Linear Scheduling Strategy for Resource Allocation in Cloud Environment

Abirami S.P.[1] and Shalini Ramanathan[2]

[1]Department of Computer Science andEngineering, PSG College of Technology, Coimbatore.

abiramii.sp@gmail.com
sha.cse@gapps.psgtech.ac.in

ABSTRACT

Cloud computing technology virtualizes and offers many services across the network. It mainly aims at scalability, availability, throughput, and resource utilization. Emerging techniques focus on scalability and availability. However, cloud computing must be advanced to focus on resource utilization and resource management. The cloud environment, embedded with the nimbus and cumulus services will contribute more in making the responsibility of resource utilization in Cloud Computing.

Considering the processing time, resource utilization based on CPU usage, memory usage and throughput, the cloud environment with the service node to control all clients request, could provide maximum service to all clients. Scheduling the resource and tasks separately involves more waiting time and response time. A scheduling algorithm named as Linear Scheduling for Tasks and Resources (LSTR) is designed, which performs tasks and resources scheduling respectively. Here, the combination of Nimbus and Cumulus services are imported to a server node to establish the IaaS cloud environment and KVM/Xen virtualization along with LSTR scheduling is used to allocate resources which maximize the system throughput and resource utilization.

KEYWORDS

Cloud Computing; Resource Allocation; Resource Utilization, Virtualization; KVM.

1. INTRODUCTION

Cloud computing is a technology that numerous IT organizations extend their hands in order to improve their financial ability. This is done by improving the various QoS parameters such as performance, throughput, reliability, scalability, load balancing, persistence, etc,. The services such as disk storage, virtual servers, application design, development, testing environment are added advantages of the Cloud Computing technology.

The cloud computing technology makes the resource as a single point of access to the client and is implemented as pay per usage. Though there are various advantages in cloud computing such as prescribed and abstracted infrastructure, completely virtualized environment, equipped with dynamic infrastructure, pay per consumption, free of software and hardware installations, the major concern is the order in which the requests are satisfied. This evolves the scheduling of the

resources. This allocation of resources must be made efficiently that maximizes the system utilization and overall performance. Cloud computing is sold on demand on the basis of time constrains basically specified in minutes or hours. Thus scheduling should be made in such a way that the resource should be utilized.

Virtualization deals with the existence of the resources that are not physical. Many virtualization techniques have been evolved in which Nimbus deals with the Xen and KVM type of virtualization. Nimbus Platform is an integrated set of tools that deliver the power and versatility of infrastructure clouds to scientific users. Nimbus Platform combines the services offered by the Nimbus, OpenStack, Amazon, and other clouds. Nimbus Infrastructure is an open source EC2/S3-compatible Infrastructure-as-a-Service implementation specifically targeting features of interest to the scientific community such as support for proxy credentials, batch schedulers, best-effort allocations and others. Cumulus is an open source implementation of the S3 REST API. The Cumulus services also deploy the file system usage services from the client node over the configured central service node.

2. RELATED WORK

The system throughput and resource utilization is based on the grid middleware, evaluation model and the real time scheduler [1]. The virtualization overhead and the middleware overhead is examined by varying the types of virtualization and the number of processes respectively to evaluate the grid middleware utilization.

The evaluation model predicts the availability of the resources submitted to the cloud and the real time scheduler formulated by Hyukho Kim et al, deals with the resource scheduling and task scheduling based on the Shortest Task First and First In First Out resulted specifying that FIFO takes longer time. The results were based on the waiting time, response time and the processing time. The experiment was carried out in four different types of environments namely normal, Xen based, GT based and GTX based environments. The evaluation predicts that the results are purely based on the type of virtualization and the scheduler used.

The scheduling can be either static or dynamic. The static scheduling is a pre scheduled operation regardless of the request submitted. Dynamic scheduling depends and varies as the request changes. The task scheduling algorithm was designed as a combinatorial problem which did not converge at the global optimum [2]. Thus a scheduling algorithm to result in maximum resource utilization and high performance should be based on both the tasks and the resources to obtain global optimum solution.

There are various combinations of commonly used computing nodes that are used for the Cloud Computing technology [3]. The Nimbus and the Cumulus services are implemented in one such node (probably a dual core) to obtain the server configuration and few other nodes contribute to the client side. The table1 lists the set of commonly used computing nodes.

Table.1 Popular computing cores

ID	Processors	Memory	Hard disk capacity
0	1	256	10
1	1	512	15
2	1	1024	20
3	1	2048	20
4	2	1024	20
5	2	2048	40
6	4	2048	80
7	4	4098	160

Ad hoc parallel data processing is one of the major obstacles for the Infrastructure as a Service (IaaS) clouds [5]. Thus many cloud computing organizations have started to integrate the framework for parallel data processing, which brings easy access to the customers for using the services and for deploying their application codes. The processing of parallel frameworks focuses only on the static infrastructure rather than dynamic. Daniel Warneke et al evaluated the opportunities and the challenges in establishing dynamic resource allocation offered in the IaaS cloud environment.

Anirban Kundu et al evaluated the memory utilization in cloud computing based on transparency [7].Transparency is used as a virtualization in which the user is unaware of the origin of the resource unit, rather the request is satisfied as a single point resource satisfaction. The resource allocation is made based on the selection criteria which will improve the efficiency of the cloud environment. The memory manager is responsible for allocating memory resources to the clients. Through the use of virtualization and resource sharing based on time constrains, clouds serve with a single set of physical resources to a large user base demanding high requests [8]. Moreover, the use of virtualization and resource time sharing introduces significant performance penalties for the demanding scientific computing workloads. Thus clouds have the potential to provide to the owners, the benefits of an economy of scale in technologies like grids, utility and parallel production environments. Based on the requests submitted to the cloud environment, cloud performance changes directly that relies on the performance of the workloads and cost metrics.

The scheduling algorithms are termed as NP completeness problems in which FIFO scheduling is used by the master node to distribute resources to the waiting tasks. This master unit point outs the remaining tasks in the dispatcher queue when the execution of the current task completes [9]. This master node is responsible for scheduling and processing the entire process in the queue. This master node is termed as the service node in our evaluation.

An important notifying advantage of infrastructure-as-a-service (IaaS) clouds is that it provides users on-demand access to resources. To provide on-demand access, cloud providers must either

significantly overprovision their infrastructure such as paying a high price for operating resources with low utilization or to reject a large proportion of user requests in which case the access is no longer on-demand. At the same time, the important concept is that not all users require truly on-demand access to resources of IaaS. Many applications and workflows are designed for recoverable systems where interruptions in service are expected [10].

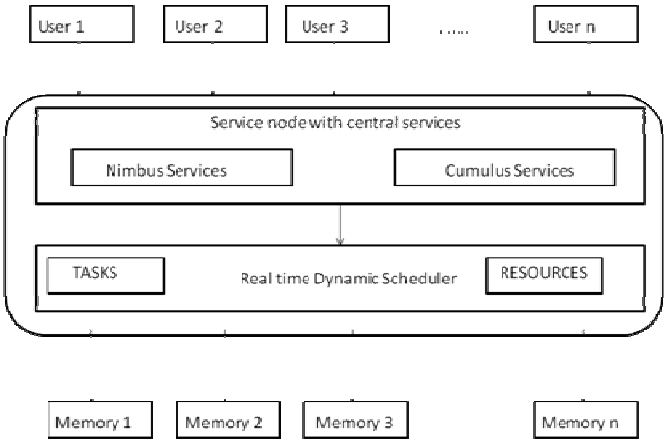
Resource consumption and resource allocation have to be integrated so as to improve the resource utilization. The percentage of resource used and the type of resource allocated protrudes directly to the resource utilization, ie., the number of request submitted and the availability of the VM's for allocation should be integrated and taken into consideration [11].

3. Resource Allocation

The cloud environment is formed with the Nimbus services and cumulus services [13] in a Intel Xeon 2.66 GHz CPUs consisting 8 CPU cores and a total main memory of 32 GB. All servers are connected through regular 1 GBit/s Ethernet links. The host operating system was Linux (kernel version 2.6.30) with KVM as the virtualization technique. The cloud environment is established with a central service node upon which the scheduling script is posed for evaluation.

The service node infrastructure consists of four layers in which the first layer collects all the resource requests from various clients at remote site. The second layer is composed of the Nimbus and the Cumulus services along with the other central server node's configuration. The third layer is composed of the real time scheduler that considers both tasks and the resources. This third later collects the requests from its upper layer for every specific time interval. The fourth layer is responsible for distributing memory to the remote clients based upon the scheduling criteria as designed by the middle layer.

Fig:1 Layered infrastructure of the service node



The scheduling algorithms mainly focus on the distribution of the resources among the requestors that will maximize the selected QoS parameters. The QoS parameter selected in our evaluation is the cost function. The scheduling algorithm is designed considering the tasks and the available

virtual machines together and named LSTR scheduling strategy. This is designed to maximize the resource utilization.

Algorithm:

Step1: The requests are collected between every pre-determined interval of time.

Step 2: Resources R_i -> $\{R_1,R_2, R_3,\ldots,R_n\}$

Step 3: Requests RQ_i -> $\{RQ_1, RQ_2, RQ_3,\ldots, RQ_n\}$

Step 4: Threshold (static at initial)

Step 5: Th= $\sum R_i$

Step 6: for every unsorted array A & B

Step 7: sort A & B

Step 8: for every RQi

Step 9: if RQ_i<Th then

Step 10: add RQ_i in low array, $A[RQ_i]$

Step 11: else if RQ_i>Th then

Step 12: add RQ_i in high array $B[RQ_i]$

Step 13: for every $B[RQ_i]$

Step 14: ** allocate resource for RQ_i of B

Step 15: $R_i= R_i-RQ_i$; Th= $\sum R_i$

Step 16: satisfy the resource of $A[RQ_i]$

Step 17: for every $A[RQ_i]$

18 ** allocate resource for RQ_i of A

Step 19: $R_i= R_i-RQ_i$; Th= $\sum R_i$

Step 20: satisfy the resource of $B[RQ_i]$

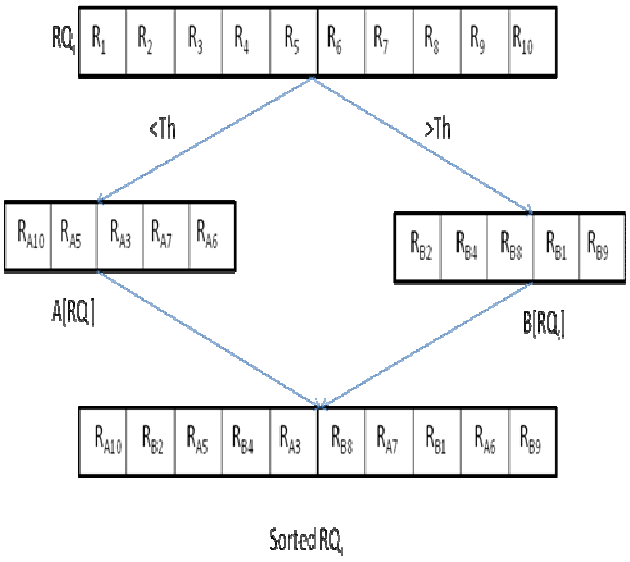
** Best fit strategy is used to satisfy the request alternatively in $A[RQ_i]$ and $B[RQ_i]$ based on the available VM.

The algorithm is explained with a simple input of memory request in GB's such as $R_i=(R_1,R_2,R_3\ldots..R_n)$.

The scheduling algorithm is carried out based on the prediction that the initial response to the request is made only after collecting the resource for a finite amount of time (say 1 day or 1 hr) but not allocating the resource as they arrive. The dynamic allocation could be carried out by the scheduler dynamically on request for additional resources. This is made by the continuous evaluation of the threshold value.

The resource requests are collected and are sorted in different queues based on the threshold value (say 50 for the sample memory request). The requests are satisfied by the VM's. Evaluation is made by creating VM of 512 MB in which the virtual memory is allocated to the longer and shorter queues based on the best fit strategy. This scheduling approach and the calculation of dynamic threshold value in the scheduler are carried out by considering both task and the resource. This improves the system throughput and the resource utilization regardless of the starvation and the dead lock conditions.

Fig.2 Scheduling strategy

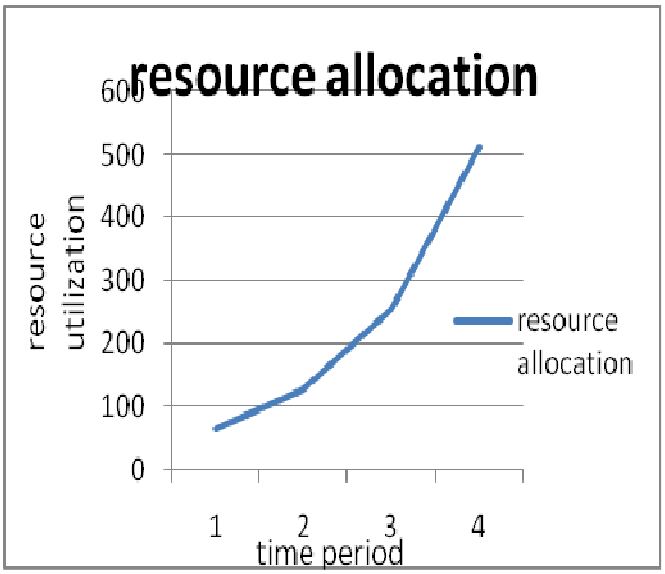


The scheduling algorithm was made for the set of inputs which specifies memory requirements, under the prediction that memory requests, CPU requests or i/o requests are satisfied only by the allocation of VM's.

Experimental verification of resource allocation in a grid environment evaluated that Shortest task first is comparatively better when compared to the First Come First Serve [1]. Hence the proposed scheduling algorithm sorts the collected request excluding the arrival time. The shortest request in both $A[RQ_i]$ and $B[RQ_i]$ is processed first which results in the allocating resource to more number of requests than in FIFO technique.

The experimental analysis was made by forming a cloud with the service node to control all client requests that are collected between a pre- defined time intervals. The service node embedded with the nimbus and cumulus services is responsible for resource allocation and the evaluation of resource allocation based on the LSTR scheduling algorithm is plotted as a graph with the resource utilized and the unit time along the X and Y axis.

Fig.3 Resource Utilization Graph



Conclusion

The cloud infrastructure is formed with the server based services including Nimbus and Cumulus services and an effective scheduling algorithm named LSTR scheduling algorithm which schedules both the task and the resources is designed. The algorithm mainly focuses is eradicating the starvation and deadlock conditions. The virtualization technique along with the scheduling algorithm will yield higher resource utilization, system throughput, thus improving the performance of the cloud resources. The result analysis of the implementation specifies the amount of resource utilized in the cloud environment when requests are satisfied by LSTR technique.

Future Work

Cloud demand and cloud resource utilization are factors that most of the IT industries and other organization will demand the most in future. Hence, apart from scheduling the task and resources, the factors such as time, and wastage of demanded resources from every client should be monitored. The provision of resource may be made with various virtualization techniques. This may ensure a higher throughput and usage than the existing cloud resource services. Our future work deals in involving few evolutionary techniques that will further result in better resource allocation, leading to improve resource utilization.

References

[1] Hyukho Kim, Woongsup Kim and Yangwoo Kim, “Experimental Study to Improve Resource Utilization and Performance of Cloud Systems Based on Grid Middleware “,Journal of Communication and Computer, USA, , Volume 7, No.12 (Serial No.73), December 2010.

[2] Chenhong Zhao, Shanshan Zhang, Qingfeng Liu, "Independent Tasks Scheduling Based on GeneticAlgorithm in Cloud Computing", IEEE Computer society,978-1-4244-3693-4,2009.

[3] Hai Zhong, Kun Tao, Xuejie Zhang, "An Approach to Optimized Resource Scheduling Algorithm for Open-source Cloud Systems" , IEEE Computer Society, 978-0-7695-4106-8, 2010.

[4] B. Sotomayor, K. Keahey, I. Foster," Combining batch execution and leasing using virtual machines", ACM 17th International Symposium on High Performance Distributed Computing, pp. 87-96,2008.

[5] Daniel Warneke, Member, IEEE, and Odej Kao," Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud", IEEE Transactions On Parallel And Distributed Systems, vol. 22, No. 6, JUNE 2011.

[6] P. Ruth, P. McGachey, D. Xu," VioCluster: virtualization for dynamic computational domain", IEEE International on Cluster Computing, pp. 1-102005.

[7] Anirban Kundu, Chandan Banerjee, Sutirtha Kr. Guha, Arnab Mitra, 4Souvik Chakraborty, Chiranjit Pal, Rahul Roy," Memory Utilization in Cloud Computing using Transparency".

[8] Alexandru Iosup , Simon Ostermann, M. Nezih Yigitbasi, Radu Prodan Thomas Fahringer Dick H.J. Epema,"Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing", IEEE Transactions On Parallel And Distributed Systems, Vol. 22, No. 6, June 2011.

[9] Sandeep Tayal,"Task scheduling optimization for the cloud computing systems", International Journal Of Advanced Engineering Sciences And Technologies Vol No. 5, Issue No. 2, pg no:111 – 115.

[10] Paul Marshall, Kate Keahey and Tim Freeman,"Improving Utilization of Infrastructure Clouds", IEEE /ACM International Symposium on Cluster, Cloud and Grid Computing, 2011.

[11] Ying Zhang, Gang Huang, Xuanzhe Liu, and Hong Mei," Integrating Resource Consumption and Allocation for Infrastructure Resources on-Demand", IEEE 3rd International Conference on Cloud Computing,2010.

[12] Yuan Yuan, Wen-Cai Liu, "Efficient resource management for cloud computing", International Conference on System Science, Engineering Design and Manufacturing Informatization, 2011.

[13] Nimbus Project, http://www.nimbusproject.org.

[14] Shalini Rmanathan, savitha Goel,subramaniyan Alagumalai,"comparison of cloud Database:Amazon's SimpleDB and Google's Bigtable",International Joutnam of Computer Science Issues, vol.8, issue 6,no 2, November 2011.

[15] Kernal Virtual Machine, http://www.kvm.org/admin.

[16] Marc Eduard Frˆıncu," Scheduling Algorithms for Distributed Systems", West University of Timisoara Romania, Nov 27th 2009.

[17] William K. Cheung, Jiming Liu, Kevin H. Tsang, Raymond K. Wong, "Dynamic Resource Selection For Service Composition in The Grid", Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI'04), 0-7695-2100-2/04.

Authors

1. Abirami S.P.

Ms.Abirami S.P. received the Bachelor's degree in Computer Science and Engineering from K.S.Rangasamy College of Technology, Tiruchengode and pursuing final year Master's degree in Computer Science and Engineering in PSG College of Technology, Coimbatore.

2. Shalini Ramanathan

Ms R.Shalini received the Bachelor's degree in Computer Science and Engineering from Park College of Engineering and Technology, Coimbatore and Master's degree in Computer Science and Engineering from Government College of Technology, Coimbatore, Tamil Nadu, India, in 2005.She is currently working as an Assistant Professor (Senior Grade) in Department of Computer Science and Engineering at PSG College of Technology, Coimbatore, India. Her research interests include bio-informatics, data mining, scientific knowledge discovery, parallel processing, distributed computing and cloud computing. She has published Four international Journals, Four paper in International Conference and three papers in National conference. She received best project award from Park College of Engineering and Technology in 2003. She is a member of Indian Society for Technical Education.