RESEARCH REVIEW JOURNALS
Volume-04
Issue-06
June-2019

ISSN: 2455-3085 (Online)
RESEARCH REVIEW International Journal of Multidisciplinary
www.rrjournals.com[UGC Listed Journal]

Performance Centric Model for Resource Sharing in Cloud

Joshi Narayan

Professor, Department of MCA, Dharmsinh Desai University, Nadiad, Gujarat (India)

ARTICLE DETAILS

Article History
Published Online: 12 June 2019

Keywords
cloud computing, virtualization, resource allocation, load balancing, cloud collaboration.

***Corresponding Author**
Email: narayan.joshi[at]yahoo.com

ABSTRACT

Progressions in storage and communication technologies have encouraged usage of distributed computing technologies such as cloud computing. Moreover, implementation and consumption of collaborated cloud computing based services also is increasing day by day. There is very high requirement of efficient resource management and utilization at service provider and consumer sides in collaborated cloud computing environments. An efficient technique for workload balancing in collaborated cloud computing environments is suggested here. The technique works on resource management in both intra cloud and inter cloud computing environments for reducing waiting time for load balancing. The technique yields significant reduction in waiting time.

1. Introduction

Day by day, cloud computing technology is getting more and more attraction from nearly all sectors of IT industry. Not only IT industry but also large number of IT enabled service providers and end users also have started to bank upon the cloud computing based different types of services. Availability of cost effective and advanced computing power, progressive storage and communication technology are some of the many driving forces behind exponential growth of the cloud computing technology.

Reliance on cloud based solutions helps the individual users and organizations to bank upon the remote cloud infrastructure and thereby stay away from procurement of on premise costly computing infrastructure. It also saves cloud consumers from maintenance cost of on premise high-end servers and issues such as software licensing and updates.

However, increasing demand of various cloud computing based services requires efficient resource allocation and management failure in which may affect the quality of service level agreements. In order to keep the qualitative execution of service level agreements cloud service providers often join hands for formation of collaborated cloud computing environments. Such cooperative cloud computing platforms help in not only maintaining 24X7 availability but also offering fault tolerance.

Availability of cooperative cloud computing infrastructure can be utilized for efficient resource management by means of inter cloud load balancing. A novel load balancing approach for collaborated cloud computing environments is presented here. The technique presented here enables load balancing among overloaded nodes and under loaded nodes in cloud computing platforms.

Related work in domain of cloud load balancing is presented in section 2. The suggested load balancing mechanism is described in section 3. Section 4 represents experiment and results. Section 5 represents concluding remarks.

2. Related Work

With aid of genetic algorithm and fuzzy theory, S. Javanmardi et. al. have presented a hybrid job scheduling approach. The approach considers cloud load balancing and reduces total execution time and execution cost by modifying the genetic algorithm. The mechanism assigns jobs to resources with considering the VM MIPS and job length [8]. I. Gupta et. al. have proposed a hybrid meta-heuristic approach for workflow scheduling for IaaS cloud load balancing. The algorithm is based on hybridization of genetic algorithm and particle swarm optimization. The algorithm takes advantages of both the algorithms by avoiding slower convergence rate of GA and local optimum problem in PSO [5]. A new VM load balancing algorithm suggested for an IaaS framework by A. Manasrah. The technique is implemented in a virtual machine environment of cloud analyst to achieve better performance in terms of response and processing time [1].

R. Awatif et. al. have proposed an enhanced load balancing strategy by means of ensuring efficient response time with reduced cost. The suggested work is based on simulation work carried out within Cloud Analyst [9]. A. Tripathi et. al. have proposed a load balancing technique to form a new ACO method. The hybrid algorithm is combined with the setting of other parameters of the upgraded bee colony algorithm to form a new ACO method. The technique uses the Cloud Analyst [3]. N. Joshi proposed a dynamic technique for load balancing. The technique is based on job relocation among virtual machines having varying workloads [6]. A load balancing technique is suggested by S. Vasudeven et. al. The honeybee theory based technique assigns the available cloud resources to network for bringing down the service makespan [10].

For efficient task allocation, a Hybrid GA-PSO algorithm is proposed by A. Manasrah [2]. The technique is based on Hybrid GA-PSO algorithm for reducing makespan and cost. The technique aims to load balance the dependent tasks over heterogonous resources in cloud computing environments. H. Ji et. al. have proposed adaptive priority based workflow scheduling technique based on multi objective scheduling with varying objective weights to self-regulate task priorities to adapt to different objectives [4].

Few of the techniques are based on reducing response time. On other hand some techniques are experimented on simulated environment such as cloud analyst. Some of the techniques tackle the load balancing issue by dealing with the genetic algorithm approach whereas few techniques work on customizing the ant colony and bees algorithms.

3. Mechanism

The architecture of proposed resource management technique and relevant modules is shown in figure 1. The resource sharing technique presented here extends the work available in [6] by means of providing an add-on local load balancer to the centralized load balancer.

In addition to the centrally available inter-cloud load balancer `InterLB`, each of the participating cloud environment is equipped with a local load balancer also known as intra cloud load balancer, `IntraLB`. The `IntraLB` scrutinizes the load balancing request initially and tries to satisfy the request within the home cloud environment first. However, in case of non-availability of suitable destination virtual machine within home environment at required time, local load balancer `IntraLB` forwards request to the centralized load balancer `InterLB`. The improved algorithm mentioned here is based on the work suggested in [6] and [7] in domain of collaborated cloud computing.

In order to protect the under loaded virtual machines from spontaneous over burdening the mechanism maintains the `UNDERLOADED_PASSIVE` state. Similarly, the virtual machine state `OVERLOADED_PASSIVE` is used to protect from spontaneous low in work load. Some of the important steps of the suggested mechanism are mentioned here.

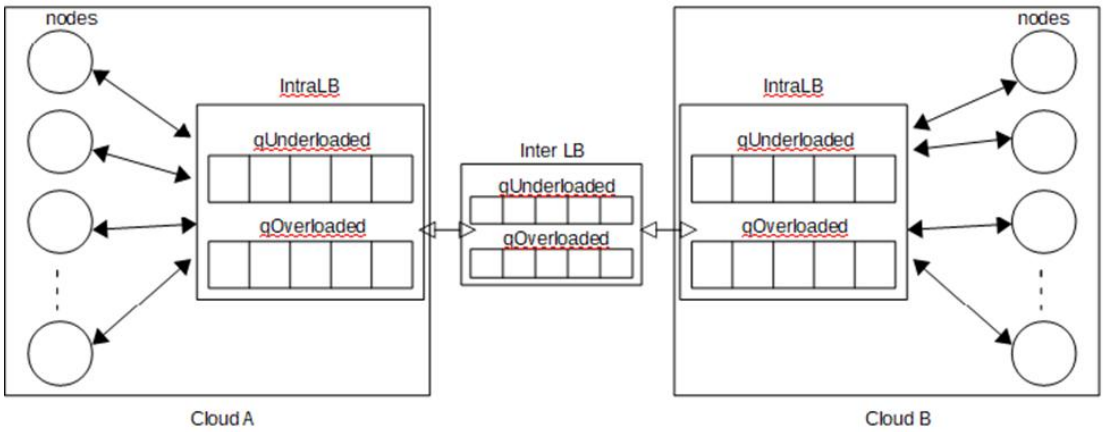

Figure 1: Block diagram for Inter & Intra cloud load balancing

1. The module `IntraLB` maintains two queues: `qUnderloaded` and `qOverloaded` for maintaining track of under loaded virtual machines and overloaded virtual machines respectively.
2. The queue `qUnderloaded` is handled by a thread `thread_UnderloadedVM`. The `thread_UnderloadedVM` pops information about under loaded virtual machines into `qUnderloaded`. In same way, the queue `qOverloaded` is handled by a thread `thread_OverloadedVM`. The `thread_OverloadedVM` pops information about overloaded virtual machines into `qOverloaded`.
3. Moreover, for the purpose of inter cloud load balancing, the module `InterLB` also maintains its own two global queues `qUnderloaded` and `qOverloaded`. The `qUnderloaded` contains information about underloaded virtual machines from all the cloud platforms who are participating in the inter cloud load balancing. Likewise, the `qOverloaded` contains information about overloaded virtual machines from all the cloud platforms who are participating in the inter cloud load balancing.
4. The module `IntraLB` handles two more threads of execution: `IntraLB.thread_LoadBalancer` and `IntraLB.thread_ManageState`.
5. The continuously running thread `IntraLB.thread_LoadBalancer` keeps on fetching information about over loaded virtual machines and under loaded virtual machines from `IntraLB.qOverloaded` and `IntraLB.qUnderloaded` respectively. The thread simply remains idle, if there are no more overloaded virtual machines in local cloud. Otherwise, the thread looks for availability of some under loaded virtual machine in order to carry out load balancing. In such a case, the thread toggles both machines to `PASSIVE` state and removes their information from respective queues. However, in case of unavailability of under loaded virtual machine, the thread forwards the load balancing request to `interLB` module by adding the overloaded machine's information to the global queue `InterLB.qOverloaded`.
6. On other hand, the continuously running thread `threadManageState` keeps watching virtual machines' state in both local queues. The thread resets machine state as `ACTIVE` if time duration since when the machine is in `PASSIVE` state is more than the maximum time duration.
7. Likewise, the centralized module `InterLB` also handles its own thread of execution: `InterLB.thread_LoadBalancer`.
8. At the central level, the global thread `InterLB.thread_LoadBalancer` keeps on fetching information about overloaded virtual machines and under loaded virtual machines from the queue

`InterLB.qOverloaded` and `InterLB.qUnderloaded` respectively. The central load balancing thread executes work load shifting if the information about overloaded and under loaded virtual machines is available. In such a case, the thread removes such participating machines' information from their respective central queues.

Some of the foremost statements of the suggested improved algorithm are mentioned here:

```
IntraLB::thread_IntraLoadBalancer
{
	...
	while(true)
	{
		Fetch overloaded virtual machine from local qOverloaded;
		Continue loop if there are no more overloaded VMs available;

		Continue loop if
			the overloaded VM passive time exceeds maximum threshold or
			the VM is no more overloaded or
			the VM is already passive

		Fetch under loaded virtual machine from local qUnderloaded;
		If there is no such under loaded VM available in local cloud
			& if the local cloud is participating with InterLB
		then append overloaded VM's information into central load
			balancer InterLB;
			update state of overloaded VM
		else
			load balance from overloaded VM to under loaded VM;
			toggle state of overloaded and underloaded virutal
			machines to PASSIVE;
		...
	}
	...
};

InterLB::thread_InterLoadBalancer
{
	...
	while(true)
	{
		Fetch overloaded virtual machine from global qOverloaded;
		Continue loop if there are no more overloaded VMs available;

		Continue loop if
			the VM is no more overloaded or
			the VM is already passive

		Fetch under loaded virtual machine from global qUnderloaded;
		Continue loop if there are no more underloaded VMs available;

		Load balance from overloaded VM to under loaded VM;
		Acknowledge receipt to respective local cloud platforms;
		...
	}
	...
};
```


4. Experimental observations

In local cloud load balancing environments, for reducing workload of overloaded virtual machines, availability of appropriate destination virtual machines is required. Unless the destination virtual machine is available in local cloud, it is not be possible to carry out further load balancing task.

Therefore, the overloaded virtual machine is required to wait in a queue for local load balancing until appropriate destination is not available.

Vice versa, the presence of global load balancer InterLB offers wider load balancing scope to overloaded virtual machines and thereby helps in bringing down the waiting time for load balancing. The load balancing environment is set with two private cloud platforms, each platform equipped with medium scaled virtual machines. Compared to the local cloud load balancing, there is 13% reduction in average waiting time in the global cloud load balancing.

5. Conclusion

A load balancing technique for collaborating cloud computing environments is suggested here. The technique works by running load balancing at global and local cloud level. The presence of centralized global load balancer, helps in bringing down the waiting time for load balancing workload of overloaded virtual machines. In future, the algorithm may be improved for load balancing cases such as spontaneous availability of under loaded virtual machine in local cloud environment. Moreover, the work may be further extended for studying security aspects. There is a scope for study on commercial aspects for the suggested global load balancing for collaborated cloud platforms.

References

1. A. Manasrah, Dynamic weighted VM load balancing for cloud-analyst, International Journal of Information and Computer Security, Vol. 9, Issue 1-2, 2017
2. A. Manasrah and H. Ali, Workflow Scheduling Using Hybrid GA-PSO Algorithm in Cloud Computing, Wireless Communications and Mobile Computing, Vol. 2018
3. A. Tripathi, S. Shukla and D. Arora, A Hybrid Optimization Approach for Load Balancing in Cloud Computing, Advances in Computer and Computational Sciences, 2017
4. H. Ji, W. Bao, and X. Zhu, Adaptive workflow scheduling for diverse objectives in cloud environments, Transactions on Emerging Telecommunications Technologies, Vol. 28, no. 2, 2017
5. I. Gupta, S. Gupta, A. Choudhary and P. Jana, A Hybrid Meta-heuristic Approach for Load Balanced Workflow Scheduling in IaaS Cloud, International Conference on Distributed Computing and Internet Technology, 2019
6. N. Joshi, Dynamic Load Balancing in Cloud Computing Environments, International Journal of Advanced Research in Engineering and Technology, Vol. 5, Issue 10, October 2014
7. N. Joshi, Dynamic Load Balancing in Cloud Computing Platform, International Journal of Computer Engineering and Technology, Vol. 10, Issue 3, May 2019
8. S. Javanmardi, M. Shojafar, D. Amendola, N. Cordeschi, H. Liu and A. Abraham, Hybrid Job Scheduling Algorithm for Cloud Computing Environment, Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA, 2014
9. R. Awatif,, E. Omri Amina, A. Nouredine, M. Khalid and R. Mohammed, A Performed Load Balancing Algorithm for Public Cloud Computing Using Ant Colony Optimization, Recent Patents on Computer Science, Vol. 11, Number 3
10. S. Vasudevan, S. Anandaram, A. Menon, A. Aravinth, A novel improved honey bee based load balancing technique in cloud computing environment, Asian Journal of Information Technology, Vol. 15, Issue 9, 2016

