A Novel Cloud Partitioning Algorithm Using Load Balancing In Public Cloud Environment

Suhas Chitturi, Surya Teja Marella, Swaroop Chitturi , Sk.Hasane Ahammad

Abstract: ***Cloud computing is a growing technology which is termed basically as use of various services like storage, servers, software and other software development platforms over the internet, often referred as the term "cloud", which is used to provide different resources in order to perform complicated tasks. As we know in present scenario, the traffic on the internet is growing rapidly and the number of services to be offered to the users is increasing due to the rapid increase in popularity of these cloud services. This results in the increase workload on the server ultimately slowing down the process and leading to overloading of the servers. Thus, in order to prevent this, load balancing is used. [1]A proper load balancer results in cloud computing services to be efficient and also leads to positive user satisfaction. The paper aims to provide a load balance model related to the concept of cloud partitioning with the help of a switch mechanism which gives a choice to switch various strategies for different situations which is to be used for the public cloud. The concept of game theory is applied in this algorithm in order to improve the efficiency in public cloud environment.***

Keywords: Cloud Computing, Load Balancing, VM Scheduling

I. INTRODUCTION

Problem Statement: In present scenario, the traffic on the internet is growing rapidly and the number of services to be offered to the users is increasing due to the rapid increase in popularity of these cloud services. This results in the increase workload on the server ultimately slowing down the process and leading to overloading of the servers. Thus, in order to prevent this, the load balancing method is used. Hence the paper provides a new balancing model which not only follows the common way of balancing jobs but also automates it in order to make it more sufficient.

Overview: The Cloud computing term can be generally defines as anything that involves delivering of services that are hosted on the internet. It is a new paradigm in software development which involves smaller organizations having access to resources like storage, processing power and business processes that were once considered to be available to only large enterprises. As more and more enterprises consider the cloud services as their regular business resource go-to, more scalable and stable their services have to be. Due to loads of services and requests, it becomes difficult for the service providers to maintain the stability of the processing ultimately leading to server overload causing the server to crash, which basically affects the business part too. In order to prevent this, certain measures are to be taken to handle load and help in maintaining the stability of processing and continue the services. Hence, Load balancing comes to of great use. Load balancing in cloud is basically defined as the process of splitting the workloads and essential computing properties and able to manage the workload and application demands by distributing resources among different computers, servers or networks. It plays an important role in improving the system performance and also maintains its stability. Since here the arrival of the job pattern prediction cannot be done beforehand and the node capacities of each one of the cloud vary, so, for this case of load balancing, workload control is required. The schemes of load balancing depend on the system dynamics namely static and dynamic schemes. The Static scheme doesn't use any system related information and are far less complex whereas dynamic can change the system status but brings additional costs for the system. In this algorithm, here, the dynamic scheme is used for the flexibility nature. Many different load balancing algorithms [2] for private cloud are present and were even used before, but the balancing model used in this paper aims at the public cloud primarily as it has many nodes containing resources for distributed computing in various distinct locations. Here, the balancing model divides the cloud into several parts termed as cloud partitions and the environment when comparatively is large enough and complex, the partitions here makes the load balancing more intelligible. The model basically contains a basic controller that helps in choosing partitions which are suitable for the requests incoming or jobs while the model balancer of each partition selects the definitive required load balancing strategy.

Revised Manuscript Received on May 06, 2019

Suhas Chitturi Computer Science & Engineering, K L University, Guntur, India

Surya Teja Marella Computer Science & Engineering, K L University, Guntur, India

Swaroop Chitturi , Computer Science & Engineering, SRKR Engineering College, India

II. SYSTEM MODEL

As we know, the work done here is largely focussed on public cloud. This type of cloud service is basically based on basic paradigm cloud computing model. Since it consists of multiple nodes of different locations geographically, cloud partitioning is done here in order to manage the huge cloud. A cloud partition is basically a part within the cloud where each part belongs to a distinct location. After creating the partitions, load balancing process begins.

A. General Introduction:

As the task is given by the user and as it arrives to the system model, the system main controller selects the partitions which will get [3] the task or job. Then, the partition balancer assigns the jobs. If the status of load isn't normal, job is transferred in another partition else, partitioning is done locally.

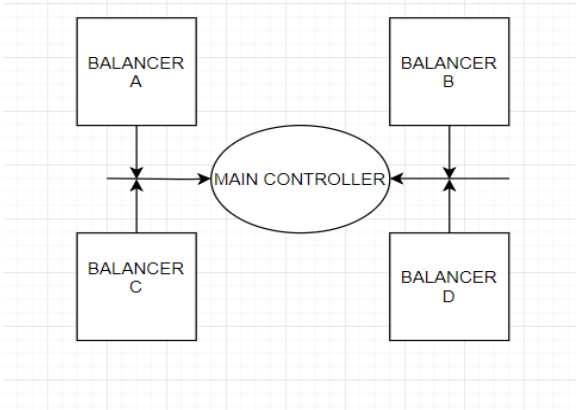


FIG 2.1: SHOWING RELATIONSHIP BETWEEN CONTROLLER MAIN AND BALANCERA, B, C, D

The controller here in the model, assigns jobs to cloud partitions and communicates with the system balancers present in every partition in order to get the updated status information. As the controller responds to the data for each partition, the data sets which are smaller results in achieving more high processing rates. The balancers in the partition gets the information of the present status from the nodes present in the model and selects the appropriate required strategy of distributing the tasks or jobs among partitions of the model. The relationship between system balancers and main controller is given below,

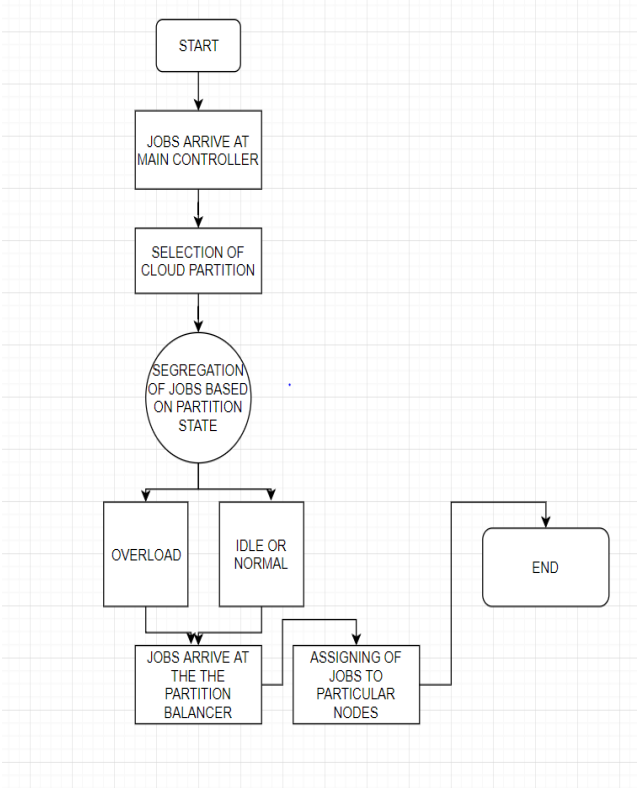


FIGURE 2.1: SHOWING WORKING ARCHITECTURE OF THE SYSTEM

B. Jobs assignment to partitions:

The working of model is as follows,

As the job reaches the public cloud, the right partition is chosen. Here, the status of cloud partition is identified which is basically divided into three types namely, idle, normal and overload. This is done by the main controller. The status is said to be idle, normal or overload if, the percentage of the particular node exceeds the given value given by cloud balancers. The main system controller transmits a request to system balancers to get the updated status information. If status is idle or normal, the partitioning or assignment of jobs is done normally else, the model finds for another partition which is not overloaded.

C. Assignment of the jobs to nodes:

As jobs arrive at the partition balancer, it finds out the load information i.e., the degree of load of every particular job. The degree of load consists of different parameters which include memory size, number of CPUs, processing speeds of CPUs etc and also parameters which are dynamic like, memory and utilization ratio of CPU, network bandwidth[4] etc. After computing of the degree, based on the values, node load status is given as follows:

If, degree of load=0, it is termed as idle status meaning no job is being processed at the moment.

If, 0<degree of load<degree of load(high), it is termed with status as normal and eligible to process other jobs too.

If, degree of load>degree of load(high), it is defined as overload state and receiving jobs is impossible till it becomes normal.

Here, degree of load(high) value is set value defined by the cloud partition balancer. As there are numerous partitions with load partition balancer present in each of them, tables are created by each cloud partition balancers based on the inputs of the load degree at particular time intervals say, T. Then, the tables are used by the partition balancers in order to figure out the partition status where the status derived show different solutions for balancing the load. As the job reaches the partition, the balancer here ascribes the task to the nodes depending upon the present strategy of the load.

III. STRATEGIES TO EXISTING SYSTEM

Several strategies can be applied to automate the system easily [5]. The main goal is to provide a good balancer for balancing of jobs as a good balancer improves the performance of the entire cloud. The use of common method for automation cannot be done for all the cases or situations as it is difficult to adapt. Here, the automation is done by using simple algorithm like Round Robin algorithm for the idle state and complex algorithm like game theory for normal state. Here, the notion is to integrate different methods but also switch between methods for load balance dependent on the given status of system model.

A. Strategy for load balancing in idle state:

As idle state indicates that, the resources for computation here are present idle and merely fewer jobs present have to be processed, so, the cloud partition can process jobs quickly [6], hence, a simple algorithm like Round Robin algorithm can be used here for balancing of jobs. As we know that in round robin algorithm, each and each node has the same chance to be selected for the transaction, but in cloud, since configurations of each node varies, hence improvised Round Robin algorithm is used here.

Working: After the creating of table by the balancers

using the degree of load values ranging from the least to the highest, the model creates a queue in circular and proceeds through the queue repeatedly and selects only those nodes which have lowest load degree value. The node order changes as the balancer refreshes the load status table. Since balance table is refreshed every now and then, this leads to inconsistency problem as the status of the system would be changed but the information still remains to be old. In order to debug this problem, separate tables are created namely, load status table i and load status table ii. A flag is used to show the operation read or write and the process goes as follows, as the flag shows 'read', the algorithm on the evaluation of the load degree uses the table and sd the flag shows 'write', then the table is presented with the updated values stored in the other table. Basically, at every moment. One of the tables gives the exact correct locations of nodes in queue for the process and the table prepares the updated values. When refreshed, the table flag interchanges and the node locations are refreshed. This method solves the problem of inconsistency here.

B. Load balancing strategy in normal state:

As status of partition is termed normal, the job arrives faster when compared to idle state. As this is very complex, for this problem, another algorithm is used for load balancing here. In order to provide reasonable response time for the job completion, game theory algorithm is used here.

Game theory [8] basically is a static load balancing strategy for distributed systems which consists of two modes, co-operative mode and non-co-operative mode. In the cooperative mode, the makers of the decision reach to a final agreement. Each maker agrees to a common point or objective by the comparison of notes done with each other. In the non-cooperative mode, the decision maker agrees to only decisions made by himself and thought and for the own benefit [9]. Here, the system after reaching Nash equilibrium, gives the final result. Here the Nash equilibrium, which is basically a condition where each decision maker makes his own, optimized solution or decision. It is basically a condition where each player in game selects a particular strategy remembering not any player here may benefit with changing their own strategy as other strategies of the players are not changed at that particular moment.

Working: In this model, nodes and the jobs are the players of the game. Several parameters are then defined according to the given values. Parameters like, each node processing ability, time spent in completion of particular job, time spent by the entire cloud partition[10]. The job assigned fraction value to given node is computed using these parameters. Once computed for a single node, a greedy algorithm is used to compute the values of other nodes. This result provides the Nash equilibrium a boost in minimizing the time of response for each job. This strategy is subjected to change as node's status change.

By including these strategies, the automation becomes more simplified and make the system much more efficient and less time consuming in doing the partition of the job in the public cloud and better than other models which are only meant for processing in private cloud. [11]

IV. SCHEDULING ALGORITHM

Input: List of Physical Machines, List Of Virtual Machines
Output: Schedule of VMs

i. Compute of Optimal Performance Power Ratio of all the Physical Machines in the list and sort the PM_List in the decreasing order of ratios.

ii. For every time period (t),
Check the VM List for VMs machines to be allocated in that time period.

For every VM to be allocated in (t) Check for idle cores in PM_List If found:
Allocate the VM to that core.
Start the VM execution and set VM.start_time=t
Add(VM.PM,Core) to Allocation List
Else:
Check in the next PM and continue until the VM is allocated Or no more PMs are left.

If a VM is not allocated successfully:

Update the required resource for VM (VM.rr)

Add the VM to the head of VM List to allocate in the next time Period.

iii. After allocating VMs in (t), set the Optimal Frequency for each active core using the number of active cores in the PM.
For every (VM,PM,Core) in Allocation List:
If VM has completed its execution:
Make this core inactive
If all the cores in the PM are inactive: Set PM to Sleep State.
Remove(VM.PM,Core) from Allocation List.

iv. For each (VM,PM,Core) in Allocation List : Check for PMs with higher oppr than PM.
If found: Check if any of its cores are idle or has enough resource for VM
If Found: Migrate the VM to the new PM and Core.
Remove (VM,PM,Core) from Allocation List.
Add (VM.PM',Core') to allocation List.
Repeat from step 2 until all the VM's are either done or failed.

V. DEMONSTRATION

A sample test case is chosen to demonstrate the scheduling process of the proposed algorithm. It includes a set of 5 heterogeneous physical machines and 30 virtual machines that arrive at the cloud in a particular time with a deadline constraint. Each time period is set to be 1 second and the VM's are allocated a PM core at the end of each time period. The VM's that could not be allocated in its time period are left to be allocated in

the next period and are given the first priority. The parameters of the machines are as follows:

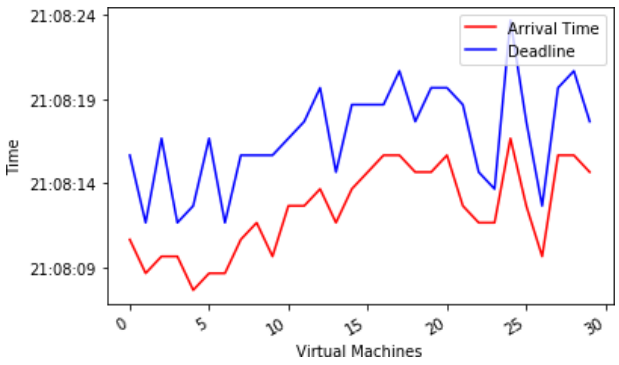
C	*S*	*F*	*U*	*I*	*P*
4	4	[2.4, 2.7, 3.0, 3.2]	63	47	148
2	4	[1.2, 1.5, 1.8, 2.2]	65	32	101
6	4	[1.5, 1.8, 2.3, 2.9]	73	55	172
4	4	[2.3, 2.7, 2.9, 3.2]	67	55	173
2	4	[1.5, 1.7, 2.3, 2.7]	79	51	159

Table: Physical Machines

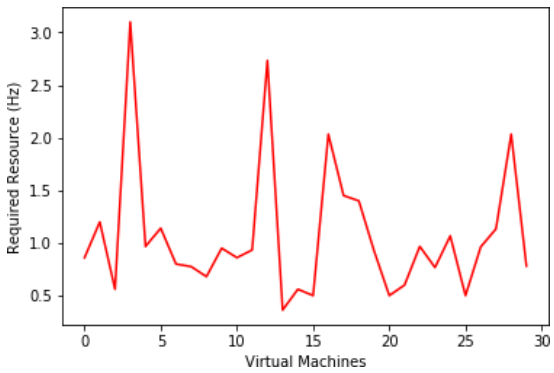
C= ***Number of Cores***
S= Number of States
F= Set of Frequencies (Hz)
U= CPU Power (W)
I = Idle Power (W)
P= Peak Power (W)

VI. PERFORMANCE RESULTS

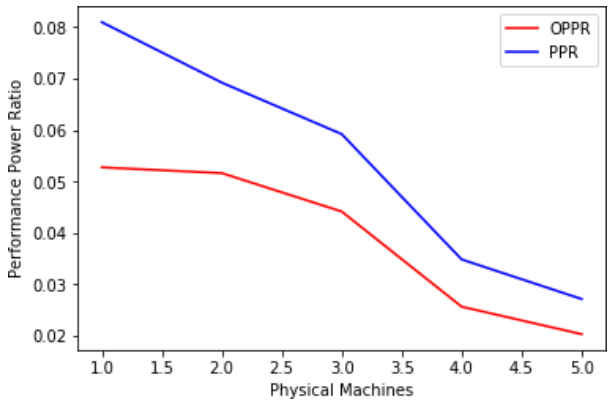
A. VM Vs Time



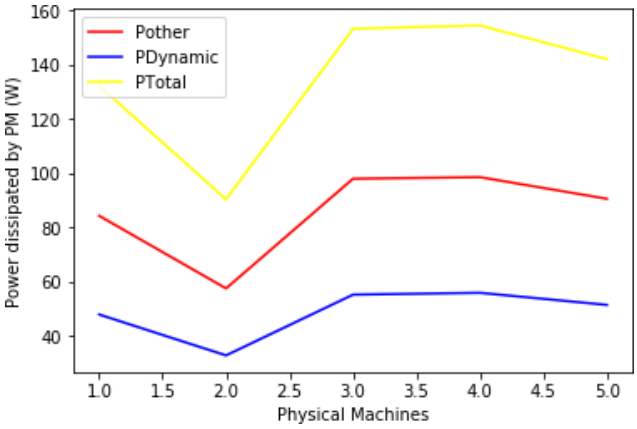
B. VM Vs RR(Required Recourses)



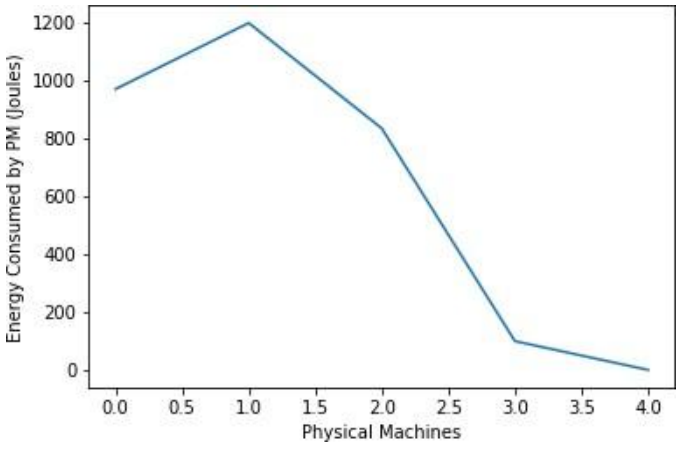
C. PM Vs PPR & OPPR(Optimal)



D. PM Vs Power Consumption



E. PM Vs Energy Consumption



VII. CONCLUSION

The biggest challenge faced while attempting to reduce the data center's electrical consumption is to link data center activities to electrical use. Data center consumption is based on IT and infrastructure loads or when cooling systems remove heat from the data center to keep the temperature optimal. The fuel or energy is used to generate electricity is the most significant factor affecting the year to year changes in CO_2 emissions. The proposed algorithm shows the reduced energy consumption of data center.

Figuring out ways to minimize the energy consumption of servers in the data centres is proving to be an open challenge for academic researchers and corporate industries. The main objective is to tackle the heterogeneity of physical machines, varying execution times of virtual machines, the power consumed by each physical machine. It is clear from the performance results that the proposed algorithm works better than the traditional first come first serve or round robin scheduling methods. This way the power consumed and the energy released can be drastically reduced. More virtual machines are able to compete their tack in a much faster time, ultimately minimizing the number of failed VMs (That could not be completed within their respective deadlines). Though some of the practical cases may not work as efficiently as expected, scheduling based on this algorithm can reduce the energy consumption significantly.

APPENDIX

I am thankful to the university and in particular to my research supervisor Dr. T Gunasekhar, for a wonderful few months of work. I hope that everything I have learned in this time will hold me in good stead for the future.

REFERENCES

1. G. Laszewski, L. Wang, A.J. Younge, X. He, Power-aware scheduling of virtual machines in DVFS-enabled clusters, in: IEEE International Conference on Cluster Computing and Workshops, 2009, pp. 1–10.
2. X. Fan, W.-D. Weber, L.A. Barroso, Power provisioning for a warehouse-sized computer, in: Proceedings of the 34th Annual International Symposium on Computer Architecture, 2007, pp. 13–23.
3. W. Lang, J.M. Patel, J.F. Naughton, On energy management, load balancing and replication, SIGMOD Rec. 38 (4) (2009) 35–42.
4. Zhen Xiao, Weijia Song, Qi Chen, Dynamic resource allocation using virtual machines for cloud computing environment, IEEE Trans. Parallel Distrib. Syst. 24 (6) (2013) 1107–1117.
5. S. Bazarbayev, M. Hiltunen, K. Josh, Content-based scheduling of virtual machines (VMs) in the cloud, in: International Conference on Distributed Computing Systems, 2013, pp. 93–101.
6. I. Hwang, M. Pedram, Hierarchical virtual machine consolidation in a cloud computing system, in: IEEE Sixth International Conference on Cloud Computing, 2013, pp. 196–203.
7. W. Vogels, Beyond server consolidation, Queue 6 (1) (2008) 20–26.
8. X. Li, Z. Qian, S. Lua, J. Wu, Energy efficient virtual machine placement algorithm with balanced and improved resource utilization in a data center,Math. Comput. Modelling 58 (5–6) (2013) 1222–1235.
9. G. Lovász, F. Niedermeier, H. Meer, Performance tradeoffs of energy-aware virtual machine consolidation, Cluster Comput. 16 (3) (2013) 481–496.
10. J. Dong, X. Jin, H. Wang, Y. Li, P. Zhang, S. Cheng, Energy-saving virtual machine placement in cloud data centers, in: IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2013, pp. 618–624.
11. N.Q. Hung, P.D. Nien, N.H. Nam, N.H. Tuong, N. Thoai, A genetic algorithm for power-aware virtual machine allocation in private cloud, in: International Conference on Information and Communication Technology, 2013, pp. 183–193.
12. X. Liao, H. Jin, H. Liu, Towards a green cluster through dynamic remapping of virtual machines, Future Gener. Comput. Syst. 28 (2) (2012) 469–477.
13. A. Beloglazov, J. Abawajy, R. Buyya, Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing, Future Gener. Comput. Syst. 28 (5) (2012) 755–768.
14. https://www.environmentalleader.com/2016/02/google-amazon-underestimate-data-centers-carbon-footprints/

AUTHORS PROFILE

Author-1 Photo

First Author personal profile which contains their education details, their publications, research work, membership, achievements, with photo that will be maximum 200-400 words.

Surya Teja Marella , Computer Science Graduate from KL University and very much interested in the concepts of Cloud Computing & network security.

Author-3 Photo

Third Author personal profile which contains their education details, their publications, research work, membership, achievements, with photo that will be maximum 200-400 words.