Load Balancing Issues in Cloud Environment Using Virtual Machines to Handles Future Load Imbalances with Service Level Objects

Mahesh. K, Dr. Yogesh Kumar Sharma & Dr. M. Laxmaiah

PhD Scholar, CSE Dept, JJT University, Rajasthan.
Associate Professor, JJT University, Rajasthan.
Professor& Head of IIIC, CSE Dept, CMR Engineering College, Telangana.

Abstract:
To give powerful framework as a help (IaaS), mists right now perform load adjusting by relocating virtual machines (VMs) from intensely stacked physical machines (PMs) to daintily stacked PMs. The one of a kind highlights of mists present considerable difficulties to accomplishing compelling and productive burden adjusting. To start with, VMs in mists utilize various assets (e.g., CPU, data transfer capacity, memory) to serve a assortment of administrations (e.g., elite processing, web administrations, record administrations), bringing about various over utilized assets in various PMs. Additionally, the over utilized assets in a PM may change after some time because of the time-shifting heterogeneous help demands. Second, there is escalated organize correspondence between VMs. Notwithstanding, past burden adjusting techniques statically allot equivalent or predefined loads to various assets, which lead to debased execution as far as speed and cost to accomplish load balance. Additionally, they don't endeavour to limit the VM interchanges between PMs. We propose a Resource Intensity Aware Load adjusting technique (RIAL). For every PM, RIAL powerfully appoints various loads to various assets as indicated by their utilization force in the PM, which altogether lessens the time and cost to accomplish load balance and keeps away from future burden unevenness. It likewise attempts to keep every now and again conveying VMs in a similar PM to diminish data transfer capacity cost, and moves VMs to PMs with least VM execution debasement. We likewise propose an all-inclusive variant of RIAL with three extra calculations. To start with, it ideally decides the loads for considering correspondence cost and execution debasement due to VM relocations. Second, it has an increasingly severe movement activating calculation to maintain a strategic distance from superfluous relocations while as yet fulfilling Service Level Objects (SLOs). Third, it conducts goal PM choice in a decentralized way to improve versatility. Our broad follow driven reenactment results and genuine world test results show the better execution of RIAL looked at than other burden adjusting strategies.

Keywords: Load balancing, service level objects, VMs, PMs, Load forecasting, Exponential smoothing, Threshold, Migration.

1. INTRODUCTION

The cloud is turning into a significant assistance in web registering. Framework as a Service gives on-request virtual machines to clients. Burden adjusting plays an significant job in the arrangement of virtual machines onto physical hosts. Asset necessity of virtual machine is difficult to anticipate. In this way to manage this issue, heaps of burden adjusting strategies are available. So in this paper we talk about these heap adjusting strategies to give outline of most recent approaches in IaaS. Distinctive burden adjusting strategies have various parameters. In this way, we looked at them based on parameters utilized for their technique. We have proposed our claim load adjusting strategy subsequent to looking at all accessible

strategies. In our proposed technique, we first use load estimating to realize load on has. At that point based on edge esteem we need to move VMs to another host. On the off chance that load is underneath or higher then we need to close down or move VM individually. Along these lines, by utilizing load anticipating we are attempting to conquer downsides of existing burden adjusting techniques.

2.RELATED WORK

Many burden adjusting strategies have been proposed to bargain with PM over-burden issue utilizing VM relocation [1]–[5]. Sandpiper [1] attempts to move load from the most over-burden servers to the most under loaded servers. To choose VMs to relocate (or select goal PM), it first structures a weighted standardized choice lattice with the uses of VMs of a PM (or PMs) concerning every basis. It at that point decides the perfect arrangement by utilizing the most extreme usage for the advantage criteria and the least use for the cost criteria.

IaaS has significant issues like asset the board, arrange foundation, virtualization, information the executives and so forth. IaaS gives benefits like: adaptability, QoS, decrease in overheads, cost adequacy. Various kinds of assets like physical and legitimate are given in IaaS. Physical assets incorporates CPU, memory, stockpiling. Consistent assets incorporates working framework, energy.

These days, distributed computing is concentrating on how productively foundation is made accessible and accessible assets are utilized. Burden adjusting is fundamental factor in distributed computing. Burden adjusting implies circulate accessible outstanding burden over different hubs to guarantee that no asset is underutilized or overpowered. So great burden balancer is required to adjust changing condition techniques and sorts of tasks.

Live VM movement is utilized for load adjusting. It is utilized to move dynamic VM starting with one physical host then onto the next without disturbing the VM. VM movement accomplishes load adjusting.

2. LITERATURE SURVEY

Khanna et al. [4] treated various assets similarly. They proposed to choose the VM with the most minimal result of asset uses from the over-burden PM and move it to the PM that has the least remaining limit sufficiently large to hold this VM. Arzuaga et al. [2] utilized foreordained asset loads to ascertain the result of weighted usages of various assets of a PM or a VM as its heap. It at that point picks the VM with the most noteworthy burden from an over-burden PM to move to a chose PM that yields the best improvement of the framework irregularity metric. Tang et al. [8] proposed a heap adjusting calculation that endeavors to augment the absolute fulfilled application request and equalization the heap crosswise over PMs. They characterize load-memory proportion of an occasion as its CPU load partitioned by its memory utilization to quantify its asset use.

Be that as it may, every past strategy statically expect equivalent or on the other hand predefined loads for various assets, which may not be right because of the distinctive time-fluctuating requests on various assets in every PM. RIAL is recognized from these strategies in that it powerfully decides the asset weight dependent on the interest on the asset in every PM, which prompts quick and consistent intermingling to the heap adjusted state.

Xu et al. [9] evaluated the best in class inquire about on dealing with the presentation overhead of VMs, and abridge them under differing situations of the IaaS cloud, going from the single-server virtualization, a solitary mega data center, to various redistributed data centers. Li et al. [10] proposed viable VM arrangement techniques to lessen the system cost in cloud server farms. Xu et al. [9], [11] proposed techniques that consider VM execution, (for example, VM execution debasement brought about by VM relocation) when settling on the VM provisioning or movement choice. Lim et al. [12] demonstrated a movement procedure of a VM example as a couple of employments that run at the hosts of sender and recipient also, proposed a strategy to investigate the relocation time and the presentation sway on multi-asset shared frameworks for finishing given VM task plan.

A few works manage load adjusting on one asset for example, stockpiling [13] and transfer speed [14]–[16]. Hsiao et al. [13] proposed a heap adjusting calculation for appropriated record frameworks in mists by moving document lumps from over-burden servers to delicately stacked servers. Oktopus [14] gives static reservations all through the system to actualize data transfer capacity ensures. Popa et al. [16] explored the tradeoffs space of prerequisites instalment proportionality, asset least assurance and framework usage when sharing cloud arrange data transfer capacity. Xie et al. [15] proposed PROTEUS for data transfer capacity provisioning utilizing anticipated transmission capacity use profile so as to build the framework data transfer capacity usage and lessen the expense to the inhabitants.

In any case, by concentrating on just a single asset, these methodologies can't be straightforwardly utilized for PM load adjusting where VMs utilize various kinds of assets.

Numerous different works for asset the board in mists manage booking approaching outstanding task at hand demands or beginning position of VMs with the worry of cost and vitality proficiency. Lin et al. [10] proposed a calculation to accomplish dynamic right-measuring in data centers so as to spare vitality. It utilizes a forecast window of future appearances to choose when to kill an inert server.

Table.1.Comparison of Various Load Balancing Methods

Models / Usages	Model based	Agent based	Genetic algorithm	Task based system	Trust and reality	Load balancing	Load balancing in cloud	Analysis of issues	Improving resource	RIAL	L3B
CPU	Yes	Yes	No	No	Yes	No	Yes	Yes	Yes	Yes	Yes
Memory	Yes	Yes	No	No	Yes	No	Yes	Yes	Yes	Yes	Yes
Network I/O	Yes	No	No	No	Yes	No	Yes	Yes	Yes	No	Yes
Disk I/O	Yes	No	No	No	No	No	No	Yes	Yes	No	Yes
MIPS	Yes	No	No	No	No	Yes	No	No	No	No	No

3.LOAD BALANCING ALGORITHMS

Balancing In Cloud Computing Using Modified Throttled Algorithm

Shridhar G.Domanal and G.Ram Mohana Reddy et al an efficient approach to handle the load at servers by considering both availability of VMs for a given request and uniform load sharing among the VMs for the number of requests served. The work aimed at efficient method for load balancing, depicted from its two different objectives. One being the response

time required to serve the requests and other being the distribution of load among the existing VMs. This algorithm is efficient than round-robin and throttled algorithm. Distribution of load among the virtual machines in Round-Robin algorithm was nearly uniform, but was found less efficient considering response time. Throttled algorithm with better response time than Round Robin failed to distribute load uniformly, overloading initial VMs and leaving others underutilized. Proposed algorithm distributes load nearly uniform among VMs, with improved response time compared to existing algorithms. Simulation results have demonstrated that the proposed algorithm has distributed the load uniformly among virtual machines.

Resource Intensity Aware Load Balancing in Clouds

Liuhua Chen, Haiying Shen, Karan Sapra et al a new method of load balancing. Resource Intensity Aware load balancing system is used for each physical machine. For each physical machine, RIAL dynamically assigns weights to different resources according to their usage in physical machine instead of assigning it statically. It reduces cost and future load imbalance. Frequently communicating VMs are kept in same PM. This algorithm migrates VMs from overloaded PMs to lightly loaded PMs. It is distinguished by its resource weight determination based on resource intensity. RIAL takes into account the communication dependencies between VMs in order to reduce the communication between VMs after migration, and also tries to minimize the VM performance degradation when selecting destination PMs. RIAL is found more superior than other load balancing algorithms.

Low Level Load Balancer in the Cloud

Monika Simjanoska, Sasko Ristov, Goran Velkoski, and Marjan Gusev et al new low level load balancing in IaaS cloud. This method preserves the cloud's elasticity by dynamic activation of cloud resources and load balancing the traffic over the resources on low network level. In addition, L3B tends to load the instances in the region where they provide maximum performance. L3B improves the overall cloud performance, reduces power consumption and customers cost for renting cloud resources etc. L3B generates additional latency in delivering the request and response packets in the direction from client to server and vice versa, it provides several benefits especially if the server is hosted in VM instances. But if the central node fails, it will result in L3B decline. L3B is the fact that all L3B modules, agents and repositories require additional hardware resources, i.e. computing, memory and storage capacity. L3B in the cloud will improve the performance of the client-server model, but we also determine the condition how it can be achieved.

Load Balancing in Public Cloud

This model divides the public cloud into several cloud partitions. When the environment is very large and complex, these divisions simplify the load balancing. The cloud has a main controller that chooses the suitable partitions for arriving jobs while the balancer for each cloud partition chooses the best load balancing strategy. The load balancing strategy is based on the cloud partitioning concept. After creating the cloud partitions, the load balancing then starts. When a job arrives at the system, with the main controller deciding which cloud partition should receive the job. The partition load balancer then decides how to assign the jobs to the nodes. When the load status of a cloud partition is normal, this partitioning can be accomplished locally. If the cloud partition load status is not normal, this job should be

transferred to another partition. Load balancing model uses static and dynamic parameters i.e. CPU, Memory respectively. It initialize parameters and checks load degree.

4. IMPEMENTATION WORKS

In IaaS cloud, there are physical servers with a large number of virtual machines. These virtual machines are hosted with many heterogeneous applications. In order to optimize the utilization of computing resources and also saving energy consumption of cloud data centers, the applications running on the virtual machines will be migrated either to the same server or to another physical or virtual server. Identifying when it is best to migrate an application in a virtual machine has a direct impact on resource optimization. Performance optimization can be best achieved by an efficiently monitoring the utilization of computing resources. So, we need a comprehensive intelligent monitoring agent to analyze the performances of virtual machines.[

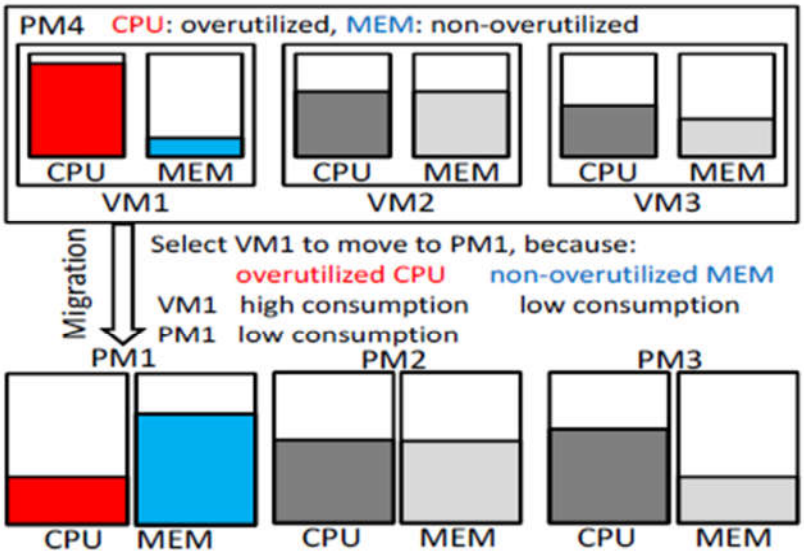


Fig. 1: Migration VM and destination PM selection.

Reducing VM Performance Degradation by Migrations

At the point when a VM is being relocated to another PM, its presentation (reaction time) is debased. We additionally point to limit the VM execution debasement brought about by relocations. We ascertain the presentation debasement of VM Vij relocating to PM Pp dependent on a strategy presented
in:

$$D_{ijp} = \sum d_{ip} \cdot \int_{t}^{t+\frac{M_{ij}}{B_{ip}}} u_{ij}(t)\mathrm{d}t$$

where t is the time when migration starts,Mij is the amount of memory used by Vij , Bip is the available network bandwidth, Mij Bip indicates the time to complete the migration, uij(t) is the CPU utilization of Vij , and dip is the migration distance from Pi to Pp. The distance between PMs can be determined by the cloud architecture and the number of switches across the communication path [16], [20].

Our concern of VM movement is a variation of the various backpack issue, which is NP-finished. An easier plan of our concern has been demonstrated to be NPcomplete in [11]. Our concern varies from them fundamentally in that it limits the quantity of VM movements. We can build an exceptional occurrence of our concern that is comparative to them and henceforth demonstrate that our VM movement issue is NP-finished. We will display a technique for unravelling this issue underneath.

Like all past burden adjusting strategies, RIAL occasionally finds over-burden PMs, recognizes the VMs in over-burden PMs to relocate out and recognizes the goal PMs to relocate the VMs to. We initially acquaint a technique with decide the heaviness of each kind of asset dependent on asset force. We expect to discover VMs to relocate out of each over-burden Pi to rapidly decrease its remaining task at hand.

To accomplish the previously mentioned target, we give over utilized assets generally higher loads than non over utilized assets. Among the non-over utilized assets, we allot lower loads to the assets that have higher uses so as to all the more completely use assets in the PM. Among the over utilized assets, the assets that have higher uses ought to get higher loads than those with moderately lower uses. For the over utilized assets that have comparative yet unique usage esteems, we want to appoint a lot higher loads to the assets with higher uses and appoint a lot of lower loads to the assets with lower use. That is, we overstate the contrast between the loads of assets dependent on the contrast between their use.

Thus, we use a power function with a basic form to determine the weight for an over utilized resource with resource utilization uik:

$$w_{ik} = \frac{1}{au_{ik}^{\alpha} + b};$$

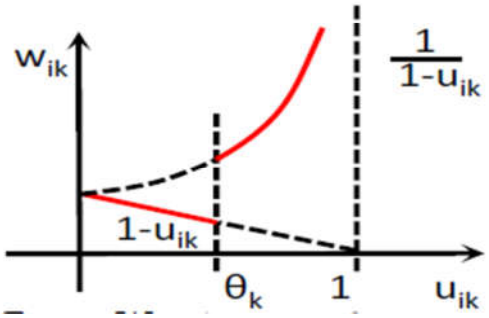


Fig.2. Weights vs Utilization

5.PERFORMANCE COMPARISON ANALYSIS

Contrasted with Sandpiper [1] and TOPSIS [5], RIAL produces less relocations. Since RIAL decides the asset weight dependent on asset force, it can rapidly assuage over-burden PMs by moving out less VMs with high Fig. 3: VM and PM choice process.

Additionally, the relocation VMs have low use of low-power assets, which helps completely use assets and abstains from over-burdening other PMs. Also, the relocation goal has a lower likelihood of being over-burden in this way as it has adequate limit to deal with the high-

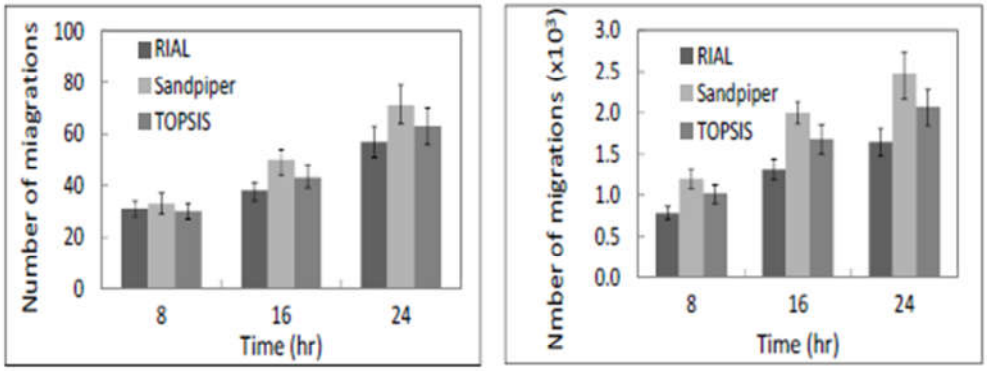
power assets. At long last, RIAL prompts less VM movements in a long haul. the framework load adjusted state for a more drawn out time span.

Table 1. No of moves needed for load balance

	Sandpiper	TOPSIS	RIAL	Proposed Work
#selected migrations VMs	2	2	2	1
#of overload destination PMs after VM migrations	0	1	0	0
Total # of migrations	2	3	1	1

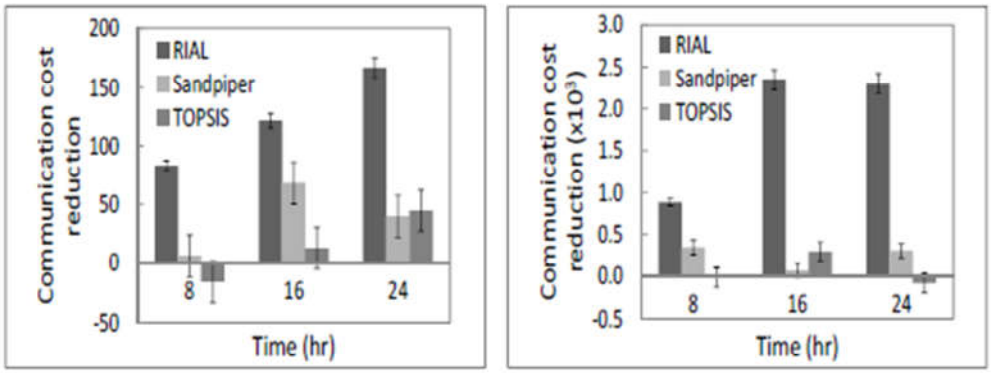
Table 1 records the quantity of chose VMs to calm over-burden PM0, the quantity of over-burden goal PMs after the VM relocations, and the all out number of movements to accomplish the heap adjusted state in one burden adjusting activity. We see that RIAL produces minimal number of relocations because of its preferences referenced beforehand.

We use an example with 3 PMs (PM0, PM1, PM2) to demonstrate the advantage of RIAL. In practice, the overloaded threshold should be close to 1. To make the example simple with few VMs, we set the threshold to0.5, and only consider the CPU and memory resources. We assume that PM0 has 4 VMs (VM0, VM1, VM2, VM3) with the same capacity and PM0's capacity is four times of the VM's. PMs have the same capacity. As in [5], the weight of CPU and memory in TOPSIS is 9 and 4, respectively. Figure 3 shows the CPU and memory utilizations of the 4 VMs, VM0(0.2,0.9), VM1(0.9,0.4), VM2(0.75,0.75), VM3(0.1,0.75) and the 3 PMs, PM0(0.49,0.7), PM1(0.3,0.15), PM2(0.1,0.32). PM0 is overloaded in memory resource usage since 0.7>0.5.



(a) 100 PMs and 250 VMs (b) 1000 PMs and 5000 VMs

Fig. 3. Total number of VM migrations.



(a) 100 PMs and 250 VMs (b) 1000 PMs and 5000 VMs

Fig. 4. Total VM communication cost reduction

6. CONCLUSION

In this paper, we propose a Resource Intensity Aware Load adjusting (RIAL) technique in mists that moves VMs from over-burden PMs to daintily stacked PMs. It is recognized by its asset weight assurance dependent on asset power. In a PM, a higher-concentrated asset is appointed a higher weight and the other way around. By thinking about the loads while choosing VMs to relocate out and choosing goal PMs, RIAL accomplishes quicker and lower-cost intermingling to the heap adjusted state, and diminishes the likelihood of future load lopsidedness. Further, RIAL considers the correspondence conditions between VMs so as to decrease the correspondence between VMs after relocation, and furthermore attempts to limit the VM execution corruption when choosing goal PMs. The loads allotted to correspondence cost and execution debasement are ideally decided so that the over utilized asset is mitigated and both correspondence cost and execution corruption are limited. We additionally propose RIAL with a progressively severe movement activating calculation to keep away from superfluous movements while fulfilling SLOs. At last, we make RIAL decentralized to improve its adaptability. Both follow driven reenactment what's more, genuine tested tests show that RIAL outflanks other burden adjusting approaches with respect to the number of VM movements, VM execution corruption and VM correspondence cost. In our future work, we will ponder how to all inclusive guide movement VMs and goal PMs in the framework to improve the adequacy and effectiveness of burden adjusting. We will likewise gauge the overhead of RIAL and investigate strategies to accomplish an ideal tradeoffs between overhead and adequacy.

REFERENCES

[1] T. Wood, P. J. Shenoy, A. Venkataramani, and M. S. Yousif, “Blackbox and gray-box strategies for virtual machine migration.” In Proc. of NSDI, vol. 7, 2007, pp. 17–17.

[2] E. Arzuaga and D. R. Kaeli, “Quantifying load imbalance on virtualized enterprise servers.” in Proc. of WOSP/SIPEW, 2010, pp. 235–242.

[3] A. Singh, M. R. Korupolu, and D. Mohapatra, “Server-storage virtualization: integration and load balancing in data centers.” In Proc. of SC, 2008, p. 53.

[4] G. Khanna, K. Beaty, G. Kar, and A. Kochut, "Application performance management in virtualized server environments," in Proc. of NOMS, 2006, pp. 373–381.

[5] M. Tarighi, S. A. Motamedi, and S. Sharifian, "A new model for virtual machine migration in virtualized cluster server based on fuzzy decision making." arXiv preprint arXiv:1002.3329, 2010.

[6] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines." inProc. of NSDI, 2005, pp. 273–286.

[7] A. B. Nagarajan, F. Mueller, C. Engelmann, and S. L. Scott, "Proactive fault tolerance for hpc with xen virtualization." in Proc. of ICS, 2007, pp. 23–32.

[8] C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A scalable application placement controller for enterprise data centers." In Proc. of WWW, 2007.

[9] F. Xu, F. Liu, and H. Jin, "Managing performance overhead of virtual machines in cloud computing: A survey, state of art and future directions," in Proc. of IEEE, 2014.
[10] X. Li, J. Wu, S. Tang, and S. Lu, "Let's stay together: Towards traffic aware virtual machine placement in data centers," in Proc. of INFOCOM, 2014.

[11] F. Xu, F. Liu, and H. Jin, "Heterogeneity and interference-aware virtual machine provisioning for predictable performance in the cloud," IEEE Transactions on Computers, 2016.

[12].Ravindra Changala, "Data Mining Techniques for Cloud Technology" in International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE),Volume 4, Issue 8, Pages 2319-5940, ISSN: 2278-1021, August **2015**.

[13] H. Hsiao, H. Su, H. Shen, and Y. Chao, "Load rebalancing for distributed file systems in clouds." TPDS, vol. 24, no. 5, pp. 951–962, 2012.

[14] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards predictable datacenter networks." in Proc. of SIGCOMM, vol. 41, no. 4, 2011, pp. 242–253.

[14]. Ravindra changala, "Clustering and Indexing for Uncertain Objects using Pruning techniques of Voronoi Diagram and R-trees" published in International Journal of Engineering and Innovative Technologies (IJEIT), *ISSN:* 2277-3754, ISO:9001:2008 Certified, Volume2, Issue 3, October 2012.

[15] D. Xie, N. Ding, Y. C. Hu, and R. R. Kompella, "The only constant is change: incorporating time-varying network reservations in data centers." in Proc. of SIGCOMM, vol. 42, no. 4, 2012, pp. 199–210.

[16] L. Popa, G. Kumar, M. Chowdhury, A. Krishnamurthy, S. Ratnasamy,and I. Stoica, "Faircloud: sharing the network in cloud computing." in Proc. of SIGCOMM, 2012, pp. 187–198.

[17] S. Lim, J. Huh, Y. Kim, and C. R. Das, "Migration, assignment, and scheduling of jobs in virtualized environment," in Proc. Of HotCloud, 2011.

18]. Ravindra Changala, "Retrieval of Valid Information from Clustered and Distributed Databases" in Journal of innovations in computer science and engineering (JICSE), Volume 6, Issue 1,Pages 21-25, September **2016**.ISSN: 2455-3506.