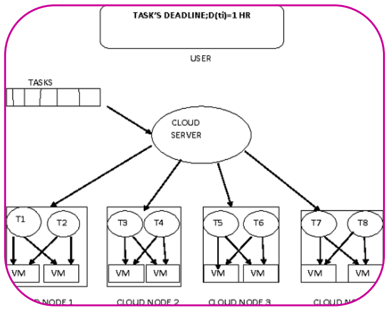
REVIEW OF RESEARCH
IMPACT FACTOR : 5.7631(UIF) UGC APPROVED JOURNAL NO. 48514 ISSN: 2249-894X
VOLUME - 8 | ISSUE - 1 | OCTOBER - 2018

ERROR-TOLERANT RESOURCE ALLOCATION AND PAYMENT MINIMIZATION IN CLOUD SYSTEM

Roopa R. L.
Department of Computer Science and Engineering, PDA College of Engineering, Gulbarga.

ABSTRACT



Resources in cloud system can be partitioned and provisioned on demand to different users with virtual machine technology. Sharing of resources among different users can lead to performance inaccuracies due to which efficient resource allocation and guaranteeing tasks execution is must in cloud systems. Different resources and services have different payment and time frame demand in cloud. With virtual machine (VM) technology being increasingly mature, compute resources in cloud systems can be partitioned in fine granularity and allocated on demand. In this work a solution which combines two techniques in order to guarantee execution of tasks within specified time deadline and try to minimize cost by using cost threshold. Equally spread current execution load and optimise response time these are the two techniques used to implement the system. A toolkit called CloudSim is used to support both system and behavior modeling of Cloud System components such as data centers, virtual machines (VMs) and resource provisioning policies. It implements generic application provisioning techniques that can be extended with ease and limited effort. In future, the proposed work can be incorporated with stricter/original deadlines into some excellent management tools like OpenNebula, for maximizing the system-wide performance.

More complex scheduling constraints like the compatibility, security issue and different task categories can be considered to achieve higher performance.

KEYWORDS: Cloud computing, equally spread current execution load, and optimize response time, resource allocation.

1. INTRODUCTION

Cloud computing is a distributed computing paradigm that focuses on providing a wide range of users with distributed access to scalable, virtualized hardware and/or software infrastructure over the internet. All the resources provisioned by the Cloud system are supposed to be under a payment model, in order to avoid users over demand of their resources against their true needs. Each task's workload is likely of multiple dimensions. First, the computer resources in need may be multiattribute (such as CPU, disk-reading speed, network bandwidth, etc), resulting in multi-dimensional execution in nature .Second, even though a task just depends on the resource type like CPU, it may also be split to multiple sequential execution phases, each calling for a different computing ability and various price on demand , also leading to a potentially high-dimensional execution scenario. As the cloud is made up of datacenters; which are very much powerful to handle large numbers of users still then the essentiality of efficient resource allocation and guaranteed execution of users tasks is must. Hence in this paper we deisgn a system using two techniques called optimize response time and equally spread current execution based on the Load balancing .Since load balancing is a technique of distributing the loads among the various nodes of a distributed system to minimize the response time, minimize the cost, minimize the resource utilization , and minimize the overhead.

1.1 Literature survey

In [1], survey on cloud computing and its characteristics is done. Cloud Computing is a "buzz word" en-compassing a wide variety of aspects such as deployment, load balancing, provisioning, and data and processing out-sourcing.

Clouds are generally used in three different scenarios, infrastructure as a service, software as a service and platform as a service.

In [2], survey on virtual machines is done. Virtualization, in computing, is the creation of a virtual version of something, such as a hardware platform, operating system, and a storage device or network resources.VM live migration is a widely used technique for dynamic resource allocation in a virtualized environment. The process of running two or more logical computer system so on one set of physical hardware. Dynamic placement of virtual servers to minimize SLA violations

In [3], survey on performance isolation mechanisms in Xen, a popular open source VMM. Xen supports per-VM CPU allocation mechanisms. However, it like many other VMMs—does not accurately account for resource consumption in the hypervisor on behalf of individual VMs, e.g., for I/O processing. The key contribution of this paper is the design of a set of cooperating mechanisms to effectively control total CPU consumption across virtual machines in Xen

2. SYSTEM DESIGN

2.1 Proposed Solution

The main aim of the proposed work is to provide efficient resource allocaton and to guarantee tasks execution within specified deadline and minimize cost for resource consumption. Therefore , two techniques optimize response time and equally spread current execution load techniques based on load balancing are proposed. With the help of these two techniques execution times are within specified deadlines is shown in the simulation results**.**

2.2 Working of Equally Spread Current Execution load technique

This technique is also called as equally spread current execution load balancing. It uses active monitoring load balancer for equally spreading the execution of loads on different virtual machines. Active monitoring load balancer (AMLB) maintains an index table of virtual machines and the number of allocations assigned to each virtual machine. Data Center Controller receives a new request from a client. When a request for allocation of new VM from Data Center Controller arrives at AMLB, it parses the index table from top until the least loaded VM is found. When it finds, it returns the VM id to the Data Center Controller. If there is more than one found, AMLB uses first come first serve (FCFS) basis to choose the least loaded. Simultaneously, it also returns the VM id to the Data Center Controller. Then the Data Center communicates the VM identified by that id. The Data Center Controller notifies the AMLB about the new allocation. After that AMLB updates the allocation table by increasing the allocation count by 1 for that VM. When a VM suitably finishes processing the assigned request, it forwards a response to the Data Center Controller. On receiving the response it notifies the AMLB about the VM de-allocation. The AMLB updates the allocation table by decreasing the allocation count for that VM by 1.

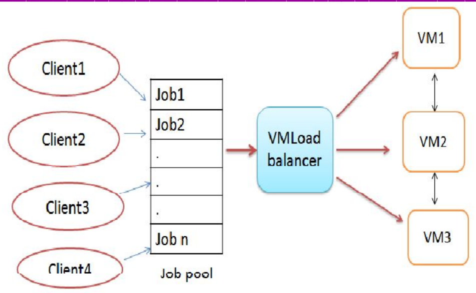


Figure1. Equally spread current execution load

EQUALLY SPREAD CURRENT EXECUTION LOAD

1. Find the next available VM
2. Check for all current allocation count is less than max length of VM list allocate the VM
3. If available VM is not allocated create a new one
4. Count the active load on each VM
5. Return the id of those VM which is having least load.
6. The VMLoadBalancer will allocate the request to one of the VM.
7. If a VM is overloaded then the VMLoadBalancer will distribute some of its work to the VM having least work so that every VM is equally loaded.
8. The datacenter controller receives the response to the request sent and then allocate the waiting requests from the job pool/queue to the available VM & so on.
9. Continue from step-2.

2.3 Optimise response time policy

As per this strategy, the data center selection is not made randomly and vm cost in each data center is compared with other datacenters in the same region. The data center with lowest vm cost is selected. Now the requests will be sent to this data center to be processed.

This strategy gives cost effective user request routing.

Algorithm: Data Center Selection (proposed)

```
01: Get the datacenter index of selected region
02: regionaList regionalDataCenterIndex.get (region) // store regionaList of selected datacentre
03: if regionaList is not NULL then
04: listSize size (regionalist)
05: if listSize is 1 then
06: dcName regionalist.get(0)
07: else
08: for all p in dcVMCostList do
09: if(dcVmCostList.get (smallest)>dcVmCostList.get (p)) then
10: smallest=p;
11: end if
12: end for
13: dcName regionalist.get (smallest)
14: end if
15: end if
16: return dcname
```

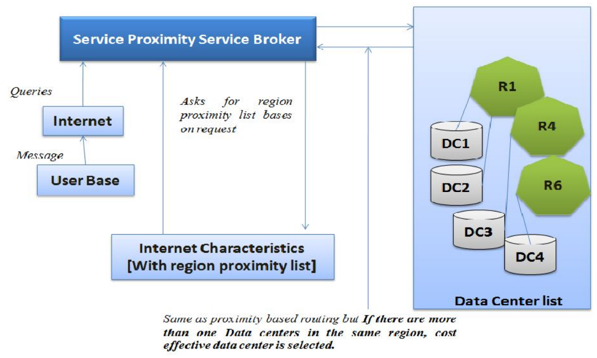

Service Proximity Service Broker
Queries
Internet
Message
User Base
Asks for region proximity list bases on request
Internet Characteristics [With region proximity list]
R1
R4
R6
DC1
DC2
DC3
DC4
Data Center list
Same as proximity based routing but If there are more than one Data centers in the same region, cost effective data center is selected.

Figure2. Optimise response time

2.4 Response Time Calculation

We use the following formula for calculation

Response Time = Fint - Arrt + Tdelay

where,

Arrt is the arrival time of user request

Fint is the finish time of user request

Tdalay is the transmission delay

However, Tdelay can be calculated as,

Tdelay = Tlatency + Ttransfer

Here,

Tlatency is the network latency

Ttransfer is the time taken to transfer the size of data of a single request (D) from source location to destination.Tlatency is taken from the latency matrix (after applying Poisson distribution on it for distributing it) held in the internet characteristics.

Ttransfer = D / Bwperuser

Where

Bwperuser = Bwtotal / N

Bwtotal is the total available bandwidth (held in the internet characteristics) and N is the number of user requests currently in transmission. The internet characteristics also keep track of the number of user requests in-flight between two regions for the value of N.

3 . Simulation Results

Following table shows various parameters used and their respective values

A. Simulation Parameters

Table1. Simulation parameters

Parameters	Values
User base	2-40
Simulation time	6 hours
Data centre	2-40
VM/ Data centre	5-10
Cost threshold	0.01-5 $/hr
Time threshold	140-150 sec

Load balancing technique	Equally spread current execution load
Scheduling technique	Optimise response time

A) Performance metrics

The Performance metrics is used to measure the simulation of the work are explained below.

➢ Execution time

Execution time is defined as the amount of duration taken by the datacenter to complete executing all submitted tasks from different user bases.

It includes individual datacenter processing time and overall response times.

➢ Cost

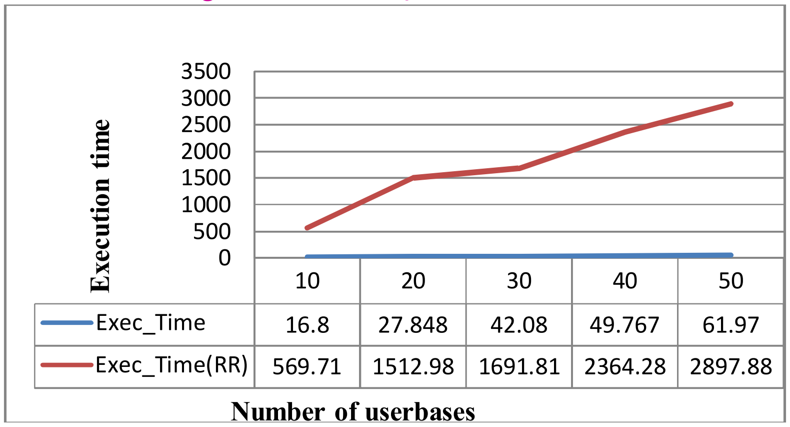
Cost is the amount to be payed by users for consumption of resources in order to get the jobs executed by the datacenters. These costs are defined by payment models.

Cost = storage cost + transfer cost (RAM to hard disk) + processing cost/instruction + network cost

C. Simulation results

In this work we have developed a cloud network model in CloudSim to incorporate core middleware (PaaS) layer fundamentals of execution management, pricing, metering, into the simulation to realistically analyse the behaviour of this network and concept. We compare the performance of round robin and equally spread current execution load for various parameters as shown in the graph.

Figure3. Userbase v/s execution time

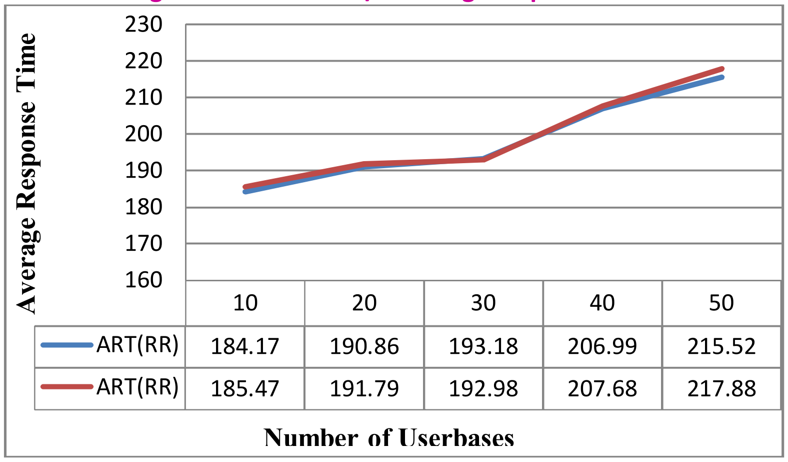


	10	20	30	40	50
Exec_Time	16.8	27.848	42.08	49.767	61.97
Exec_Time(RR)	569.71	1512.98	1691.81	2364.28	2897.88

Figure3. illustrates that as the number of Userbase increase the execution time also increases.

But the same number of tasks submitted will require more time to get completed with the round robin technique than compared to the equally spread current execution load. Hence it is concluded that with the proposed system high number of tasks are executed within minimised time and deadline.

Figure4. Userbase v/s Average response time



	10	20	30	40	50
ART(RR)	184.17	190.86	193.18	206.99	215.52
ART(RR)	185.47	191.79	192.98	207.68	217.88

Figure4 illustrates the average response time for tasks is high for same number of tasks for round robin technique compared to the optimised response time technique.

CONCLUSION

The response time and data transfer cost is a challenge of every engineer to develop the products that can increase the business performance in the cloud based sector. The several strategies lack efficient scheduling and load balancing resource allocation techniques leading to increased operational cost and give customer satisfaction. This work aims to development of error tolerant resource allocation strategy through improved job and load balancing resource allocation techniques. Equally spread current execution algorithm dynamically allocates the resource to the job in a queue leading reduced cost in data transfer and virtual machine formation. Optimize response time service broker policy tries to minimize the response time for user jobs by selecting suitable datacentre for user requests. This improves the business performance and retention to the total customer satisfaction.

REFERENCES

[1] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner,"A break in the clouds: towards a cloud definition," *SIGCOMM Comput. Commun. Rev*, vol. 39, no. 1, pp 50–55, 2009.
[2] J. E. Smith and R. Nair, *Virtual Machines: Versatile Platforms For Systems And Processes*. Morgan Kaufmann, 2005.
[3] D. Gupta, L. Cherkasova, R. Gardner, and A. Vahdat, "Enforcing performance isolation across virtual machines in xen," in *Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware (Middleware'06)*, New York, USA, 2006, pp. 342–362.
[4] Amazon elastic compute cloud: on line at http://aws.amazon.com/ec2/.
[5] L. Huang, J. Jia, B. Yu, B.G. Chun, P. Maniatis, and M. Naik, "Predicting Execution Time of Computer Programs Using Sparse Polynomial Regression," in *24th Conference on Neural Information Processing Systems (NIPS'10)*. 2010, pp. 1–9.
[6] V. Petrucci, O. Loques, and D. Moss´e, "A dynamic optimization model for power and performance management of virtualized clusters," in *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking (e-Energy'10)*. New York, NY, USA: ACM, 2010, pp. 225–233.
[7] X. Meng, C. Isci, J. Kephart, L. Zhang, E. Bouillet, and D. Pendarakis, "Efficient resource provisioning in compute clouds via vm multiplexing," in *Proceeding of the 7th international conference on Autonomic computing (ICAC'10)*. ACM, 2010, pp. 11–20.

[8] M. Mao, J. Li, and M. Humphrey, "Cloud auto-scaling with deadline and budget constraints," in *11th IEEE/ACM International Conference on Grid Computing (Grid'10)*, 2010, pp. 41–48.
[9] Y. Wu, K. Hwang, Y. Yuan, and W. Zheng, "Adaptive workload prediction of grid performance in confidence windows," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 7, pp. 925 –938, july 2010.
[10] F. Chang, J. Ren, and R. Viswanathan, "Optimal resource allocation in clouds," *IEEE International Conference on Cloud Computing*, pp. 418–425, 2010.