BioGecko Vol 12 Issue 03 2023

ISSN NO: 2230-5807

Horizontal Scalability based Load Balancing in Cloud Computing for Efficient Task Allocation using Group Task Algorithm

Aruna Mailavaram
Asst. Prof. in Keshav Memorial Institute of Technology, Hyderabad, India.
Dr. B.Padmaja Rani
Prof. JNTUH College of Engineering, Hyderabad, India.

Abstract :
Cloud computing is a new paradigm with the goal of providing on-demand, pay-as-you-go, network access to shared resources in a metered, self-service, dynamically expandable manner. . In this research work, some concerns in task scheduling were explored, such as task processing time, execution delay, server failure and complexity. The tasks that are arriving can be classified as high priority and normal priority tasks and they are allocated to respective server groups with Virtual Machines (VM). The virtual machine if allotted with any task fails, the traditional models fails to allot a new virtual machine to avoid the delay in the cloud environment. The proposed model considers with dynamic resource allocations using machine learning technique that are required to process tasks without delay. In this research, a Horizontal Scalability Model using Grouped Task Allocation (HSM-GTA) is proposed. The proposed model reassigns the tasks of failed virtual machines to the new available virtual machines to reduce delays in processing. The proposed model is compared with the traditional model and the results represent that the proposed model performance is enhanced.

Keywords: Horizontal Scalability, Task Allocation, Resource Allocation, Machine Learning, Failed Servers, Dynamic Scheduling.

1. Introduction : The name "Cloud Computing" is derived from the reference to a "cloud" that is commonly used to describe the Internet in most technical designs. When a firm uses cloud computing, it is offered a variety of resources that are actually held by a third-party supplier and located within their own data centres [1]. Physical hosts' resources are virtualized in the virtual machines through VMs [3]. Because of this, task allocation is critical [4]. Our focus in this study is to minimise the time it takes to complete a task and maximize the utilization of resources and also to add required resources dynamically to enhance computing power. There are two types of resource allocation algorithms: static and dynamic [5]. The on-demand feature and elasticity will not be taken into account in traditional models [6]. Horizontal scaling is the process of adding more servers to handle your current workload while distributing it among them in order to keep the load on each one to a manageable level[9].There is a focus on the system's ability to adapt an increasing problem scope, which is referred to as the system's scalability [10].

2. Literature survey : As a Cloud Computing term goes, load balancing refers to ensuring that all of the available resources are not overburdened or under loaded [11]. Meta-scheduler was proposed by Dubey et al. [1] in order to remove the limitations of the Improved Bat.

According to Chen et al. [2], min-min load balancing can reduce the execution time of activities while simultaneously eliminating the downside of min-min scheduling. The scheduler selects the smallest task Ti and calculates the waiting time at all operating virtual machines to assign the task to the virtual machine that can complete it in the shortest amount of time.

Auto-scaling is defined bySinghet al., [7] by examining the classification system of auto-scaling techniques. This opens up a wide range of new research areas, particularly in the area of self-scaling technology. Kumar et al. [8] proposed a novel simulated cluster architectural style for dynamic scaling of cloud services in a virtualization cloud computing environment.

3. Proposed Work

According to the proposed paradigm, Cloud Computing applications can prevent unbalanced workloads by providing efficient resource allocation. Workload movement and task rejection concerns in the cloud can be addressed by this strategy. To submit requests to the cloud, customers can use a variety of different devices to connect to the Internet. Using the Cloudlet Schedule Time Shared algorithm, the model submits tasks in random and assigns them to VMs based on the Deadline and Complete Time parameters along with resource utilization time levels. The Data Center (DC) in Cloud Computing can be thought of as a huge storage facility for cloud storage and their associated data. Whenever a request comes in, the DC forwards it to the currently active load balancer. If the suggested algorithm is found to be violated, it will be implemented as a Scheduler in this model layer, which will be the first in the literature to handle migrations in cases of violation. When it comes to virtual machine allocation [12], the bottom layer is responsible for this task (VMs). To maximize CPU and cloud resources, the proposed framework offers dynamic scheduling and load balancing using horizontal scalability model. In this research, a Horizontal Scalability Model using Grouped Task Allocation (HSM-GTA) is proposed. The failed tasks are handled effectively by rescheduling such tasks to available server groups that is added using horizontal scalability model using machine learning technique. The model is trained with the strategy failed and further resource allocations can be performed using the trained model. The proposed model with group task allocation procedure is explained in the algorithm.

Algorithm HSM-GTA

{Step-1: Initialize the resource pool with ResPool{RID, Rtype, RLock}, TaskList{TID, Ttime, TResReq}

Step-2: The input tasks are considered and all the tasks are inserted into a Task Input Queue (TIQ) by allocating the TID and then calculates the Ttime and number of required resources to complete the operation.

Step-3: The resources that enters into the ResPool are registered for efficient allocation and distribution to avoid resource sharing collisions. The resource registration is performed as

$$\mathrm{RID}(R(N)) = \frac{SerGroup(N)}{N(i)} * \sum_{i=1}^{R} set(ID(R(i))_i^N + (TimeInstance) + Th$$

Here SerGroup(N) represents the server groups considered address, ID is the task ID, R is the resource considered, Th is threshold time to delay the waiting levels.

Step-4: The incoming tasks are arranged in the TIQ and then the tasks are classified based on the Ttime and TResReq. The task classification is performed as

$$TaskClassific(T[N]) = \frac{\sum_{i=1}^{N} \min\left\{exec\_time_{Task(i)}\left\{(MaxLimit_i^{Length})\right\}\right\}}{\sum_{i=1}^{N}\sqrt{\frac{TID(i)+MinLimit_i^{Length})}{TotalTime(T(i))}}}$$

Here minimum execution time tasks are considered as a separate cluster from maximum time taking tasks. The min limit and max limit are the tasks IDs that can have a maximum limit for scheduling.

Step-5: The tasks after classification are arranged as high priority and normal priority tasks based on the Ttime and TResReq. The identification of normal and high priority tasks are performed as

$$HighPri(Task[N])) = \frac{TaskClassific(T[i]) + \min limit_{task(i)}}{\min(RID(i))} + Ttime_{\max Res Req.}^{(N)}(RID(i))$$

$$NormPri(Task[N])) = \frac{TaskClassific(T[i]) + \max limit_{task(i)}}{\min(RID(i))} + Ttime_{\min Res Req.}^{(N)}(RID(i))$$

The task classification is done based on the burst time of the tasks. The resources are allocated to the scheduled tasks after classification.

Step-6: The resources are allocated to the required group tasks based on the server groups allotted for successful execution without delays. The group tasks allocation to server group is performed as $GroupTask(\text{TaskClassific(i)}) = \left(\sum_{i\in R_i^N}\sum_{i,j\varepsilon RID}\frac{|\min(RID(i)|}{|MaxTIme(T(i))|} * \min(\text{HighPri(T[i])}) + \max(\text{NormPri(T[i])})\right)$

Here T is the current task considered for scheduling. The priority is allocated to the tasks based on burst time and required resources to avoid delay in the cloud environment. The high priority tasks are scheduled prior to normal tasks to avoid delays.

Step-7: The horizontal scalability model is applied to enhance the services to reduce the delay. The tasks that are in the waiting state because of lack of resources will trigger the horizontal scalability model that improves the system capabilities dynamically. The process is performed as

$$Horz.Scal(RID(N), TID(M)) = \sum_{i=1}^{N} maxResusage\big(T(i)\big) + VM(i+1) + add(RID(i+1)) + Res_{avg}$$

Here VM is the virtual machines considered and the maximum resource usage is considered for efficient scheduling. To the VMs new VMs are added to improve the cloud capabilitiles.

Step-8: The Error rate is calculated for the load balancing by performing strategic resource allocation and task allocation. The error rate is calculated as

$$Errorrate = \frac{1}{T_{IDi} RID_m} \frac{\sum_{i=1}^{N_1} \sum_{j=1}^{N_2} min(waittime(RID(i))\hat{} + max(Ttime(i))}{\sum_{i=1}^{R} Total\_exec\_time(T(i))}$$

}

4. Results

To increase cloud computing performance levels, scale up or down of IT resources can be performed in the cloud as needed to meet changing requirements . The proposed model implements horizontal scalability model using machine learning . The proposed model is implemented in CloudSim and can be executed in Eclipse platform. The proposed Horizontal Scalability Model using Grouped Task Allocation (HSM-GTA) model is compared with the traditional cloud-native Multi-Agent Platform (cloneMAP) model and the results are evaluated by evaluating the parameters like Cloud Setup Time, Scaling out Time Levels, Resource Registration Time Levels, Task Classification Accuracy, All the parameters are measured in terms of milli seconds. Task Classification accuracy is a percentage. The cloud setup time levels of the proposed and traditional models are shown in Figure 1.

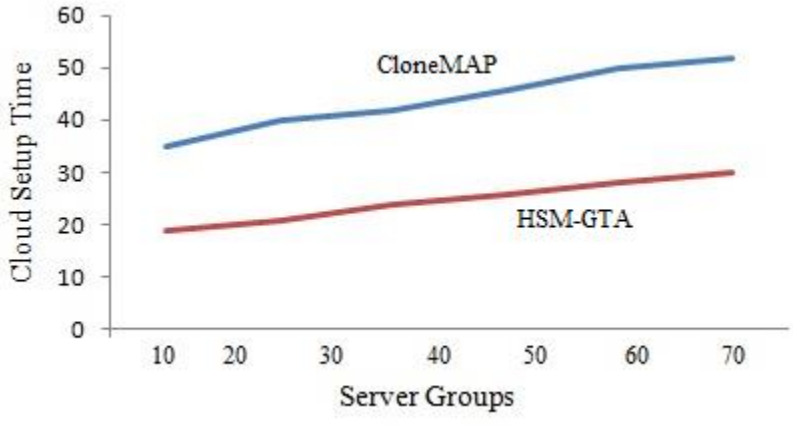

Fig 1: Cloud Setup Time

The scaling out levels of the proposed and traditional model are represented in Figure 2. The results represent that the proposed model scale out time levels are less when compared to traditional models.

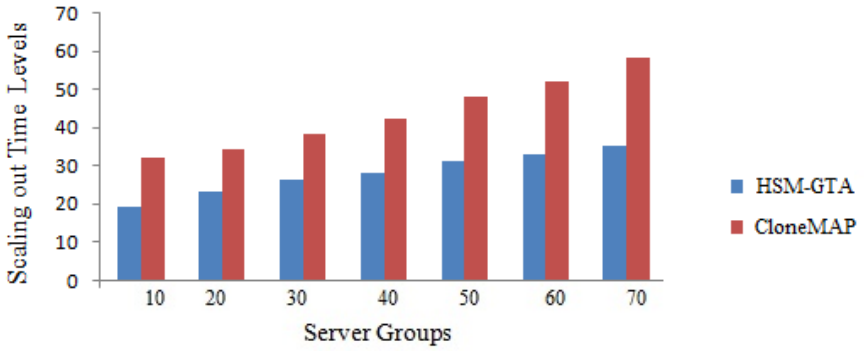

Fig 2:Scaling out Time Levels

The resources are initially registered in the cloud resource pool for efficient allocation and distribution for handling tasks. The resource registration time levels of the proposed and traditional model is indicated in Figure 3.

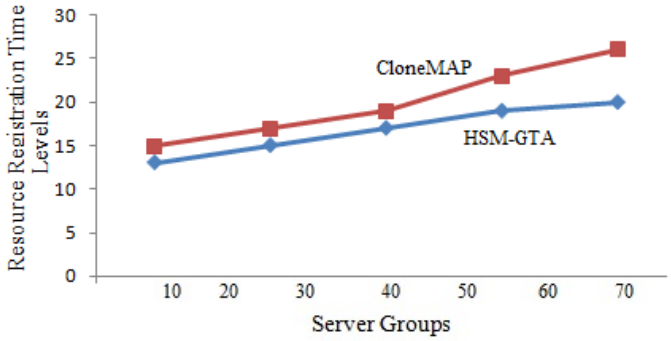

Fig 3: Resource Registration Time Levels

The tasks received are classified based on the execution time and the resource utilization time and such tasks are allotted to respective server groups to avoid delays. The task classification accuracy levels of the proposed and traditional models are shown in Figure 4.

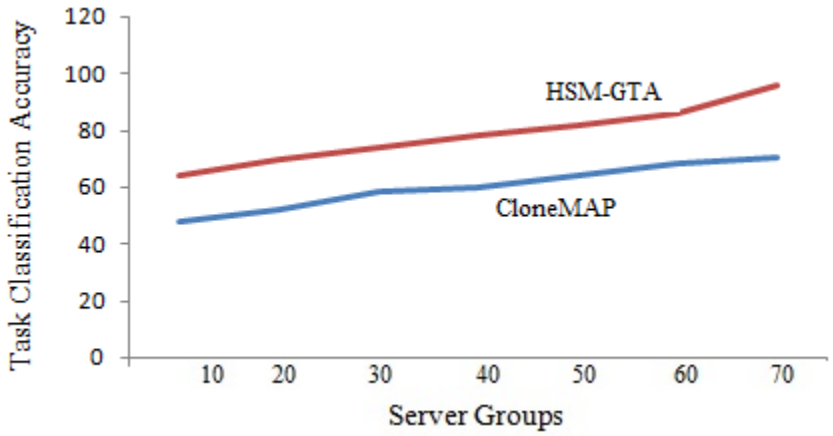

Fig 4: Task Classification Accuracy

5. Conclusion

Scalability is one of the most exciting features of cloud computing, allowing customers to use the resources they need as they see fit. The challenge of dealing with the unpredictability of cloud services and applications. More than anything else, the ability to scale is critical. When the need arises, resources can be scaled in and scaled out. It can be either static or dynamic. In the Cloud, however, dynamic or Auto scaling is the most commonly implemented and preferred method of scaling. The ability of a system to handle an increase in data processing demands is referred to as system scaling. Scaling platforms for big data processing uses Horizontal Scaling in the proposed research work, Horizontal Scalability Model using Grouped Task Allocation (HSM-GTA) with dynamic resource allocations using machine learning technique that are required to process tasks without delay is proposed. With scaling, load balancing is taken into account by dynamically allocating and reassigning overall load among all nodes to maximise resource utilisation. As a result, the system's performance is enhanced by distributing resources in an equitable and efficient manner. In future dynamic resource handling model can be optimized and the number of resource pools can also be improved for still enhancing the performance levels.

References

1. Dubey K, Kumar M,Chandra M, "A priority based job scheduling algorithm using IBA and EASY algorithm for cloud metaschedular," In: International conference on advances in computer engineering and applications, pp. 66- 70, 2015.
2. C. Chen, K. Li, A. Ouyang, Z. Tang, and K. Li, "Gpu-accelerated parallel hierarchical extreme learning machine on flink for big data," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 47, pp. 2740-2753, 2017.
3. Tsai H, Fang J,Chou J, "Optimized task scheduling and resource allocation on cloud computing environment using improved differential evolution algorithm.," Computer and Opertation Research, vol. 40, no. 12, pp. 3045-3055, 2013.
4. Somasundaram T, Govindarajan K, RajagopalanM,Rao SM, "A broker based architecture for adaptive load balancing and elastic resource provisioning and deprovisioning in multi-tenant based cloud environments," In: Advances in intelligent systems and computing, vol. 174, pp. 561-573, 2013.
5. S. K. Mishra, B. Sahoo, and P. P. Parida, ``Load balancing in cloudcomputing: A big picture," J. King Saud Univ._Comput. Inf. Sci., vol. 32,no. 2, pp. 149_158, 2020, doi: 10.1016/j.jksuci.2018.01.003

6. S. H. H. Madni, M. S. AbdLatiff, M. Abdullahi, S. M. Abdulhamid, andM. J. Usman, ``Performance comparison of heuristic algorithms for taskscheduling in IaaS cloud computing environment," PLoS ONE, vol. 12,no. 5, May 2017, Art. no. e0176321, doi: 10.1371/journal.pone.0176321
7. B. Singh and G. Singh, ``A study on virtualization and hypervisor in cloudcomputing," Int. J. Comput. Sci. Mobile Appl., vol. 6, no. 1, pp. 17_22,2018.
8. M. Kumar and S. C. Sharma, ``Dynamic load balancing algorithm tominimize the makespan time and utilize the resources effectively in cloudenvironment," Int. J. Comput. Appl., vol. 42, no. 1, pp. 108_117, Jan. 2020,doi: 10.1080/1206212X.2017.1404823
9. A. Jyoti, M. Shrimali, and R. Mishra, ``Cloud computing and load balanc-ing in cloud computing -survey," in Proc. 9th Int. Conf. Cloud Comput.,Data Sci. Eng. (Con_uence), Jan. 2019, pp. 51_55, doi: 10.1109/con_u-ence.2019.8776948
10. Mohit Kumar, S.C.Sharma, "Deadline constrianen based dynamic load balancing algorithm with elasticity in cloud environment.," Computer and Electrical Engineering, vol. 64, pp. 395-411, July 2018.
11. A. K. G. M. D'silva, Gaurav and S. Bari, "Real-time processing of IoT events with historic data using Apache Kafka and Apache Spark with dashing framework," 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, 2017, pp. 1804- 1809.
12. C. M. Zaharia M, Das T, Dave A. Fast and interactive analytics over Hadoop data with Spark. USENIX and L. 2012;37(4):45–51.