Enhanced Load Balancing in Clustered Cloud-based Multimedia System

Suresh Babu Kuntumalla[1], Lakshumaiah Maddigalla[2]

[1]M.Tech Scholar (Software Engineering), [2]Working as Assistant Professor and Head of Department (CSE) at Tadipatri Engineering College (TECH), Kadapa Road, Veerapuram (V), Tadipatri, Ananthapur (Dist), Andhra Pradesh (India).

Department of CSE & Tadipatri Engineering College (TECH), Kadapa Road, Veerapuram (V), Tadipatri, Ananthapur (Dist), Andhra Pradesh (India)

Abstract *- In any typical centralized hierarchy cloud-based multimedia system (CMS) contains a resource manager, cluster heads and clustered servers. All multimedia service requests of clients, first directed to resource manager then it forwards to appropriate cluster head based on the task characteristic and then again cluster head forwards the client task to the one of the its clustered servers.. In such a complex CMS, obviously, there are prominent design challenges to overcome load balancing issues and enhance the cost effectiveness in transmitting multimedia content between CMS and clients' systems without exceeding maximum load capacity of cluster server. Unlike to earlier paper works, this research work considered additional real time CMS use cases with performance monitoring at various time periods and modelled them into a mathematical optimization program, which intern to derive a better approach to utilize maximum capacity of servers performance and also ensuring not to exceed the cluster server load. As a result, this enhanced work got resolved runtime issues of load balancing by mathematical derived models. The imitation of intended mathematical optimization programs over CMS multi-server load balancing resulted significantly enhanced.*

Keywords — *load balancing, cloud computing, multimedia system, mathematical optimization, heuristic.*

I. INTRODUCTION

With the fast development of technology and communications, cloud-based multimedia systems (CMS) are advancing to meet a large number of customer needs for several types of multimedia content processing and storage operations through the internet simultaneously. It also has to support various protocols and devices to access the data. In general, all cloud-based multimedia systems and its platform softwares are providing connectivity for a large number of customers at a time for storing and processing media content in a clustered fashion with set of expected Quality of Services though internet. With these highly configured cloud-based systems, customers will benefit to minimize the cost of IT infrastructure and maintenance by paying only for resources that are consumed by them over given period of time.

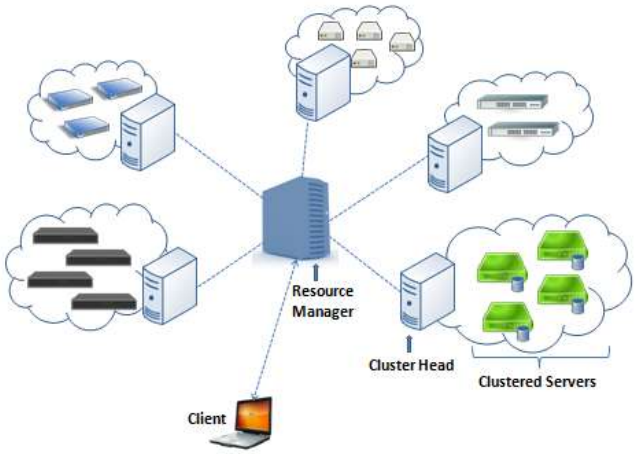

Fig. 1 Characterization of a centralized hierarchy cloud-based multimedia system.

The centralized hierarchy CMS that is taken for this research work contains, one resource manager connected to multiple cluster heads and each cluster head associated with multiple clustered servers as shown in Fig. 1. The flow of a client request in CMS as follows: whenever client sent any multimedia request for fetching or processing of video/audio/still images, etc from his device to CMS, firstly it is directed to resource manager to identify the appropriate cluster servers based on task characteristic and forwards to specific cluster head to distribute the request to one of the its associated clustered servers. It is not difficult to monitor each clustered server importantly impacts the whole CMS performance. Generally, the CMS resource manager responsible for performing load balance over clustered servers with proper distribution of client results. So, there is a lot of scope for resource manager in improving load balancing over CMS.

II. RELATED WORK

In the earlier submitted papers, there are a lot more research studies has been done on load balancing mechanisms, especially over wireless networks based on various factors, policies, and theories. Among these, a few of them also applied to leverage these studies on existing load balancing on CMS to enhance the multimedia process over

clustered servers by reducing transmitting payload information between clients and CMS clustered servers and again without any loss of QoS and overloading on the servers. In general, CMS provides many services to the clients to operate various tasks like storing, organizing, sharing, searching the multimedia content like images, animations, audio, video, etc. It also supports various devices like laptops, tablets, smartphones etc for performing these operations. With lots of innovation in communications and technologies, some of the vendors are also delivering advanced features like camera-to-cloud to store the live recording data into cloud directly, multi-user concurrent operations (for team collaboration), live streaming, single & multi user 3D graphical gaming etc. There is a lot of variation in utilization of server resources based on type of multimedia task for e.g., server need more CPU, RAM and storage space for processing gaming or videos when compare to images or web pages. The earlier works which are on time based are won't be considerable for this research work.

To meet above dynamic real time load balancing requirements, we reinforced each CMS cluster group with particular multimedia process tasks so that each customer request of specific type always redirected to that particular cluster group based on the task type or characteristic. Thereafter, each customer requests multiple types of services at specific times and captured load balancing issues for every trial and modelled them into mathematical optimizing program models for analyzing the problem in mathematical approach. This modeling approach helps to learn or discover the problem solution with practical methods and it also observed that the results of this approach have shown significant improvement in load balancing of CMS by distributing the load equally across clustered servers. This approach may or may not impact other problem solving techniques of cloud distributed techniques.

III. Problem Description

In general, there are various types of multi-media services are supported by CMS, but for this experiment we have adapted only one type of client requests always to deal with load balancing problems of CMS, by the way it will avoid additional load balancing time on change of service type. The details of overview of system and problem modelling are as follows.

A. Overview of System

All CMSs are classified into two groups: *centralized* and *decentralized*, but here we have selected centralized CMS as depicted in Fig. 1 which contains one *resource manager*, number of *clustered servers* and each cluster is associated with one *cluster head*. Unlike decentralized CMS, any request that from clients, first directed to centralized CMS *resource manager* which has all *clustered server's* service information, then distribute to *cluster head* based on the task type and cluster head is responsible for redistributing client request to one of the associated clustered servers by balancing load over the servers as much as possible. The centralized CMS is scalable by adding less overhead when we compare to decentralized framework. But still, centralized CMS platforms are not guaranteed over on load balancing when there are issues in resource manager distribution logic.

B. Problem Modelling

To model the CMS that can adjust to time progress and divided into set of time periods. At the *t*-th time period, the CMS can be formulated as a complete weighted bipartite graph mentioned below.

$G_t = (U, V, E, \Phi, \Psi^t, q, r^t, \omega^t)$ in which

- U is the set of vertices that represents the server clusters of the CSM;
- V is the set of vertices that represent clients;
- E is the set of edges between U and V, in which each edge $e_{ij} \in E$ represents the link between clustered server $i \in U$ and client $j \in V$;
- $\Phi: U \rightarrow$ N is a function used to restrict that clustered server i can only cope with multimedia tasks of type Φ_i;
- $\Psi^t: V \rightarrow$ N is a function used to represent that client j requests the multimedia service of type Ψ^t_j at the t-th time period;
- $q : U \cup V \rightarrow$ N is a function used to represent that server cluster i can provide the multimedia service of QoS q_i;
- $r^t : U \cup V \rightarrow$ N is a function used to represent that client j requests the multimedia service of QoS requirement r^t_j at the t-th time period;
- $\omega^t : E \rightarrow R^+$ is the weight function associated with edges, in which ω^t_{ij} donates the ω^t value that represents the cost for transmitting multimedia data between server cluster i and client j at the t-th time step, which is defined as follows:

$$\omega_{ij}^t = \begin{cases} \infty, & \text{if } d_{ij}^t \rightarrow \infty \text{ or } \Phi_i \neq \varphi_j \\ d_{ij}^t\, l_{ij}^t, & \text{otherwise} \end{cases} \quad \text{..... (1)}$$

where d^t_{ij} is the network proximity between server cluster i and client j; l^t_{ij} is the traffic load on the link between server cluster i and client j that is defined as follows:

$$l_{ij}^t = \sum_{k \in K_i} u_{ikj}^t C_{ik} \quad (2)$$

where K_i is the set of servers in clustered servers i; u^t_{ikj} is the server utilization ratio of server k in

clustered server i due to client j and C_{ik} is its capacity.

By the way the proximity d_{ij}^t between the clustered server i and client j in equation 1 is mandatory to calculate at every time period due to chance in CMS network topology. The measure of proximity between the clustered server and client considered as distance between them. Here is the example that describes the measuring of proximity: Initially, we calculate the distance of a node to a given set of prominent nodes in the topology by the latency between those nodes. Imagine if there are 3 prominent nodes in the network topology, the latencies from referred node to those prominent nodes are 35, 8, and 21 respectively. Nodes are rated with latency information as mentioned: latencies range 0 in [0, 13], latencies range 1 in [13, 43], and latencies range 2 in greater than 43. Thus, the prominent order of the considered node is '201'. With help of prominent order, overall nodes are categorized into multiple bins.

By the above notes, the mathematical model of the load balancing issue at the t-th time can be defined in following mathematical expression:

Minimize

$$\lambda \frac{\sum_{i\in U}\sum_{j\in V} x_{ij}^t \omega_{ij}^t}{\sum_{j\in V} \omega_{\max}} + (1-\lambda)\left(1-\frac{\sum_{i\in U}\sum_{j\in V} x_{ij}^t}{|V|}\right) \quad (3)$$

subject to

$$\sum_{i\in U} x_{ikj}^t \le 1, \forall_j \in V, \quad (4)$$

$$\sum_{i\in U} x_{ikj}^t \, l_{ij}^t \le \sum_{i\in U} C_{ik}, \forall_i \in U \quad (5)$$

$$x_{ij\,i}^t = x_{ij}^t \Psi_j^t, \forall_i \in U, j \in V \quad (6)$$

$$x_{ij}^t q_i \ge x_{ij}^t r_j^t, \forall_i \in U, j \in V \quad (7)$$

$$x_{ij}^t \in \{0,1\}, \forall_i \in U, j \in V \quad (8)$$

Where, x_{ij}^t is an indicator variable defined as mentioned below:

$$x_{ij}^t = \begin{cases} 1, & \text{if client } j \text{ is assigned to server cluster } i \text{ at the } t-\text{th time period;} \\ 0, & x \ge 0 \end{cases}$$

In the above mathematical model, the indicator variable x_{ij}^t is meant to identify whether to fill the gap e_{ij} between clustered server i and client j in the complete bipartite graph $U \times V$. The main intent of the model is a weighted in the addition of two terms: the first is to reduce the total weighted values of the bipartite graph, i.e., to reduce the overall content of multimedia data at given t-th time period, while the other one is to increase the number of connections between servers and clients.

IV. GENETIC ALGORITHM

A. *Genetic Algorithm with Immigrant Scheme*

The fundamental theme of the GA is to simulate the behaviour changes of a population of chromosomes to figure out the resolution nearer to the global optimal result by using three primary evolutional actions: *selection*, *crossover*, and *mutation*. The main concept of the GA is that standard chromosomes can only exist for the next generation, and mainly the standard chromosome in the last generation represents the ending result that has a better performance. We first derive how a *chromosome* denotes a resolution of particular scenario or use case, and how the *suitableness* of each chromosome interprets the targeted value of that specific result. The first step of the pseudocode is to keep a generation of chromosomes either in random or any specific order. Secondly, a set of the chromosomes from overall population are *selected* as the parental pool and *crossovered* each pair of chromosomes to generate child chromosomes. Thirdly, subset of initial chromosomes and its child chromosomes contains the next generation. With this gradual *mutation* by the number of repeated cycles, the result provided by the chromosome with the best values in the final cycle is resulted as the best resolution.

B. *Algorithm 1: Dynamic Load Balancing Algorithm*

1: **for** t = 1,2,3 . . . do
2: consider complete weighted bipartite graph G_t
3: remove the links in G_t violating Constraints (6) and (7)
4: calculate $\{l_{ij}^t\}$ and $\{\omega_{ij}^t\}$ By calling *Algorithm 2*
5: assign $\{x_{ij}^t\}$ by calling *Algorithm 3*
6: **end for**

C. *Algorithm 2: Calculate Weights*

1: **for** each client $j \in V$ **do**
2: measure the latency from client j to each prominent node
3: compute the prominent order l_j of client j
4: obtain the set of available clustered server U_j
5: **for** each $i \in Uj$ **do**
6: measure the latency from clustered server i to each prominent
7: compute the prominent order l_i of clustered server i
8: **if** $l_i = l_j$ **then**
9: measure the network proximity d_{ij}^t between clustered server i and client j

10: measure server utilization ratios u_{ikj}^t for all $k \in Ki$
11: calculate l_{ij}^t and ω_{ij}^t by equations (2) and (1) respectively
12: **else**
13: $\omega_{ij}^t = l_{ij}^t = \infty$
14: **end if**
15: **end for**
16: **end for**

D. Algorithm 3 Genetic Algorithm (Graph Gt, Time Period t)

1: **if** $t = 1$ then
2: generate and evaluate the initial population P_0^t of size η in which each chromosome has to satisfy Constaints (4) and (5).
3: **else**
4: $P_0^t \leftarrow P_r^{t-1}$
5: **end if**
6: $i \leftarrow 0$
7: **while** $i < \tau$ or the convergent condition is not achieved **do**
8: select the parental pool Q_i^t from P_i^t
9: reproduce a new population P_i^t of size of η by performing *crossover* procedure on pairs of chromosomes in Q_i^t with probability p_c
10: perform *mutation* procedure on chromosome in P_i^t with probability p_m
11: fix each infeasible chromosome in P_i^t
12: evaluate P_i^t
13: **if** $r_e > 0$ **then**
14: a number of the best chromosome E_{i-1}^t, called elite, are selected from the previous generation P_{i-1}^t
15: generate and evaluated $r_e \cdot \eta$ elite immigrations
16: **end if**
17: **if** $r_r > 0$ **then**
18: generate and evaluate $r_r \cdot \eta$ random immigrant
19: **end if**
20: replace the worst chromosomes in P_i^t with the above elite and random immigrants
21: $P_{i+1}^t \leftarrow P_i^t$
22: $i \leftarrow i + 1$
23: **end while**
24: output the best found chromosome as the solution at the *t*-th time period.

E. Basic Elements of GA

To use GA in order to solve the CMS-dynMLB problem, we first define the basic elements of GA (i.e., population, chromosome, fitness function) for the problem as follows.

1) Population: A population consists of a number of chromosomes, and the number of chromosomes depends on the given initial population size.

2) Chromosome: A solution for the CMS-dynMLB problem consists of all the indicator variables $\{ x_{ij}^t \mid \forall \mid i \in U, j \in V\}$.The solution for indicator variables $\{ x_{ij}^t\}$ is encoded as a sequence of decimal numbers of length n: $\{\sigma 1 , \cdots , \sigma j , \cdots , \sigma n \}$ where $\sigma j \in \{0, 1, 2, \cdots ,m\}$ represents the link assignment of client j with the following two cases. 1) If $\sigma j = 0$, then each $x_{ij}^t = 0$ for any $i \in U$, i.e., client j is not linked at the *t*-th time period

3) Otherwise, $\sigma j = k \neq 0$, meaning that $x_{ik}^t = 1$ and $x_{ij}^t = 0$ for any $i \neq k$, i.e., client j is linked to server cluster k uniquely. 3) Fitness Function: Fitness function is the measure for determining which chromosomes are better or worse. We let the objective (3) of our concerned problem as our fitness function as follows: $f(X(t)) = \lambda \cdot \sum i \in U \sum j \in V\ x_{ij}^t \omega_{ij}^t / \sum j \in V\ wmax + (1 - \lambda) \cdot (1 - \sum j \in V \sum i \in U\ x_{ij}^t / |V|)$ where $X(t)$ is the profile of $\{x_{ij}^t\}$.

F. Main Components of GA

1) Initialization: In Lines 1–5 of Algorithm 3, we use two cases to get initial population at each time step. If $t >= 1$, then just take the final population at the previous time step as the initial population at the current time step. For the other case (i.e., $t = 1$), we randomly produce the initial population by using Algorithm 4, which is explained as follows. Lines 1–3 find the set *arg(Uj)* that collects the indices of the available server clusters for each client *j*. Next, the set *arg(Uj)* is used to construct a population of η chromosomes in Line 5, and then we repair each chromosome to be feasible in Lines 6–21. The idea of the repairing operation is that if (5) is violated, then we find another available server cluster to serve the violated load; otherwise, we drop the load.

2) Selection, Crossover: The range selection procedure is used. Each individual in the population is assigned a numerical rank based on their fitness, and selection is based on this ranking rather than absolute differences in fitness. The advantage of this method is that it can prevent very fit individuals to gain dominance at first at the expense of the less fit, which would reduce the genetic diversity of the population and could hamper the search for an acceptable solution [7]. One-point crossover operation is used in which two selected chromosomes are *crossovered* for generating two new child chromosomes in order to keep some characteristics of its parent chromosomes.

3) Mutation: Changing some genes on some chromosomes. let p_m be given probability to mutate. In the mutated chromosome, a nonzero gene σ_x is chosen randomly, and then we randomly find a zero gene σ_y that can accommodate σ_x and do not violate any problem constraint. Then, we swap the values of σ_x and σ_y .

4) Repair: The modified chromosomes may become infeasible, and hence repair is required. The repairing operation is referred to Lines 6–21 in Algorithm 4, which have been used in initializing chromosomes.

5) Termination: If the difference of average fitness values between successive generations in the latest ten generations is not greater than 1% of the average of the average fitness values of these ten generations, or the maximum generations are achieved, then GA stops. After termination, the best chromosome from the latest population is chosen, and its corresponding load assignment is outputted as the final solution.

V. Implementation and Experimental Results

In addition to GA as an immigration scheme as mentioned earlier papers, we have considered two additional factors: the best multimedia compression algorithm to efficiently compress the data stream between cloud server and client, and applying an efficient encoding & caching algorithm based on type of multimedia content, to still minimize the data transmission.

A. Implementation

CloudSim 3.0.3 (latest version) is used for simulation with Java as a programming language. CloudSim has support for modelling and simulation of large scale Cloud computing environments, including data centers, on a single physical computing node. It has necessary classes in Java for simulating every entity that involved in our project like Datacenter, Host, VM, Cloudlet, DatacenterBroker etc. The CloudSim toolkit supports both system and behaviour modelling of Cloud system components. It has availability of a virtualization engine that aids in the creation and management of multiple, independent, and co-hosted virtualized services on a data centre node.

Our simulation was tested on an Intel Core i7-4910MQ CPU at 3.90 GHz with 32-GB memory.

B. Results

By adapting various other techniques to minimize the multimedia data storage and data transmission between servers and client, though efficient encoding, compression and caching algorithms over existing genetic algorithm for load balancing of centralized CMS outputted even better results.

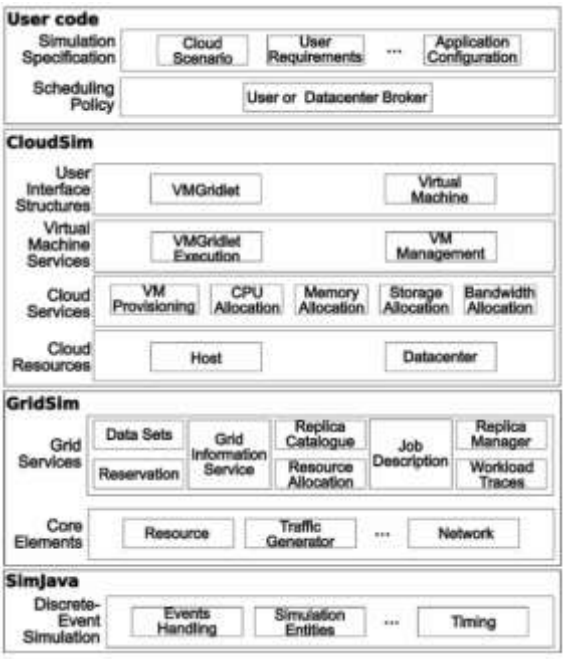

Fig 2 Layered CloudSim Architecture

VI. Conclusion

The load balancing implementation in the cloud computing environment is to provide on demand resources with high availability. But the existing load balancing approaches suffers from various overhead. The enhanced load balancing approach using genetic algorithm with other best efficient compression, encoding and caching techniques over *Platform as a service & Application as a service layers* are proposed to overcome the aforementioned limitations.

References

[1] Chun-Cheng Lin, Hui-Hsin Chin, Der-Jiunn Deng, "Dynamic Multi-Service Load Balancing in Cloud-based Multimedia System" IEEE Systems Journal, vol 8, Issue 1, pp.225 - 234, 2014

[2] W. Zhu, C. Luo, J. Wang, and S. Li, "Multimedia cloud computing: An emerging technology for providing multimedia services and applications," IEEE Signal Processing Magazine, vol. 28, no. 3, pp. 59–69, 2011.

[3] K.-P. Chow and Y.-K. Kwok, "On load balancing for distributed multiagent computing," IEEE Transactions on Parallel and Distributed Systems, vol. 13, no. 8, pp. 787–801, 2002.

[4] C.-F. Lai, Y.-M. Huang, and H.-C. Chao, "DLNA-based multimedia sharing system over OSGI framework with extension to P2P network," IEEE Systems Journal, vol. 4, no. 2, pp. 262–270, 2010.

[5] X. Qin, H. Jiang, A. Manzanares, X. Ruan, and S. Yin, "Communicationaware load balancing for parallel applications on clusters," IEEE Transactions on Computers, vol. 59, no. 1, pp. 42–52, 2010.

[6] X. Nan, Y. He, and L. Guan, "Optimal resource allocation for multimedia cloud based on queuing model," in Proceedings of 2011 IEEE 13th International Workshop on Multimedia Signal Processing (MMSP 2011). IEEE Press, 2011, pp. 1–6.

[7] W. Hui, H. Zhao, C. Lin, and Y. Yang, "Effective load balancing for cloud-based multimedia system," in Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology. IEEE Press, 2011, pp. 165–168.

[8] C.-Y. Chen, H.-C. Chao, S.-Y. Kuo, and K.-D. Chang, "Rule-based intrusion detection mechanism for IP multimedia subsystem," Journal of Internet Technology, vol. 9, no. 5, pp. 329–336, 2008.

[9] R. Van den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-optimal scheduling in hybrid IaaS clouds for deadline constrained workloads," in Proceedings of 2010 IEEE 3rd International Conference on Cloud Computing. IEEE Press, 2010, pp. 228–235.

[10] Y.-M. Huang, M.-Y. Hsieh, H.-C. Chao, S.-H. Hung, and J. H. Park, "Pervasive, secure access to a hierarchical-based healthcare monitoring architecture in wireless heterogeneous sensor networks," IEEE Journal on Selected Areas of Communications, vol. 27, no. 4, pp. 400–411, 2009.

[11] T.-Y. Wu, H.-C. Chao, and C.-Y. Huang, "A survey of mobile IP in cellular and mobile ad-hoc network environments," Ad Hoc Networks Journal, vol. 3, no. 3, pp. 351–370, 2005.

[12] L. J. Wu, A. E. AL Sabbagh, K. Sandrasegaran, M. Elkashlan, and C. C. Lin, "Performance evaluation on common radio resource management algorithms," in Proceedings of 2010 IEEE 24th International Conferenceon Advanced Information Networking and Applications Workshops(WAINA 2010). IEEE Press, 2010, pp. 491–495.

[13] C.-Y. Chang, T.-Y. Wu, C.-C. Huang, A. J.-W. Whang, and H.-C. Chao, "Robust header compression with load balance and dynamic bandwidth aggregation capabilities in WLAN," Journal of Internet Technology, vol. 8, no. 3, pp. 365–372, 2007.

[14] J. Sun, X. Wu, and X. Sha, "Load balancing algorithm with multiservice in heterogeneous wireless networks," in Proceedings of 6th International ICST Conference on Communications and Networking in China (ChinaCom 2011). IEEE Press, 2011, pp. 703–707.

[15] H. Son, S. Lee, S.-C. Kim, and Y.-S. Shin, "Soft load balancing over heterogeneous wireless networks," IEEE Transactions on Vehichular Technology, vol. 57, no. 4, pp. 2632–2638, 2008.

[16] L. Zhou, H.-C. Chao, and A. V. Vasilakos, "Joint forensics-scheduling strategy for delay-sensitive multimedia applications over heterogeneous networks," IEEE Journal on Selected Areas of Communications, vol. 29, no. 7, pp. 1358–1367, 2011.

[17] CloudSim : A Framework For Modeling And Simulation Of Cloud Computing Infrastructures And Services https://code.google.com/p/cloudsim/

[18] P.Gayathri Atchuta, L.Prasanna Kumar, Amarendra Kothalanka"Improving Performance and Reliability Using New Load Balancing Strategy with Large Public Cloud", International Journal of Engineering Trends and Technology (IJETT), V21(8),400-404 March 2015. ISSN:2231-5381. www.ijettjournal.org. published by seventh sense research group