Journal of Theoretical and Applied Information Technology
30th November 2013. Vol. 57 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

CLUSTER BASED BEE ALGORITHM FOR VIRTUAL MACHINE PLACEMENT IN CLOUD DATA CENTRE

[1]AJITH SINGH. N AND [2]M. HEMALATHA

[1, 2]Department of Computer Science, Karpagam University, Coimbatore, Tamil Nadu, India

E-mail: [1]ajithex@gmail.com, [2]csresearchhema@gmail.com

ABSTRACT

The utilization of cloud data centres in combination with Virtualization technology has advantages of running more than one virtual machine in a single server. The data centres are a collection of many servers, allocation of VM to Host is known as VM placement. VM placement problem was examined in this paper with focus for maximum utilization of the resources and energy reduction. Switching off the idle server or in sleep mode can save energy consumption highly wasted in data centres. Technique for solving Virtual machine placement problem is implemented with the HoneyBee algorithm with hierarchical clustering in order to minimize energy consumption in servers. Cluster formation with the HoneyBee algorithm supports easy relocation of Virtual Machine migration and reduces the network latency. Further, simulation work with PlanetLab workload was experimented and revealed that the proposed HCT algorithm reduced energy consumption significantly while reducing the SLA and VM migration.

Keywords: *Virtual Machine, Cloud Computing, Live Migration, Server Consolidation, HoneyBee Algorithm, hierarchical Cluster*

1. INTRODUCTION

Cloud computing is an emerging topic, advancing rapidly in IT due to the flexibility of using computing without buying any infrastructure but by using as pay-per-use model. Some of the services which are available include Software as a Service (SaaS), Platform as a Service (PaaS) & Infrastructure as a Service (IaaS), all this service can be deployed either in public, private or hybrid model. Today's servers are highly configured with the advancing technology of hardware and software. Server in data centres remains idle when not in use, this server can be switched off or put in sleep mode to reduce energy consumption.

Success of cloud computing is by using Virtualization, where multiple instances of a Virtual Machine (VM) run on the same physical hardware. Instances of VM are created in the physical machine by using Virtualization, Amazon EC2, eucalyptus, cloud stacks, VMware, KVM and Xen are software involved, some of which are freely available and are also listed in free distribution with GNU/Linux.

The workload in the cloud is heterogeneous because the work is CPU intensive or I/O intensive [1]. High energy consumption in data centres is a recent topic where energy cost is very high. Studies have shown that data centres alone consume 61 billion kWh of U.S. energy in 2006. This is enough energy to power 5.8 million average U.S households and approximately costing $4.5 billion/year on energy costs. The numbers are more likely to increase more than 120 billion kWh [2] in case no further energy conservation steps are taken. Different approaches to reduce the energy consumption have been studied by many researchers, the use of lower power hardware and energy based hardware are available. The best alternative is switching off or putting in sleep mode the systems when host idle. The cloud data centres are different from traditional data centres in which, the server is running with different OSs, running different kinds of application.

The use of nature inspired algorithm like Honeybee which matches the data centre allocation is proposed and experimented. Honeybee algorithm in combination with the hierarchical clustering technique [4] was designed to solve the Virtual Machine placement problem in the cloud data centres. In this paper, we model the dynamic virtual machine allocation problem and propose a biologically inspired approach to this optimization problem in a cloud data centre. Specifically, our work has been inspired by the study of honeybee colonies and the behaviour of bees, characterized by decentralized and elementary interactions that

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

affect a complex collective behaviour to solve the problem of adequate food collection to ensure survival of the colony. Therefore, in this research a new algorithm HCT (Honey Bee Clustering Technique) has been proposed for Virtual Machine placement.

2. LITERATURE REVIEW

The cloud data centre is comprised of many servers and systems, all the systems differ from one another by using Virtualization technology like Xen Virtual machine which are run on it.

Jinhua et al., 2010 [5] proposed the scheduling of resource by using genetic algorithm based on the historical data and current states of the system having all the knowledge of the centre, the proposed idea is not suitable where user requirement of the differ from time to time, a possible way would be record keeping the of available VM and based on cloudlet requirement it would be faster to process all the resource allocation.

Yee et al., 2012 proposed a modified PSO is known as discrete particle swarm optimization for task allocation which reduces the execution time for data transfers between resources [6].

In contrast to the above methods where the main aim is to avoid or minimize VM migration, Clark et al., 2005 deeply studied the live migration of virtual machines and pointed out that Live migration of OS can be achieved by minimizing the downtime as low as 60ms , hence it helps in maintaining load balancing in cloud computing[7].

Sara Casolari et al., 2009 managed the process of VM migration by analysing separately each physical host and its related virtual machines with the main goal of minimizing migrations just to the most severe instance [8].

A network-aware migration scheduler was described by Stage and Setzer which consider the workload type of each VM and the migration takes place by explicitly considering the network topology and the bandwidth requirements to move VM images within a given deadline [9].

Wood et al., 2007 described a system called Sandpiper which relies an application independent OS and it in monitoring disk and network usage inside the Xen VM monitor. This system automatically identifies performance bottlenecks by identifying a new VM allocation which removes the previously selected VMs and finally initiate the required migrations to instantiate the new allocation [10].

Then, Das et al., 2008 proposed a multi-agent system approach to the problem of green performance in the data centre. The authors presented a framework which was based on a hierarchy wherein a resource arbiter assigns resources to the application managers, which in turn become in charge of managing physical servers[11].

Srikantaiah et al., 2009 studied the impact of consolidation of multiple workloads with different resource usage on performance, energy usage, and resource utilization. The authors focused on consolidating the workload so that each server receives a “balanced mix” of requests instead of simply migrating applications [12].

Khanna et al., 2006 [14] monitored the resources (CPU and memory) of physical and virtual machines. If a resource exceeds a predefined threshold and some SLA is at risk, then the system shifts the virtual machine to another physical host, if there is no available host, it activates a new physical a machine.

Barbagallo et al., 2010 [13] describe a bio-inspired algorithm based on the scout-worker migration method, in which some entities (the scouts) are allowed to move from one physical node to another in order to cooperatively identify a suitable destination for VMs (the workers) which are migrated.

A distributed algorithm like Compare And Balance proposed by Yi Zhao &Wenlong Huang, 2013 [18] to achieve a stable solution and highlight their migration of virtual machine is slow as shown by Xen.

A study from Fang et al., 2010 [23] describes that when the virtual machine is overloaded and host cannot allocate more resource, possibly two possible systems are either loaded in VM which is migrated to another host which have more free resources and the other one is migrated to the other VM, from this paper it comes to the point that in both the solution migration is required.

In the study of [21] proposed a load balancing policy with a central dispatcher called central load balancing policy for virtual machines where the load balancing decision is made by a centralized server by using global state information. Gandhi et al [25] findings revealed

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

that using maximum speed on the server is not optimal for best performance.

3. PROBLEM FORMULATION DEFINITION

Traditional load balancing cannot be implemented in the cloud data centre due to its nature of deployment, the cloud data centre has the special feature of Virtualization where many small Virtual Machine instances are created above the physical machine. Each virtual machine has a different load which leads to cross the threshold of a physical machine and result load imbalance, variables like CPU, memory and speed causes the system imbalance. IaaS layer of cloud computing serves as a foundation for the other two layers (i.e. PaaS and SaaS), for their execution therefore it has to focus on the IaaS. IaaS delivers computer infrastructure typically a platform Virtualization environment as a service. Efficient scheduling of the VMs instance request which meet user's requirements and improve the resource utilization and subsequently increases the overall performance of the cloud computing environment. VM instance scheduling in IaaS is one of the crucial cloud computing questions to address.

In general, the problem of dynamic VM consolidation can be split into 4 sub-problems as cited [16]:

1. Check whether the host is under loaded.
2. Check whether the host is overloaded.
3. Selection of policy to migrate VMs from an overloaded host.
4. VM Placement

VM placement for placing the VM in allocation or migration to other active or re-activated hosts.[16]

4. PROPOSED METHOD

Cloud computing is self-possessed of N node, each of which can host at most m^{th} VMs. N nodes are of different size and as for m^{th}. The most efficient algorithm must be able to allocate as many as VMs on a single host and reduce the utilization of the host as well as reduce the migration of VMs when the host is over utilized or underutilized. It would be easy for the providers realize kind of application running in Virtual Machines, hence we can differentiate the VMs based on its capability to handle the same job. So the approach leads to the formation of a cluster of VMs base on its CPU, SPEED, MEMORY. As mentioned above Host are of different properties. In combination that honeybee is formulated with clustering methods to reduce the latency while migration to find the best cluster for virtual machine placement. The proposed work is simulated with PlanetLab workload data which is of more 1000 VMs and running on 800 hosts [17].

4.1 Hierarchical Clustering Algorithm

1. Begin with the disjoint clustering having level L(0) = 0 and sequence number m = 0.
2. Find the least dissimilar pair of clusters in the current clustering, say pair (r), (s), according to d[(r),(s)] = min d[(i),(j)], where the minimum is over all pairs of clusters in the current clustering.
3. Increment the sequence number : m = m +1. Merge clusters (r) and (s) into a single cluster to form the next clustering m. Set the level of this clustering to L(m) = d[(r),(s)]
4. Update the proximity matrix, D, by deleting the rows and columns corresponding to clusters (r) and (s) and adding a row and column corresponding to the newly formed cluster. The proximity between the new cluster, denoted (r,s) and old cluster (k) is defined in this way:
5. d[(k), (r,s)] = min d[(k),(r)], d[(k),(s)]
6. If all objects are in one cluster, stop. Else, go to step 2.[4]

4.2 Bees Algorithm

Nature-inspired algorithm have received a lot of research attention in seeking distributed method, to address the increasing scale and complexity in such systems [19]. This nature based algorithm has its own feature which can be directly be applied or modified to suit the goal of any research. Honey bees behave in search of honey is applied in many applications. It is one of a number of applications inspired by the believed behaviour of a colony of honey bees foraging and harvesting food. The honey bee is sent to search for suitable sources of food, when one is found, they return to the hive to advertise this using a display to the hive known as a "waggle dance". The suitability of the food source may be derived from the quantity or quality of nectar, the bee harvested, or its

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

distance from the hive. This is communicated through the waggle dance display. Honey bees then follow the back to the discovered food source and begin to harvest it. Upon the bees' return to the hive, the remaining quantity of food available is reflected in their waggle dances, allowing more bees to be sent to a plentiful source, or exploited sources to be abandoned. This biologically-inspired technique is now used as a search algorithm in a variety of computing applications; seeming particularly scalable on a fluctuating underlying system[15].

1. Initialize population (N bees) at random
2. Evaluate fitness of population (fittest bee is the queen, D fittest following bees are drones, W fittest remaining bees are workers)
3. While stopping criteria are not satisfied (Forming new population) /* reproduction behavior */
4. Generate N broods by crossover and mutation
5. Evaluate fitness of broods
6. If the fittest brood is fitter than the queen then replace the queen for the next generation
7. Choose D best bees among D fittest following broods and drones of current population (Forming next generation drones)
8. Choose W best bees among W fittest remaining broods and workers of current population (to ensure food foraging) /* food foraging behavior */
9. Search of food source in W regions by W workers
10. Recruit bees for each region for neighborhood search (more bees (FBest) for the best B regions and (FOther) for remaining regions)
11. Select the fittest bee from each region
112. Evaluate fitness of population (fittest bee is the queen, D fittest following bees are drones, W fittest remaining bees are workers)

Figure.1: Bees Life Algorithm pseudo-code [20]

Considering the food source as the Host (Physical Machine) and bees as a VM (Virtual Machine) and hive as a load balancing server. Cloud computing is a heterogeneous distributed system, host as well as VM is in heterogeneous and even the cloudlet (task) of the user. Request of processing a cloudlet is different from one another, based on the cloudlet properties cloudlet are assigned to different VM, and VMs are created on different hosts. Allocation of VM is done on host based on the VM constant (size, memory, mips, storage), and a proper load balancing method is required so that all VM are created successfully in the Host with minimum time and achieving the SLA of the user. Allocation of VM to host can be done by a method called VmAllocationPolicy the policy is responsible for proper allocation of VM to host and if needed, perform VM migration and reduce the energy consumption and fulfil the SLA [19. Pseudo code of honey bees as depicted in Figure1 [20].

4.3 HCT - HoneyBee Cluster Technique

1. Cluster resources using hierarchical clustering by CPU, SPEED, MEMORY

2. Each cluster is considered as a single resource.

3. Job is categorized as minimum required, maximum required

4. Initialize honeybee parameters

n= number of employed bees

m = Number of onlooker bees (m>n)

s= number of scout bees

Iteration : Maximum iteration number

α : initial value of penalty parameter

5. Construct initial Employed Bee for initial solution

All employed bees find suitable vms for each task.

6 .Evaluate the fitness value for each employed Bee

Set of tasks T = {T_1, T_2, . . ., T_n}.

Deadline of tasks $D = \{ D1,D2,\ldots Dn \}$

Let VM = {VM_1,VM_2, . . . VM_m}

Total task completion - TCT.

Completion time of task T_i on VM_j as CT_{ij}

TCT= max {CT_{ij} |i ∈ T, i = 1, 2 . . . n and j ∈ VM, j = 1, 2, . . . m}

$$\min \sum_{i=1}^{n} CT_{ij} \quad j = 1,\ldots,m$$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Capacity of VM C_j

$C_j = PE_{numj} \times PE_{mipsj} + VM_{bwj}$

PE_{numj} is the number processor in VM_j

PE_{mipsj} is million instructions per second of all processors in VM_j

VM_{bwj} is the communication bandwidth ability of VM_j

Capacity of all VMs in Host

$$C = \sum_{i=1}^{m} C_i$$

Total length of tasks that are assigned to a Host is called a load

$$L_{V,M_i,t} = \frac{N(T,t)}{S(VM_i,t)}$$

Load of a VM can be calculated as the Number of tasks at time t on the service queue of VM_i

Divided by the service rate of VM_i at time t.

Processing time of a VM

$$PT_i = \frac{L_{VM_i}}{C_i}$$

Load of all VMs in a data centre is calculated as

$$L = \sum_{i=1}^{m} L_{VM_i}$$

Processing time of all VMs:

$$PT = \frac{L}{C}$$

$$Fitness = \frac{CT_{max}}{CT_i} + \frac{1}{PT_j}$$

if $\min(CT_{ij}) < D_i \parallel PT > 1$

Migrate VM as per policy

else

Calculate probabilities related to fitness values

$$P_i = \frac{1}{\sum_{i=1}^{n} Fitness_i}$$

Assign Onlooker Bees to Employee Bees according to probabilities

For all Onlooker Bees

Construct solution using Onlooker bees

All Onlooker bees find suitable vms for each task.

If fit(Best Onlooker)>fit(Employed)

Find best Onlooker, replace with respective Employed Bee

If fit (BestFeas Onlooker)<fit(Best)

Find best Feasible Onlooker, replace with Best solution,

End For

Initialize scout bees

Construct solution using Onlooker bees

If fit(Scout)> fit(Employed)

The scout bee replace Employed Bee

n=n+1

Until (n=Employed Bee)

I=I+1

Until (I=MaxIteration)

5. RESULT & DISCUSSION

The proposed VM placement algorithm based on CPU, SPEED, MEMORY using hierarchical clustering, each cluster is considered as a single resource. The benefit of clustering technique in VM placement, it's easy to search for available resource and helps us to reduce migration response time. The simulation is done with planet workload [16], the proposed HCT is compared with the HoneyBee placement technique. From the simulation it's found that, applying clustering methods in VM placement can reduce the energy consumption, SLA and VM migration in figure 2 energy consumption with HCT and honeybee is compared and its

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

found that HCT with LRR and MU gives the minimum energy consumption of 33.47kWh while 41.45kWh is given a minimum with honeybee combine with LRR and MMT. SLA comparison is shown in figure 3, SLA was not varied, HCT violated SLA upto 0.00008% and honeybee upto 0.00009%. Migration was able to reduce to 852 VM by using HCT where as honeybee 857 VM was migrated.

Further experimental result of Honeybee and HCT Placement in terms of Energy, SLA & Migration is shown in figure 5, results indicate that the HCT method improves the VM placement while reducing the migration and not violating the SLA (service level agreement) [3]. Simulation was done with 800 Hosts and 1052 VM in CloudSim and the experimental setup included detection of overload algorithm and Virtual Machine.

6. CONCLUSION

Table 1 gives a summary of the proposed VM placement run with different previous overload detection and VM selection. A Honeybee algorithm and HCT was implemented and experiment with various overload detection and VM selection algorithm. Parameter like CPU, SPEED, MEMORY were considered while forming the cluster. The result is based upon the energy consumption, VM migration and SLA. The proposed algorithms HCT showed a reduced in energy consumption, minimal in migration and SLA. Thus, our current findings strongly recommended the adoption of HCT.

REFRENCES:

[1] GunhoLeey, Byung-Gon Chunz, Randy H. Katz, "Heterogeneity-Aware Resource Allocation and Scheduling in the Cloud", University of California, Berkeley, Yahoo! Research

[2] U.S. Environmental Protection Agency. Epa report to congress on server and data center energy efficiency appendices, 2007.

[3] http://en.wikipedia.org/wiki/Service-level_agreement

[4] http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/hierarchical.htm

[5] Jinhua Hu, Jianhua Gu, Guofei Sun &Tianhai Zhao, "A Scheduling Strategy on Load Balancing of Virtual Machine Resources in Cloud Computing Environment", 3rdInternational Symposium on Parallel Architectures, Algorithms and Programming, 2010 pp89-96.

[6] Yee Ming Chen & Shin-Ying Tsai, "Optimal Provisioning of Resource in a Cloud Service", International Journal of Computer Science Issues, Vol. 7, No 6, 2012, pp95-99.

[7] Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I., Warfield, A.: Live migration of virtual machines. In: NSDI 2005: Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation, USENIX Association, Berkeley, 2005, pp273–286.

[8] Mauro Andreolini, Sara Casolari, Michele Colajanni, and Michele Messori, " Dynamic load management of virtual machines in a cloud architectures", CloudComp Munich, Germany, 2009, pp201-21.

[9] A. Stage and T. Setzer, "Network-aware migration control and scheduling of differentiated virtual machine workloads," in Proc. CLOUD'09. IEEE Computer Society, 2009, pp9–14.

[10] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Black-box and gray-box strategies for virtual machine migration," in Proc. NSDI'07. USENIX Association, 2007, pp229–242.

[11] R. Das, J. O. Kephart, C. Lefurgy, G. Tesauro, D.W. Levine, and H. Chan, "Autonomic multiagent management of power and performance in data centers," in PRoc. AAMAS'08. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2008, pp 107–114.

[12] S. Srikantaiah, A. Kansal, and F. Zhao, "Energy aware consolidation for cloud computing," Cluster Computing, vol. 12, 2009, pp. 1-15.

[13] D. Barbagallo, E. Di Nitto, D. J. Dubois, and R. Mirandola, "A bio-inspired algorithm for energy optimization in a

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

self- organizing data center," in Proc. SOAR'09. Berlin, Heidelberg: Springer-Verlag, 2010, pp127–151.

[14] G. Khanna, K. Beaty, G. Kar, A. Kochut, Application performance management in virtualized server environments, in: Network Operations and Management Symposium, 2006. NOMS 2006, 10th IEEE/IFIP, pp. 373–381, 2006. http://dx. doi.org/10.1109/NOMS.2006.1687567.

[15]http://en.wikipedia.org/wiki/Bees_algorithm

[16] Anton Beloglazov and RajkumarBuyya, "Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers," Software: Practice and Experience, 41(1), pp23–50, 2011

[17] Calheiros, R. N. , et al. : Cloudsim: A Novel Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services, Technical Report, GRIDS-TR-2009-1, Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Australia, 2009.

[18] Y. Gao et al., "A multi-objective ant colony system algorithm for virtual machine placement in cloud computing", J. Comput. System Sci, 2013

[19] Martin Randles, David Lamb, A. Taleb-Bendiab, "A Comparative Study into Distributed Load BalancingAlgorithms for Cloud Computing," IEEE 24th International Conference on Advanced Information Networking and Applications Workshops, 2010, pp551-556.

[20] S. Bitam, "Bees Life Algorithm for job scheduling in cloud computing", ICCIT 2012, 186-191, 2012.

[21] A. Bhadani, and S. Chaudhary, "Performance evaluation of web servers using central load balancing policy over virtual machines on cloud", Proceedings of the Third Annual ACM Bangalore Conference (COMPUTE), 2010

[22] A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy, "Optimal power allocation in server farms," in Proceedings of the 11th International Joint Conference on Measurement and Modeling of Computer Systems. ACM New York, NY, USA, 2009, pp157–168.

[23] Yiqiu Fang, Fei Wang, and JunweiGe, "A Task Scheduling Algorithm Based on Load Balancing in Cloud Computing", WISM LNCS 6318, 2010, pp271–277.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table1: HCT And HONEYBEE With Overload Detection And VM Selection

Overload Detection/ VM Selection	ENERGY		SLA		MIGRATION	
VM Placement→	HCT	HONEYBEE	HCT	HONEYBEE	HCT	HONEYBEE
DVFS	66.87	102.64	0.00000	0.00000	0	0
IQR/MC	34.71	41.90	0.00009	0.00012	854	889
IQR/MMT	36.52	41.47	0.00009	0.00013	865	931
IQR/MU	34.35	42.41	0.00010	0.00012	887	907
IQR/RS	34.29	44.44	0.00010	**0.00009**	**852**	869
LR/MC	34.02	44.56	0.00010	0.00010	856	900
LR/MMT	33.99	**41.45**	0.00010	0.00012	882	**857**
LR/MU	36.85	42.19	**0.00008**	0.00013	869	896
LRR/MC	36.52	44.17	0.00010	0.00011	896	885
LRR/MMT	35.17	46.62	0.00010	0.00010	874	841
LRR/MU	**33.47**	44.82	0.00011	0.00011	908	918
LRR/RS	35.71	45.38	0.00009	0.00010	861	879
LR/RS	34.75	41.90	0.00009	0.00012	867	948
MAD/MC	34.27	43.32	0.00011	0.00011	916	884
MAD/MMT	34.84	43.01	0.00010	0.00012	882	893
MAD/MU	34.70	40.31	0.00009	0.00013	873	875
MAD/RS	34.84	44.82	0.00009	0.00011	866	906
THR/MC	35.88	43.46	0.00009	0.00012	895	894
THR/MMT	34.30	43.45	0.00010	0.00011	868	921
THR/MU	34.88	43.46	0.00010	0.00012	854	905
THR/RS	34.27	44.51	0.00011	0.00010	908	911

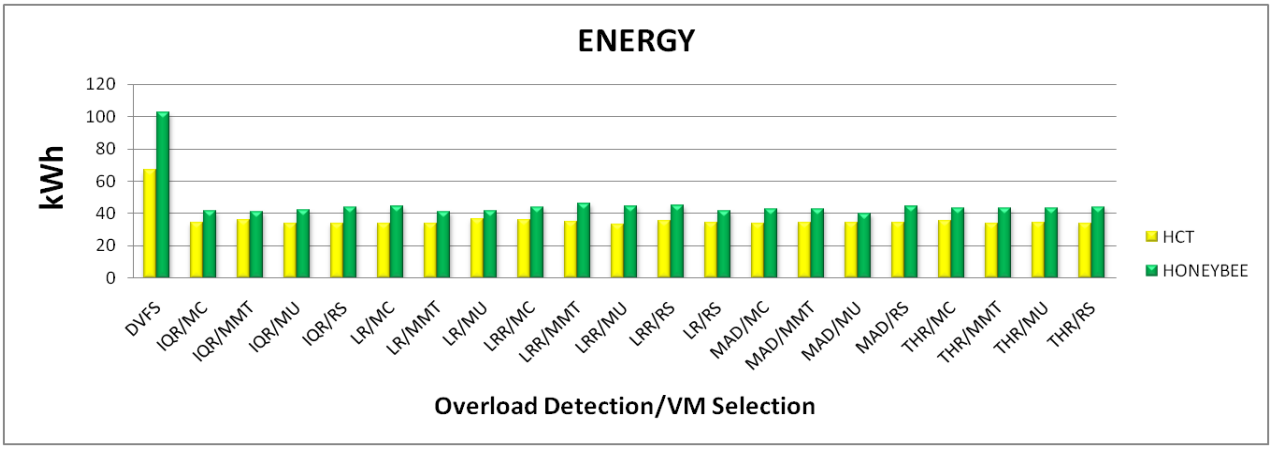

Figure 2 Energy Consumption Using HCT & Honeybee

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

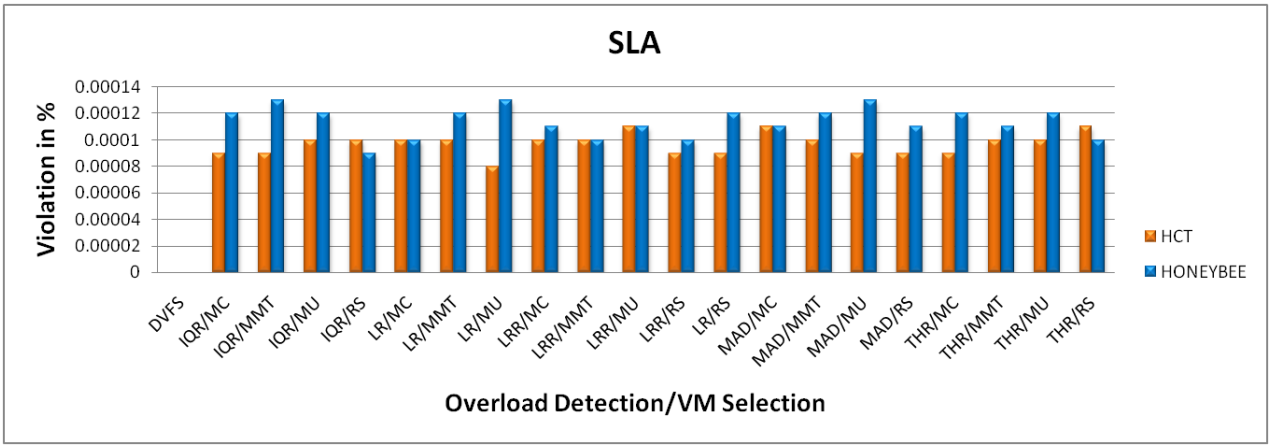

Figure 3 SLA Using HCT & Honeybee

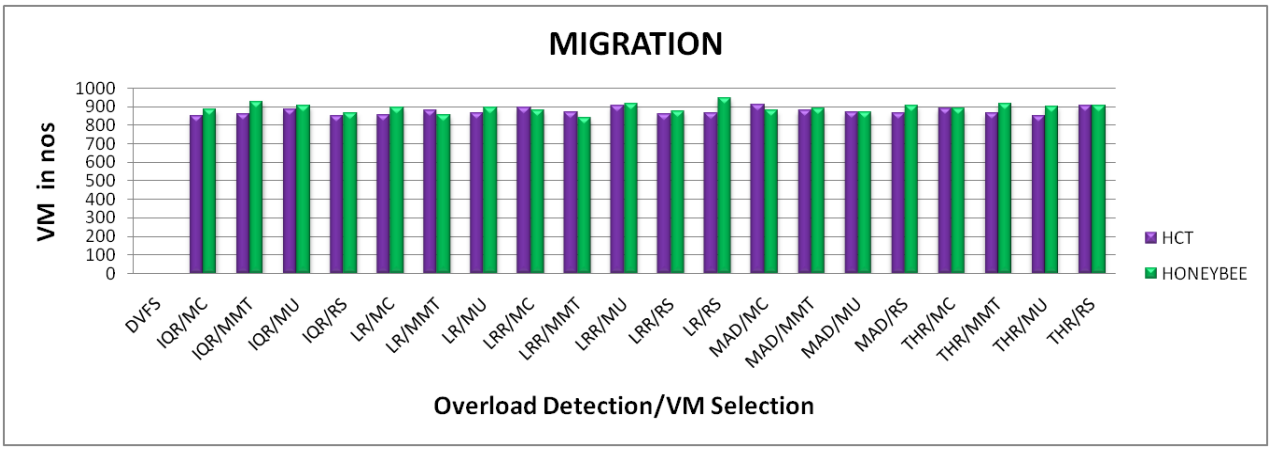

Figure 4 Migration Using HCT & Honeybee

Overload Detection: DVFS-Dynamic Voltage Frequency Scaling, IQR-Interquartile Range, LR-Local Regression, LRR-Robust Local Regression, MAD-Median Absolute Deviation, THR- CPU utilization threshold,

VM Selection Policy: MC-Maximum Correlation, MMT-Minimum Migration Time, RS-Random Selection

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Overload Detection/VM Selection	ENERGY kWh		SLA %		MIGRATION	
	HCT	HONEYBEE	HCT	HONEYBEE	HCT	HONEYBEE
LRR-MU	33.47					
LR-MMT		41.45				
LR-MU			0.00008			
IQR-RS				0.00009		
IQR-RS					852	
LRR-MMT						857

Figure 5. Experimental Result Of Honeybee And HCT Placement In Terms Of Energy, SLA & Migration