ISSN 2277-3061

A New Approach For Load Balancing In Cloud Computing

Rahul Rathore, Bhumika Gupta, Vaibhav Sharma, Kamal Kumar Gola
M.Tech CSE Student Uttarakhand Technical University, Dehradun,
rahul.rathore15@gmail.com
Assistant Professor, GB Pant Engineering college, Pauri Garhwal
bhumikamit6@gmail.com
Assistant Professor, Teerthanker Mahaveer University, Moradabad
vaibhavaatrey@gmail.com
Lecturer, Teerthanker Mahaveer University, Moradabad
Kkgolaa1503@gmail.com

ABSTRACT

Cloud computing is the dynamic delivery of information technology resources and capabilities as a service over the Internet. Cloud computing is a style of computing in which dynamically scalable and often virtualized resources are provided as a service over the Internet. It generally incorporates infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS). [1] Load balancing is one of the biggest challenges in the cloud computing. The concept of load balancing is to equally distribute the workload, resources across all the nodes to guarantee that all the nodes have equal load i.e. no single node is over loaded. As we all know that cloud computing services are mainly product based so in this approach we are using different product based priority queues for different services.

Indexing terms/Keywords

cloud computing; load balancing; priority queue; cloud services; load balancing policies.

Academic Discipline And Sub-Disciplines

Computer Science & Engineering.

SUBJECT CLASSIFICATION

Load balancing in cloud computing.

TYPE (METHOD/APPROACH)

Literary Analysis

Council for Innovative Research
Peer Review Research Publishing System

Journal: INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY

Vol 13., No 12.

www.ijctonline.com , editorijctonline@gmail.com

INTRODUCTION

Cloud computing provides different services such as IAAS, PAAS, SAAS for which users pay different amount as per their requirement. Today's scenario users are willing to pay more for better services. Sometimes it is necessary that availability of specific services is crucial for specific users. For such scenario we created methodology in which the privileged user will get access to the resources even if others users are waiting for the same resource. In this mechanism we are maintaining different priority queues at a remote location which may be geographically situated at the centroid of the available servers. Each server will maintain its different product based priority queues & when its waiting list gets over it will transfer the request to remotely located priority queue

Existing Load Balancing Techniques in Cloud Computing

In [2] Authors discussed the, a heuristic algorithm based on ant colony optimization has been proposed to initiate the service load distribution under cloud computing architecture. The pheromone update mechanism has been proved as a efficient and effective tool to balance the load. This modification supports to minimize the make span of the cloud computing based services and portability of servicing the request also has been converged using the ant colony optimization technique. This technique does not consider the fault tolerance issues. Researchers can proceed to include the fault tolerance issues in their future researches.

In [3] Authors discuss the JIQ algorithms for web server farms that are dynamically scalable. The JIQ algorithms significantly outperform the state-of-the-art SQ(d) algorithm in terms of response time at the servers, while incurring no communication overhead on the critical path. The overall complexity of JIQ is no greater than that of SQ(d). The extension of the JIQ algorithms proves to be useful at very high load. It will be interesting to acquire a better understanding of the algorithm with a varying reporting threshold. We would also like to understand better the relationship of the reporting frequency to response times, as well as an algorithm to further reduce the complexity of the JIQ-SQ(2) algorithm while maintaining its superior performance.

In [4] author proposed a novel load balancing algorithm called VectorDot. It handles the hierarchical complexity of the datacenter and multidimensionality of resource loads across servers, network switches, and storage in an agile data center that has integrated server and storage virtualization technologies. VectorDot uses dot product to distinguish nodes based on the item requirements and helps in removing overloads on servers, switches and storage nodes.

In [5] Authors proposed a mechanism CARTON for cloud control that unifies the use of LB and DRL. LB (Load Balancing) is used to equally distribute the jobs to different servers so that the associated costs can be minimized and DRL (Distributed Rate Limiting) is used to make sure that the resources are distributed in a way to keep a fair resource allocation. DRL also adapts to server capacities for the dynamic workloads so that performance levels at all servers are equal. With very low computation and communication overhead, this algorithm is simple and easy to implement

In [6] Authors addressed the problem of intra-cloud load balancing amongst physical hosts by adaptive live migration of virtual machines. A load balancing model is designed and implemented to reduce virtual machines' migration time by shared storage, to balance load amongst servers according to their processor or IO usage, etc. and to keep virtual machines' zero-downtime in the process. A distributed load balancing algorithm COMPARE AND BALANCE is also proposed that is based on sampling and reaches equilibrium very fast. This algorithm assures that the migration of VMs is always from high-cost physical hosts to low cost host but assumes that each physical host has enough memory which is a weak assumption.

In [7] Authors presented an event-driven load balancing algorithm for real-time Massively Multiplayer Online Games (MMOG). This algorithm after receiving capacity events as input, analyzes its components in context of the resources and the global state of the game session, thereby generating the game session load balancing actions. It is capable of scaling up and down a game session on multiple resources according to the variable user load but has occa- sional QoS breaches.

In [8] Authors proposed a scheduling strategy on load balancing of VM resources that uses historical data and current state of the system. This strategy achieves the best load balancing and reduced dynamic migration by using a genetic algorithm. It helps in resolving the issue of load imbalance and high cost of migration thus achieving better resource utilization.

In [9] Authors proposed a Central Load Balancing Policy for Virtual Machines (CLBVM) that balances the load evenly in a distributed virtual machine/cloud computing environment. This policy improves the overall performance of the system but does not consider the systems that are fault- tolerant.

In [10] Authors proposed a load balancing virtual storage strategy (LBVS) that provides a large scale net data stor- age model and Storage as a Service model based on Cloud Storage. Storage virtualization is achieved using an architecture

that is three-layered and load balancing is achieved using two load balancing modules. It helps in improving the efficiency of concurrent access by using replica balancing further reducing the response time and enhancing the capacity of disaster recovery. This strategy also helps in improving the use rate of storage resource, flexibility and robustness of the system.

In [11] Authors discussed a two-level task scheduling mechanism based on load balancing to meet dynamic requirements of users and obtain high resource utilization. It achieves load balancing by first mapping tasks to virtual machines and then virtual machines to host resources thereby improving the task response time, resource utilization and overall performance of the cloud computing environment.

In [12] Authors investigated a decentralized honeybee-based load balancing technique that is a nature inspired algorithm for self-organization. It achieves global load balancing through local server actions. Performance of the system is enhanced with increased system diversity but throughput is not increased with an increase in system size. It is best suited for the conditions where the diverse population of service types is required.

In [12] Authors investigated a self- aggregation load balancing technique that is a self- aggregation algorithm to optimize job assignments by connecting similar services using local re-wiring. The performance of the system is enhanced with high resources thereby in- creasing the throughput by using these resources effectively. It is degraded with an increase in system diversity.

In [12] Authors investigated a distributed and scalable load balancing approach that uses random sampling of the system domain to achieve self- organization thus balancing the load across all nodes of the system. The performance of the system is improved with high and similar population of resources thus resulting in an in- creased throughput by effectively utilizing the increased sys- tem resources. It is degraded with an increase in population diversity.

In [13] Authors proposed a two-phase scheduling algorithm that combines OLB (Opportunistic Load Balancing) and LBMM (Load Balance Min-Min) scheduling algorithms to utilize better executing efficiency and maintain the load balancing of the system. OLB scheduling algorithm, keeps every node in working state to achieve the goal of load balance and LBMM scheduling algorithm is utilized to minimize the execution time of each task on the node thereby minimizing the overall completion time. This combined approach hence helps in an efficient utilization of resources and enhances the work efficiency.

In [14] Authors proposed a new content aware load balancing policy named as work- load and client aware policy (WCAP). It uses a parameter named as USP to specify the unique and special property of the requests as well as computing nodes. USP helps the scheduler to decide the best suitable node for processing the requests. This strategy is implemented in a decentralized manner with low overhead. By using the content information to narrow down the search, it improves the searching performance overall performance of the system. It also helps in reducing the idle time of the computing nodes hence improving their utilization.

In [15] Authors proposed a new server-based load balancing policy for web servers which are distributed all over the world. It helps in reducing the service response times by using a protocol that limits the redirection of requests to the closest remote servers without overloading them. A middleware is described to implement this protocol. It also uses a heuristic to help web servers to endure overloads.

In [16] Authors proposed a lock-free multiprocessing load balancing solution that avoids the use of shared memory in contrast to other multiprocessing load balancing solutions which use shared memory and lock to maintain a user session. It is achieved by modifying Linux kernel. This solution helps in improving the overall performance of load balancer in a multi-core environment by running multiple load-balancing processes in one load balancer.

Challenges In Existing Techniques

A. Transfer policy
B. Selection policy
C. location policy
D. Information policies

Different algorithms working fine on different parameters but they have certain dependencies. Here one more challenge is that if someone wants to get access to the resources and better services and for this he is willing to pay more, then he/she should be treated as privileged user and he/she should get access to resources before others.

.

ISSN 2277-3061

Proposed Algorithm

A new approach for load balancing in Cloud Computing algorithm is as follows :

1. For each server there will be different product based priority queues i.e. if a server can have 10 services then it will have 10 different priority queues.
2. Whenever a server receives the request it will be in the respective product queue of the server.
3. If the server product priority queue is full or the server does not have the requested service then it will transfer that request to the priority queue, maintain by the request manager at remote location.
4. Priority will be decided on the following parameter:
    a. Product Id-20%
    b. Cost of Product-50%
    c. Privileged user- 30%

If two requests have same priority then the request will be scheduled according to first come first serve basis.

5. Whenever the request manager sees that the any of the server is free then it will dispatch the request to the corresponding server.
6. If any of the server does not free then it will prompt the server who process the request with lower priority than the request exist in the priority queue.
7. If priority of the request in the remotely managed priority queue & the request in product based priority queue are equal or less than it should not be pre-empt the existing running request on the server, else it will pre-empt the running request and get access to the server.

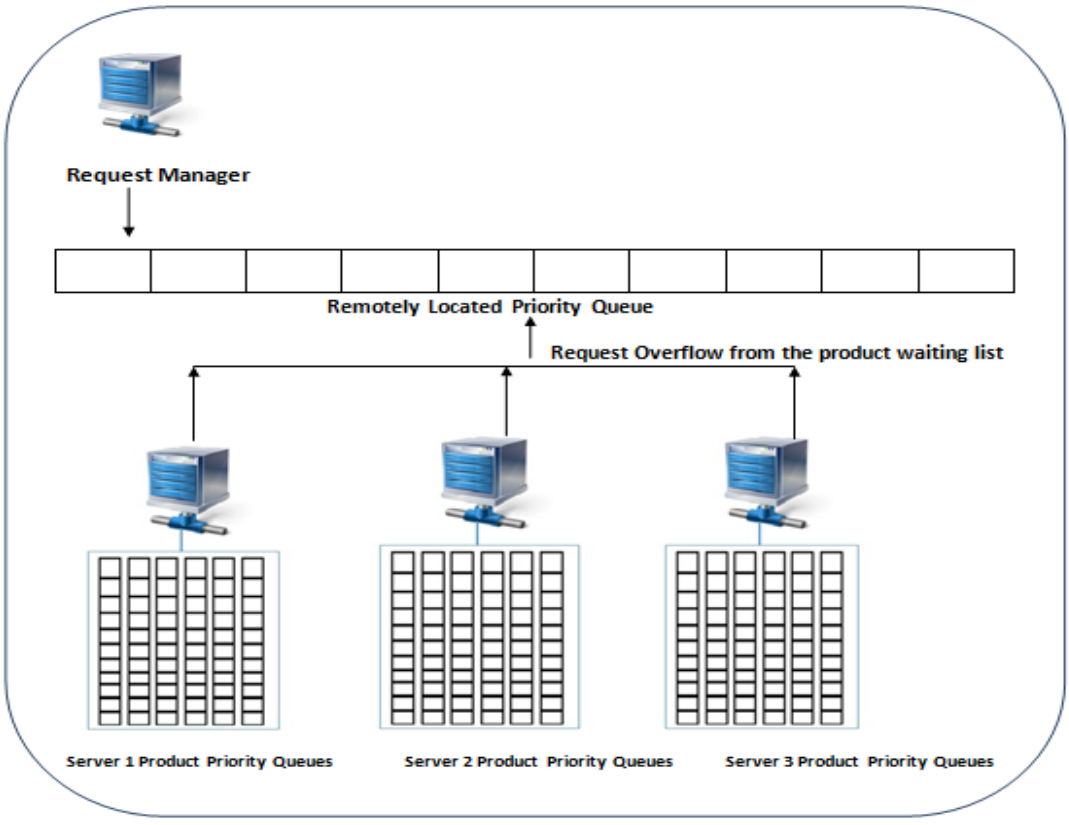

Fig 1: Proposed Architecture

IMPLEMENTATIONS

To implement priority queue at different level we used binary heap. It may take a little bit more time than normal queue to fetch the request, but it will ensure that the privileged user will get the access of the server before any normal user. Request Manager work is just to process the priority queue. It ensures insertion of request to the priority queue and fetching request from the priority. It also ensures that which server is processing the request with which priority. By finding the server who is processing request with lower priority than the priority of request available in the remotely managed priority queue, it transfers the corresponding request to that server. To deal with location policy we used Request manager which is remotely located at the centroid of all the servers so that every server will access the remotely located priority queue server in least request and response time. Load balancing is done by migrating request of one server to the same or any other server with the help of priority queue managed at both the ends i.e. server end and the priority queue managed at remote location.

RESULTS & CONCLUSIONS

This algorithm provides the load balancing technique to the cloud as well as increase the efficiency of the cloud by using the concept of priority queue to fulfil the request of privileged user without wait. Even if the privileged user request the server for service and its waiting queue is full, then also the waiting time of the user will be minimum. Load balancing is done by Request Manager by migrating the requests to different servers who will be free at that instant. This is meant for those cloud providers who wants to provide better services to their privileged users. We can also change the number of parameters and their weightage in terms of percentage who define priority of the user. By this we can define different level of priority of the users.

REFERENCES

[1] Implementing and Developing Cloud Computing Applications David E.Y. Sarna

[2] Ant colony Optimization: A Solution of Load balancing in Cloud Ratan Mishra 1 and Anant Jaiswal 2

[3] Join-Idle-Queue: A Novel Load Balancing Algorithm for Dynamically Scalable Web Services Yi Lua, Qiaomin Xiea, Gabriel Kliotb, Alan Gellerb, James R.Larusb, Albert Greenbergc

[4] Singh A., Korupolu M. and Mohapatra D. (2008) ACM/IEEE conference on Supercomputing.

[5] Stanojevic R. and Shorten R. (2009) IEEE ICC, 1-6.

[6] Zhao Y. and Huang W. (2009) 5th International Joint Conference on INC, IMS and IDC, 170-175

[7] Nae V., Prodan R. and Fahringer T. (2010) 11th IEEE/ACM International Conference on Grid Computing (Grid), 9-17.

[8] Hu J., Gu J., Sun G. and Zhao T. (2010) 3rd International Symposium on Parallel Architectures, Algorithms and Programming, 89-96

[9] Bhadani A. and Chaudhary S. (2010) 3rd Annual ACM Bangalore Conference

[10] Liu H., Liu S., Meng X., Yang C. and Zhang Y. (2010) International Conference on Service Sciences (ICSS), 257-262

[11] Fang Y., Wang F. and Ge J. (2010) Lecture Notes in Computer Science, 6318, 271-277

[12] Randles M., Lamb D. and Taleb-Bendiab A. (2010) 24th International Conference on Advanced Information Networking and Applications Workshops, 551-556.

[13] Wang S., Yan K., Liao W. and Wang S. (2010) 3rd International Conference on Computer Science and Information Technology, 108-113

[14] Mehta H., Kanungo P. and Chandwani M. (2011) International Conference Workshop on Emerging Trends in Technology, 370-375

[15] Nakai A.M., Madeira E. and Buzato L.E. (2011) 5th Latin- American Symposium on Dependable Computing, 156-165.

[16] Liu Xi., Pan Lei., Wang Chong-Jun. and Xie Jun-Yuan. (2011) 3rd International Workshop on Intelligent Systems and Applications, 1-4

ISSN 2277-3061

Author' biography with Photo

Rahul Rathore born on 15th Dec. at Moradabad**..** He did his B.Tech in Information Technology from IFTM ,Moradabad. He is pusung his M.Tech degree in Computer Science & Engineering from UttaraKhand Technical University, Dehradun, India.

Bhumika Gupta born on 31-March-1982 . she is currently working as Assistant Professor in Department of Computer science & Engineering, GB Pant Engineering College, Pauri Garhwal, Uttarakhand, India.

Vaibhav Sharma born on 13th may 1987.He did his B.Tech in Information Technology GNIT greater Noida.He received his M.Tech degree in Computer Science from Jaypee Institute of information Technology university. Presently he is working as Assistant Professor in COE, Teerthanker mahaveer University.

Kamal Kr. Gola born on 15th March. at Rampur**..** He did his B.Tech in Computer Science from MIT ,Moradabad in 2008. He is pusung his M.Tech degree in Computer Science from from UttaraKhand Technical University, Dehradun, India.Presently he is working as a lecturer in COE, Teerthanker mahaveer University.