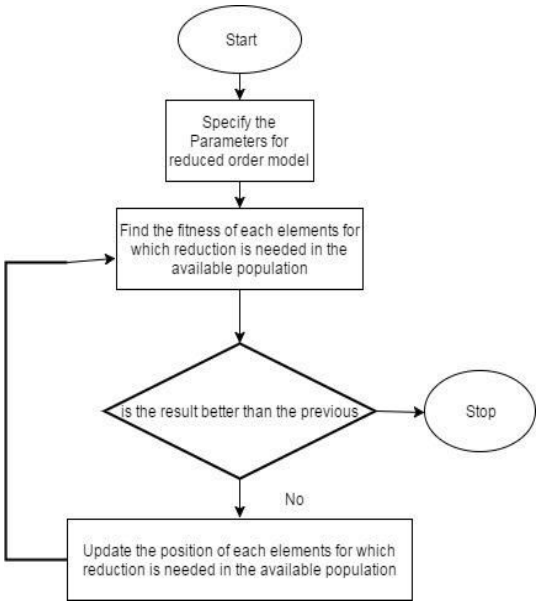


Figure-4. VM optimization using MLPSO for Cloud Computing Services.

The reason behind the restricted variety of resources that may be set to the frequency of use of the processor and also the indisputable fact that every particle remembers the coordinates of the most effective resolution (gbest) achieved to date. The coordinates of current international best (pbest) are hold on. The coordinates of the particle area unit updated consequently. Moreover, these studies have shown that on the average associate degree idle server consumes just about less variety of the resources consumed by the server running at the complete processor speed. This truth justifies the technique of shift idle servers to the sleep mode to cut back the entire power consumption by truthful load leveling. Therefore, during this work we tend to use the facility model outlined in (1).

$$NC = C + C1 \times r \times (pbest - C) + C2 \times r \times (gbest - C) \quad (11)$$

Where NC represents new coordinates, C represents coordinates and C1, C2 are learning factors and r is a random number between 0 and 1.

The PSO use the following equations to update its velocity and position (31)

$$v(k+1) = w\ v(k) + c1r1(pbest - x(k)) + c2r2(gbest - x(k)) \quad (12)$$

$$x(k+1) = x(k) + v(k+1) \quad (13)$$

where x and represent the rate and also the position of the particle, severally, c1 and c2 area unit positive constants cited as acceleration constants; r1 and r2 are unit random numbers between zero and 1; pbest refers to the most effective position found by the particle and gbest refers to the worldwide best position and eventually w is that the inertia weight.

Algorithm 2: Modified Load Balancing using PSO (MLPSO)
Input: hostList, vmList, NoOfTasks, schedulingPolicy
Output: VM allocation, New Coordinates

1. Numbers of particles← maximum iteration, particles
2. IDs List of VMs←VMList (this solution is not the final solution but it just an initial solution)
3. for every VM in vmList do
a. allocatedHost← NULL
4. for every host in hostList do
a. if host[R]≥Resource[VM]
5. Pbest←estimatePower(host, VM) (pbest: best position of every Task)
6. Repeat steps two to six till a stopping condition is met. Stopping condition is also the utmost variety of iterations or no modification in fitness worth of particles for consecutive iterations.
a. If allocated Host≠ NULL then
i. assign VM to allotted Host
7. return allocation

4.1 Allocation of data center

4.1.1. Experimental setup

Here, we will discuss a performance analysis of two resource allocation policies i.e. Round Robin and proposed Modified Load Balancing using PSO. Here in experiments, we calculate the migration time of VMs. This is even on modify live migration the pictures and information of VMs should be hold on on a Network connected Storage (NAS); and thus, repeating the VM's storage isn't needed. Live migration creates an additional mainframe load; but, it's been shown that the performance overhead is low [32]. Here we have represented the number of tasks as the workload of the cloud application. We prepared the dataset with all virtual machines having the same configuration. As per input dataset the number of virtual machines is 5 per host machine and total 20 hosts are provided which is allotted with 6 different zones currently available with datacenter.

4.1.2 Implementation

One of the oldest, simplest, fairest and most generally used algorithms is Round-robin (RR). The fundamental purpose of this algorithmic program is to support time-sharing systems. This algorithmic program is comparable to FCFS formula currently it's a preventative FCFS scheduling. The preemption takes place when a hard and fast interval of your time referred to as the time quantum or time slice. Its implementation needs timer interrupts supported that the preemption takes place. The performance of this algorithm depends upon many factors: Size of time quantum.

a) If time quantum's size is large than this algorithm becomes same as FCFS and thus performance degrades
b) If time quantum's size is extremely little then the quantity of context switches will increase significantly and also the time quantum nearly equals the time taken to modify the system from one process to a different, that isn't fascinating the least bit variety of context switches
- The variety of context-switches should not be too several to block the general execution of all the method. Time quantum ought to be massive with relation to the context switch time.

MLPSO algorithm is enforced with the assistance of simulation package Cloudsim and CloudSim based mostly tool. Java language is employed for implementing VM load equalisation algorithmic program. This Simulation could be a technique wherever a program models the behavior of the system (CPU, network etc.,) by calculative the interaction between its totally different entities victimisation mathematical formulas, or really capturing and taking part in back observations. Here it's done underneath a framework that allows seamless modeling; simulation and experimenting on planning while not cloud infrastructures. Here we have a tendency to sculpturesque datacenters, host, service brokers, programming and allocation policies of an outsized scaled cloud platform.

It supports VM provisioning at 2 levels:

a) At the host level: it's attainable to specify what proportions of the process power of every core are going to be assigned to every VM referred to as VM policy Allocation.

b) At the VM level: The VM assigns a set quantity of the accessible process power to the individual application services (task units) that square measure hosted at intervals its execution engine referred to as VM programming.

4.1.3 Parameters

We assume that tasks square measure reciprocally freelance i.e., there's no precedence constraint between tasks and tasks aren't preventive and that they can't be interrupted or enraptured to a different processor throughout their execution.

We have simulated 20 data centers comprising each with 100 heterogeneous physical nodes. Each node is having either linux or window operating system with one core CPU mapped with 10 VMs each. The CPU performance equivalent to 1k or 2k MIPS, 1 TB of storage and 4 GB of RAM. Each VM runs a web-application or any quite application with a variable work that is sculptured to get the employment of central processor in keeping with a uniformly distributed stochastic variable. Every experiment has been run for 60 minutes.

Simulation results

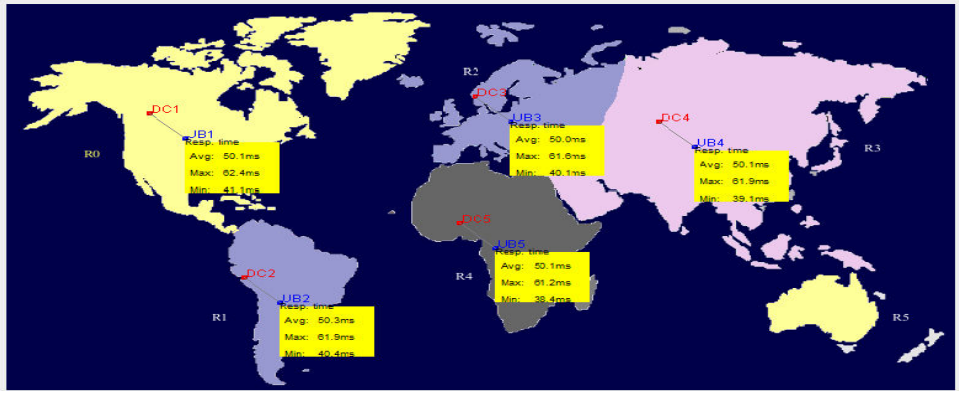


Figure-5. Optimization of VM placement according to the utilization of multiple system resources using RR algorithm.

Overall Response Time Summary

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	50.13	38.39	62.36
Data Center processing time:	0.49	0.02	0.90

Response Time by Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.06	41.11	62.36
UB2	50.29	40.39	61.89
UB3	50.03	40.13	61.63
UB4	50.12	39.13	61.88
UB5	50.14	38.39	61.17

Figure-6. Optimization of VM placement according to the utilization of multiple system resources using RR algorithm.

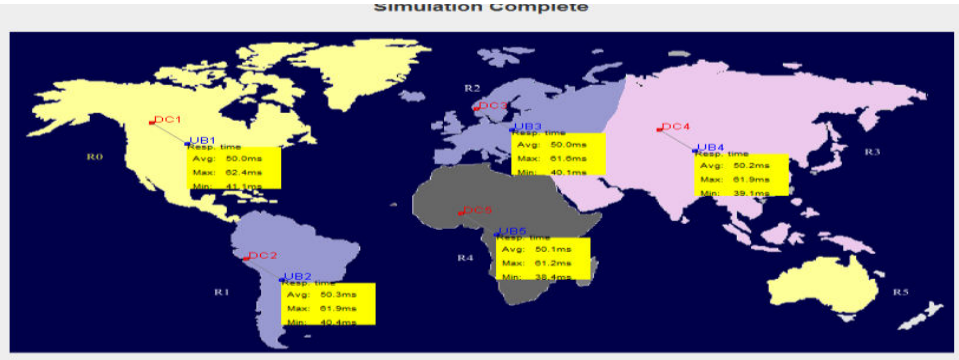


Figure-7. Optimization of VM placement according to the utilization of multiple system resources using MLPSO algorithm.

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	50.13	38.39	62.36
Data Center processing time:	0.49	0.02	0.90

Response Time by Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.02	41.11	62.36
UB2	50.30	40.39	61.89
UB3	49.98	40.13	61.63
UB4	50.20	39.13	61.88
UB5	50.14	38.39	61.17

Figure-8. Optimization of VM placement according to the utilization of multiple system resources using RR algorithm.

5. PERFORMANCE METRICS

Among Various performance metrics we choose throughput into thought so as to live and judge the chosen job programming algorithms. This performance metric is further explained below.

The Ligo real dataset (Brown *et al.*, 2007) is employed. The dataset was drawn by a group of XML files that change within the range of jobs. Every file contains a group of jobs and their necessities like job length, job ID, lists of input and output files and their sizes. Set of six files has been taken to be the employment of this analysis wherever the amount of jobs is 50, 100, 200… 1000. The missing parameters in job's characteristics like the point in time, file location and also the begin and end deadlines square measure completed at random supported the job's length and information size found within the XML files.

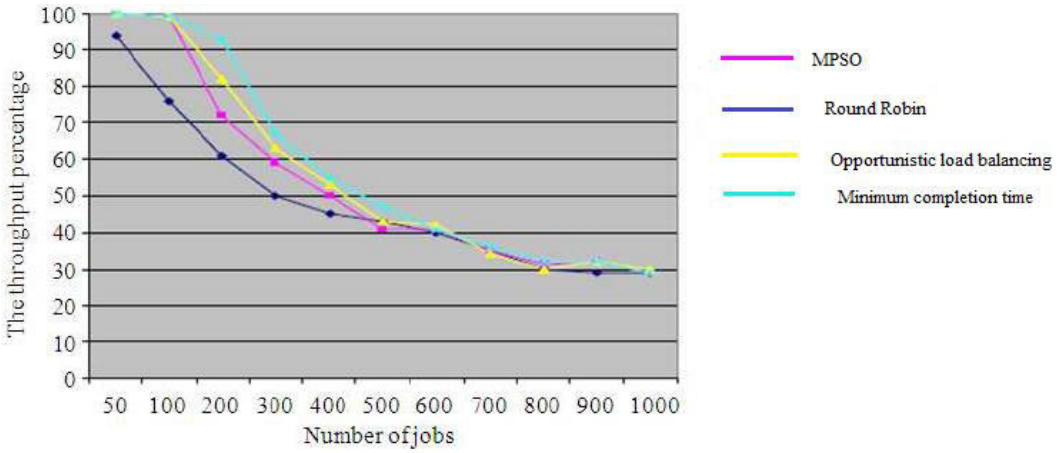
5.1. Throughput

In this study, every job is assumed to own exhausting dead-lines that represent the finishing time. Therefore, the outturn is that the range of executed jobs, that is calculated to check its potency in satisfying the

roles dead-lines. The outturn is calculated victimization Equation 14:

$$\text{Throughput } J = \sum_{j \in J} Xj \quad (14)$$

Where:
j = Job from the list of jobs
J = List of jobs



6. CONCLUSIONS

This paper presented a complete overview of the cloud resource allocation, its applications and also explained conventional techniques for performing the optimized resource allocation in a cloud environment. From the conventional researchers, it is well known that there is an essential requirement of dynamically providing the energy-efficient cloud services to the end-users. Hence, the proposed approach presented a novel idea for maintaining the trade-off between the efficient resource allocation and the performance. It employs the Markov chain model to predict the resource utilization of the workloads and PSO technique to allocate the cloud resources optimally in an effective and efficient manner, associated with the dynamic load balancing based VM migration method.

REFERENCES

[1] Rings T., Caryer G., Gallop J., Grabowski J., Kovacikova T., Schulz S. and Stokes-Rees I. 2009. Grid and cloud computing: opportunities for integration with the next generation network. Springer transaction on Journal of Grid Computing. 7(3): 375-393.

[2] Buyya R., Yeo C.S., Venugopal S., Broberg J., Brandic I., Broberg J. and Brandic I. 2009. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Elsevier transaction on Future Generation Computer Systems. 25: 599-616.

[3] Buyya R, Yeo CS, and Venugopal S. 2008. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. 10th IEEE International Conference on High Performance Computing and Communications. pp. 5-13.

[4] Buyya R, Beloglazov A, and Abawajy J. 2010. Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges. arXiv preprint arXiv:1006.0308.

[5] B. Guenter, N. Jain, and C. Williams. 2011. Managing cost, performance, and reliability tradeoffs for energy-aware server provisioning. IEEE International Conference on Computer Communications (INFOCOM). pp. 1332-1340.

[6] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya. 2011. A taxonomy and survey of energy-efficient data centers and cloud computing systems. Advances in Computers. 82(2): 47-111.

[7] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy and G. Jiang. 2009. Power and performance management of virtualized computing environments via lookahead control. Springer transaction on Cluster Computing. 12(1): 1-15.

[8] S. Srikantaiah, A. Kansal and F. Zhao. 2008. Energy-aware consolidation for cloud computing", ACM Proceedings of the Conference on Power aware computing and systems. 10: 1-5.

[9] Ghribi C., Hadji M. and Zeghlache D. 2013. Energy efficient VM scheduling for cloud data centers: exact allocation and migration algorithms. 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing.

[10] Kaur T. and Chana I. 2015. Energy efficiency techniques in cloud computing - a survey and taxonomy. ACM Proceedings of the Computing Surveys. 48(2).

[11] Beloglazov A. and Buyya R. 2010. Energy efficient allocation of virtual machines in cloud data centers. IEEE/ACM 10th International Conference on Cluster, Cloud and Grid Computing (CCGrid). pp. 577-578.

[12] Al Nuaimi K, Mohamed N, Al Nuaimi M and Al-Jaroodi J. 2012. A survey of load balancing in cloud computing: Challenges and algorithms. IEEE Second Symposium on Network Cloud Computing and Applications (NCCA). pp. 137-142.

[13] Ren X, Lin R, and Zou H. 2011. A dynamic load balancing strategy for cloud computing platform based on exponential smoothing forecast. IEEE International Conference on Cloud Computing and Intelligence Systems. pp. 220-224.

[14] Lee, R, and B. Jeng. 2011. Load-balancing tactics in cloud. IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC). pp. 447-454.

[15] Mohamed N, Al-Jaroodi J and Eid A. 2013. A dual-direction technique for fast file downloads with dynamic load balancing in the Cloud. Elsevier transaction on Journal of Network and Computer Applications. 36(4): 1116-1130.

[16] Wang, S-C., K-Q. Yan, W-P. Liao, and S-S. Wang. 2010. Towards a load balancing in a three-level cloud computing network. IEEE 3rd International Conference on Computer Science and Information Technology (ICCSIT). 1: 108-113.

[17] K. Li, G. Xu, G. Zhao, Y. Dong and D. Wang. 2011. Cloud Task Scheduling Based on Load Balancing Ant Colony Optimization. IEEE Sixth Annual Conference on China grid. pp. 3-9.

[18] Luo L, Wu W, Di D, Zhang F, Yan Y and Mao Y. 2012. A resource scheduling algorithm of cloud computing based on energy efficient optimization methods. IEEE International Conference on Green Computing. pp. 1-6.

[19] Feller E., Rilling L. and Morin C. 2012. Snooze: a scalable and autonomic virtual machine management framework for private clouds. 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. pp. 482-489.

[20] Tawfeek MA, El-Sisi A, Keshk AE, and Torkey FA. 2013. Cloud task scheduling based on ant colony optimization. IEEE 8th International Conference on Computer Engineering and Systems (ICCES). pp. 64-69.

[21] B. Li, J. Li, J. Huai, T. Wo, Q. Li, and L. Zhong. 2009. EnaCloud: An Energysaving Application Live Placement Approach for Cloud Computing Environments. IEEE Proceedings of the International Conference on Cloud Computing. pp. 17-24.

[22] Fu X and Zhou C. 2015. Virtual machine selection and placement for dynamic consolidation in Cloud computing environment. Springer transaction on Frontiers of Computer Science. 9(2): 322-330.

[23] Tarighi M., Motamedi S.A. and Sharifian S. 2010. A new model for virtual machine migration in virtualized cluster server based on fuzzy decision making. Journal on Telecommunications. 1(1): 40-51.

[24] Xiaoli W. and Zhanghui L. 2012. An energy-aware VMs placement algorithm in cloud computing environment. IEEE Proceedings of the Second International Conference on Intelligent System Design and Engineering Application. pp. 627-630.

[25] A. Beloglazov, J. Abawajy, and R. Buyya. 2012. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Elsevier transaction on Future Generation Computer Systems. 28(5): 755-768.

[26] A. Beloglazov and R. Buyya. 2013. Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. IEEE Transactions on Parallel and Distributed Systems. 24(7): 1366-1379.

[27] Feller, Eugen, Louis Rilling, and Christine Morin. 2011. Energy-aware ant colony based workload placement in clouds. IEEE Computer Society In Proceedings of the 12th International Conference on Grid Computing. pp. 26-33.

[28] Liu Z and Wang X. 2012. A PSO-based algorithm for load balancing in virtual machines of cloud computing environment. Springer transaction on Swarm Intelligence. pp. 142-147.

[29] Kolman, Bernard, Robert C. Busby, and Sharon Cutler Ross. 2004. Discrete Mathematical Structures. Fifth edition, Pearson Education, Inc.

[30] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. L. Harris, A. Ho, R. Neugebauer, I. Pratt and A. Warfield. 2003. Xen and the art of virtualization. in SOSP. pp. 164-177.

[31] S. Panda, J. S. Yadav, N. P. Patidar, and C. Ardil. 2009. Evolutionary Techniques for Model Order Reduction of Large Scale Linear Systems. International Journal of Applied Science, Engineering and Technology. 5(1): 22-28.

[32] W. Voorsluys, J. Broberg, S. Venugopal, R. Buyya. 2009. Cost of virtual machine live migration in clouds: a performance evaluation, in: Proceedings of the 1st International Conference on Cloud Computing, CloudCom 2009, Springer, Beijing, China.

[33] Shweta Jaiswal, Atish Mishra, Praveen Bhanodia. 2014. Grid Host Load Prediction Using GridSim Simulation and Hidden Markov Model, in: Proceedings International Journal of Emerging Technology and Advanced Engineering 2014. 4(7).