Mathematical Modelling of Joint Routing and Scheduling for an Effective Load Balancing in Cloud

Suriya Begum
Research Scholar
Visvesvaraya Technological University
Belgaum, India

Prashanth C.S.R, PhD
Prof. & Head of Department
Computer Science & Engineering
New Horizon College of Engineering
Bangalore, India

ABSTRACT
Due to the increasing adoption of the cloud in majority of the business, the level of traffic intensity is increasing leading to a challenging situation for traffic management in cloud. There were various algorithms in past that has discussed about the load balancing techniques concerning the cloud environment, but very few of them are found to be actually effective. The proposed system therefore presents a mathematical model exclusively considering virtual machine for performing load balancing. The system jointly addresses the routing as well as task scheduling and also focuses on the issues pertaining to resource allocation. The model is designed and compared with existing load balancing algorithms, where the simulation results shows better throughput by highlighting minimized waiting time for jobs with faster completion of task.

Keywords
Cloud Computing, Resource Allocation, Traffic.

1. INTRODUCTION
Cloud computing can be illustrated as the deployment of computing resources including hardware and software that are delivered as a service over a large scale network [1]. The name 'cloud' has been originated from the use of a cloud-shaped symbol as an abstraction for the complex infrastructure that is contained in a system diagrams. The origination of the term 'cloud computing' is incomprehensible, but it seems to be derived from the practice of using schematic drawings of stylized clouds to denote networks in diagrams of computing and communications systems. The word cloud is used as a parable for the large scale network that is based on the standardized use of a cloud-like shape to denote a network on telephony schematics and later to depict the Internet in computer network diagrams as an abstraction of the underlying infrastructure it represents. The cloud symbol was used to represent the Internet as early as 1994 [1]. The fundamental concept of cloud computing basically dates back to the 1950s when there was an availability of large scale mainframe in institutions and corporations. Due to the expensive nature of mainframe, there arise a need to find an alternative solution for permitting the multiple users for accessing and sharing equivalent CPU time thereby truncating the possibility of periods of inactivity (also termed as time-sharing). As computers became more prevalent, scientists and technologists explored ways to make large-scale computing power available to more users through time sharing, experimenting with algorithms to provide the optimal use of the infrastructure, platform and applications with prioritized access to the CPU and efficiency for the end users. The high cost of these powerful computing systems has forced many prime organizations to take an initiative to explore better cost effective solution using time sharing. The prime organization will include IBM, GE, National CSS etc who took the initiative to launch and marketed time sharing. With the rapid growth of internet technology and standards, various products and demands of distributed computing were on high escalation. The presence of pervasive high computing network, cost effective computing devices, storage devices along with large scale use of virtualization of hardware, service oriented architecture has paved the path for high demands in new technology, so called as 'cloud computing.'

Load balancing [2] is a process of reassigning the total load to the individual nodes of the collective system to make resource utilization effective and 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. A load balancing protocol is dynamic in nature doesn't contemplate the previous state or behavior of the system, that is, it depends on the current behavior of the system. It is common these days in redundant high-availability computer systems that incoming network traffic is distributed on network level by deploying one of the frequently used network load balancing algorithms like:- random-allocation, round-robin allocation, weighted round-robin allocation, etc). These algorithms use solely network parameters of incoming traffic to create selections wherever to forward traffic, with none data from different elements of database system, like current load of application or info servers. Since these days it is extremely common to possess internet servers acting as application servers, it is usual that load balancers use session-switching technique, which suggests that once a user opens website on one server, it will stay on it server whereas the session lasts. Central to many other issues lies the establishment of an effective load balancing algorithm. The load can be CPU load, memory capacity, delay or network load. Load balancing is the process of distributing the load among various nodes of a distributed system to improve both resource utilization and job response time while also avoiding a situation where some of the nodes are heavily loaded while other nodes are idle or doing very little work. Load balancing ensures that all the processor in the system or every node in the network does approximately the equal amount of work at any instant of time. This technique can be sender initiated, receiver initiated or symmetric type (combination of sender initiated and receiver initiated types). The domain of cloud computing is still surfaced by many issues which will be discussed in this paper in later section. The prime focus of the paper will be to analyze the research issues in load balancing protocols or understanding its requirement in cloud platform.

Since, with the scene of rapid use of cloud computing resources, the demand along with provisioning of the cloud resources has to be effectively design to claim better SLA (Service Level Agreement) with zero downtime claims. Currently, there is a presence of multiple vendors offering cloud services (Amazon, Microsoft, IBM, Google, Salesforce, HP, Oracle, Citrix, EMC etc.) and there are growing numbers of clientele too. Hence, it is quite obvious that catering the massive and dynamic requirements of such exponentially growing clients will become one of the most challenging issues. And to mitigate this issue, an effective load balancing technique should be explored. The proposed paper will introduce a thorough analysis of the evolution of cloud platform right from the origination of the initial distributed computing system. The paper will mainly focus on the research issues of load balancing. The main contributions of this paper are summarized below.

- The system depicts the efficient region in cloud framework by incorporating its connection to the efficient region of the wireless system application. The efficient region of the cloud framework can be states to be group of the dynamic loads of traffic that can stabilize and manage the queue system.
- The next contribution of the system is to use the best fit provisioning technique and thereby utilize frequently use MaxWeight algorithm in ideal scenario where incoming jobs can be preempted and possibly drift among the clusters where each clusters can perform auto re-configuration based on the incoming load. However, the system is less dependent on preemption and virtual machine migration, as it is an expensive process and hence the proposed system adopts non-preemptive policy that biases the allocation of new task to a cluster using optimal queuing mechanism and intermediate processing among the processing elements. The system represents the throughput of load balancing and exhibits that it can actually accomplish any random fraction of the efficient region if the framework parameters are wisely selected.
- The computation discussed in the previous state actually requires central queues. However, much efficient technique is to direct the jobs to clusters exactly after the arrival. The system considers shortest queuing technique that directs the task to the cluster with the shortest queue.
- The shortest queuing mechanism is required to maintain a track of queuing length at all the clusters. The system also ensures the faster job swapping between the processing elements that can ensure shorter waiting time, less delay, as well as better processing speed. The system can be well-integrated within the racks of server to maintain the routing keeping the existing dynamic load of the uncertain users of the cloud services.

2. RELATED WORK

Our previous paper [3] has discussed some of the lead balancing techniques with respect to cloud environment along with evidential based discussion on issues in cloud platform. This section discusses about the past work being carried out to present a framework that mitigates issues in load balancing in cloud. Xu et al. [4] introduced a better load balance model for the public cloud based on the cloud partitioning concept with a switch mechanism to choose different strategies for different situations. The algorithm applies the game theory to the load balancing strategy to improve the efficiency in the public cloud environment.

Nishant et al. [5] proposed an algorithm for load distribution of workloads among nodes of a cloud by the use of Ant Colony Optimization (ACO). This is a modified approach of ant colony optimization that has been applied from the perspective of cloud or grid network systems with the main aim of load balancing of nodes. This modified algorithm has an edge over the original approach in which each ant build their own individual result set and it is later on built into a complete solution.

Thazhathethil et al. [6] introduced a system which has main controller, balancers and servers. The main controller selects the appropriate balancer for a particular job. The balancer further selects the server having minimum load. Hence, this system will help dynamically allocate jobs (data) to the least loaded server which will result in an efficiently balanced cloud system.

Sethi et al. [7] presented the novel load balancing algorithm using fuzzy logic in cloud computing, in which load balancing is a core and challenging issue in Cloud Computing. The processor speed and assigned load of Virtual Machine (VM) are used to balance the load in cloud computing through fuzzy logic. Bhargava et al. [8] demonstrated a Load balancing is one of the major and important part in cloud computing which is necessary to share out the dynamic work load across numerous nodes to make sure that no single node is staggered. It helps in most favorable consumption of resources and hence in augmenting the performance of the system.

Tiwari et al. [9] introduced a better approach for public cloud load distribution using partitioning and game theory concept to increase the performance of the system. And also proposed the Load balancing is the process of distribution of workload among different nodes or processor. The purpose of load balancing is to improve the performance of a cloud environment through an appropriate distribution strategy. Game theory is the formal study of conflict and cooperation. Game theoretic concepts apply whenever the actions of several agents are interdependent.

Cheng et al. [10] formulated the problem as a constrained kmedoids clustering problem, and propose a novel Weighted Partitioning Around Medoids (wPAM) solution. They present a dissimilarity/similarity metric to facilitate the preservation of the social relationship. They compare their solution with other state-of-the-art algorithms, and the preliminary results show that it significantly decreases the access deviation in each cloud server, and flexibly preserves the social relationship. Priyadarshinee et al. [11] discussed only two divisible load scheduling algorithms that can be applied to clouds, but there are still other approaches that can be applied to balance the load in clouds. The performance of the given algorithms can also be increased by varying different parameters.

Singhal et al. [12] focused on a two level task distribution system over three tier cloud architecture. This paper studies the use of a hybrid task scheduling algorithm which combines two commonly used scheduling methods, the MM (Min-Min) and OLB (Opportunistic Load Balancing) to create their hybrid Balanced Load Min-Min algorithm (BLMM) algorithm.

Mashaly et al. [13] introduced a new approach towards load balancing in cloud-based content delivery networks. By applying adaptive server activations/deactivations at each data center in the cloud, overloaded data centers can move the extra load to lightly loaded ones without affecting the

performance of any of the data centers or violating service level agreements (SLA) of users.

Mohana Priya and Subramani [14] proposed algorithm it uses the active monitoring load balancing algorithm and resource aware scheduling algorithm for improved resource utilization and scheduled load balancing for high performance in cloud systems. Galloway et al. [15] presented a load balancing approach to IaaS cloud architectures that is power aware. Since the cloud architecture implemented by local organizations tends to be heterogeneous, they take this into account in their design. Their Power Aware Load Balancing algorithm, PALB, maintains the state of all compute nodes, and based on utilization percentages, decides the number of compute nodes that should be operating.

Probhuling [16] discussed with the cloud computing requirements for access control, migration, security, data availability, trust issues and sensitive information. Beside this reviewed the algorithms used in the cloud computing for load balancing; weighted active monitoring load balancing algorithm, dynamic load balancing, static algorithms, dynamic algorithms, ant colony optimization algorithms, and load balancing in distributed systems. Mahapatra and g [17] proposed two alternate load balancing mechanisms that utilize the realtime network state information to achieve load balancing. Their experimental results indicate that these techniques can improve the performance over flow-level VLB in many situations. They also investigate the flow-level Valiant Load Balancing (VLB) technique that uses randomization to deal with highly volatile data center network traffics.

Adnan et al. [18] presented an offline formulation for geographical load balancing problem with dynamic deferral and give online algorithms to determine the assignment of workload to the data centers and the migration of workload between data centers in order to adapt with dynamic electricity price changes. They compare their algorithms with the greedy approach and show that significant cost savings can be achieved by migration of workload and dynamic deferral with future electricity price prediction. They validate their algorithms on MapReduce traces and show that geographic load balancing with dynamic deferral can provide 20-30% cost-savings.

3. PROBLEM DESCRIPTION

Cloud computing services are emerging as an important resource for personal as well as commercial computing applications. The simplest architecture for serving the jobs is to queue them at a central location. In each time slot, a central scheduler chooses the configuration at each server and allocates jobs to the servers, in a preemptive manner. As discussed in the review of literature, this problem is then identical to scheduling in an adhoc wireless network with interference constraints. In practice, however, jobs are routed to servers upon arrival. Thus, queues are maintained at each individual server. It was shown that join-the-shortest queue-type algorithms for routing, along with the MaxWeight scheduling algorithm at each server is throughput optimal.

Characterizing the exact delay or queue length in general is difficult. The random arrival of load in such an environment can cause some server to be heavily loaded while other server is idle or only lightly loaded. Equally load distributing improves performance by transferring load from heavily loaded server. Efficient scheduling and resource allocation is a critical characteristic of cloud computing based on which the performance of the system is estimated. The considered characteristics have an impact on cost optimization, which can be obtained by improved response time and processing time. One of the major issues of cloud computing is load balancing because overloading of a system may lead to poor performance which can make the technology unsuccessful. So there is always a requirement of efficient load balancing algorithm for efficient utilization of resources.

Hence, the primary target of the proposed system is to address these issues of load balancing in the proposed system. The notations used in the proposed system are highlighted in the Table 1.

Table 1 List of Notation Used

Notation	*Meaning*
η_{VM}	Number of VMs of M-dimension
Re_p	Amount of resource (e.g. memory) of *p*-numbers of processing elements
S_iRe_p	Specific amount of type p-resource of i^{th} server
Size	Size of jobs for VM
t	Time-slots
$\gamma_m(t)$	Load Process
λ_m	Number of jobs arrival rate
Prob	Probability of random events
σ	Expected value
$\theta_m(t)$	Quantity of type *m*-jobs
$\delta_m(t)$	Workload (or queue) of *m*-type
F_i	Feasible configuration of VM in i^{th} server
δ_{mi}	Queue length of *m*-jobs at i^{th} server

4. PROPOSED MODEL

The prime purpose of the proposed system is to introduce a novel mathematical model for load-balancing in cloud environment. A cloud system consists of a number of networked servers. Each of the servers may host multiple Virtual Machines. Each Virtual Machine requires a set of resources, including CPU, memory, and storage space. Virtual Machines are classified according to the resources they request. We assume there are *M* distinct Virtual Machine configurations where each Virtual Machines configuration is specified in terms of its requirements for *p* different resources in processing elements.

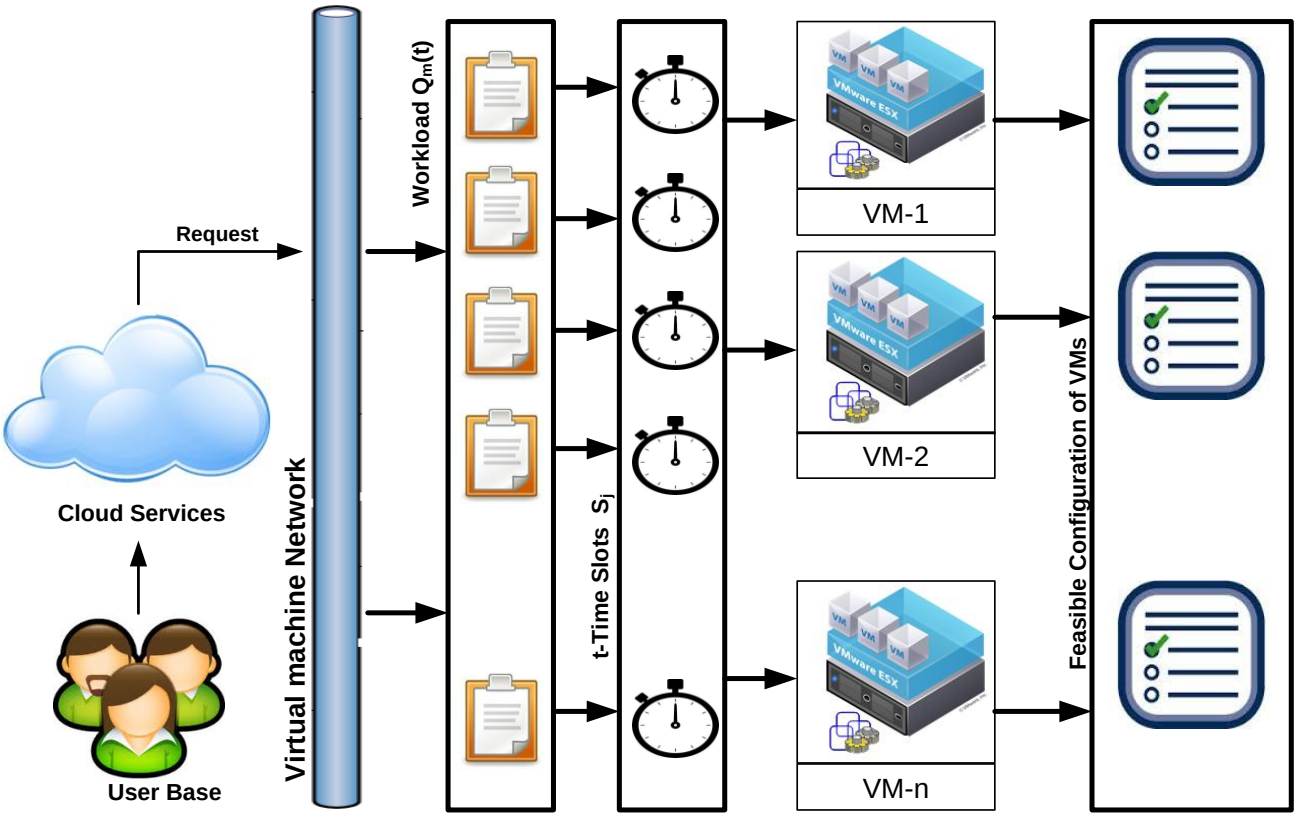

Fig.1 Proposed Load Balancing Mechanism using Job Allocation for VMs

Let Re_p be the amount of type-p resource (e.g. memory) required by a type-m VM (e.g., a standard extra large Virtual Machines). Further, we assume that the cloud system consists of L different servers. Let S_iRes_p denote the amount of type-p resource at server i. Given a server, an M-dimensional vector η is said to be a feasible VM-configuration if the given server can simultaneously host η_1 type-1 VMs, η_2 type-2 VMs… and η_M type-M VMs. In other words, η is feasible at ith server if and only if,

$$\sum_{m=1}^{M} \eta_{(VM)m} \operatorname{Re}_p \le S_i \operatorname{Re}_p \qquad (1)$$

For all p. We let η_{max} denote the maximum number of VMs of any type that can be served on any server. In the proposed system, a cloud system is considered that hosts VMs for clients. The request generated by VM by user represents the specification of the VM that is actually demanded by the client's operational need. For easiness in computation, the study represents the term "jobs" for every request of VM. Fig.1 gives pictorial representation of this process. The system considers m-type jobs, if m number of specified request arrives at VMs. For better articulation of jobs, the proposed study considers time-slotted features whereby the size of the jobs arriving in VMs are considered as *Size* if the time required for hosting by the VM is also *Size*. Consider category of the type-m jobs arriving at the VMs in t-slot is $|A_m(t)|$, which represents the quantity of the type-m jobs arrived before the time slot t. Therefore, the stochastic method ($\gamma_m(t)$), which is independently and identically distributed across the considered time slot t can be represented as

$$\gamma_m(t) = \sum_{j \in A_m} Size_j \qquad (2)$$

In the above Eq.(2), $Size_j$ can be signified as cumulative quantity of time-slots demanded by the arrived jobs in VMs. Therefore, expected value of the stochastic process can be said to be equivalent to jobs arrival rate of m-dimension ($\sigma[\gamma_m(t)] = \lambda_m$) . Similarly, the probability of random events occurring at VM side should be more than corresponding stochastic methods w [i.e. Prob($\gamma_m(t) = 0$) > ϵ w]. This is one of the simplest processing of designing the stochastic method in our study. Consider quantity of the type-m jobs are represented by $\theta_m(t)$ served by the cloud environment, which tends to minimize by the advancement of end of the t-time slot. The workload owing to the jobs can be represented as the summation of the remnant jobs to be executed. Denoting $\delta_m(t)$ as the workload of jobs in the networking environment before arrival of any jobs, the possible size of $\delta_m(t)$ can be represented as

$$\delta_m(t+1) = \delta_m(t) + \gamma_m(t) - \delta_M(t) \qquad (3)$$

With support of stochastic model formulation, it can be said that the cloud system is stable if

$$\limsup_{t \to \infty} \sigma[\sum_m \delta_m(t)] < \infty \qquad (4)$$

i.e., the anticipated cumulative workload in steady-state is highly bounded. The entire processing is done considering various resource charecteristics e.g. operating system, resources, processing elements, architecture, VMs, allocation policy, and time slot. Moreover, a vector of incoming loads λ

can be indexed as supportable if there exist a resource allocation mechanism under which the cloud is stable. The group of supportable arrival rates is signified and a new set is formulated as follows,

$$Group = \{\lambda : \lambda = \sum_{i=1}^{L} \lambda^{(i)}, \lambda^{(i)} \in Conv(\eta_i)\} \quad (5)$$

In the above Eq.(5), η_i is operational configuration of VM in i^{th} server and Conv represents a convex hull in the stochastic modelling process. The proposed system attempts to design a distributed solution by ensuring that once queues are formulated on the server side, it should be effectively processed as sooner as possible by routing the jobs to specific VMs. However, such issues of load balancing cannot be directly said to be eligible for scheduling or routing framework, but in our proposed system, it is showed that it is possible to use simple stochastic modelling for scheduling of the servers are well balanced by the load using effective and shortest queuing process as job routing role. In the proposed model, the study assumes that every server should maintain *M* discreet queues for specific and unique jobs arriving. The scheduling decision is formulated based on this queuing processing information with respect to size of the queue vector. For the purpose of mathematical formulation, we consider δ as the vector of jobs (or queue) where δ_{mi} represents length of queue of m-specific job at i^{th} server. The algorithm is discussed below:

Algorithm 1 Load Balancing Algorithm

Input: m, t, δ_{mi}

Output: Load balancing with updated queue in traffic

Start:

1. Define *m*-type of jobs in *t*-time slot.
2. Find the size of queue for *m*-type of jobs
3. Route jobs to shortest queue size using

$$\text{server } i^*_m(t) = \text{argmin } \delta_{mi}(t)$$

4. Formulate a condition of job arrival to Q_{mi} as

$$\gamma_{mi}(t) = \begin{cases} \gamma_m(t) & i = i^*_m(t) \\ 0 & otherwise \end{cases}$$

5. Perform grouping for *t*-time slots.
6. Apply stochastic model before job arrival in VM using eq. (3)
7. Formulate a condition for feasible configuration of VM as

$$\bar{\eta}^{(i)}(t) \in \arg\max \sum_m \delta_{mi}(t)\eta_m$$

8. Update the feasible configuration for VM to perform load balancing using

$$\overline{\eta_{up}}^{(i)}(t) \in \arg\max \sum_m Q_{mi}(t)\eta_m$$

9. Update the length of queue using

$$\delta_{mi}(t+1) = \delta_{mi}(t) + \gamma_{mi}(t) - \eta^{(i)}_m(t)$$

End

In the above algorithm formulation, $\eta^{(i)}(t^-)$ is the configuration of jobs in i^{th} server that are still in processing at the end of the previous time slot. As there are availability of multiple jobs, which are selected for processing from the queue upto maximum level of $\eta^{(i)}_m(t)$ jobs of type *m*. The formulations also consider the constraint that a novel job of *m*-type is processed if that job can be executed and dispatched by the VM at the end of defined time slot. Therefore, only if $Size_j \leq T - (t - |T|)$: Let $\overline{\eta_{up}}^{i}_{m}(t)$ denote the actual number of type *m* jobs selected at i^{th} server and mathematically represent as:

$$\eta^{(i)}_{up}(t) = \eta^{(i)}(t^-) + \overline{\eta_{up}}^{(i)}(t) \quad (6)$$

Hence, step-9 of the algorithm updates the queue length using Eq.(6).

5. RESULT ANALYSIS

The proposed algorithm which is an outcome of mathematical modelling is evaluated in open source platform Java for better supportability of networking APIs. The simulation is performed and the outcome of the proposed algorithm is compared with existing scheduling algorithm. The simulation study was carried out considering 100 similar types of servers where each server is assumed to have specific hardware profiles within 20 iterations. The analysis also considers that jobs arriving at VM are of specific types. For the purpose of performance comparison, we theoretically compared with the study of Pop et al.[19]. The authors have proposed an algorithm for dynamic load balancing in cloud using Monte Carlo simulation considering with and without check points. The proposed system is simulated for 20 iterations, where numbers of loads were on constantly increasing level. So, the analysis is carried out to see how the proposed algorithm performs.

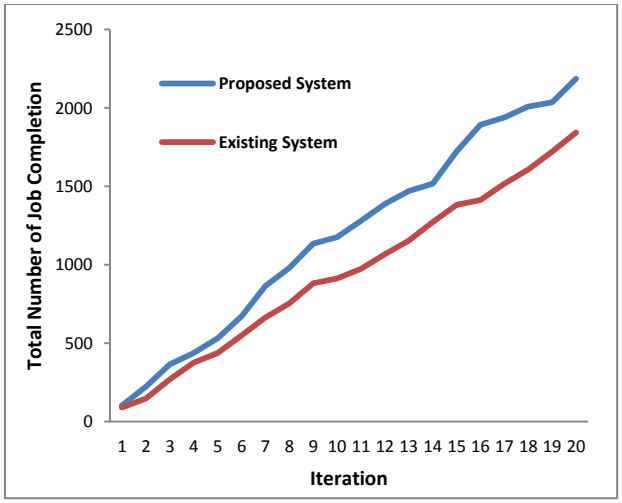

Fig. 2 Total Number of Job Completion

The proposed study considers the variable size of the jobs being seeked by dynamic number of users over cloud. The proposed system performs the load balancing by an excellent switching mechanism among the processing elements, for which purpose the queuing time is highly reduced even with increasing number of traffic. The outcome exhibited in Fig.2 shows that proposed system is capable of undertaking and accomplishing higher number of jobs as compared to existing system. All the values of iterations and total number of job completed are recorded from the simulation study.The simulation was carried out by designing a task that is a

complete sequence of instructions. Task execution starts when a task is selected by task dispatcher and one of the system's processors starts to run task's instructions. Tasks are classified according to their deadline, priority, arrival characteristic, and computation cycles requests.

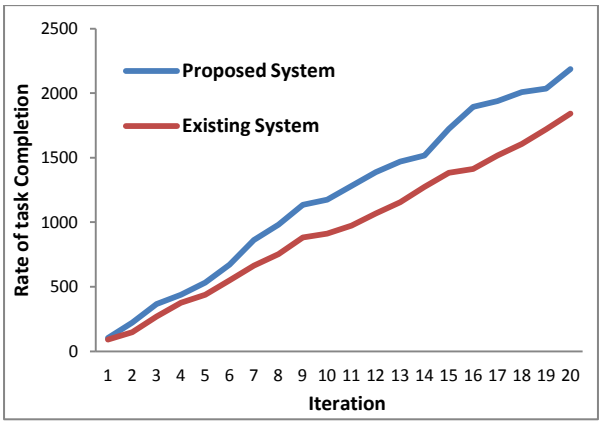

Fig.3 Rate of task Completion

Interestingly, processing speed of the proposed system is evaluated using rate of task completion. For this purpose, the simulation was carried out for maximum number of time-slots, where the study monitors the differences between dynamic routing selection and proposed routing selection by comparing their average rate of task completion at various traffic intensities. The outcome of the simulation study depicts that delay performance of both the algorithms is very discreet. The outcomes exhibited in Fig.2 and Fig.3 shows the serious attempt to prove specific dynamic, adaptive, and decentralized load balancing algorithms for cloud environment that are shown to be applicable in balancing loads depending on the efficiency of the underlying Virtual Machine. The application is basically a framework design where the user chooses to simulate for the multiprocessor application. The simulation results in display of the interface design considering all the parameters like waiting jobs, jobs in executions compared with diversified time span relation. The user selects the simulation results in multiple modes. The outcome exhibited in Fig.2 shows that proposed system processes the task very faster leading to smooth gradient in curve as compared to existing load balancing system. While Fig.3 exhibits that rate of task completion is pretty faster as compared to existing system.

Hence, proposed system can be also considered to have faster processing time. The initial phase of the study checks each processor performance along with the time span. It is basically design for much functional information for any uni-processor system in the network. Not only this, the simulation will ultimately furnish the all the fundamental information about this e.g. cloud IDs, user ID, status, start and finish time of total multiprocessor system. The system will also furnish the information pertaining to existing work status and also assist for overall simulated results in a very fast track. Since in a cloud environment, the network topology is varying, our model captures this constraint as well by considering an arbitrary topology. The data transfer rate is not fixed and varies from link to link. The processors that are directly connected to a processor constitute its buddy set. We also assume that each processing element has knowledge about its virtual machines and the communication latency between them, and load balancing is carried out within virtual machines only.

6. CONCLUSION

The main aim of the project work is to design a framework application where a scheduling approach to arrange real-time periodic and non-periodic tasks in multiprocessor systems in order to balances task loads of the processors successfully while consider starvation prevention and fairness which cause higher priority tasks have higher running probability. We considered a stochastic model for load balancing and scheduling in cloud computing clusters. A primary contribution is the development of frame-based non-preemptive VM configuration policies. These policies can be made nearly throughput-optimal by choosing sufficiently long frame durations, whereas the widely used best fit policy was shown to be not throughput optimal. Simulations indicate that long frame durations are not only good from a throughput perspective but also seem to provide good delay performance.

7. REFERENCES

[1] Buyya, R., Broberg, J., Goscinski, A. M.2010.Cloud Computing: Principles and Paradigms. John Wiley & Sons, Computers,pp.664.

[2] Membrey,P., Plugge,E., Hows, D.2012.Practical Load Balancing: Ride the Performance Tiger. Apress, pp.272.

[3] Begum, S., Prashanth.2013.Review of Load Balancing in Cloud Computing. International Journal of Computer Science Issues. Vol. 10, Issue 1, No 2

[4] Xu,G., Pang, J., Fu, X.2013.Load Balancing Model Based on Cloud Partitioning for the Public Cloud. IEEE Transactions on Cloud Computing

[5] Nishant,K., Sharma, P., Krishna, V., Gupta, C., Singh, K.P., Rastogi, R.2012.Load Balancing of Nodes in Cloud Using Ant Colony Optimization.14th International Conference on Modelling and Simulation

[6] Thazhathethil,D.,Katre,N.,Deshmukh,J.M.,Kshirsagar,M, Nadaph, A.2014. International Journal of Innovative Research in Computer and Communication Engineering. Vol. 2, Issue.1

[7] Sethi,S., Sahu, A.,Jena, S.K.2012. Efficient load Balancing in Cloud Computing using Fuzzy Logic. IOSR Journal of Engineering (IOSRJEN) ISSN: 2250-3021 Vol.2, Issue. 7, pp. 65-7

[8] Bhargava,S., Goyal,S. 2013. Dynamic Load Balancing in Cloud Using Live Migration of Virtual Machine. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Vol. 2 Issue. 8

[9] Tiwari, M.N., Gautam, K.K.,Katare, R.K.2014. Analysis of Public Cloud Load Balancing using Partitioning Method and Game Theory.International Journal of Advanced Research in Computer Science and Software Engineering, Vol.4, Issue 2

[10] Cheng,X.,.Liu,J.2011.Load-Balanced Migration of Social Media to Content Clouds. NOSSDAV'11,Vancouver, British Columbia, Canada.Copyright ACM

[11] Priyadarshinee,P., Jain,P.2012. Load Balancing and Parallelism in Cloud Computing. International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Vol.1, Issue.5

[12] Singhal,P., Shah, S. Load Balancing Algorithm over a Distributed Cloud Network.

[13] Mashaly,M., Kuhn, P.J.2012. Load balancing in cloud-based content delivery networks using adaptive server activation/deactivation.Engineering and Technology (ICET) International Conference, pp.1- 6

[14] Priya,S.M.,Subramani, B.2013. A new approach for load balancing in cloud computing. International Journal Of Engineering And Computer Science ISSN:2319-7242 Vol. 2, Issue. 5, pp. 1636-1640

[15] Galloway, J.M., Smith, K.L., Vrbsky, S.S.2011. Power Aware Load Balancing for Cloud Computing., Proceeding of the World Congress on Engineering and Computer Science, Vol Wcecs

[16] D P L.2013.Load Balancing Algorithms in Cloud Computing, International Journal of Advanced Computer and Mathematical Sciences, Vo.l4, Issue.3, pp.229-233

[17] Mahapatra,S.,Yuan,X.2010.Load balancing mechanisms in data center networks. In the 7th Int. Conf. & Expo on Emerging Technologies for a Smarter World (CEWIT)

[18] Adnan,M.A., Sugihara, R., Gupta,R.K.2012. Energy Efficient Geographical Load Balancing via Dynamic Deferral of Workload.IEEE Fifth International Conference on Cloud Computing

[19] Pop,S.C., Glatard,T., Silva, R. F.2013. Monte Carlo simulation on heterogeneous distributed systems: A computing framework with parallel merging and checkpointing strategies. Future Generation Computer Systems, Elsevier, Vol.29, pp.728-738