Indian Journal of Science and Technology, Vol 8(26), DOI: 10.17485/ijst/2015/v8i26/77179, October 2015
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

An Evolutionary Computing based Energy Efficient VM Consolidation Scheme for Optimal Resource Utilization and QoS Assurance

Perla Ravi Theja[1*] and S. K. Khadar Babu[2]

[1]School of Computing, Science and Engineering, VIT University, Vellore - 632014, Tamil Nadu, India; ravithejaperla9048@gmail.com
[2]School of Advanced Sciences, VIT University, Vellore - 632 014, Tamil Nadu, India; khadar.babu36@gmail.com

Abstract

Background: The increase in Cloud applications have demanded efficient cloud computing systems like Virtual Machine (VM) consolidation that intends to facilitate optimal resource utilization, energy conservation and quality of service. **Methods:** In this paper, an evolutionary computing technique called Adaptive Genetic Algorithm (A-GA) has been proposed for VM consolidation that encompasses under load and overload utilization detection, VM selection and placement, where the modified robust local regression and interquartile range schemes estimate the dynamic CPU utilization threshold for overload detection, minimum migration time works as VM selection policy, while A-GA optimizes VM placement across network to reduce energy consumption and SLA violation. **Findings:** PlanetLab Cloud benchmark data based simulation results confirms that the proposed VM consolidation scheme exhibits better than other existing approaches such as Ant Colony Optimization (ACO), Static Threshold (THR), Local Regression (LR), Conventional Inter Quartile Range (IQR) and Median Absolute Deviation (MAD) based virtualization schemes. The proposed system has exhibited minimal host shutdown, VM migration, energy consumption and SLA violation as compared to other existing approaches. **Applications:** Thus, the efficiency of the proposed VM consolidation scheme signifies that it can be a potential VM consolidation solution for large scale Cloud data centers.

Keywords: Adaptive Genetic Algorithm, Evolutionary Computing, Resource Utilization, VM Consolidation

1. Introduction

The high pace increase of Cloud computing has caused the establishment of the large scale data centres comprising thousands of computing nodes that requires huge energy consumption. There is an inevitable need of energy efficient cloud computing techniques. In addition, the high pace increase in the number of users in Cloud computing has raised varied challenges for Cloud service provider to ensure optimal Quality of Service (QoS) and reliability. Employing live migration[1] VM consolidation can be performed to leverage the fine-grained instabilities in the workload that ultimately can maintain minimal active hosts to conserve energy. In general, the dynamic VM consolidation comprises two fundamental processes; VM migration from underutilized physical machine and offloading VMs from PMs, when it suffers from overloading to avoid any possible performance degradation or SLA violation. In this process, the idle nodes are switched off to eliminate the static or active mode power consumption and as per requirements the nodes (PMs) are reactivated to accommodate migrated VMs. The modern Cloud infrastructures employ virtualization for optima resource utilization, minimal computational cost and energy efficient computing. Virtualization enables Cloud service providers to deal

* Author for correspondence

with the energy inefficiency by means of creating multiple Virtual Machine (VMs) instances on a physical server and thus enhancing the resources utilization. Ensuring optimal QoS defined through Service Level Agreements (SLA) is of great significance for Cloud computing environments; and hence the Cloud service providers need to optimistically and efficiently deal with the energy-performance trade off.

In general, there are four predominant phases in an efficient VM consolidation scheme. These are, under load host detection, overload host detection, VM selection and VM allocation. Certain efficient approaches for these key consolidation processes might lead to the efficient virtualization, thus providing optimal service provisioning and Return on Investment (RoI) for Cloud service providers. In this paper, a highly robust evolutionary computing technique, Adaptive Genetic Algorithm based VM consolidation has been developed that ensures optimal energy efficiency with enhanced SLA performance metrics. In this paper the modified Linear Regression (LR) and Inter-Quartile Regression (IQR) technique has been used for dynamic threshold based overload detection. In the proposed model, the Minimum Migration Time (MMT) policy has been considered for VM selection, which has been followed by Adaptive Genetic Algorithm based VM allocation for consolidation. The results obtained has exhibited that A-GA based consolidation performs better as compared to Ant Colony Optimization (ACO) based VM allocation policy. The proposed system has performed better energy efficiency while ensuring minimal SLA violation during virtualization.

The emergence of Cloud computing techniques and associated applications have motivated industries, researchers and a academician to develop certain optimal technique to make it more productive and energy efficient. Researcher[2–5] explored the VM consolidation and made effort to alleviate the issues of PM underutilization. Authors further stressed on the optimal situation when the relocation of VMs from certain overloaded PM for dynamic VM consolidation, which may enhance the QoS as well as resource utilization. Researchers employed static threshold based scheme[6] for VM allocation where the overall CPU utilization of the host was maintained to be in the range of defined thresholds. Later, it was observed that the scheduling with static threshold can't be the optimal solution due to dynamic Cloud pattern and workloads. Buyya et al.[7] optimized this concept by means of proposing an adaptive threshold for upper and lower bounds which was based on the statistical analysis of the CPU historical pattern. Researcher's developed regression approach for CPU utilization of certain PM in Cloud environment and employed linear regression and the K-nearest neighbour regression approach for approximating the data retrieved throughout the VMs lifetime[8,9]. Researchers employed this approach for SLA optimization. VM consolidation is the optimization issue and performed bin-packing problem optimization of the VM consolidation constraints such as capacity and SLA[10–12]. These algorithm tries to reduce the number if hosts while packing more and more VMs. Some other researchers have tried to work on certain combinatorial optimization issue by means of certain adaptive optimization technique[13–15].

A bio-inspired optimization technique was proposed for the minimizing the overall energy consumption by means of load distribution among servers[16]. Similarly a foraging of ant based resource allocation was proposed for cloud infrastrcture[17]. The evolutionary concepts such as Ant Colony Optimization (ACO)[18,32] and Genetic algorithm (GA)[19,31] were employed for VM consolidation problem. The decentralize schemes were developed using ACO based scheduling for VM consolidation[20,21]. The evolutionary approach GA has been explored in Cloud computing[22–24], where it has been found exhibiting better for resource scheduling and VM allocation. In this paper a robust and energy efficient VM consolidation scheme has been developed using dynamic threshold based CPU utilization estimation and adaptive genetic algorithm based VM placement schemes.

2. Our Contribution

This section discusses the proposed research model and its implementation.

2.1 System Architecture

In this paper, the Infrastructure as a Service (IaaS) has been considered as the targeted system which is comprised of a large-scale data canters having heterogeneous Physical Machines (PMs). The individual PMs has been characterised in terms of the CPU performance, which itself is defined in terms of the Millions Instructions Per

Second (MIPS), network bandwidth and RAM utilization. In fact, the considered servers don't possess any inbuilt storage facility and hence in the proposed Cloud model the storage has been facilitated using a Network Attached Storage (NAS) or Storage Area Network (SAN) that enables the live virtualization. The considered Cloud environment assumes that there is no existing information about the applications, functional time period and associated workloads for which virtualization has to be accomplished. It refers that the proposed resource management approach is an application-agnostic paradigm. In this paper, multiple independent users requests for heterogeneous VMs in terms of the required RAM space, resource bandwidth and MIPS. Here, the VM management by certain independent user can be stated to be the fact that the resulting workload generated because of the VMs consolidation on a single PM host is mixed workload, which is formed by combining numerous applications, which employs resources simultaneously. In general, the users form certain Service Level Agreements (SLAs) with the Cloud service provider to ensure Quality of Service (QoS) provisioning and in case of QoS violation; the service provider pays certain penalties. This research work is motivated towards developing a novel green computing based virtualization paradigm for large scale data centres while ensuring optimal QoS provisioning. In our paper, the proposed dynamic VMs consolidation approach considers four predominant sub-problems of live VM consolidation. These are:

- PM under load detection.
- PM overload detection.
- VM selection, and
- VM placement.

To ensure optimal scalability of the proposed model, the detection of under loaded or overloaded PMs and the optimal VM placement has been incorporated which is performed locally by individual computing host. Thus it facilitates that the application layer of the Cloud system is tiered encompassing the Local Managers (LMs) and Global Managers (GMs), as illustrated in the following figure (Figure 1). As illustrated in Figure 1, the LMs operate as Virtual Machine Management (VMM) module on the individual node. The prime objective of LMs is to perform continuous monitoring of the CPU utilization, overload or under load detection of the host machine or PMs. In case of overload at certain PM, it initiates the VM selection mechanism to estimate which VMs to be offloaded from that host node or PM. On the other hand, the Global Manager (GMs) collects information from the LMs so as to maintain the optimal Cloud resource utilization. On the basis of the decisions made by LMs, the Global Manager (GM) issues commands for VM migration for optimal VM placement. During this process, the VMMs exhibits virtualization as well updates the variation in the power modes of the host machines.

In our proposed model, the multi-core CPU architecture has been employed, which is briefed in the following section.

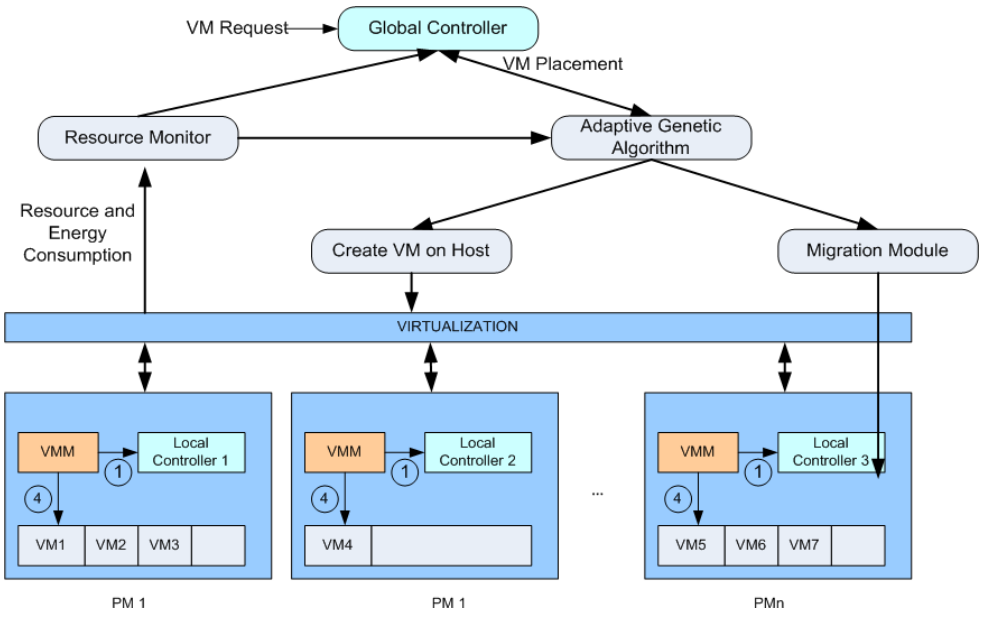

Figure 1. System architecture.

2.1.1 Multi-Core CPU Architectures

In our proposed model, the physical servers have been equipped with multi-core CPUs. The developed CPU architecture possesses cores each having MIPS which has been further modelled as a single-core CPU with the overall capacity of MIPS. In fact all the allied applications and VMs in the network are not much dependent on the CPU core and hence can be performed on certain arbitrary core by means of a time-shared scheduling mechanism, provided a limitation that the overall capacity of the individual virtual processing core allocated to a VM should be less than or equal to the overall capacity of a single physical CPU core. The predominant reason behind such limitation is that in case the required CPU capacity for a virtual CPU core becomes more than the actual capacity of a single physical CPU core, then the VM is needed to be executed on multiple parallel physical CPU cores, which seems impractical as the automatic VMs parallelization with a single virtual CPU seems infeasible.

2.1.2 The Power Model

In data centers the power consumption by computing nodes is determined by various factors such as CPUs, disk storage, RAM, power supplies and network cooling systems[25]. Researchers[26,27] have found that in Cloud servers the overall power consumption can be precisely defined in terms of a linear relationship between the CPU power consumption and CPU utilization, even with techniques like dynamic voltage and frequency scheduling. The prime reason behind this fact is that the performance and voltage scaling can't be incorporated with the other component of the Cloud architecture such as, RAM and network interfaces. On the other hand, the proliferation of multi-core CPUs and VMs consolidation techniques have made modern Cloud servers equipped with large scale memory that causes huge power consumption[28]. The power consumption can be defined as a function of the CPU utilization (P(u)), the mathematical expression for power consumption is as follows:

$$P(u) = k.P_{\max} + (1-k).P_{\max}.u = P_{\max}(0.7 + 0.3.u) \quad (1)$$

Where, P_{max} refers standard power consumption in modern computing servers, k refers the power consumed by an idle server while the CPU utilization is given by u. As the CPU utilization varies over time due to the workload variation, it can be a function of time u(t) and thus the overall energy consumption by a server is obtained as:

$$E = \int_t P(u(t))dt \quad (2)$$

2.2 The Cost of VM Live Migration

Virtualization or the live VMs migration enables relocating a VM between varied host nodes or PMs with minimal downtime and without causing functional suspension of the applications. However, this approach influences the performance of certain applications running on a VM during migration. In fact, the downtime and allied performance degradation rely on the specific application characteristics such as how many memory slots the application updates during its execution. In case of dynamic workload based applications, the average performance degradation caused due to downtime is approximately 10% of the total CPU utilization. It is significant to model the resource consumption by a VM while being migrated to the host node. The individual VM migration causes a defined SLA violation and hence the reduction in the number of VM migrations can be a significant approach to reduce energy consumption. The duration of the VM migration depends on the total amount of memory used by the VM and available network bandwidth as the workload images and VM's data are stored on a shared storage (NAS or SAN), which is required to enable live migration and hence the replication of the VM's storage is not needed. The migration time for certain VM VM_j can be calculated by the following equation.

$$T_{m_j} = \frac{M_j}{B_j}, \quad (3)$$

Where the amount of memory used by VM_j is M_j, and the available bandwidth is B_j.

Similarly, the overall performance degradation during migration can be obtained as follows:

$$U_{dj} = 0.1.\int_{t_0}^{t_0+T_{m_j}} u_j(t)dt \quad (4)$$

Where U_{dj} represents the overall performance degradation during migration, t_0 is the migration initiation time, T_{mj} refers the total time consumed in performing migration, $u_j(t)$ represents the total CPU utilization by node VM_j.

3. SLA Violation Metrics

In Cloud computing fulfilling the QoS requirements is an inevitable need and is of great significance. In Cloud infrastructure, the QoS requirements are in general formalized in terms of the SLAs. The SLAs can be estimated in terms of the network characteristics such as, minimum throughput or maximum response time delivered by the deployed system. These characteristics vary as per applications and therefore it is significant to define certain workload independent metric that could be employed for QoS estimation for any VM deployed on the IaaS. Furthermore, to ensure optimal QoS of the Cloud network, which is nothing else but the SLA violation, it is inevitable to maintain minimal SLA violation. In this paper, the overall SLA violation has been defined to be the difference between the requested MIPS by all VMs (U_{r_j} (t)) and the actual allocated MIPS (U_{a_j} (t)) moderately to the total requested MIPS over the life time of the VMs. Mathematically it can be obtained as:

$$SLA = \frac{\sum_{j=1}^{M} \int_{t} U_{r_j}(t) - U_{a_j}(t) dt}{\sum_{j=1}^{M} \int_{t} U_{r_j}(t) dt} \quad (5)$$

Where M represents the total number of VMs

In addition to the MIPS, in our proposed model, the CPU utilization (%) that couldn't be allocated when demanded by certain application relatively to the overall demand created by the VMs has been considered as an SLA metrics.

Sophistically, in our proposed model, two SLA metrics, one the duration through which the active host nodes or PMs have experienced 100% CPU utilization, called Overload Time Fraction (OTF); and second the performance degradation by VMs due to VMs migrations. Mathematically, these SLA metrics have been obtained by the following equations.

$$OTF = \frac{1}{N} \sum_{i=1}^{N} \frac{Ts_i}{Ta_i}, PDM = \frac{1}{M} \sum_{j=1}^{M} \frac{Cd_j}{Cr_j} \quad (6)$$

where N represents the total number of PMs or the hosts; M represents the total number of virtual machines; T_{si} refers the overall duration during which the PM i has experienced 100% resource causing an SLA violation; the total number of active hosts are T_{ai}; C_{dj} is the performance degradation of VM_j due to migration and the total CPU requested by VM_j during its life time is given by C_{rj}. Since, both the metrics OTF as well as PDM characterize the SLA violation individually and hence in our research model the overall SLA violations has been estimated using a combined SLA metrics, SLAV that encompasses both performance degradation cause because of the host overloading as well as the VM migrations. Mathematically, it is obtained as:

$$SLAV = OTF.PDM \quad (7)$$

The proposed energy efficient dynamic VM consolidation using genetic algorithm is given in the following section.

4. Energy Efficient Dynamic VM Consolidation using Evolutionary Computing

In this section, the proposed dynamic VMs consolidation has been discussed for IaaS Cloud environment. In our proposed research, the overall resource allocation and energy efficient VMs consolidation problem has been divided into four steps: first, to determine when a PM can be declared to be under loaded and thus migrating all the VMs from this host PM and turning it OFF or sleep mode to conserve energy; second is to determine when a PM can be stated to be overloaded asking for the migration of one or multiple VMs from this PM so as to minimize the load; third selecting specific VMs that might be migrated from certain overloaded PM; and fourth determining a robust and efficient VM placement or migration scheme to relocate VMs from the PMs. These problems have been discussed in the following sections.

4.1 PM under Load Detection

In this paper, the overloaded physical machines or the hosts have been found using the overload detection algorithm, which has been followed by the migration of the selected VMs to the destination PMs or hosts. Further, a compute host is found with the minimal utilization as compared to the other hosts and all the allied VMs are migrated from

this PM host to the other hosts, while ensuring them to be not overloaded. Completing the migration, the source PM is switched OFF or is turned into sleep mode so as to conserve energy. In case, all the VMs from the source PM can't be relocated to the other hosts, then it is continued to be active and these scheme is continued iteratively for all non-overloaded PMs.

4.2 Host Overload Detection

In overload host or PM detection, the individual host executes an overload detection algorithm periodically so as to perform de-consolidation of the VMs that avoids the degradation SLA performance. In order to detect the overloaded hosts, the CPU utilization threshold can be employed. In our research, a dynamic adaptive threshold estimation based overload detection scheme has been developed, where the proposed dynamic threshold scheme adjusts CPU utilization threshold on the basis of the strength of the deviation in the CPU utilization. It follows the principle that higher the deviation, lower the upper CPU utilization threshold. In fact, it is justified by the observation that higher deviation increases the probability of 100% CPU utilization causing SLA violation. Thus, the robust statistics can facilitate an optimal alternative to the traditional statistical approach. In this paper, we have used the Inter Quartile Range (IQR) and modified form of local regression technique for dynamic threshold estimation for CPU utilization. IQR represents the statistical dispersion which is equal to the differences between the third and first quartile. The CPU utilization threshold can be obtained by IQR estimation by means of following expression:

$$IQR = Q_3 - Q_1, \quad T_u = 1 - s.IQR \tag{8}$$

In addition to the IQR based CPU thresholding approach, in this paper we have employed Robust Local Regression (LRR) technique which is functional on the basis of modified Loess method[29]. In our paper, the LRR scheme, an enhanced form of LR to be used for fitting a trend polynomial to the last k observations of the CPU utilization and a modified method with a new trend line has been derived as $g(x) = a + b\,x$ for each new observation, which is later employed for estimating the next observation $g(x_{k+1})$. In case the following equation is satisfied, this scheme suggests offloading some VMs from the host.

$$s.\ g(x_{k+1}) \geq 1, \quad x_{k+1} - x_k \leq t_m \tag{9}$$

Where $s \in R^+$ represents the maximum extent of the tolerability of a host node and t_m is the maximum time needed for migration of any VM from the host node.

In fact the conventional Loess method is vulnerable to the outliers caused because of leptokurtic or heavy-tailed distributions. In order to eliminate such limitations and to make Loess efficient and robust, a modification has been proposed to bisquare to the Least-Squares (LR) scheme and a modified LRR scheme has been employed. Here, such modification has been done in an iterative manner and the initial fit has been performed with weights estimated by means of tricube weight function. In this approach, the fit is estimated at x_i to get the fitted values using y_i, and thus the residuals values would be $\varepsilon_i = y_i - y_i$. In ascending phase, the individual observation (x_i, y_i) is assigned certain robustness factor R_i which depends on the magnitude of ε_i. Thus, each observation has been assigned a weight factor $R_i w_i(x)$, where the robustness factor is presented as:

$$R_i = B\left(\frac{\varepsilon_1}{6s}\right) \tag{10}$$

Where $B(.)$ states the bisquare weight function and s represents the Mediun Absolute Deviation (MAD) for the least square fit. Mathematically,

$$B(.) = \begin{cases} (1-u^2)^2 \ if\ |u|, < 1, \\ 0\ Otherwwise \end{cases} \tag{11}$$

And

$$s = mediun|\varepsilon_i| \tag{12}$$

Employing the estimated trend line, equation (9) is used for calculating the next observation and in case the inequalities in (9) are satisfied, then the PM or host is stated to be overloaded.

6. VM Selection

Performing the host's utilization levels, under load and overload, in this paper, we have performed VMs selection so as to offload from the host to avoid certain

SLA performance degradation. In this paper, we have implemented the Minimum Migration Time (MMT) policy for VM selection. The Minimum Migration Time (MMT) policy performs the migration of a Virtual Machine (VM) v that needs minimal time to exhibit complete migration as compared to the other VMs allocated to the PM node or host. We have estimated the migration time in terms of the RAM being utilized by the VM divided by the additional network bandwidth accessible for the PM_j. Consider, V_j represents a set of VMs presently associated with the host node j. Thus, the proposed MMT policy determines a VM v by means of fulfilling the following conditions:

$$\upsilon \in V_j \mid \forall_a \in V_j, \frac{RAM_u(\upsilon)}{NET_j} \leq \frac{RAM_u(a)}{NET_j} \quad (13)$$

Where $RAM_u(a)$ represents the amount of RAM currently being utilized by the VM a; and NET_j refers the bandwidth available for migration from the host node j.

4.4 VM Placement

In general, VM placement is considered to be a case of bin packing with varying bin sizes and prices, where bins state the PMs; items are the VMs that have to be allocated; bin sizes represent the available resource or CPU capacities of the PMs; and the power consumption by PMs is referred by price. In fact, the problem of bin packing is NP-hard and therefore to alleviate the non-convexity problem, in this paper evolutionary computing technique name Adaptive Genetic Algorithm (A-GA) has been implemented. Genetic Algorithm has been integrated with the simulator tool CloudSim that comprises multiple data centers having multiple hosts. The individual PM has one or more Processing Elements (PE). The running VMs on the hosts PM have one or multiple running Cloudlets. In the implemented CloudSim, the user requests are represented as Cloudlets, where the processing power requirement of the individual Cloudlet is defined in terms of Million Instructions Per Second (MIPS). In our proposed A-GA based VM placement scheme, the scheduler module intakes the all PM nodes or hosts and generates a node mapping. It divides the complete MIPS into varied network components such as VMs and PMs running in parallel. In this paper, we made effort to minimize the issues of premature convergence by means of a robust and novel adaptive or self-adjusting mutation operator. A discussion of the implemented A-GA paradigm for VM placement is given as follows:

4.4.1 A-GA based VM Placement

The proposed A-GA scheme functions VM scheduling based on upper threshold values in order to satisfy transient variations in resource demand by existing VMs on certain specific PM. Here we have employed CPU utilization history for VMs placement onto the hosts or PMs. The A-GA approach intends to reduce migration and uses history of PMs for placing VMs on optimal one (hosts or PMs). Once placing the VM on certain PM, the proposed A-GA estimates the energy of data center and accordingly performs further scheduling so as to reduce energy consumption. In fact, our proposed A-GA algorithm focuses on reducing number of VM migration, SLA violation, SLA performance degradation, SLA time per active host and tries to achieve higher count of PMs Shutdown so as to conserve energy. A-GA algorithm initiates with a definite set of population where individual can be stated to be a tree comprising global controller as the root, PMs from the next level nodes and VMs as the Child nodes. For each such mapping it estimates the total energy consumption in the data center. The population consists of previously allocated mapping of virtual machines to the physical machines along with the data center energy for that mapping between some time interval and the future mapping of virtual machines, PMs and their estimated energy values. From the considered population A-GA chooses two VMs mapping having minimum energy values and then the genetic operators (Mutation and crossover) are applied on them. Thus, the mapping generated for VMs to the PMs is added to the overall population on the basis of the fitness values that in our research represents the CPU utilization. In our proposed A-GA, the crossover operator of performs selection of the PMs with the best CPU utilization based on the previous VMs mapping. Here, the crossover probability (P_r), and mutation rate (P_r) tries to minimize the PMs needed by means of shutting down one of the PMs, in case there are two PMs having CPU utilization less than the mutation rate of 0.5. The algorithms developed for crossover, mutation and overall VM placement are given in pseudocode-1, pseudocode-2 and pseudocode-3.

Pseudo Code 1. Algorithm for crossover

Input: *VM Mapping*
Output: *Crossover Propagation*
ParentVMMap1=previousVMMap
ParentVMMap2=findBestVMMap()
for all *PMs in ParentVMMap1*
If $(P_i.CPU_{Utilization} \geq P_r)$
PMsinChild.add(P_i)
end for
VMsleftfromChild=VMList-VMsinChild // VMsleftfromChild are the list of VMs which are not present in child mapping during crossover
if*(newVMs≠0)*
VMsleftfromChild.add(newVMs)
PMsnotInChild=PMsinParentVM.Map2-PMsinChild
While*(VMsleftfromChild≠0)*
VMi=max CPU UtilizationAmong(VMs leftfromChild)
Is*VMplaced=0*
For eachPM *in PMsinChild*
O_j=max $CPU_{Utilization}$*Among(PMsinChild)*
//remove Host temporary
If(P_j*.avg* ***CPU*** *Utilization≤Threshold &&((after Placing*VM_i *to* P_j*)* P_j*.CPU Utilization (t)<Threshold))*
Assign VMi to P_j
Is*VMPlaced≠****1***
end for
if*(isVMPlaced-False)*
if*(PMsnotInChild≠0)*
select P_j *from PMsnotInChild*
Assign VMi to P_j
isVMPlaced=1
PMsInChild.add(P_j)
PMsnotInChild.remove(P_j)
*if (isVMPlaced=****0****)*
//Select one switched off host P_k *from datacenter*
Assign VMi to P_k *PMsinChild.add(*P_k*)*
end while
end Crossover

Pseudo Code 2. Algorithm for mutation

From partial Child Map
//Select 2 PMs I and j such that
$P_i.CPU_{Utilization}(t) < P_m \&\& P_j.CPU_{Utilization}(t) < P_m$
//From host I and j select one of the best as s (s for select) and other as remaining (r for reject) such that
$P_s.avgCPU_{Utilization} <$ *Threshold* $\&\& P_s.CPU_{Utilization}(t) > P_r.CPU_{Utilization}(t)$
Remove all VMs from P_r *and Place* P_s.
End Mutation

Pseudo Code 3. Algorithm for VM placement using Adaptive Genetic Algorithm

Input: *VMs, PMs Energy utilization history*
Output: *VM placement map*
Generate the population size, Pop
Obtain the Best Individual in Population
while, *the termination condition is not true, do*
for *each Individual in Population Pop do*
Estimate the fitness value
end
For *each individual in Pop* ***do***
Use the Roulette selection to select another Individual to pair up;
end
for *each pair of parent* ***do****,*
Probabilistically employ the crossover function to generate an off spring
end
For *each individual in P* ***do***
Probabilistically apply the mutation function on the individual
end
Find the best individual in Pop;
if *the best individual in Pop is better than the current best individual then*
Replace the current Best Individual with the new best individual;
end
end
VM Migration Map=child VM Map-Previous VMMap

3. Experimental Setup and Results

In order to assess the efficiency of certain algorithm in real time scenarios, the simulation-based evaluation is of great significance and it becomes important to perform simulation with the workload traces retrieved from certain real system where it is intended to perform higher VMs consolidation on a PM. In addition, it is significant to employ full-fledged VM images exhibiting real time Cloud requirements. In this paper, we have used CoMon data project, which is a part of Cloud monitoring infrastructure for PlanetLab[30]. The implemented benchmark data comprises the CPU utilization by 1000+

VMs from the servers located at 100s of different places and the data has been collected during 10 randomly selected days in March and April 2011, where the measurement interval of the CPU utilization is 5 minutes. In order to evaluate the efficiency of our proposed system, we have developed a simulation model using the CloudSim toolkit, which has become highly popular and in the Cloud computing community because of its flexibility, scalability and reliable performance evaluation with real time Cloud data benchmarks. Because of the intricacies of the power consumption model in case of multi-core CPUs in our work, we have employed real benchmark data retrieved from CoMon data project, which is a part of Cloud monitoring infrastructure for PlanetLab30. In our simulation model, two server configurations with dual-core CPUs, one HP ProLiant ML110 G5 with Intel Xeon 3040, 2 cores x 1860 MHz processors, 4GB RAM, another and HP ProLiant ML110 G5 having specification for Intel Xeon 3075, 2 cores x 2660 MHz, 4 GBRAM have been used. We have mapped the frequency of the servers onto MIPS ratings where HP ProLiant ML110 G4 server is mapped with 1860 MIPS and 2660 MIPS for the HP ProLiant ML110 G5 server. Here the individual server is having 1 GB/s network bandwidth. In this paper, we have compared the performance of the proposed A-GA based VM consolidation and other existing approaches such as conventional genetic algorithm31Ant Colony Optimization (ACO)[32], Static Threshold (THR). Local Regression (LR), Inter Quartile Range (IQR) and Median Absolute Deviation (MAD). The simulation has been done while considering Minimum Migration Time as the VM selection policy.

In fact, VM migration incurs computational overhead on the data center and causes unwanted energy utilization. The results obtained in this paper exhibited minimum VM migration (Figure 2.) and maximum host shut down (Figure 6.), which depicts that the proposed (Adaptive Genetic Algorithm) can be a potential candidate for energy efficiency cloud resource utilization and VM consolidation. On the other hand, Figures (Figures 3, 4, 5, 7) illustrates the robustness and higher efficiency of proposed A-GA based VM consolidation over other exiting approaches. Figure 7 depicts better energy consumption efficiency of the proposed system as compared to other approaches where, in addition it also justifies the concept that higher host shutdown and lower VM migration might reduce energy consumption significantly. Thus, the proposed model can accomplish both the QoS (SLA performance) as well as energy efficiency for optimal resource utilization and green computing system for cloud data centers.

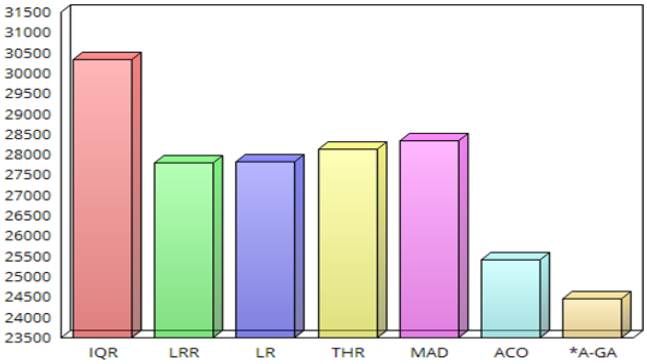

Figure 2. Number of VM migration.

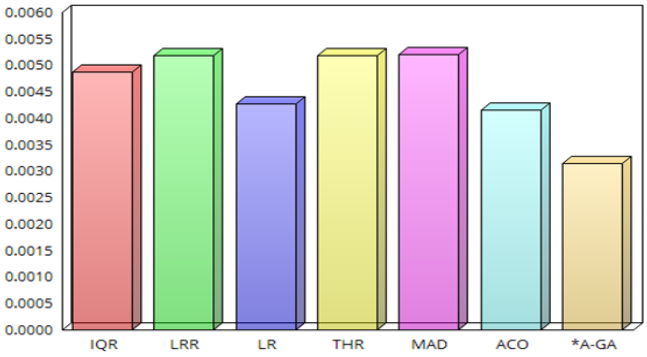

Figure 3. SLA Violations.

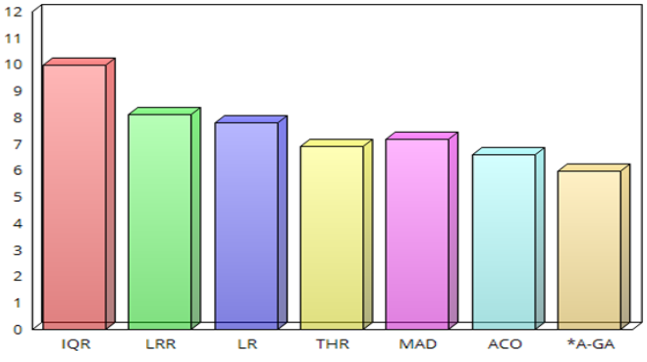

Figure 4. SLA performance degradation.

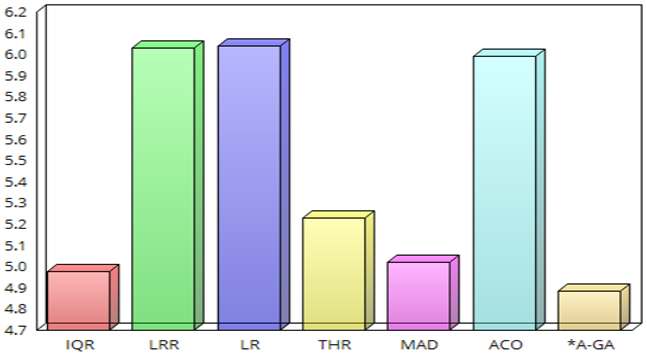

Figure 5. SLA time per active host.

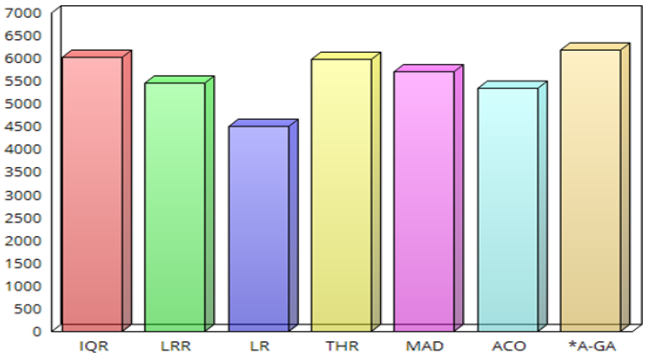

Figure 6. Number of host shutdown.

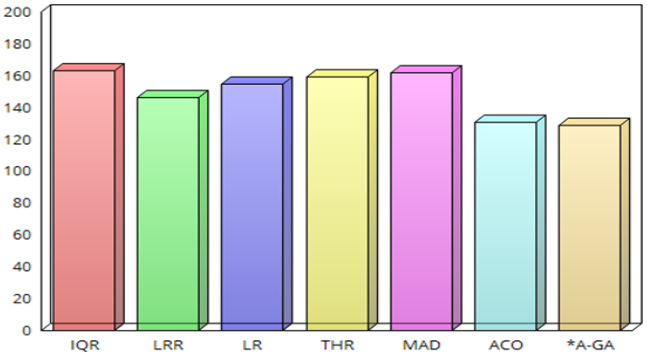

Figure 7. Energy consumption.

9. Conclusion

In this paper, a highly robust and efficient dynamic Virtual Machine (VM) consolidation mechanism has been developed to minimize the energy consumption in the Cloud data centers by means of consolidating more VMs into minimal active Physical Machines (PMs) as per the current resource requirements. The proposed system has incorporated a distributed controller that divides VM consolidation issues into four sub-problems: under load node detection, overload node detection, VM section and VM placement optimization. In this paper, the overload PM has been detected using dynamic thresholding approaches using linear regression and inter quartile range technique. The simulation made with CloudSim benchmark data images, the proposed Adaptive Genetic Algorithm (A-GA) based VM consolidation scheme has exhibited better results as compared to other heuristic approaches and Ant-Colony Optimization based VM consolidation. The developed model has exhibited higher energy efficiency and higher SLA performance with minimal violation as compared to existing approaches of the VM consolidation. This research has focussed only on the optimal consolidation and energy efficiency while ensuring optimal SLA. In this paper, Minimum Migration Time (MMT) policy has been used for VM selection but in future, other selection policies such as random selection; maximum correlation etc can be incorporated for respective performance assessment to ensure effective VM consolidation.

10. References

1. Clark C, Fraser K, Hand S, Hansen JG, Jul E, Limpach C, Pratt I, Warfield A. Live migration of virtual machines. Proceedings of the 2nd USENIX Symposium on Networked Systems Design and Implementation (NSDI); Boston. 2005. p. 273–86.
2. Vogels W. Beyond server consolidation. ACM Queue. 2008 Mar; 6(1):20–6.
3. Feller E, Morin C, Esnault A. A case for fully decentralized dynamic VM consolidation in Clouds. Cloud Computing Technology and Science (CloudCom). 2012 IEEE 4th International Conference; Taipei. 2012 Dec 3-6. p. 26–33.
4. Murtazaev A, Oh S. Sercon: Server consolidation algorithm using live migration of virtual machines for green computing. IETE Technical Review. 2011 May; 28(3):212–31.
5. Marzolla M, Babaoglu O, Panzieri F. Server consolidation in Clouds through gossiping. IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks (WoWMoM); Lucca. 2011 Jun 20-24. p. 1–6.
6. Beloglazov A, Abawajy J, Buyya R. Energy-aware resource allocation heuristics for efficient management of data centres for Cloud computing. Grid Computing and eScience. Future Generation Computer Systems (FGCS). 2012 May; 28(5):55–68.
7. Beloglazov A, Buyya R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centres. Concurrency and Computation: Practice and Experience. 2012 Sep; 24(13):1397–420.

8. Farahnakian F, Liljeberg P, Plosila J. LiRCUP: Linear regression based CPU usage prediction algorithm for live migration of virtual machines in data centres. 39th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA); Santander. 2013 Sep 4-6. p. 357–64.
9. Farahnakian F, Pahikkala T, Liljeberg P, Plosila J. Energy aware consolidation algorithm based on K-nearest neighbor regression for Cloud data re. 6th IEEE/ACM International Conference on Utility and Cloud Computing (UCC); Dresden. 2013 Dec 9-12. p. 256–9.
10. Wood T, Shenoy P, Venkataramani A, Yousif M. Sandpiper: Black-box and gray-box resource management for virtual machines. Computer Networks. 2009 Dec; 53(17):2923–38.
11. Ajiro Y, Tanaka A. Improving packing algorithms for server consolidation. Proceedings of the International Conference for the Computer Measurement Group (CMG); San Diego. 2007. p. 399–406.
12. Wang M, Meng X, Zhang L. Consolidating virtual machines with dynamic bandwidth demand in data centres. Proceedings on IEEE in INFOCOM; Shanghai. 2011 Apr 10-15. p. 71–5.
13. Harman M, Lakhotia K, Singer J, White DR, Yoo S. Cloud engineering is search based software engineering too. Journal of Systems and Software. 2013 Sep; 86 (9):2225–41.
14. Dorigo M, Caro GD, GambardellaL M. Ant algorithms for discrete optimization. Artif Life. 1999 Apr; 5(2):137–72.
15. Dorigo M, Gambardella L. Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation. 1997 Apr; 1(1):53–66.
16. Barbagallo D, Nitto D, Dubois DJ, Mirandola R. A bio-inspired algorithm for energy optimization in a self-organizing data centre. Self-Organizing Architectures. Springer; 2010. P. 127–51.
17. Chen H, Xiong L. Cloud task scheduling simulation via improved ant colony optimization algorithm. Journal of Convergence Information Technology (JCIT). 2013; 8(7):1139–47.
18. Dong YS, Xu GC, Fu XD. A distributed parallel genetic algorithm of placement strategy for virtual machines deployment on Cloud platform. The Scientific World Journal. 2014; 2014(2014):1–12.
19. Esnault A, Feller E, Morin C. Energy-aware distributed ant colony based virtual machine consolidation in IaaS Clouds bibliographic study. Informatics Mathematics (INRIA). 2012 Jan;1–13.
20. Feller E, Morin C, Esnault A. A case for fully decentralized dynamic VM consolidation in Clouds. IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom); Taipei. 2012 Dec 3-6. p. 26–33.
21. Ferdaus MH, Murshed M, Calheiros RN, Buyya R. Virtual machine consolidation in Cloud data centres using ACO metaheuristic. Euro-Par 2014 Parallel Processing; Porto. 2014. p. 306–17.
22. Zhong H, Tao K, Zhang X. An approach to optimized resource scheduling algorithm for open-source Cloud systems. IEEE Fifth Annual ChinaGrid Conference (ChinaGrid); Guangzhou. 2010 Jul 16-18. p. 124–9.
23. Madhusudhan B, Sekaran KC. A genetic algorithm approach for virtual machine placement in Cloud; 2013. p. 115–22
24. Tang M, Pan S. A hybrid genetic algorithm for the energy-efficient virtual machine placement problem in data centers. Neural Processing Letters. 2015 Apr; 41(2):211–21.
25. Minas L, Ellison B. Energy efficiency for information technology: How to reduce power consumption in servers and data centres. Intel Press; 2009.
26. Fan X, Weber WD, Barroso LA. Power provisioning for a warehouse-sized computer. Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA); San Diego. 2007. p. 13–23.
27. Kusic D, Kephart JO, Hanson JE, Kandasamy N, Jiang G. Power and performance management of virtualized computing environments via look ahead control. Cluster Computing. 2009 Mar; 12(1):1–15.
28. Minas L, Ellison B. Energy Efficiency for Information Technology: How to Reduce Power Consumption in Servers and Data Centers", Intel Press, 2009;
29. Cleveland WS. Robust locally weighted regression and smoothing scatterplots. Journal of the American Statistical Association. 1979; 74(368):829–36.
30. Park KS, Pai VS. CoMon: A mostly-scalable monitoring system for Planet-Lab. ACM SIGOPS Operating Systems Review. 2006 Jan; 40(1):65–74.
31. Thiruvenkadam T, Kamalakkannan P. Energy efficient multi dimensional host load aware algorithm for virtual machine placement and optimization in Cloud environment. Indian Journal of Science and Technology. 2015 Aug; 8(17):1–11.
32. Ashwin KS, Rahul R, Dheepan P, Sendhil KS. An optimal ant colony algorithm for efficient VM placement. Indian Journal of Science and Technology. 2015 Jan; 8(S2):156–9.