Performance Analysis of Load Balancing Algorithm in Cloud Computing

Swikruti Dash[1], Amrutanshu Panigrahi[2], Nihar Ranjan Sabat[3]
[1,2,3] *Department of CSE, CIME, Bhubaneswar, Odisha, India*

***Abstract*- "Distributed computing" is a term, which includes virtualization, circulated registering, systems administration, programming, and web administrations. A cloud comprises of a few components, for example, customers, datacenter and dispersed servers. It incorporates adaptation to internal failure, high accessibility, versatility, adaptability, the diminished overhead for clients, the decreased expense of possession, on-request benefits and so forth. Key to these issues lies the foundation of a viable load adjusting calculation. The heap can be CPU load, memory limit, deferral or system load. Burden adjusting is the way toward circulating the heap among different hubs of a conveyed framework to improve both asset use and employment reaction time while likewise maintaining a strategic distance from a circumstance where a portion of the hubs are vigorously stacked while different hubs are inactive or doing next to no work. Burden adjusting guarantees that all the processor in the framework or each hub in the system does around the equivalent measure of work at any moment of time. This procedure can be sender started, collector started or symmetric sort (the blend of sender-started and recipient started types).The objective is to implement various dynamic load balancing algorithm such as Round Robin (RR), Throttled, Equally Spread Current Execution (ESCE) and Shortest Job First (SJF) algorithms with some sample data center loads.**

***Index terms*- SJF, ESCE, RR, Throttled**

I.INTRODUCTION

Cloud computing is an on interest administration in which shared assets, data, programming, and different gadgets are given by the customers prerequisite at the explicit time. Its a term which is commonly utilized if there should arise an occurrence of Web. The entire Web can be seen as a cloud. Capital and operational expenses can be cut utilizing distributed computing. Load balancing in cloud computing frameworks is extremely a test now. Continuously a conveyed arrangement is required. Since it isn't in every case basically plausible or cost effective to keep up at least one inactive administration similarly as to satisfy the required requests. Jobs cannot be assigned to fitting servers and customers independently for proficient burden adjusting as the cloud is a mind-boggling structure and segments are available all through a widespread zone. Here some vulnerability is appended while occupations are doled out. Cloud computing is progressively being received by substantial organizations, just as little and medium estimated organizations, for "on-request" and "utility figuring", which holds colossal guarantee for the eventual fate of administration registering [1].

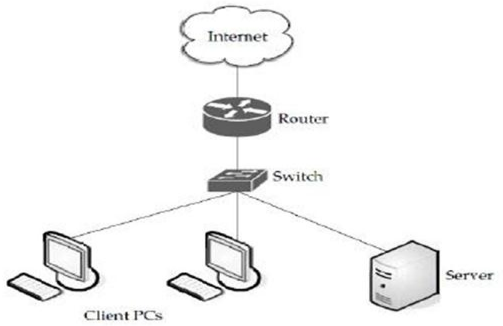


Fig 1: A cloud is used in network diagrams to depict the Internet [1]

Virtualization is a key empowering innovation for distributed computing conditions, which makes it conceivable to run various working frameworks and different applications on the equivalent equipment in the meantime, to give benefits by a virtual unit [2]. Through virtualization innovation, not exclusively can by and large equipment usage improve and lower costs for calamity recuperation, yet it can likewise accomplish programmed checking for all hosts. Be that as it may, it is extremely hard to relegate an expansive number of undertakings to dynamic assets for dispersed registering. There are an assortment of

variables that may prompt a few hubs in the over-burden state while others stay in the under load state, for example, uneven designation of assets, the client needs changing after some time, recently joining hubs, and a high probability of disappointment in the over-burden hubs, and so forth [3– 5]. Load adjusting is the best method to take care of the above issue in a distributed computing foundation, which guarantees that administrations are conveyed straightforwardly paying little heed to the physical usage and area inside the "cloud". In ongoing decades, extraordinary advancement has been accomplished for load balancing, and a standout amongst the most encouraging branches is swarmed insight calculations, for example, insect settlement streamlining [6– 8], fake honey bee state [9,10], molecule swarm improvement [11,12], and so on. Subterranean insect settlement streamlining, proposed by Marco Dorigo in 1992 [13], is a class of stochastic advancement algorithms in view of the actions of an ant colony. Cloud computing is a huge idea. A considerable lot of the calculations for load balancing in cloud computing have been proposed. A portion of those calculations has been outlined in this work. The entire Web can be considered as a cloud of numerous connections less an association arranged administrations. So the distinguishable load scheduling hypothesis for Wireless systems depicted in [9] can likewise be connected for mists. The execution of different calculations have been considered and compared.

II. LOAD BALANCING

It is a procedure of reassigning the all-out burden to the individual hubs of the system framework to make asset usage successful and to improve the reaction time of the activity, all the while expelling a condition in which a portion of the hubs are over stacked while some others are under stacked. A heap adjusting calculation which is dynamic in nature does not think about the past state or conduct of the framework, that is, it relies upon the present conduct of the framework. The imperative interesting points while growing such calculation are : estimation of burden, examination of burden, security of various framework, execution of framework, association between the hubs, idea of work to be exchanged, choosing of hubs furthermore, numerous different ones [4] . This heap considered can be regarding CPU load, measure of memory utilized, postponement or System load. A site or a web-application can be gotten to by a lot of clients anytime of time. It ends up troublesome for a web application to deal with all these client asks for at one time. It might even outcome in framework breakdowns. For a site proprietor, whose whole work is reliant on his entryway, the sinking feeling of site being down or not available additionally brings lost potential clients. Here, the heap balancer assumes a vital job. Cloud Burden adjusting is the way toward disseminating remaining tasks at hand and figuring assets crosswise over at least one servers. This sort of dissemination guarantees most extreme throughput in least reaction time. The outstanding burden is isolated among at least two servers, hard drives, organize interfaces or other figuring assets, empowering better asset use and framework reaction time. Along these lines, for a high traffic site, compelling utilization of cloud load adjusting can guarantee business coherence. The normal goals of utilizing load balancers are:

- To keep up framework solidness.
- To improve framework execution.
- To ensure against framework disappointments.

Cloud suppliers like Amazon Web Administrations (AWS), Microsoft Purplish blue and Google offer cloud load adjusting to encourage simple dispersion of outstanding tasks at hand. For ex: AWS offers Versatile Burden adjusting (ELB) innovation to appropriate traffic among EC2 examples. The greater part of the AWS controlled applications have ELBs introduced as key compositional segment.

How does Load Balancer work?

Here, load refers to not only the site activity but too incorporates CPU stack, organize stack and memory capacity of each server. A stack adjusting strategy makes beyond any doubt that each framework within the organize has same sum of work at any moment of time. This implies not one or the other any of them is unreasonably over-loaded, nor under-utilized. The stack balancer disperses information depending upon how active each server or hub is. Within the nonattendance of a stack balancer, the client must wait whereas his prepare gets prepared, which may be as well tiring and demotivating for him. Various

data like occupations holding up in line, CPU handling rate, job entry rate etc. are traded between the processors amid the stack adjusting handle. Disappointment within the right application of stack balancers can lead to genuine results, information getting misplaced being one of them.

Goals of Load balancing

As given in [4], the goals of load balancing are:

- To make strides the execution substantially
- To have a reinforcement arrange in case the framework falls flat indeed partially
- To keep up the framework stability
- To suit future alteration within the system

Different kinds of Load balancing

Depending on who initiated the process, load balancing algorithms can be of three categories as given in [4]:

- Sender Initiated: If the load balancing algorithm is initialized by the sender.
- Receiver Initiated: If the load balancing algorithm is initiated by the receiver.
- Symmetric: It is the combination of both sender initiated and receiver initiated.

Depending on the current state of the system, load balancing algorithms can be divided into 2 categories as given in [4]:

- Static: It doesn't depend on the current state of the system. Prior knowledge of the system is needed.
- Dynamic: Decisions on load balancing are based on current state of the system. No prior knowledge is needed. So it is better than static approach.

III. LITERATURE SURVEY

Load balancing assumes a fundamental job in giving Quality of Service (QoS) ensures in cloud registering, and it has been producing generous enthusiasm for the exploration network. There are a lot of methodologies that have adapted to the heap adjusting issue in distributed computing. We talk about the past related work of burden adjusting by partitioning them into two classes as per the basic calculation.

The top notch comprises of differing customary methodologies without using any sort of swarm insight calculations. Many load balancing approaches were proposed as of late and each centered around various angles of calculations and strategies, e.g., utilizing a focal burden adjusting approach for virtual machines [17], the planning methodology on burden adjusting of virtual machine (VM) assets dependent on hereditary calculations [18], a mapping arrangement dependent on multi-asset load adjusting for virtual machines [19], versatile dispersed calculation for virtual machines [20], weighted least-association methodology [21], and two-stage booking calculations [22]. Also, a few techniques for burden adjusting were displayed for various cloud applications, for instance, an administration based model for extensive scale stockpiling [23], information focus the executives design [24], and a heterogeneous cloud [25]. Despite the fact that these commitments have gained incredible ground in burden adjusting under distributed computing, it has a high level of centralization and isn't anything but difficult to expand. Besides, these displayed methodologies did not completely mirror the qualities of asset hubs and are increasingly reasonable to the static circumstance of distributed computing.

The inferior contains approaches use swarm knowledge calculations, for example, subterranean insect state enhancement [6– 8], fake honey bee state [9, 10], and molecule swarm streamlining [11, 12], which is better for the dynamic circumstance of distributed computing. With self-sorted out conduct, these social bugs can be imitated all things considered, or with important changes, to take care of undifferentiated from issues in distributed computing. In [6], Nishant, K. et al. proposed a calculation for burden conveyance of an outstanding burden with an adjusted methodology of ACO from the point of view of cloud or matrix organize frameworks. In this methodology, the ants just refreshed a solitary outcome set constantly in the procedure, as opposed to refreshing their own outcome set. In [7], a heap adjusting system was proposed in light of subterranean insect state and complex system hypothesis in an open distributed computing league. This is the first time that ACO and complex systems were brought together into burden adjusting in distributed computing what's more, acquired great execution. In [8], Mishra, R. et al. gave an answer for

burden adjusting in the cloud by ACO, to expand or limit diverse execution parameters, for example, CPU burden and memory limit. Notwithstanding, few elements were considered as pheromones to discover target hubs when utilizing ACO in the over three methodologies. In [9,10], Sesum-Cavic, V. et al. exhibited a novel methodology for burden adjusting dependent on counterfeit honey bee settlement. A conventional design, named SILBA (self-activity load adjusting operators), was characterized to bolster the trading of various calculations through stopping procedures. Six calculations were connected in this design and the outcomes exhibited promising advantages in the Amazon EC2 cloud. In spite of the fact that SILBA is a decent example, it didn't consider the lower requests for hub servers in cloud processing conditions and the dynamic client needs. Molecule swarm advancement (PSO) was likewise embraced for burden adjusting in distributed computing, for example, [11,12]. In [11], it proposed another assignment planning model to evade the genuine burden awkwardness issue, with progress of the standard PSO by presenting a basic change component and a self-adjusting latency weight strategy. To take care of the improvement issue of discrete space in cloud figuring, Feng, X. et al. built a proper asset assignment show dependent on a discrete molecule swarm enhancement calculation [12]. The investigation results demonstrated that the talked about PSO strategies can improve the use in burden adjusting of assets yet they may take a lot of time with a gigantic number of errands. Different applications and research on burden offsetting with swarm insight calculations can be found in [26– 33]. Since these calculations were initially intended for disseminated load adjusting instead of distributed computing, much work should be done in the event that we need to apply these calculations into distributed computing.

IV. LOAD BALANCING ALGORITHMS

A. Equally Spread Current Execution

1. In this algorithm, the Heap Balancer keeps up a list table of VM's and the quantity of solicitations presently designated to the VM's. At begin all VM's have 0 allotments.
2. At the point when a demand to designate another VM from the Data Center Controller arrives, it parses the list table also, recognizes the least stacked VM. In the event that there are more than one, the principal distinguished is chosen.
3. The Heap Balancer restores the VM ID to the DataCenterController.
4. The DataCenterController sends the demand to the VM distinguished by that ID.
5. The DataCenterController tells the Heap Balancer of the new assignment.
6. The Heap Balancer refreshes the assignment table augmenting the allotment mean that VM.
7. At the point when the VM wraps up the demand and DataCenetrController gets the reaction cloudlet, it tells the Heap Balancer of the VM de-distribution.
8. The Heap Balancer refreshes the distribution table by decrementing the assignment mean the VM one by one.

In Equally Spread Current Execution Algorithm, a correspondence exist between the heap Balancer and the Data Center Controller for refreshing the list table prompting an overhead. Further, this overhead makes delay in giving reaction the arrived solicitations.

B. Throttled Load Balancing:

1. In this algorithm, the Heap Balancer keeps up an record table of VM's just as their states (Accessible/Occupied).
2. At the point when a demand to assign another VM from the
3. DataCenterController arrives, it parses the record table
4. from best until the most readily accessible VM is found.
5. On the off chance that VM is discovered, the Heap Balancer restores the VM ID to the Data Center Controller.
6. The DataCenterController send the demand to the VM recognized by that ID.
7. The DataCenterController informs the Heap Balancer of the new allotment.
8. The Heap Balancer refreshes the allotment table by augmenting as needs be.
9. At the point when the VM wraps up the demand and the Data Center Controller gets the reaction cloudlet, it advises the Heap Balancer of the VM de-assignment.

10. The Heap Balancer de-assign the equivalent VM whose Id is as of now imparted.

The objective of this algorithm is to find the response time of every virtual machine as VMs are of different capacity corresponding to the processing efficiency.
RT= Ft-At+Td
Where RT= Response Time, Ft=Finish Time, At= Arrival Time, Td= Transmission Delay.
Td= Tl+Tt
Where Tl= Network Latency Time, Tt= Time taken to transfer the data of single request.
Tt=D/B
B=BT/Nr
Where D= Time to transfer the data of Single Request, B= Available Bandwidth per user, BT= Total Bandwidth, Nr=Number of simultaneous user requests.

C. Round Robin (RR):
It is the least complex calculation that utilizes the idea of time quantum or cuts. Here, time is separated into numerous cuts also, every hub is given a specific time quantum and inside this time quantum the hub will play out its activities. In this calculation, the Data Center Controller dole out the demand to a rundown of VM's on a pivoting premise. The first demand is allotted to a VM picked haphazardly from the gathering and afterward Data Center Controller doles out the resulting demands in a roundabout request. Once the virtual machine is appointed the demand, the VM is moved to the end of the rundown. In this RRLB, there is a superior distribution idea known as Weighted Round Robin Designation in which one can dole out a weight to each VM so that in the event that one VM is prepared to do taking care of twice as much burden as the other, the incredible server gets a weight of 2. In such cases, Data Center Controller will allocate the two solicitations to the ground-breaking VM for each demand allotted to a more fragile one. Round Robin Calculation chooses the heap on arbitrary premise, also, in this manner prompts a circumstance where a few hubs are intensely stacked and some are daintily stacked. However, the calculation is exceptionally straightforward however there is an extra burden on the scheduler to choose the extent of quantum [5]. It has longer normal holding up time, higher setting switches, higher turnaround time and low throughput. For instance, if the schedule opening is 100 milliseconds, and job1 takes an absolute time of 250 ms to finish, the round-robin scheduler will suspend the activity after 100 ms and give other occupations their time on the CPU. When alternate occupations have had their equivalent offer (100 ms each), job1 will get another designation of CPU time and the cycle will rehash. This process proceeds until the activity completes and needs no more time on the CPU.

Pseudo Code:
1. CPU scheduler picks the procedure from the roundabout/prepared line, set a clock to intrude on it after 1 time cut/ quantum and dispatches it .
2. If process has burst time under 1 time slice/quantum
   - Process will leave the CPU after the finish
   - CPU will continue with the following procedure in the prepared line/round line, else If process has burst time longer than 1 time cut/quantum
   - Timer will be ceased. It cause interference to the OS.
   - Executed procedure is then set at the tail of the roundabout/ready queue by applying the setting switch.

D. Shortest Job First
Shortest Job First (SJF) planning is a need and Non-Preemptive booking. Non-Preemptive methods, when the allocated time a processor then the processor can't be taken the other, until the procedure is finished in the execution. Fundamentally Most limited Occupation Initially is a dynamic burden adjusting calculation which handles the procedure with need premise. It decides the need by checking the span of the procedure. This calculation appropriates the heap haphazardly by first checking the extent of the procedure and after that exchanging the heap to a Virtual Machine, which is gently stacked. All things considered that procedure measure is least, this procedure will get first need to execute whether we guess most reduced estimated process executes in least time. The heap balancer spreads the heap on to various hubs known as spread range technique. The instrument of most brief Occupation First Calculation is, to plan the procedure with the most brief time to

fruition first, hence giving high proficiency and low turnaround time. Regarding time spent in the present program (work) started to enter in to the framework until the procedure is done the framework, need a brief timeframe. Shortest Job First (SJF) planning calculation can be said to be ideal with a normal holding up time is negligible, which improves the framework execution.

Procedure:

1. Firstly start process, vmloadbalancer keep up the procedure by need checking the extent of the procedure and disseminate the heap to the virtual machine which is daintily stacked.
2. The Vmloadbalancer, first assign exhibit estimate for example A [10].
3. Take number of components to be embedded.
4. Vmloadbalancer select procedure which load has briefest blasted time among all heaps will execute first.
5. On the off chance that in the process any heap have same blasted time length, at that point FCFS (First started things out Served) booking calculation utilized.
6. Make normal holding up time length of next procedure.
7. Begin with first procedure, choice as above as most brief burden start things out which has insignificant normal time and different procedures are to be in line.
8. Calculate Burst all out number of time.
9. Show the Related qualities.

V. SIMULATION AND RESULT

The current load balancing systems in mists, consider different parameters like execution, reaction time, versatility, throughput, asset use, adaptation to non-critical failure, relocation time and related overhead. But, for an energy efficient load balancing, metrics like energy consumption and carbon emission should also be considered.

Overhead Related- It decides the measure of overhead included while executing a heap adjusting calculation. It is made out of overhead because of development of errands, inter processor and between procedure correspondence. This ought to be limited with the goal that a heap adjusting method can work productively.

Throughput-It is utilized to figure the no. of errands whose execution has been finished. It ought to be high to improve the execution of the framework.

Performance-It is utilized to check the proficiency of the framework. It must be improved at a sensible expense for example decrease reaction time while keeping satisfactory postponements. Asset Usage It is utilized to check the use of assets. It ought to be enhanced for a proficient burden adjusting.

Scalability- It is the capacity of a calculation to perform load offsetting for a framework with any limited number of hubs. This measurement ought to be improved.

Response Time-It is the measure of time taken to react by a specific burden adjusting calculation in a circulated framework. This parameter ought to be limited.

Fault Tolerance- It is the capacity of a calculation to perform uniform burden adjusting disregarding 2discretionary hub or connection disappointment. The heap adjusting ought to be a decent blame tolerant procedure.

Migration Time-It is an ideal opportunity to move occupations or assets starting with one hub then onto the next. It ought to be limited so as to upgrade the execution of the framework.

Energy Consumption- It decides the vitality utilization of the considerable number of assets in the framework. Burden adjusting helps in abstaining from overheating by adjusting the remaining burden over every one of the hubs of a cloud, subsequently decreasing vitality utilization.

Table-1 Region Configuration

Cloud Analyst Region id	Users
0	4.4M
1	1.1M
2	2.6M
3	1.3M
4	0.5M
5	0.8M

Table-2 Region Configuration

UB	Region	Online Users during peak Hours	Online Users during off-peak hour
1	0	400000	40000
2	1	100000	10000
3	2	250000	25000
4	3	120000	12000
5	4	50000	5000
6	5	70000	7000

Table-3 DC Configuration

Parameter	Value Used
VM Image Size	10000
VM Memory	1024Mb
VM Band	1000
DC-Arch	X86
DC-OS	Linux
DC-Machine	20
DC-Memory/machine	2048Mb
DC-Strorage	100000Mb
DC-Band	10000
DC-Processors/Machine	4
DC-Speed	100MIPS
DC-Policy	Time Shared/Space Shared
DC Grouping UB based	1000
DC Grouping Request based	100
Instruction Length	250

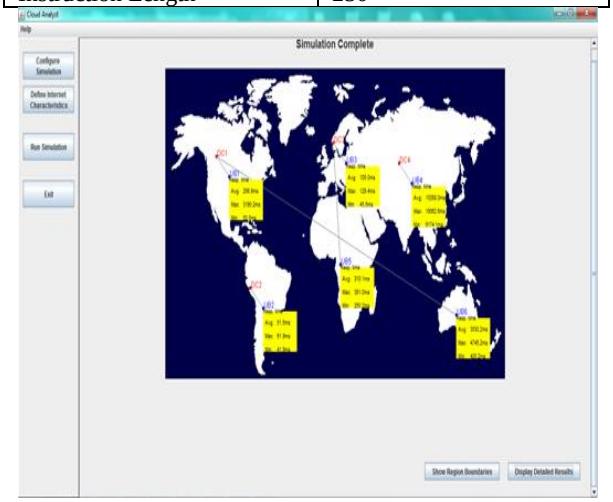


Fig 2 Cloud Analyst GUI using defined configuration

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	299.876	50.509	3,190.248
UB2	51.455	41.896	61.923
UB3	104.985	45.484	129.373
UB4	15,268.041	6,174.11	19,062.622
UB5	310.076	250.201	361.037
UB6	3,530.16	405.214	4,745.163

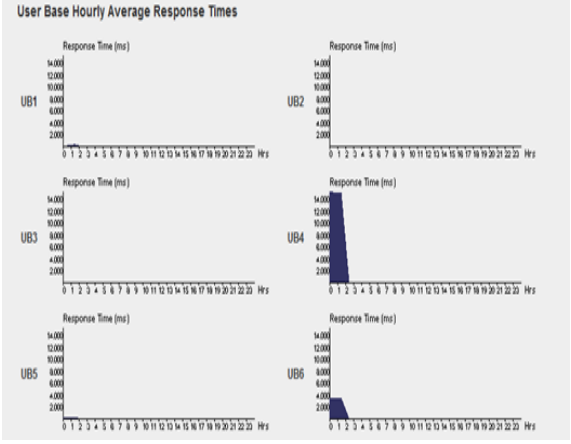


Fig 3: UB Response time using RR Algorithm

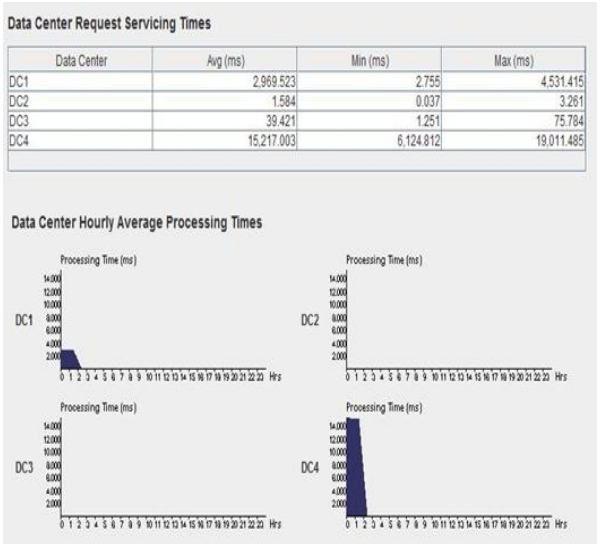


Fig 4: DC Processing time using RR Algorithm

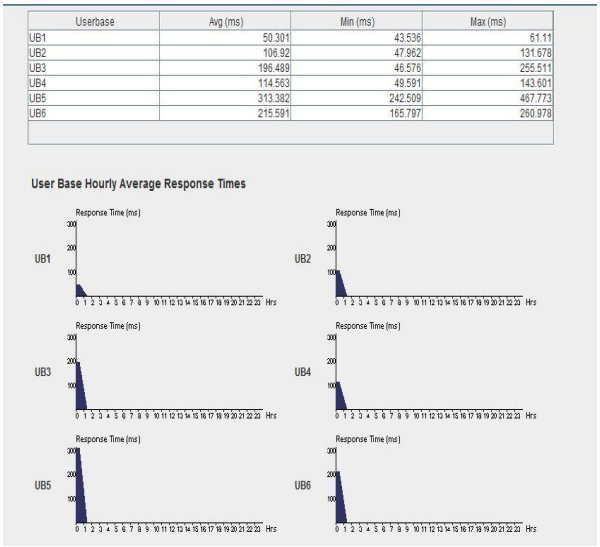


Fig 5: UB Response time using ESCE

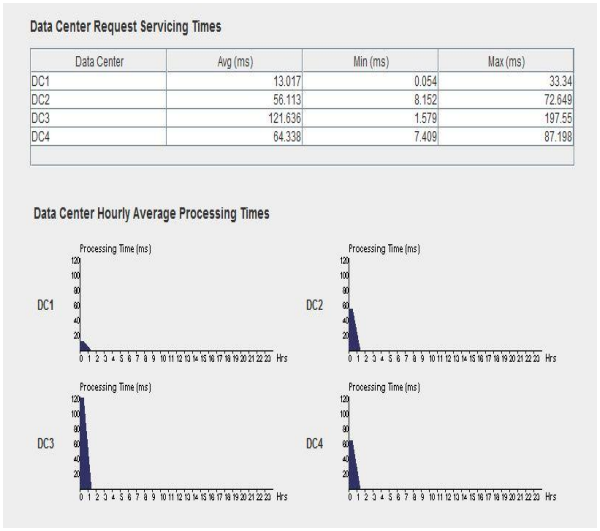
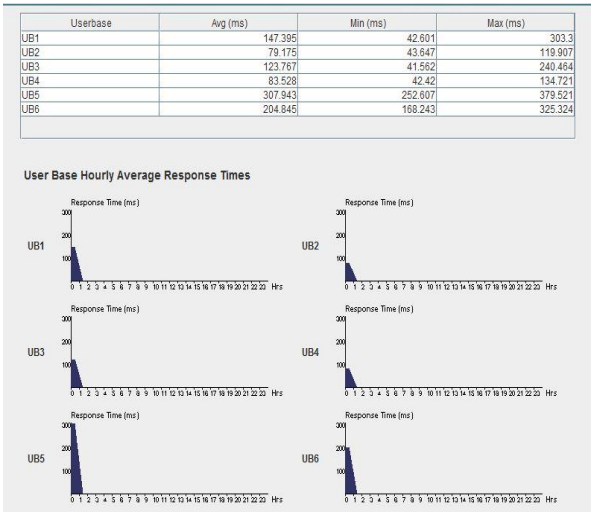
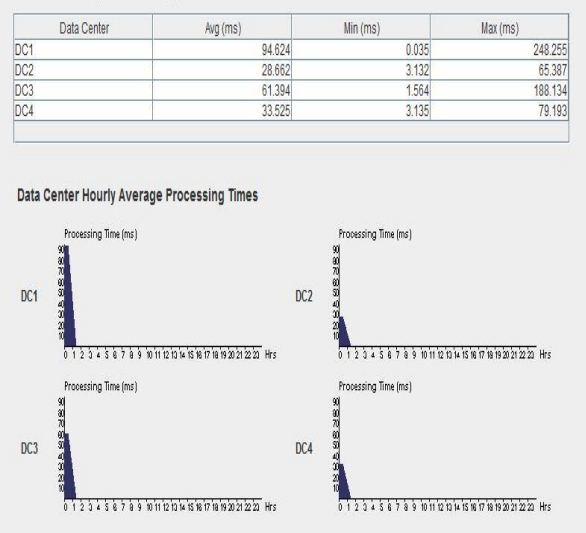


Fig 6: DC Processing time using ESCE



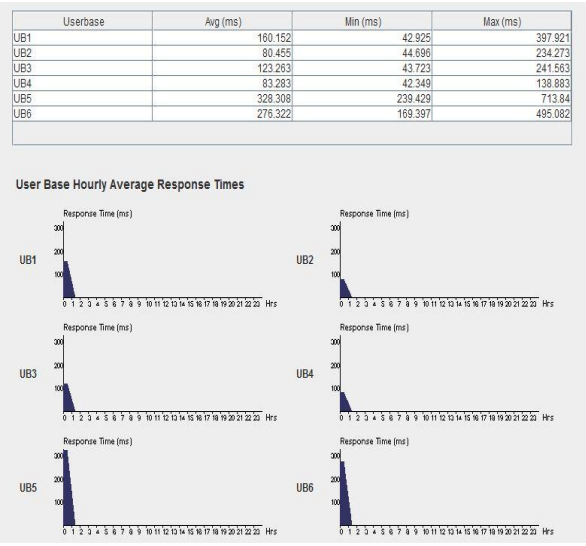
Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	147.395	42.601	303.3
UB2	79.175	43.647	119.907
UB3	123.767	41.562	240.464
UB4	83.528	42.42	134.721
UB5	307.943	252.607	379.521
UB6	204.845	168.243	325.324

Fig 7: UB Response time using Throttled Algorithm



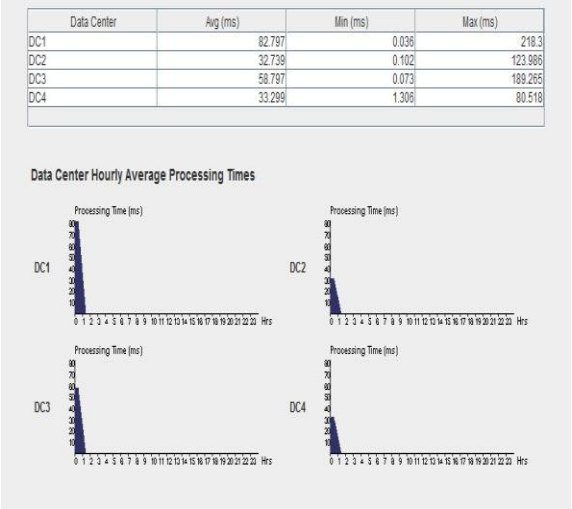
Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	94.624	0.035	248.255
DC2	28.662	3.132	65.387
DC3	61.394	1.564	188.134
DC4	33.525	3.135	79.193

Fig 8: DC Processing time using Throttled Algorithm



Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	160.152	42.925	397.921
UB2	80.455	44.696	234.273
UB3	123.263	43.723	241.563
UB4	83.283	42.349	138.883
UB5	328.308	239.429	713.84
UB6	276.322	169.397	495.082

Fig 9: UB Response time using SJF Algorithm



Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	82.797	0.036	218.3
DC2	32.739	0.102	123.986
DC3	58.797	0.073	189.265
DC4	33.299	1.306	80.518

Fig 10: DC Processing time using SJF Algorithm

VI. CONCLUSION

Cloud computing has broadly been received by industry, in spite of the fact that there are many existing issues like load balancing, Movement of Virtual Machines, Server Unification and so on. Which have not been completely tended to. Actually load balancing is the most focal issue in the framework i.e., to circulate load balancing in a proficient way. It too guarantees that each registering asset is appropriated effectively and reasonably. Existing load balancing methods/calculations that have been examined chiefly center on decreasing overhead, lessening the relocation time and improving execution and so on. The response time is a test of each specialist to build up the process that can increment the throughput in the cloud based part. The a few techniques need productive scheduling and load balancing asset distribution methods prompting expanded operational expense.

We have studied the concepts of Cloud Computing and Load balancing and studied some existing load balancing algorithms, which can be applied to clouds as well. In addition to that, the closed-form solutions for minimum measurement and reporting time for single level tree networks with different load balancing strategies were also studied. The performance of the strategies such as Throttled, Round Robin, Equal Spread Current Execution and Shortest Job First with respect to the response time and the processing time has been studied. A

comparison is also made between different strategies based on the defined parameter.

This experiment has been done with millions of users in different User Base in different region. Based on the experiment and comparison it has been observed that the Equally Spread Current Execution performs well in the presence of heavy load.

REFERENCE

[1] Velte, A. T., Velte, T. J., Elsenpeter, R. C., & Elsenpeter, R. C. (2010). Cloud computing: a practical approach (p. 44). New York: McGraw-Hill.

[2] Randles, M., Lamb, D., & Taleb-Bendiab, A. (2010, April). A comparative study into distributed load balancing algorithms for cloud computing. In 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (pp. 551-556). IEEE.

[3] A Vouk, M. (2008). Cloud computing–issues, research and implementations. Journal of computing and information technology, 16(4), 235-246.

[4] Alakeel, A. M. (2010). A guide to dynamic load balancing in distributed computer systems. International Journal of Computer Science and Information Security, 10(6), 153-160.

[5] http://www.ibm.com/press/us/en/pressrelease/22613.wss

[6] http://www.amazon.com/gp/browse.html?node=20159001

[7] Randles, M., Odat, E., Lamb, D., Abu-Rahmeh, O., & Taleb-Bendiab, A. (2009, December). A comparative experiment in distributed load balancing. In 2009 Second International Conference on Developments in eSystems Engineering (pp. 258-265). IEEE.

[8] Pacheco, P. (1997). Parallel programming with MPI. Morgan Kaufmann.

[9] Moges, M., & Robertazzi, T. G. (2006). Wireless sensor networks: scheduling for measurement and data reporting. IEEE Transactions on Aerospace and Electronic Systems, 42(1), 327-340.

[10] Pallis, G. (2010). Cloud computing: the new frontier of internet computing. IEEE internet computing, 14(5), 70-73.

[11] Armbrust M., Fox A., GriffithR., Joseph A. D., Katz R., Konwinski A., Lee G., Patterson D., Rabkin A., Stocia I. and Zaharia M. (2009) Above the Clouds: A Berkeley View of Cloud Computing, EECS Department, University of California, 1-23.

[12] Bhadani, A., & Chaudhary, S. (2010, January). Performance evaluation of web servers using central load balancing policy over virtual machines on cloud. In Proceedings of the Third Annual ACM Bangalore Conference (p. 16). ACM.

[13] Rimal, B. P., Choi, E., & Lumb, I. (2009, August). A taxonomy and survey of cloud computing systems. In 2009 Fifth International Joint Conference on INC, IMS and IDC (pp. 44-51).

[14] Lee, Y.C.; Zomaya, A.Y.(2010), Energy efficient utilization of resources in cloud computing systems.J. Supercomput. 60, 268–280.

[15] Zhang, Q., Cheng, L., & Boutaba, R. (2010). Cloud computing: state-of-the-art and research challenges. Journal of internet services and applications, 1(1), 7-18.

[16] Mahowald, R. P. (2010). Worldwide Software as a Service 2010-2014 forecast: software will never be the same. IDC Report, 223628.

[17] Mishra, R., & Jaiswal, A. (2012). Ant colony optimization: A solution of load balancing in cloud. International Journal of Web & Semantic Technology, 3(2), 33.

[18] Caron, E., Rodero-Merino, L., Desprez, F., & Muresan, A. (2012). Auto-scaling, load balancing and monitoring in commercial and open-source clouds (Doctoral dissertation, INRIA).

[19] Zhang, Z., & Zhang, X. (2010, May). A load balancing mechanism based on ant colony and complex network theory in open cloud computing federation. In 2010 The 2nd International Conference on Industrial Mechatronics and Automation (Vol. 2, pp. 240-243). IEEE.

[20] Hiranwal, S., & Roy, K. C. (2011). Adaptive round robin scheduling using shortest burst approach based on smart time slice. International Journal of Computer Science and Communication, 2(2), 319-323.

[21] Wickremasinghe, B., Calheiros, R. N., & Buyya, R. (2010, April). Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications. In 2010 24th IEEE international conference on advanced information networking and applications (pp. 446-452). IEEE.

[22] Wickremasinghe, B., Calheiros, R. N., & Buyya, R. (2010, April). Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications. In 2010 24th IEEE international conference on advanced information networking and applications (pp. 446-452). IEEE.

[23] Buyya, R., Ranjan, R., & Calheiros, R. N. (2009, June). Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities. In 2009 international conference on high performance computing & simulation (pp. 1-11). IEEE.

[24] Bo, Z.; Ji, G.; Jieqing, A.(2010), Cloud Loading Balance algorithm. In Proceedings of the 2010 2nd International Conference on Information Science and Engineering (ICISE), Hangzhou, China, 4–6; pp. 5001–5004.

[25] Randles, M., Lamb, D., & Taleb-Bendiab, A. (2010, April). A comparative study into distributed load balancing algorithms for cloud computing. In 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (pp. 551-556). IEEE.

[26] Jaspreet, K.(2012), Comparison of load balancing algorithms in a Cloud. Int. J. Eng. Res., 2,1169–1173.

[27] Shaveta, N.; Raj, G.(2012), Comparative Analysis of Load Balancing Algorithms in Cloud Computing. Int. J. Adv. Res. Comput. Eng., 120–124.

[28] Nakrani, S., Tovey, C (2004) On honey bees and dynamic server allocation in Internet hosting centers. Adapt. Behav. 223–240.

[29] Sivanandam, S.N.; Visalakshi, P. (2009) Dynamic task scheduling with load balancing using parallel orthogonal particle swarm optimisation. Int. J. Bio-Inspired Comput. 276–286.

[30] Visalakshi, P., Sivanandam, S.N. (2009), Dynamic Task Scheduling with Load Balancing using Hybrid Particle Swarm Optimization. Int. J. Open Probl. Compt. Math. 475–488.

[31] Ludwig, S.A., Moallem, A. (2011) Swarm intelligence approaches for grid load balancing. J. Grid Comput,279–301.

[32] Ali, A., Belal, M.A., al-Zoubi, M.B. (2010) Load Balancing of Distributed Systems Based on Multiple Ant Colonies Optimization. Am. J. Appl. Sci., 433–438.

[33] Liu, L., & Feng, G. (2005, August). A novel ant colony based QoS-aware routing algorithm for MANETs. In International Conference on Natural Computation (pp. 457-466). Springer, Berlin, Heidelberg.

[34] Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., & Buyya, R. (2011). CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and experience, 41(1), 23-50.