An Optimal Algorithm to Improve Resource Utilization in Cloud Data Centre

S.Kanaga suba Raja, M.Hema

Abstract*—Cloud Computing Plays significant role in all type of business due to it features of measured service. so 89% of business are moving toward cloud that increases more number of data centers. Because of this there is more carbon dioxide emission.utilizing data centres for the efficient usage of client requests is major issue in the growing scenario amongst cloud users. Waiting for the data in queue for a long period of time increases the delay to obtain specific data. Our system focuses on reducing the delay of users, increasing the bandwidth thus giving an optimized system in cloud using software defined network. The software defined network has a central control over the entire system which efficiently manages the client requests. We have used the genetic load balancing algorithm to process the client requests in an orderly manner to enhance the data distribution between the various users of cloud.

***Keywords*—software defined network, genetic load balancing, data distribution, reduce delay.**

I. INTRODUCTION

Cloud computing is the most fast growing technology in today's generation. Cloud computing is mainly based on accessing the data anytime, anywhere. The internet applications such as YouTube, face book follow cloud computing in order to manage all the client requests. These applications can be accessed anytime for any data. Using these might lead to overflow of requests since more people use the application at given time. This leads to difficulty in allocating particular client for a particular resource. According to the survey conducted on 2016, around 57% of today's environment is based on cloud. Many organizations have moved about two third of their applications to cloud. This has made the employees to easily enhance new features time to time.

Many internet applications are being used by the enormous number of users at a given time. This makes the traffic to increase when a particular data is being queried. By which many users find it difficult to access the data. This results in a difficult task for allocation of users amongst the cloud system. The major issue will be faced by the provider, who will not be able to handle a large number of requests rated, and the user, who will not be able to access the data in a given period of time. On the other hand, the data centre is dumped with more requests which increases the workload and reduces the efficiency in turn.

Revised Manuscript Received on October 15, 2019.

S.Kanaga suba Raja, Elakkya Manoharan, Deepika.U, Arunkumar.KEaswari Engineering College, RamapuramRamapuram Chennai, India

Mrs.M.Hema,Easwari Engineering College , Chennai, India

Due to this the data centres face very low bandwidth. This causes both waste of time and energy for the provider and also increases the waiting period of the users. Eventually users find it difficult to access the data and increase the delay. In order to overcome these issues, we focus on distributing the requests among the data centres across the system thus increasing the bandwidth and reducing the user's delay.

II. LITERATURE SURVEY

Use of extensive measurement study and content distribution networks(CDN) for large scale video-streaming platforms explores the various strategies and service architectures but bandwidth utilization is minimum[1].The use of hash tables which maps keys into values into the concept of content-addressable networks(CAN) is scalable and fault-tolerant but leads to data inaccuracy during storage of large data.[2].A system to extent the tenant-provider interface is developed called the Octopus but does not concentrate more on balancing the tenant requests[3].A system called NetStitcher to utilize bandwidth whenever and wherever it is available was developed but failed in propagation of bulky updates[4].A detailed research on the costs and sub components of various cloud service data centers is done to understand the most optimal but does not examine the reliability of the optimal cloud service chosen[5].An agile data centre with integrated server and storage visualization technologies was developed but could not incorporate proactive migrations and predicting workloads[6].The provisioning and de-provisioning of resources on demand is done using auto-scaling but addressing open issues and future research directions is a challenge[7].The focus is completely placed on analyzing and handling energy expenses involving both temporal and geographic variations. It is not reliable as it is not based on actual data but rather on a number of simplifying assumptions. [8].A federated cloud computing environment(Intercloud)to facilitate scalable provisioning has failed because of ambiguity during multiple requests[9].A completely cost efficient and environment-friendly cloud data centre failed to provide good performance[10].Cloud data nodes identified by their domain name space(DNS) and IP address failed to identify required data centers.[11].Each data center had to be evaluated for every request that came in leading to delay in processing of the request[12].Thus any of the existing systems do not provide high performance with the minimum time delay is the reason why this system is being proposed.

Also it is required to introduce such a system wherein the overall performance of the system must be increased and all the legitimate users must be provided with the resources efficiently.

III. PROPOSED SYSTEM

The proposed system implements an architecture which has the following main components:

1) User

2) Controller

3) Cloud service provider

The system functions by accepting user requests and these requests are sent to the service provider. The controller begins its function by analyzing the available resources and the capacity that they can handle further. And once the most suitable datacenter is identified the controller sends the user request and thus the request is processed.The working of the algorithm is represented in the fig 1.

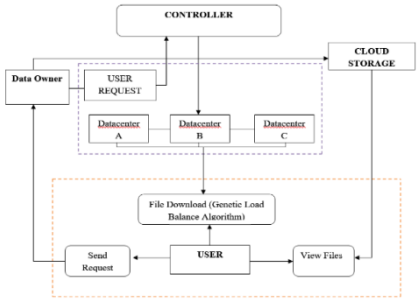

Fig 1 Genetic load balancing algorithm

Genetic Load Balancing Algorithm: The best example for dynamic load balancing algorithm is the Genetic Algorithm. Genetic Algorithm is a dynamic environment and at the same time a centralized environment. Genetic Algorithms (GAs) follows 3 steps namely, selection, crossover and mutation. These steps are used to find the computational analysis between the existing chromosomes and the new chromosomes. The values thus obtained help in obtaining the new fit offspring. The selection method is used in selecting the fit parent chromosome in-order to produce a healthy offspring. The crossover process helps in choosing the best parent chromosome. The final step is the mutation and the acceptance where the new fit offspring is produced and evaluated. The genetic algorithm helps in having a track over the existing data and helps in generating results at ease. This Proposed work consists of the following modules:

1) Selection

2) Crossover

3) Mutation and acceptance

A Selection:

Selection is the process in genetic load balancing algorithm in which the individual off springs are chosen from a larger population where breeding takes place. The procedure is carried out in the following steps:

1. The fitness function is calculated for each off spring which is normalized and sum of these fitnesses equals to 1.
2. These values are arranged in descending order.
3. From the normalized values the accumulated normalized values are calculated such that the sum of an individual's fitness value and that of the previous individual should be equal to 1.
4. Any random numeral K is chosen between 0 and 1.
5. The finally selected value of the individual is the last accumulated normalized value which is greater than K.

This specifies a method to select the best bigger fitness will be selected chromosomes is Roulette Wheel Selection. The better the chromosomes are, the more chances to be selected they have. Chromosomes with the more times. The selection process takes place by choosing the individuals and their fitnesses. Let's say the individuals be $a_1,a_2..a_m$ and their corresponding fitness be $f_1,f_2..f_m$.The relative probability of choosing an individual is given by:

$$\frac{f_j}{\sum_i f_i}$$

B Crossover:

Crossover is the process of selecting healthy parent chromosome to produce disease free offspring. Crossover procedure takes place by selecting the parent chromosome by thorough analysis to ensure that there is no negative effect on off springs. The crossover is of three types:

1. **One way crossover:**

Initially the parent chromosomes are analyzed and a particular point called the "crossover point " is chosen. The portion on the right to this point is interchanged between the two parents leading to generation of two off springs each of which is genetically related to the parents.

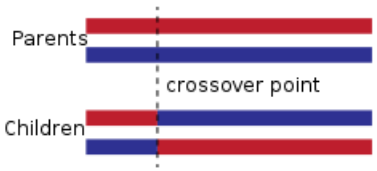

Fig 2. Two way and n-way crossover:

In this type, unlike the one way cross over, two cross over points are chosen upon analysis of parent chromosomes. The portion in between these two points is exchanged between the parent chromosomes. The two way crossover is equivalent to performing the one way crossover twice in a row.

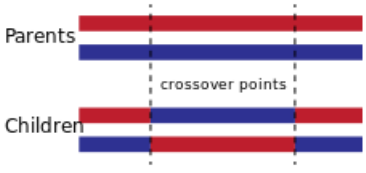

Fig 3. Consistent crossover:

This particular type of crossover involves choosing each portion of the offspring without any dependency from the parent chromosomes. Generally the proportions are chosen with equal probability from both the parents and also in some cases different mixing ratios are used.

After selection phase is completed the population is enriched with better individuals. It makes clones of good strings but does not create new ones. Cross over operator is applied to the mating pool with a hope that it would create better string.

Before crossover:

S1 = 1111010101
S2 = 1110110101
After crossover:

S1 = 1110110101 S2 = 1111010101

C Mutation and Acceptance:

Mutation is the main step in the load balancing algorithm that is used to maintain genetic diversity between each generation. It alters the gene values that differ from each of the chromosomes. Mutation of a bit involves flipping it, changing 0 to 1 and vice versa. We have the delay of the link between datacenter I and end user k for application instance i as follows:

$$\delta(j,k,i) = l(j,k) + \varphi(j,i) + \varphi(j,i)\frac{\rho(j,i)}{1-\rho(j,i)}\left(\frac{\sigma^2{}_{\varphi(j,i)} + \sigma^2{}_{\tau(j,i)}}{2}\right)$$

Where:

δ (j,k,i) is the time delay of the link between datacenter and end-user

(j, k) is the delay in transportation between end user and data centre

φ (j, i) is the average service time

ρ (j, i) is the bandwidth that is utilized

$\sigma^2\varphi$ (j, i) is the doubled coefficient of average service time

$\sigma^2\tau$ (j,i) is the doubled coefficient of average arrival time

Improved Genetic Load Balancing Algorithm:

Algorithm: Genetic load balancing
Input: The incoming resource requests from the users.
Output: Resource allocation as intended in the request
Declarations:
Step 1: Register and identify legitimate users.
Step 2: Analyze and keep track of the resources available at each server.
Step 3: Receive requests from the user with specification about the required resource.
Step 4: Select the data centre as per the request using selection strategy.
Step 5: Perform crossover in-order to choose the fit data centre for the request.
Step 6: Perform mutation and acceptance to allocate and process the request.
Step7:If(datacenter1<=datacenter2&&datacenter1<==datacenter&&datacenter1<15) then datacenter ++;
Step8: If(datacenter==15) then overload and shift;
Step9: Else all datacenters overloaded
Step10: Repeat the steps 7, 8 and 9 until all the requests are distributed and all the data centres are utilized.

Genetic load balancing algorithm is one of the easiest and simplest methods for load balancing amongst datacenters in cloud. The genetic load balancing works on the principle where the load balancer forwards the request to each data centre. Each datacenter is assigned with a specific weight. For example consider a Datacenter A already has 10 requests and the balance is 5. Datacenter B already has 8 requests and the balance is 7. Datacenter C already has 11 requests and the balance is 4. Now the datacenter B only has the maximum balance, so the new request will be forwarded to Datacenter B after checking all the Datacenters which is represented in fig 4.

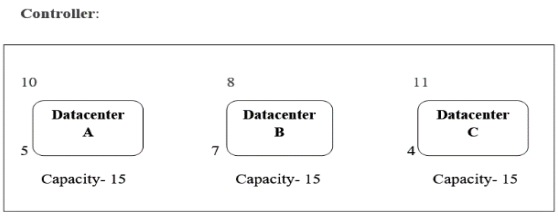

Fig 4 load balancing between data centres

IV. EXPERIMENTAL RESULTS

The existing system follows Nash bargaining algorithm for load balancing in cloud. The comparison of various factors with the existing system is represented graphically to show the improvisation in proposed system using Genetic algorithm. Our system produces 90 percent accurate results in reducing the time delay, increasing the bandwidth and also in reducing power between various virtual machines in cloud.

The proposed system helps in reducing the time delay between various users in the cloud than the existing system. It has observed that the algorithm that has been mainly used in the majority of the existing works is the Nash Bargaining algorithm. The systems that do not use this algorithm have derived their methodologies and procedural computations from the Nash Bargaining algorithm.

In the above comparison we have chosen 5 data centers and 100 application instances serving 500 end-users. From the above graph it can be inferred that the delay in the improved genetic load balancing algorithm is reduced considerably

Parameters	Nash bargaining	Smoothing technique	Improved genetic load balancing algorithm
Time delay	75%	64%	89%
Bandwidth utilization	60%	70%	90%
Power efficiency	52%	68%	87%

when compared to the already existing Nash bargaining algorithm. Thus we can conclude Table 1.1:

Performance Comparison of Proposed Algorithm
saying that when there is multiple requests for the cloud server during resource requirement it is advisable to use the genetic load balancing for the allocation of resources rather than any other algorithm that is available.

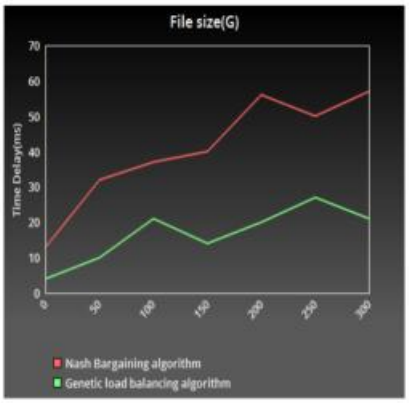

Fig5 Time delay comparison

The proposed system uses JAVA/J2EE for backend programming which is implemented using Net beans 7.4 wherein the database is maintained using MySql. The overall system runs in windows platform of any version.

$$time\ delay = \frac{1}{x\rho - \theta} mn$$

Fig 5 represents the reduction in time delay using the proposed system by comparing with the existing system. The time delay is given by the above formula where

x = total number of servers

ϱ = rate of service

Θ = rate of monitoring request

m, n = waiting time

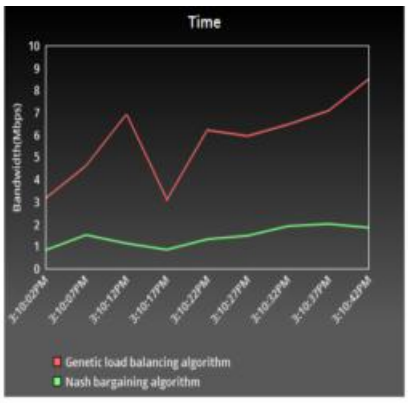

Fig 6 Bandwidth comparison

There is an increase in bandwidth using the proposed system which is represented in fig 6.The bandwidth utilization is given by :

$$BD_V = \left(BD\ in\ V_i\ /\ Total\ capacity\ of\ BD\ V_i \right) * 100$$

$$BD_P = \left(BD\ in\ P_j / \ Total\ capacity\ of\ BD\ P_j \right) * 100$$

Where:

BD_V = Bandwidth of virtual machine

V_i = Virtual machine

P_j = Physical machine

BD_P = Bandwidth of physical machine

Total Bandwidth = $BD_V + BD_P$ Another major comparison that has been done is done is how the power efficiency varies between the proposed system and the already existing ones which is represented in fig 7. The power consumption is represented by the following formula:

$$Power = P_i + \left(P_j - P_i \right) * U$$

Where:

P_i = minimum power at zero state

P_j = maximum power at highest load

U = rate of utilization

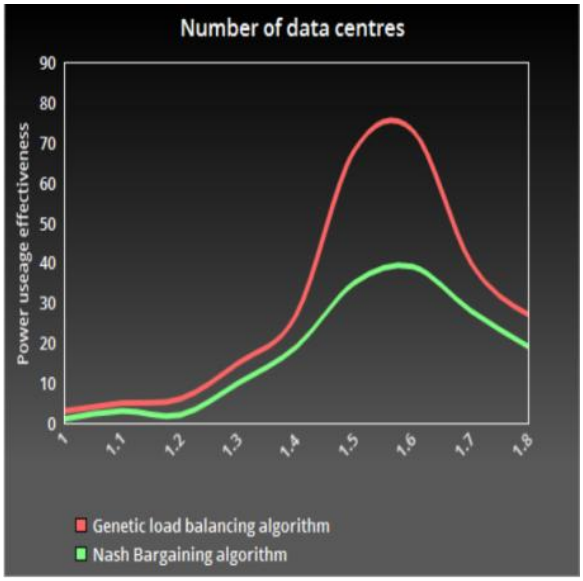

Fig 7: Power comparison

V.CONCLUSION

In our paper we have mainly focused on efficiently distributing the requests over number of users in the cloud system by applying genetic load balancing algorithm using software defined network. The major drawback in the existing system is that the working is N-P hard which is difficult to resolve for request allocation. The smoothing technique being used to solve such n-p hard problems is based on time series which is difficult to resolve and conclude to a smaller value. Our proposed system efficiently allocates the requests to the data centres which increases the bandwidth and reduces the delay of the users. We encourage developers of other large scale distributed systems to try ClosestNode.com and get the benefit of an accurate, scalable

and backwards-compatible system for directing clients to nearby servers.

Future work can proceed in couple of directions. We are interested in looking into other video streaming delivery systems such as Hulu, to see if cloud sourcing and/or multiple CDN strategy have been adopted.

ACKNOWLEDGEMENT

This work was supported by DST-FIST Programme No.SR/FST/College-110/2017, Government of India

REFERENCES

1. Amazon web services [Online]. Available: http://aws.amazon. com, 2014.
2. V. K. Adhikari, Y. Guo, F. Hao, M. Varvello, V. Hilt, M. Steiner, and Z.-L.Zhang, “Unreeling netflix: Understanding and improving multi-CDN movie
3. delivery,” in Proc. IEEE Conf. Comput. Commun., 2012, pp. 1620–1628.
4. A. Cockcroft. (2011). Netflix in the cloud [Online]. Available:http://velocityconf.com/velocity2011/public/schedule/detail/17785 B. Wong and E. G. Sirer, “Closestnode.com: An open access, scalable, shared geocast service for distributed systems,” Operating Syst. Rev., vol. 40, no. 1, pp. 62–64, 2006.
5. H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, “Towards predictable datacenter networks,” in Proc. ACM SIGCOMM Conf., Toronto, ON, Canada, 2011, pp. 242–253.
6. N. Laoutaris, M. Sirivianos, X. Yang, and P. Rodriguez, “Interdatacenter bulk transfers with netstitcher,” in Proc. ACM SIGCOMM Conf., Toronto, ON, Canada,2011, pp. 74–85.
7. A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, “The cost of a cloud:Research issues in information center networks,” ACM SIGCOMM Comput. Commun.Rev., vol. 39, no. 1, pp. 68–73, 2008.8. A. Singh, M. Korupolu, and D.Mohapat, “server virtualization: Integration and merchandise exploit in information centers,” in Proc.ACM/IEEE Conf.Supercomput., 2008, p. 53.
8. 9. R. Buyya, R. Ranjan, and R. N. Calheiros, “Intercloud: Utilityoriented Federation of cloud computing environments,” in Proc. 10th Int. Conf. Algorithms Archit. Parallel Process., 2010, pp. 13–31.
9. 10. A. Qureshi, R. Weber, H. Balakrishnan, J. V. Guttag, and B. M. Maggs, “Cut the electrical bill for internet-scale systems,” in Proc. ACM SIGCOMM Conf., 2009, pp. 123–134.
10. 11. H. Xu and B. Li, “Cost efficient datacenter selection for cloud services,” in Proc. IEEE Int. Conf. Commun. China, 2012, pp. 51–56.
11. 12. P. Wendell, J. W. Jiang, M. J. Freedman, and J. Rexford, “Donar: Decentralized server selection for cloud services,” ACM SIGCOMM Comput. Commun. Rev., vol. 40, no. 4, pp. 231–242, 2010.
12. 13. S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, A. Vahdat, S. Stuart, U. Hlzle, and J. Zolla, “B4: Experience with a globally-deployed software defined wan,” in Proc. ACM SIGCOMMConf., 2013, pp. 3–14.
13. 14. H. Xu and B. Li, “A general and practical datacenter selection framework for cloud services,” in Proc. 5th IEEE Int. Conf. Cloud Comput., 2012, pp. 9–16.
14. C. Q. Wu, X. Lin, D. Yu, W. Xu, and L. Li, “End-to-end delay minimization for scientific workflows in clouds under budget constraint,” IEEE Trans. Cloud Comput., 2014, Doi: 10.1109/TCC.2014.2358220.
15. V. Mathew, R. K. Sitaraman, and P. J. Shenoy, “Energy-aware load
16. balancing in content delivery networks,” in Proc. IEEE Conf. Comput. Commun., 2012, pp. 954–962.
17. A. Leon-Garcia and A.-H. Mohsenian-Rad, “Coordination of cloud computing and smart power grids,” in Proc. 1st Int. Conf. IEEE SmartGrid Commun., 2010, pp. 368–372.
18. Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew,“Greening geographical load balancing,” in Proc. ACM SIGMETRICS Joint Int. Conf. Meas. Model. Comput. Syst., 2011, pp. 233–244.
19. K. Boloor, R. Chirkova, Y. Viniotis, and T. Salo, “Dynamic request allocation and scheduling for context aware applications subject to a score latent period SLA in an exceedingly distributed cloud,” in Proc.2nd Int. Conf. IEEE Cloud Comput. Technol. Sci., 2010, pp. 464–472.
20. K. Le, R. Bianchini, T. D. Nguyen, O. Bilgir, and M. Martonosi,“Capping the brown energy consumption of web services at low cost,” in Proc. IEEE Green Comput. Conf., 2010, pp. 3–14.
21. M. Polverini, A. Cianfrani, S. Ren, and A. V. Vasilakos, “Thermalaware scheduling of batch jobs in geographically distributed data centers,” IEEE Trans. Cloud Comput., vol. 2, no. 1, pp. 71–84, Jan.– Mar. 2014.
22. H. Xu and B. Li, “Joint request mapping and response routing for geo-distributed cloud services,” in Proc. IEEE Conf. Comput. Commun., 2013, pp. 854–862.
23. P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, “It’s not easy being green,” ACM SIGCOMM Comput. Commun. Rev., vol. 42, no. 4, pp. 211–222, 2012.
24. DynDNS [Online]. Available: http://dyn.com/dns/, 2012.

AUTHORS PROFILE

Dr. S.KanagaSuba Raja, is working as an Associate Professor in Easwari Engineering College. He obtained his Ph.D degree in Computer Science in 2013 from Manonmaniam Sundaranar University, Tirunelveli. He has about 14 years of teaching and research experience. He has successfully guided many graduate and under-graduate students for their research projects. He has several papers published in International journals and conferences to his credit. His current research interests are Wireless Body Area Networks, Deep learning, IoT and Mobile and Agricultural Engineering.

Mrs. M.Hema,M.E,(Ph.D), is working as Assistant Professor (Sr.Gr) in Easwari engineering College. she is Pursuing Ph.D in Anna University, She has 15 years of teaching and research Experience. She has published many Papers in Journals and Conferences. she got fund of Rs.7.5 Lakhs worth from AICTE .Lifetime member in ISTE,member in IET IAENG and UCCEE.Organized various Workshops, National and International Conference. She is currently working Cloud Computing, IOT ,Deep learning and AI.