International Journal of Smart Sensor and Adhoc Network

Volume 3 | Issue 3 Article 3

March 2022

Empirical analysis of dynamic load balancing techniques in cloud computing

Arpita Mahapatra
IMIT, Cuttack, aspresearchsolution@gmail.com

Priyanka Maharana
IMIT, Cuttack, aspresearchsolution@gmail.com

Empirical analysis of dynamic load balancing techniques in cloud computing

Arpita Mohapatra[1], Priyanka Maharana[2]

[1,2] Research Scholar, IMIT, Cuttack, Odisha, India

Abstract: Virtualization, dispersed registration, systems administration, programming, and web administrations are all examples of "distributed computing." Customers, datacenters, and scattered servers are just a few of the components that make up a cloud. It includes things like internal failure adaption, high accessibility, flexibility, adaptability, lower client overhead, lower ownership costs, on-demand advantages, and so on. The basis of a feasible load adjusting computation is key to resolving these challenges. CPU load, memory limit, deferral, and system load are all examples of heaps. Burden adjustment is a method for distributing the load across the many hubs of a conveyance framework in order to optimize asset utilization and employment response time while avoiding a situation where some hubs are heavily loaded while others are idle or performing little work. Burden adjustment ensures that at any one time, each processor in the framework or each hub in the system does about the same amount of work. This method may be initiated by the sender, the collector, or the symmetric sort (the blend of sender-started and recipient started types). With some example data center loads, the goal is to create several dynamic load balancing techniques such as Round Robin, Throttled, Equally Spread Current Execution Load, and Shortest Job First algorithms.

Keyword: Cloud Computing, Round Robin, Throttled, Equally Spread Current Execution Load, Shortest Job First

1. Introduction

Cloud computing is a kind of on-demand administration in which clients request shared assets, data, programming, and other devices at a certain time. It's a word that's widely used when there's a problem with the Internet. The Internet as a whole may be seen as a cloud. Distributed computing may help you save money on both capital and operating costs. Load balancing in cloud computing frameworks is now undergoing extensive testing. A conveying arrangement is necessary on a continuous basis. Because it isn't always financially advantageous to maintain at least one inactive administration in the same way as it is to meet the needed demands. Because the cloud is a complex structure with segments accessible over a vast area, jobs cannot be distributed to appropriate servers and customers individually for effective load balancing. While professions are assigned, some vulnerability is added. Cloud computing is increasingly being used by large companies, as well as small and medium-sized businesses, for "on-demand" and "utility calculating," which has enormous promise for the future of administrative registration [1].

Virtualization is a significant enabling innovation for distributed computing environments, allowing many working frameworks and programs to operate on the same equipment at the same time, allowing a virtual unit to provide advantages [2]. Not only can virtualization technology enhance and minimize costs for disaster recovery, but it can also

perform scheduled checks for all hosts. However, relegating a large number of projects to dynamic assets for scattered registration is quite difficult. There are a variety of factors that might cause certain hubs to become overburdened while others remain underloaded, such as unequal asset allocation, changing client demands over time, newly joined hubs, and a high likelihood of disappointment in overburden hubs, among others [3– 5]. Load adjusting is the ideal way to deal with the aforesaid problem in a distributed computing foundation, as it ensures that administrations are delivered without regard for physical utilization or space within the "cloud."

Load balancing has progressed tremendously in recent decades, and one of the most promising fields is swarm insight computations, such as insect settlement streamlining [6–8], fake honey bee state [9,10], molecular swarm improvement [11,12], and so forth. Marco Dorigo introduced subterranean insect settlement streamlining in 1992 [13], which is a family of stochastic advancement algorithms based on the behaviors of an ant colony. Cloud Computing provides a client's advantage within a certain time frame by employing assets, data, computer programs, and shared hardware. Of fact, in terms of the amount of time spent on the Internet, the whole Web may be called a Cloud. Cloud Computing may help you save money on labour and capital [4]. Understanding the influence of stack modifying inside the Cloud is crucial from the standpoint of promote presence, because to the widespread adoption of Cloud Computing in recent years. Cloud Computing Stage is a fully computerised benefit stage that enables customers to buy, produce separate, energetic versatility, and framework management [5].

When one or more components of the framework are inconvenient, the stack measuring is automated to keep a strategic distance from disruption in the delivery of a benefit. In this example, the framework components are constantly monitored, and if one fails to respond, stack balancing kicks in, and no activity is delivered to it. Regularly, difficulties may be eliminated with suitable stack adjustment, which not only reduces costs and makes computing greener, but also maintains the strain on the one-of-a-kind circuits to a minimum, potentially extending their life [6]. Load balancing in Cloud Computing frameworks is currently a difficult task. It might be a tool that distributes the active local workload evenly across all hubs in the Cloud to prevent a situation where some hubs are overloaded while others are idle or underloaded. It makes a difference to achieve high customer satisfaction and framework asset usefulness. It also ensures that each computational asset is distributed efficiently and fairly [7].

The load here may be measured in terms of the CPU stack, total memory used, latency, or arrange stack [8]. Stack adjustment ensures that all of the processors in the system, or each hub within the organization, are doing nearly the same amount of work at any one time [9]. Stack adjusting calculations are divided into three categories based on who initiates the method of stack adjusting: sender started, recipient started, and symmetric (combination of sender started, collector started sorts) [6]. They are also divided into two categories based on the framework's current state: inactive and energetic. The stack may be modified using stack adjusting by actively swapping neighboring workload from one computer to another inside the inaccessible hub or a machine that is underutilized [10]. To satisfy the Green Computing

requirements, Clouds must also adapt their stacks. One of the most difficult aspects of Cloud Computing is load balancing. To achieve a high client fulfilment and asset utilization percentage, it is necessary to distribute the stack equitably at each hub, ensuring that each computing asset is distributed productively and appropriately.

As a result, in this article, we examined several load balancing calculations in Cloud Computing and came to the conclusion that we are ready to use an exceptional calculation based on our requirements. Regardless, Cloud Computing encompasses a vast range of topics, and none of the above computations currently meet all of the requirements. As a result, the need to develop a flexible approach that is appropriate for a variety of scenarios is critical.

1.1 Motivation

Cloud computing is a fantastic concept. A large number of load balancing computations in cloud computing have been suggested. This paper has presented a part of the computations. The Internet as a whole might be thought of as a cloud of many connections rather than a collection of organized administrations. As a result, the distinct load scheduling hypothesis for Wireless systems shown in [9] may also be applied to mists. The results of several computations have been examined and compared.

1.2 Objective

- To study the cloud computing environment
- To study the performance of some of the existing load balancing algorithms
    a. Round Robin
    b. Throttled
    c. Equally Spread Current Execution Load
    d. Shortest Job First

1.3 Paper Structure

The structure of the paper is given as follows. Section 2 focuses on Load balancing concept while section 3 shows the literature review done during the research work. Section 4 and 5 shows the dynamic load balancing algorithm and its corresponding implementation. Finally, section 6 shows the overall conclusion of the research work.

2. Load Balancing

It is a way of reassigning the hard and fast weight to the various centres of the framework system in order to optimise resource efficiency and action reaction time while addressing a circumstance where certain centres are over loaded while others are under stacked. A dynamically changing estimate does not take into account the system's past condition or direction, relying instead on the structure's present lead. The primary fascinating focuses when making such a count are: weight estimate, weight evaluation, security of various systems, structure execution, association between the centres, concept of work to be exchanged, centre point selection, and numerous others [4].

This pile may be measured in terms of CPU load, memory use percentage, latency, or Framework load.

A website or a web-application may be accessed by a large number of people at any time. It becomes difficult for a web application to handle all of these consumer requests one by one. It might even cause system failures. The lowering sensation of a site being down or not accessible also delivers lost anticipated clients for a site owner whose whole career is depending on his passage. The load balancer accepts an important action in this case.

Cloud Weight modifying is a method of spreading out the remaining tasks and allocating resources among at least one server. Such a spread guarantees the highest possible throughput with the shortest possible reaction time. The significant weight is distributed among at least two servers, hard drives, arrange interfaces, and other calculating resources, allowing for greater resource use and system reaction time. As a result, for a high-traffic site, effective use of cloud load balancing may assure business clarity. Load balancers are often used to achieve the following goals:

• To maintain the structural integrity.
• To enhance the execution of the structure.
• To avoid difficulties with the framework.

Cloud suppliers such as Amazon Web Services (AWS), Microsoft Azure, and Google supply cloud load acclimating to help with the essential dispersal of exceptional operations that must be completed. For example, AWS provides Flexible Weight Altering (ELB) technology to help EC2 models fit traffic. ELBs are offered as important compositional fragments in the majority of AWS-controlled applications.

Virtualization might be a crucial factor in distributed computing. Virtualization, as the name implies, isn't certifiable yet provides all of the real-world workspaces. Virtualization is the use of virtual PCs to run a variety of different projects as if they were a real system. At the end of the day, Amazon EC2 is a cloud-based IT platform where suppliers' data is housed inside the structure of virtual machines [22].

2.1 Static Load Balancing

Static algorithms are well-suited to frameworks with modest load variations. The traffic is distributed evenly across the servers via a static computation. This computation requires prior knowledge of framework assets, since the processors' execution is resolved near the start of the execution. As a result, the decision of relocating the heap does not rely on the current state of the framework. However, static load adjusting algorithms have the drawback of relegating errands to the processor or machines immediately after they are created, as well as the inability to shift errands to another machine for load adjusting during execution.

For a work-based application, the static load balancing problem comprises dividing the work among subdomains. The subdomains might then be sent across the processors and counted in parallel. Various task allotments may result in different times for the calculation to be completed. As a result, it's critical to consider the nature of the parcelling in terms of its influence on the application code. There are several components [35].

2.2 Dynamic Load Balancing

Dynamic load adjustment should be possible in two different ways in an appropriated structure: circular and non-dispersed. The dynamic weight adjusting calculation is carried out by all centre points in the system in the ringed one, and the responsibility of weight modifying is divided among them. The form of the correspondence between centre points to

facilitate load shifting might be helpful or unappealing [4]. In general, the centre points work together to accomplish a common goal, such as improving overall reaction time. Each centre point in the following structure works independently toward a target neighbourhood, for example, to increase the reaction time of a close by attempted. Because all of the system's centre points must communicate with one another, dynamic weight-altering calculations of this kind usually output a higher number of messages than non-flowed ones. If one or more centre points in the system miss the mark, the overall weight altering approach will not halt, but it will impact the structure execution considerably. Appropriate dynamic burden adjustment may add a significant amount of stress to a system in which each centre point in the structure must share status information with one another. When a big fraction of the centres demonstrates autonomously with just a few connections with others, it is dynamically valuable.

In the non-coursed version, one centre point or a social affair of centres might do the weight-changing activity. The structures of non-passed on strong weight modifying calculations may be divided into two types: unified and semi-appropriated. The stack modifying estimation is only performed in the primary structure by a single centre point in the whole system: the central centre point. This centre is entirely responsible for troubleshooting the whole system. Only the central centre point is discussed when substituting centre points. The system's centre points are segregated into bundles in a semi-passed on structure, and the pile modifying in each gathering is of brought together structure. In each bundle, an acceptable racing strategy selects a core centre point that handles problems shifting inside that pack. From now on, techniques for the core centre points of each gathering will complete the stack altering of the whole structure [4].

3. Literature Review

Load balancing is predicted to play a crucial function in ensuring cloud enrolment quality of service (QoS), and it has been giving enough energy for the investigation strategy. Several strategies have been tweaked to meet the issue of store switching in allocated processing. We categorise prior weight-loss attempts into two groups depending on the vital count. The first group consists of various traditional systems that do not make extensive use of knowledge calculations.

Numerous load balancing approaches were suggested endeavours, and each revolved around different estimations and procedures, such as using a central weight adjusting approach for VMs [17], the arranging philosophy on trouble modifying of VM resources information on racial figuring [18], a planning course of action subject to multi-resource load changing for virtual machines [19], flexible scattered count for virtual machines [20], and w An organization-based model for large-scale accumulation [23], data centre authority structure [24], and a heterogeneous cloud [25] were among the troubleshooting methods exhibited for numerous cloud applications. Regardless of how these tasks have gained remarkable headway in issue solving under circulated registering, it has a high degree of centralization and is not difficult to increase. Furthermore, the methods described do not fully capture the characteristics of advantage centre placements, and they are becoming increasingly sensitive to the static status of assigned registration.

Swarm information computations are used in underground insect state upgrading [6–8], counterfeit bumble bee state [9,10], and particle swarm smoothing out [11,12], for

example, which is better for the dynamic situation of dispersed registration. These social bugs may be repeated with all factors taken into account, or with significant changes, to deal with undifferentiated from problems in transmitted registration, utilising self-sifted through direct. In [6, Nishant, K. et al. produced a figure for issue transfer of an unusual weight using a balanced approach of ACO from the perspective of cloud or grid compose structures. Throughout the approach, rather of energising their own result set, the ants just revived a single result set. [7] proposed a pile shifting framework in light of underground creepy crawly condition and difficult framework theory in an open dispersed registration alliance. This is the first time ACO and advanced frameworks have been employed in troubleshooting in appropriate processing, resulting in incredible performance. In [8, Mishra, R., and colleagues developed a method for issue changing in the cloud by employing ACO to extend or restrict different execution constraints, such as CPU weight and memory limit. In any event, when using ACO to discover target centre points in the more than three philosophies, just a few components were regarded necessary. In [9,10], V. Sesum-Cavic et al. proposed a new method for troubleshooting based on a synthetic bumble bee colony. SILBA (self-movement load modifying administrators) was presented as a consistent way for improving the interchange of diverse figures through stopping systems. Six computations were connected together in this framework, and the results showed potential Amazon EC2 cloud regions of interest [18]. Despite the fact that SILBA is a better model, it fails to account for the reduced demand for centre servers in cloud preparation scenarios as well as changing customer needs. Particle swarm progression (PSO) has been utilised to tackle challenges in information processing in the past, for example [17]. To circumvent the real weight cumbersomeness issue, it produced another task arranging model in [11], which improved on the basic PSO by including a fundamental change segment and a self-modifying inertness weight strategy. Feng, X. et al. created a resource task display subject to a discrete particle swarm upgrade count. The findings showed that although the above PSO solutions may help troubleshooting, they can be time consuming when dealing with high numbers of processes. [26–33] contains a range of applications and research on problem balancing using swarm understanding. Using these approaches in communicated registration requires a lot of effort since they were designed for distributed processing rather than dispersed burden altering.

The author in [34] looked at almost every cloud computing architectural arrangement in which the cloud computing system is separated into two parts: front-end and back-end. Both are connected via the internet. The front conclusion is apparent for clients, while the back conclusion is for the cloud system [12]. The client's computer is linked to the cloud on the front end, while the back end consists of 'cloud computing services' such as capacity, computers, and so on. Author also examined into cloud computing administrations and tiers, such as Program as a Service, Stage as a Service, and Framework as a Service, to name a few. There are also certain challenges in terms of protection, security, and long-term quality, to mention a few. [6] describes the fundamental notions of cloud computing, as well as the organisations that are promoted and their primary components, the cloud's capabilities, and a few current stack adjusting calculations that can be created on cloud [6]. The authors of [18] investigated the art of stack modification in cloud computing. By defining the term, explaining it, and illustrating how it is used to set up distributed systems, they develop the skill of stack adapting for cloud computing. Join-Idle-Queue calculations for dispersed stack

modification in enormous systems design were presented in [7]. This algorithm produces superior results for the framework stack. At a medium to large stack, it results in a 30-fold decrease in line overhead when compared to Power-of-Two. The author of [16] presented a particular kind of long accessibility application that has been more common in conveyed computing in recent years. An improved calculation is predicted based on the weighted slightest association computation. The current algorithm considers the stack and control, as well as a single exponential smoothing estimate. The authors of [35] looked at the Stack Adjusting Technique of Cloud Computing on Manufactured Bee Calculation, which might be a method based on bumble bee collection. By replicating honeybee behaviour, it increases the degree of nectar to obtain the highest throughput. The author looked at a gadget called Cloud-Analyst in [12]. The cloud examiner is used to assess a cloud-based social application's performance. It is the most latest version of CloudSim. [9] looked at three different types of stack adjusting computations: round robin, simple lining, and randomised calculations. MIPS vs. VM and MIPS vs. Have ground work were the focus of their study. When considering the entire number of VMs in a Datacenter, their studies demonstrate that these calculations may greatly reduce response time. The pantomime's execution decision shows that modifying MIPS has an impact on reaction time. The authors of [13] concentrated on the execution of three stack adjustment calculations, assessing their shortcomings and questioning why using a Centralized Planning approach in a cloud setting is unreasonable. The designer considered three different arrangements for stack adjustment: Bumble bee Scrounging Conduct calculation, Irregular Testing calculation, and Dynamic Clustering calculation. The authors investigated the Altered Throttled approach in [14], which ensures a list table of virtual machines. A concerted effort has been made to improve response time and maximise virtual machine utilisation. The proposed motivation adopts a method for choosing a virtual machine (VM) to fulfil a client's request in which the VM in the first list is effectively allocated depending on the VM's condition. If the requested VM is free then, it is allocated to the request and its id is sent to the DC; otherwise, 1 is returned.

4. Dynamic Load Balancing

In this section few dynamic load balancing algorithms such as Equally Spread Current Execution (ESCE), Throttled, Round Robin, Shortest Job First (SJF) are discussed.

4.1 Equally Spread Current Execution (ESCE)

ESCE load balancer makes an attempt to guarantee that all virtual machines associated with the data centre earn back the initial investment with stack. The stack balancer keeps track of the Virtual Machines and the requests delivered to them (VM). If the data centre requests the advanced VM, it searches the record database for the least stacked VM. For each customer/request, the stack balancer returns the VM ID to the data centre controller and picks the first perceived VM. The data centre sends the project to the VM identified by that id. By increasing the task check of recognised VMs, the data centre rethinks the rundown table. When a VM completes a work, an assignment is sent to the data centre, which is advanced educated by the stack balancer. The stack balancer modifies the record table once again by lowering the job check for every VM by one, but this time there's an additional computation overhead to channel the line over and over.

4.2 Throttled Load Balancing (TLB)

The heap balancer keeps track of virtual machine states in a table (Accessible or Dynamic). As a result, the customer/server asks the data centre to find a suitable virtual machine (VM). The data centre wants the VM on the stack balancer. The stack balancer cycles the rundown table until it finds the requisite open VM or the list table is checked. The VM will find the stack data centre. Request sent to the VM with id. It then locates the cutting-edge assignment's stack balancer and changes the record table.

4.3 Round Robin (RR)

A famous appropriation concept known as Weighted Cooperative Assignment is employed in this RRLB, in which a load is divided to each VM, so that if one VM is configured up to carry twice as much weight as the other, the mind-boggling server receives a load of 2. DC Controller will allocate the two sales to the noteworthy VM in this instance for each request that is promoted to a more delicate one. Because Cooperative Figuring determines the pile based on subjective criteria, some centre points are placed tightly while others are stacked loosely. The computation, on the other hand, is quite straightforward, and the scheduler is also responsible for calculating the quantum degree [5]. It features a longer average wait time, more configuration changes, a slower turnaround time, and lower throughput.

4.4 Shortest Job First (SJF)

Planning for the shortest job first (SJF) for non preemptive scheduling is essential. Non-preemptive techniques imply that once a processor has been assigned time, it cannot be taken over by another until the operation has finished. Occupation is the most restrictive in terms of freedom. Initially, a dynamic burden adjustment calculation is performed, which deals with the approach based on the demand. It determines the requirement by looking at how long the procedure takes. The heap is crudely appropriated in this computation by first confirming the procedure's scope and then trading the heap to a lightly stacked Virtual Machine.

Because the procedure measure is the smallest, it will be the first to be tested to check whether the most reduced estimated process can be completed in the lowest time. The heap balancer distributes the heap over numerous hubs using the spread range method. The instrument of Most Brief Occupation First Calculation is to organise the way that takes the least amount of time to complete first, resulting in high competence and rapid turnaround. The framework demands a short period of time from the time it starts to enter the framework until the process is finished in terms of time spent in the present programme (work). Because the normal holding time is modest, the shortest job first (SJF) planning computation is great, resulting in speedier framework execution.

5. Implementation

Cloud Analyst [8] [9] [10] [11] might have been a CloudSim-based graphical user interface (GUI) application. CloudSim might be a collection of tools for modelling, replication, and other forms of research. CloudSim has a basic flaw in that it needs all work to be done programmatically. It allows the user to quickly and easily do several reenactments with modest parameter adjustments. You may define the location of the clients who are constructing the application, as well as the location of the data centres, using the cloud investigator. This method may be used to provide many configuration elements such as the number of customers, the number of requests produced per user per hour, the number of virtual machines, the number of processors, the sum of capacity, network bandwidth, and other essential features. Based on the settings, the instrument computes the simulation result and shows it in graphical form. Response time, handling time, fetched, and so on are all part of the result. In the Cloud Analyst environment, many algorithms are installed. Table 1, 2 and

3 shows the region configuration, UB configuration and DC configuration respectively. Figure 1 shows the simulation configuration. Figure 2-9 shows the UB response time (RT) and DC processing time (PT) for different algorithms.

Table 1: Cloud Analyst Region Configuration

Region id	Users (Million)
R-0	4.95M
R-1	1.75M
R-2	3.2M
R-3	1.9M
R-4	0.65M
R-5	0.88M

Table 2: UB configuration

UB id	Region id	Number of users in peak Hours	Number of users in off-peak hour
UB-1	R-0	450000	45000
UB-2	R-1	150000	25000
UB-3	R-2	300000	20000
UB-4	R-3	100000	9000
UB-5	R-4	55000	5500
UB-6	R-5	80000	8000

Table 3: DC configuration

Parameter	Value Used
VM Configuration	
Image Size	15000
Memory	2048Mb
Band	2000
DC Configuration	
Architecture	X86
Operating System	Linux
Number of Machine	20
Memory per machine	2048Mb
Storage	150000Mb
Band	15000
Processors/Machine	4
Speed	100MIPS
Sharing Policy	Time Shared/Space Shared
Grouping in contrast to UB	1000
Grouping on the basis of Request	150
Instruction per unit time	200

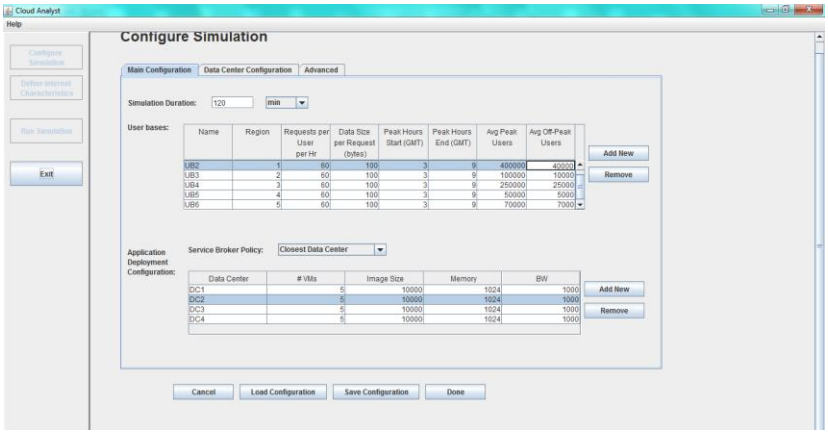

Figure 1: Simulation configuration

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

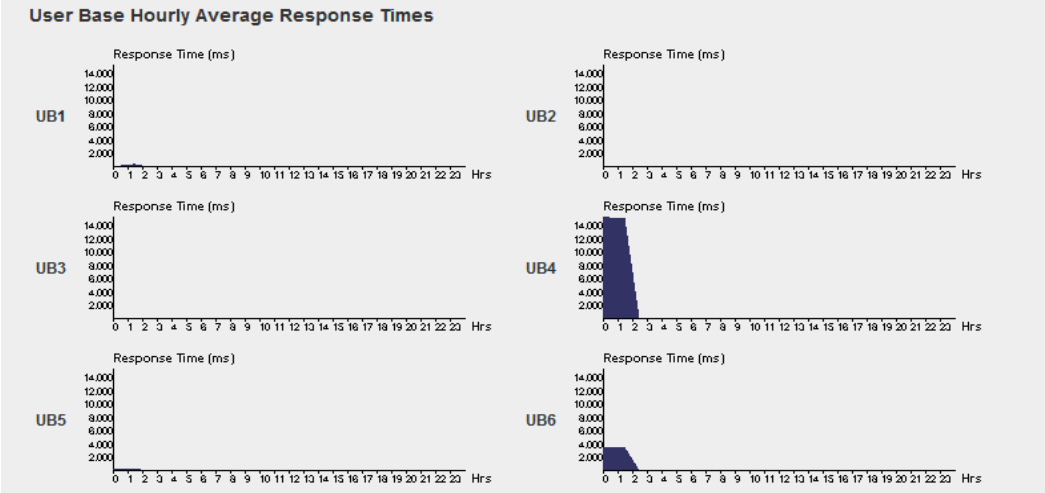

Fig 2: UBRT using RR Algorithm

Data Center Request Servicing Times

Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	2,969.523	2.755	4,531.415
DC2	1.584	0.037	3.261
DC3	39.421	1.251	75.784
DC4	15,217.003	6,124.812	19,011.485

Data Center Hourly Average Processing Times

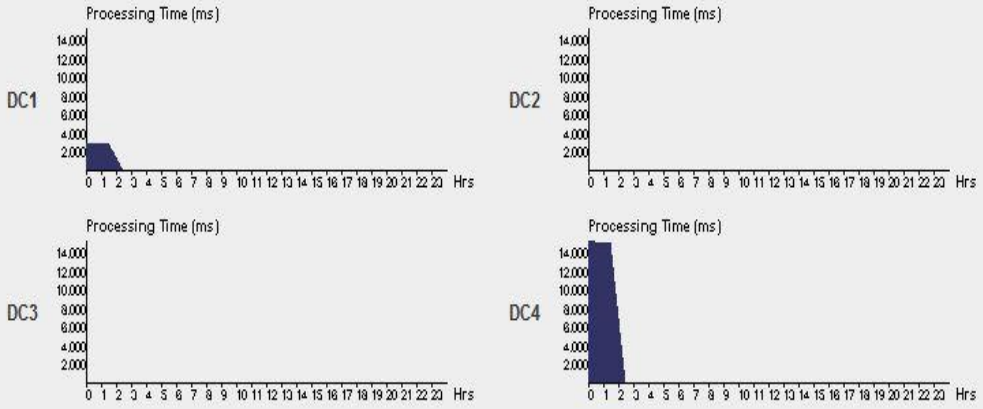

Fig 3: DCPT using RR Algorithm

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.301	43.536	61.11
UB2	106.92	47.962	131.678
UB3	196.489	46.576	255.511
UB4	114.563	49.591	143.601
UB5	313.382	242.509	467.773
UB6	215.591	165.797	260.978

User Base Hourly Average Response Times

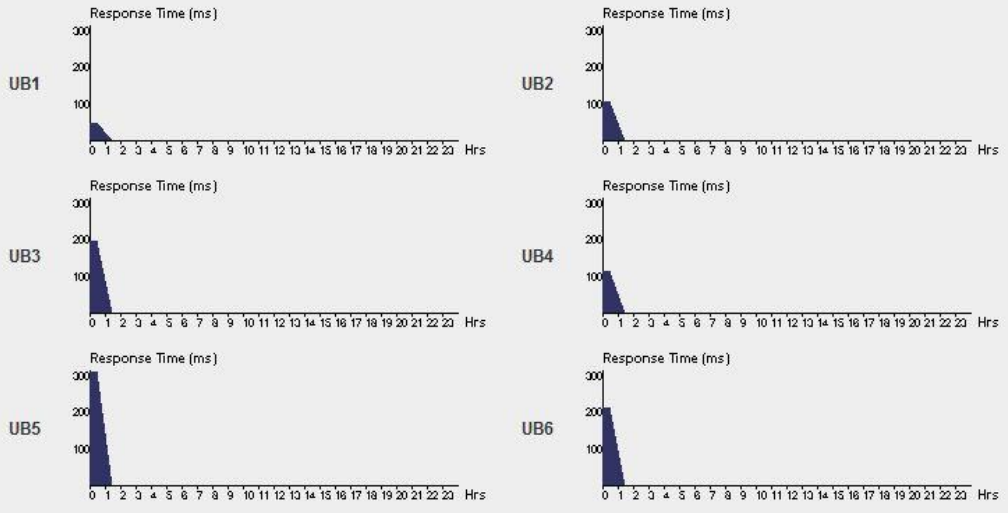

Fig 4: UBRT using ESCE Algorithm

Data Center Request Servicing Times

Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	13.017	0.054	33.34
DC2	56.113	8.152	72.649
DC3	121.636	1.579	197.55
DC4	64.338	7.409	87.198

Data Center Hourly Average Processing Times

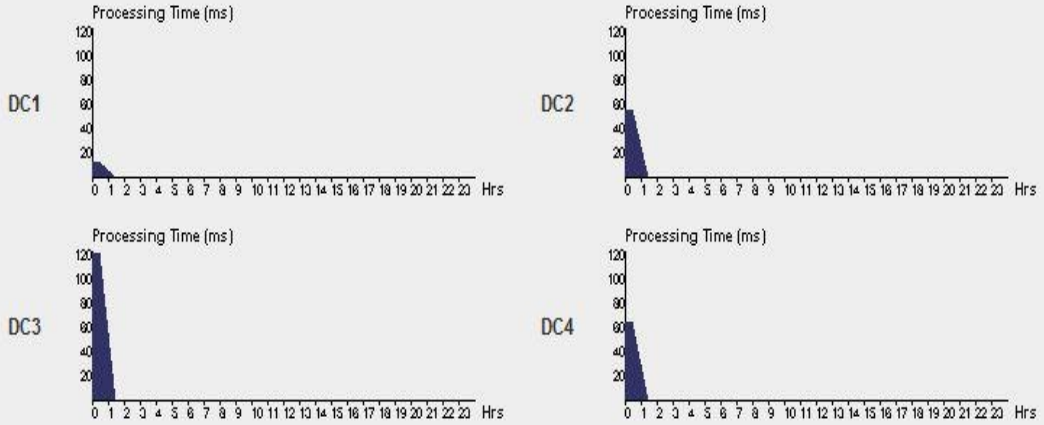

Fig 5: DCPT using ESCE Algorithm

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

User Base Hourly Average Response Times

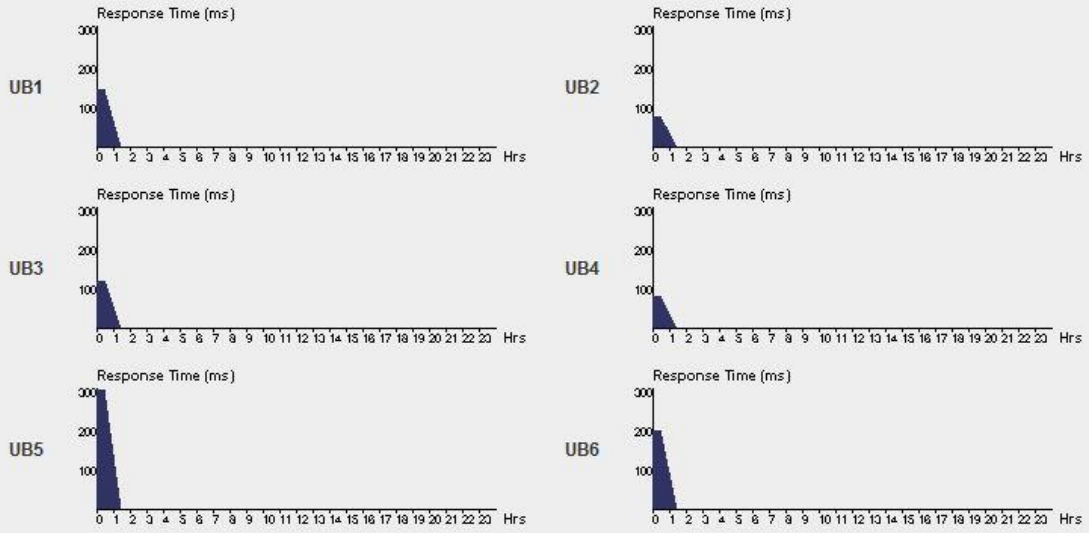

Fig 6: UBRT using TLB Algorithm

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

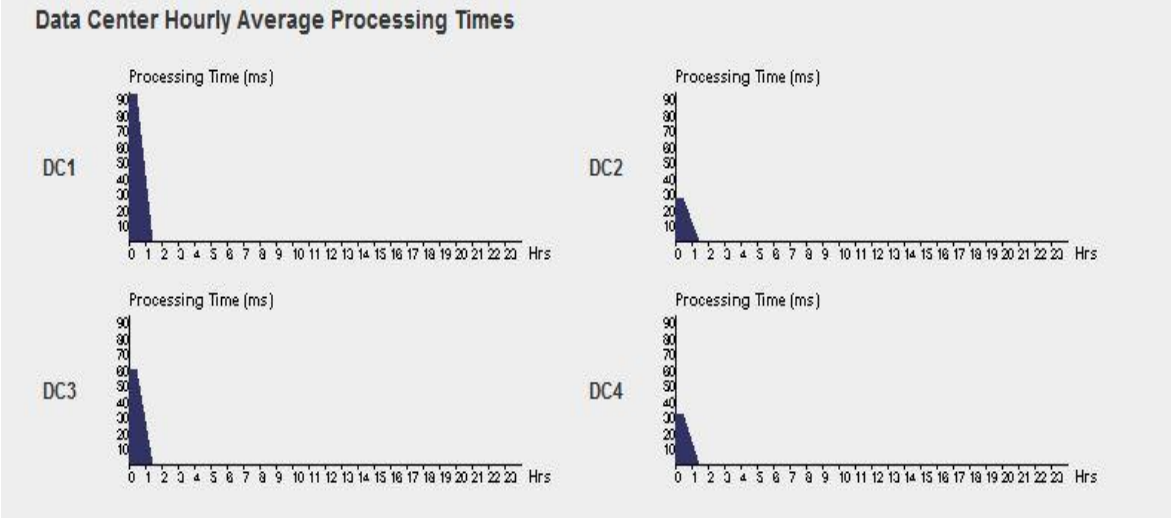

Fig 7: DCPT using TLB Algorithm

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

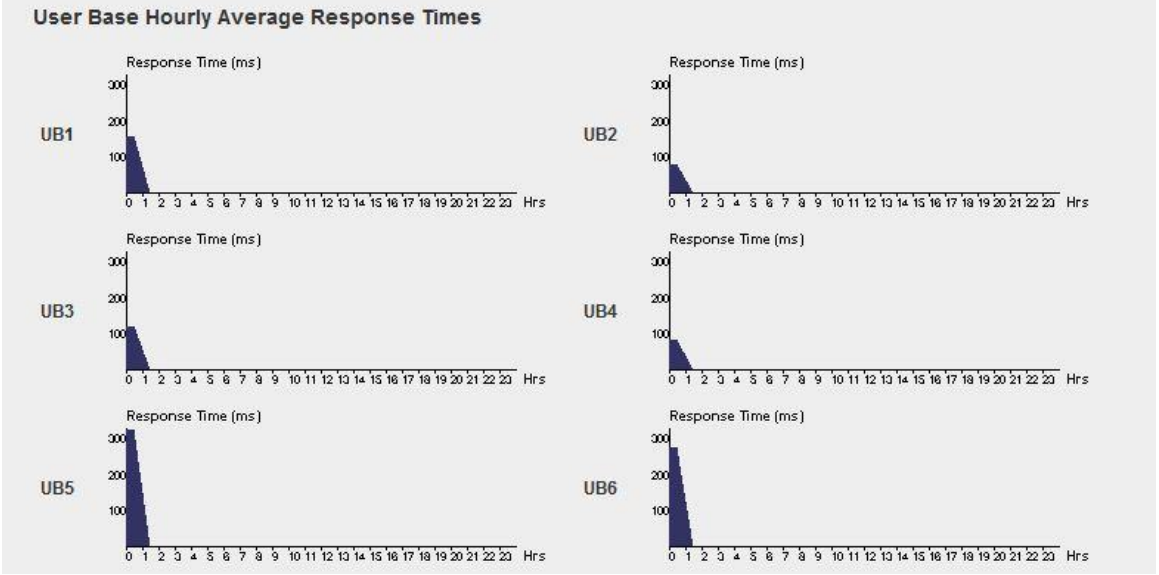

Fig 8: UBRT using SJF Algorithm

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

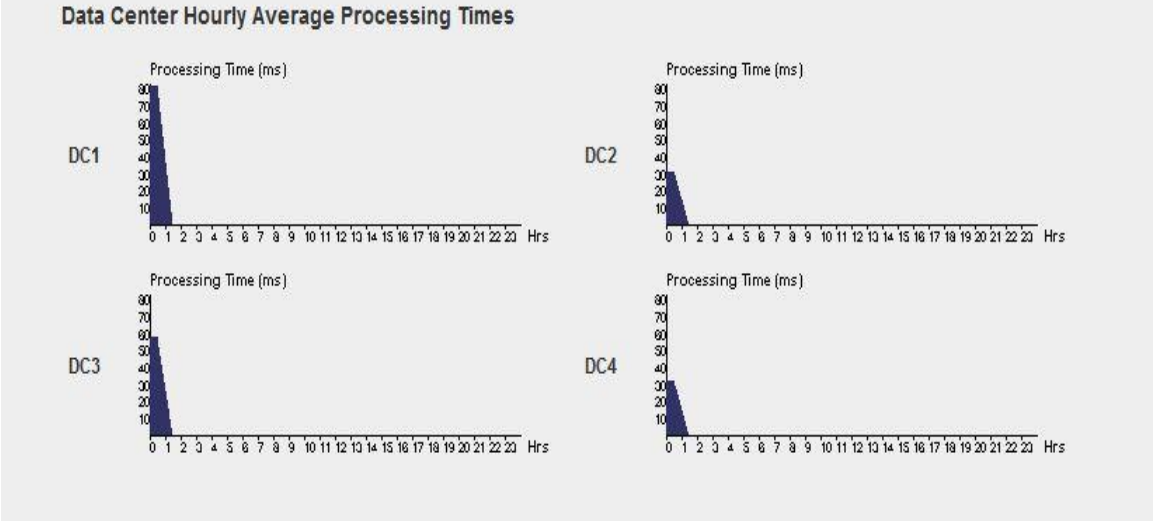

Fig 9: DCPT using SJF Algorithm

6. Conclusion

Despite the fact that many present challenges such as load balancing, virtual machine mobility, server unification, and other issues have yet to be completely solved, industry has embraced cloud computing. In fact, load balancing is the most significant challenge in the system, with the purpose of spreading load balancing as effectively as possible. It also assures that each registered asset is used correctly, on time, and at a low cost. The currently explored load balancing methods/calculations are mostly focused on decreasing overhead, reducing relocation time, and improving execution, among other things. The response time is a test of each specialist's ability to devise a way for increasing cloud-based component throughput. Due to the limited number of processes, effective scheduling and asset distribution techniques are required, resulting in higher operational expenses.

We've looked into distributed computing and load balancing, as well as several current burden-adjustment algorithms that may be used with mists. Furthermore, for single level tree systems with various load adjustment methodologies, closed structure solutions for least estimation and announcement time were investigated. The presentation of techniques such as Choked, Cooperative effort, Equivalent Spread Current Execution, and Most Brief Employment First has been investigated in terms of reaction time and preparation time. In addition, depending on the established boundary, a comparison is made between multiple systems. A vast number of clients from different Client Bases in various places were included in this research. The Similarly Spread Current Execution performs brilliantly in the face of heavy load, according to the results of the tests and exams.

Reference

1. Afzal, S., & Kavitha, G. (2019). Load balancing in cloud computing–A hierarchical taxonomical classification. Journal of Cloud Computing, 8(1), 1-24.
2. Shafiq, D. A., Jhanjhi, N. Z., & Abdullah, A. (2021). Load balancing techniques in cloud computing environment: A review. Journal of King Saud University-Computer and Information Sciences.
3. Shafiq, D. A., Jhanjhi, N. Z., & Abdullah, A. (2021). Load balancing techniques in cloud computing environment: A review. *Journal of King Saud University-Computer and Information Sciences*.
4. Shafiq, D. A., Jhanjhi, N. Z., & Abdullah, A. (2021). Load balancing techniques in cloud computing environment: A review. *Journal of King Saud University-Computer and Information Sciences*.
5. Priya, V., Kumar, C. S., & Kannan, R. (2019). Resource scheduling algorithm with load balancing for cloud service provisioning. Applied Soft Computing, 76, 416-424.
6. Milan, S. T., Rajabion, L., Ranjbar, H., & Navimipour, N. J. (2019). Nature inspired meta-heuristic algorithms for solving the load-balancing problem in cloud environments. Computers & Operations Research, 110, 159-187.
7. Nayak, D. S. K., & ShreerudraPratikb, J. (2022). IoT Ecosystems Enable Smart Communication Solutions: A Case Study. *Network*, *3*(2), 6.
8. Rajwinder Kaur and Pawan Luthra," Load Balancing in Cloud Computing", Association of Computer Electronics and Electrical Engineers, 2014, DOI: 02.ITC.2014.5.92
9. Ashalatha R; J. Agarkhed," Dynamic load balancing methods for resource optimization in cloud computing environment ", 2015 Annual IEEE India Conference (INDICON) , Pages: 1 - 6, DOI: 10.1109/INDICON.2015.7443148
10. Nayak, D. S. K., Mahapatra, S., & Swarnkar, T. (2018). Gene Selection and Enrichment for Microarray Data—A Comparative Network Based Approach. In *Progress in Advanced Computing and Intelligent Engineering* (pp. 417-427). Springer, Singapore.
11. R. Kanakala; V. K. Reddy; K. Karthik, " Performance analysis of load balancing techniques in cloud computing environment", 2015 International Journal of Computer Sciences and Engineering Vol.-5(1), Jan 2017, E-ISSN: 2347-2693 © 2017, IJCSE All Rights Reserved 100 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), Pages: 1 - 6, DOI: 10.1109/ICECCT.2015.7226052
12. PATI, ABHILASH; Parhi, Manoranjan; and Pattanayak, Binod Kumar (2022) "IoT-Fog-Edge-Cloud Computing Simulation Tools, A Systematic Review," *International Journal of Smart Sensor and Adhoc Network*: Vol. 3 : Iss. 2 , Article 3.
13. A.N. Ivanisenko; T. A. Radivilova , "Survey of major load balancing algorithms in distributed system ", Information Technologies in Innovation Business Conference (ITIB), 2015 ,Pages: 89 - 92, DOI: 10.1109/ITIB.2015.7355061
14. Sidra Aslam, Munam Ali Shah, "Load Balancing Algorithms in Cloud Computing: A Survey of Modern Techniques", 2015 National Software Engineering Conference (NSEC 2015)

15. A. Jaiswal, Dr. Sanjeev Jain," An Approach towards the Dynamic Load Management Techniques in Cloud Computing Environment", 2014 International Conference on Power, Automation and Communication (INPAC)
16. Pati, A., Parhi, M., & Pattanayak, B. K. (2022). IADP: An Integrated Approach for Diabetes Prediction Using Classification Techniques. In *Advances in Distributed Computing and Machine Learning* (pp. 287-298). Springer, Singapore.Surbhi Kapoor, Dr. Chetna Dabas," Cluster Based Load Balancing in Cloud Computing", 2015 Eighth International Conference on Contemporary Computing (IC3).
17. Shridhar G.Domanal and G.Ram Mohana Reddy, " Load Balancing in Cloud Computing Using Modified Throttled Algorithm ", 2013 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)
18. Pati, A., Parhi, M., & Pattanayak, B. K. (2021, January). IDMS: An Integrated Decision Making System for Heart Disease Prediction. In *2021 1st Odisha International Conference on Electrical Power Engineering, Communication and Computing Technology (ODICON)* (pp. 1-6). IEEE.
19. Foster, I., Zhao, Y., Raicu, I. & Lu, S. (2008). Cloud computing and grid computing 360-degree compared. IEEE Grid Computing Environment Workshops.
20. Youseff, L. Toward a Unified Ontology of Cloud Computing Retrieved from http://spsteve. wikispaces.asu.edu/file/view/unified+interface+cloud.pdf.
21. R. Buyya, R. Ranjan, and R. N. Calheiros. Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities. Proceedings of the Conference on High Performance Computing and Simulation (HPCS 2009) (pp. 21-24). IEEE Press, New York, USA, Leipzig, Germany, June, 2009.
22. Pati, A., Parhi, M., & Pattanayak, B. K. (2021). COVID-19 Pandemic Analysis and Prediction Using Machine Learning Approaches in India. In *Advances in Intelligent Computing and Communication* (pp. 307-316). Springer, Singapore.
23. Radojevic, B. & Zagar, M. (2011). Analysis of issues with load balancing algorithms in hosted (cloud) environments. In proceedings of 34th International Convention on MIPRO, IEEE. 14 International Journal of Distributed and Cloud Computing Volume 1 Issue 2 December 2013
24. Lee, R. & Jeng, B. (2011). Load-balancing tactics in cloud. In proc. International Conference on CyberEnabled Distributed Computing and Knowledge Discovery (CyberC), IEEE, (pp. 447-454).
25. Wang, S. C., Yan, K. Q., Liao, W. P. & Wang, S. S. (2010). Towards a load balancing in a threelevel cloud computing network. Proceedings of 3rd International Conference on Computer Science and Information Technology (ICCSIT), IEEE, July, 1, 108-113.
26. Randles, M., Bendiab, A. T. & Lamb, D. (2008). Cross layer dynamics in self-organising service oriented architectures. IWSOS, Lecture Notes in Computer Science, 5343, pp. 293-298, Springer.
27. Randles, M., Lamb, D., Bendiab, A. T. (2010). A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing. 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops 978-0-7695-4019-1/10 $26.00 © 2010 IEEE. DOI 10.1109/ WAINA.2010.85.

28. Shu-Ching, W., Yan, K. Q., Sheng, S. & Wei, W. C. (2011). A three-phases scheduling in a hierarchical cloud computing network. 2011 Third International Conference on Communications and Mobile Computing 978-0-7695-4357-4/11 $26.00 © 2011 IEEE DOI 10.1109/CMC.2011.2.
29. Li, K., Xu, G., Zhao, G., Dong, Y. & Wang, D. (2011). Cloud task scheduling based on load balancing ant colony optimization. 2011 Sixth Annual ChinaGrid Conference 978-0-7695-4472-4/11 $26.00 © 2011 IEEE. DOI 10.1109/ChinaGrid.2011.17.
30. Vesna Sesum-Cavic Institute of Computer Languages Vienna University of Technology Wien, Austria, Eva Kühn. Applying swarm intelligence algorithms for dynamic load balancing to a Cloud Based Call Center" 2010 Fourth IEEE International Conference on Self-Adaptive and Self-Organizing Systems. 978-0-7695-4232-4/10 $26.00 © 2010 IEEE DOI 10.1109/SASO.2010.19.
31. Naghibzadeh, M. (2007). A min-min max-min selective algorithm for grid task scheduling. 1-42440- 1007-X/07/$25.00, 2007 IEEE. Dept. of Computer Engineering Ferdowsi University of Mashad.
32. Al Nuaimi, K., Mohamed, N., Al Nuaimi, M. & Al-Jaroodi, J. (2012). A survey of load balancing in cloud computing: challenges and algorithms. 2012 IEEE Second Symposium on Network Cloud Computing and Applications 978-0-7695-4943-9/12 $26.00 © 2012 IEEE DOI 10.1109/NCCA.2012.29.
33. Zhao, C., Zhang, S., Liu, Q., Xie, J. & Hu, J. (2009). Independent Tasks Scheduling Based on Genetic Algorithm in Cloud Computing.
34. P, Mandal J K, Dam S 2013 A Genetic Algorithm (GA) based load balancing strategy for cloud computing Int Conf Comput Intell Model Tech Appl [Internet] Elsevier B.V. 10 340–7.
35. Panigrahi, A., Sahu, B., Rout, S. K., & Rath, A. K. (2021). M-Throttled: Dynamic Load Balancing Algorithm for Cloud Computing. In Intelligent and Cloud Computing (pp. 3-10). Springer, Singapore.