Research Article

International Journal of Advanced Technology and Engineering Exploration, Vol 8(83)
ISSN (Print): 2394-5443 ISSN (Online): 2394-7454
http://dx.doi.org/10.19101/IJATEE.2021.874481

An adaptive threshold policy for host overload detection in cloud data centre

Bhagyalakshmi[1*] and Deepti Malhotra[2]
Research Scholar, Department of CS&IT, Central University of Jammu, J&K, India[1]
Assistant Professor, Department of CS&IT, Central University of Jammu, J&K, India[2]

Received: 22-July-2021; Revised: 16-October-2021; Accepted: 19-October-2021
©2021 Bhagyalakshmi and Deepti Malhotra. This is an open access article distributed under the Creative Commons Attribution (CC BY) License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract
Excessive resource usage in a cloud computing system causes an increase in operational costs. In contrast, shortage of resources results in increased load on the server leading to Service Level Agreement Violation (SLAV) and reduced Quality of Service (QoS). Since, the workload is highly dynamic in nature, maintaining the utilization at optimal levels to effectively consume energy while keeping SLA integrated is a challenging task. To deal with the variability of the workload, the current research study focuses on calculating the upper threshold using past Central Processing Unit (CPU) utilization with the help of an adaptive threshold policy. The proposed P_n estimator based Adaptive Threshold policy ($P_n\_AT_HP$) scheme implements an estimator P_n that periodically analyses the previous CPU utilization data of a host machine and sets the upper threshold accordingly. The results so obtained against traditional schemes of overload detection, show improvement in the performance metrics. According to the simulation analysis, the proposed $P_n\_AT_HP$ host overload detection scheme shows an improvement of 41.83% and 44.70%, compared to existing LRMMT and SNMMT schemes, in terms of the combined metric of energy, SLAVs, and the number of Virtual Machine (VM) migrations.

Keywords
Cloud computing, Dynamic virtual machine consolidation (DVMC), Host overload detection, Dynamic threshold.

1.Introduction
Cloud computing technology exploits virtualization with the help of the internet and remote servers to provide hassle free services to its users. With the help of virtualization, the resources of a physical server are divided into multiple Virtual Machines (VMs), where each VM seems like an independent entity having its own capacity for performing computations and processing the data. The demand for cloud computing services has risen exponentially in the last decade because of its ability to easily host applications anytime and anywhere. This exponential growth leads to an enormous increase in energy consumption. Also, the pandemic situation occurred due to coronavirus has emphasized the online work environment for almost all organizations and institutions. This has taken the energy consumed by data centers to the next level. This raised energy consumption is a source of increased CO_2 emissions contributing to global warming.

The motivation behind the current research study is the linearly increasing energy consumption of the Cloud Data Centers (CDCs). A report by Cisco Systems [1] claims that due to the introduction of 5G, the mobile traffic will be seven times more in 2022 than in 2017.

Koot and Wijnhoven [2] predict the energy consumption by data centers to increase from 292 TWh in 2016 to 353 TWh in 2030. However, according to Hintemann and Hinterholzer, the amount of energy consumed by data centers in 2016 is forecasted to get doubled by 2030 [3]. According to the literature [4], data centres work at an average Central Processing Unit (CPU) utilization of 20% and an idle server consumes 70 % of its power ratio. Also, majority of the servers work at 10-50% [5] of their capacities and the remaining resources are wasted. The under-utilized servers lead to the wastage of power making the system inefficient whereas, the over-utilized servers lead to the degradation of the system performance. To guarantee Quality of Servic (QoS) while reducing the consumed energy aimed to avoid Service Level Agreement

*Author for correspondence

violation (SLAVs) is a challenging task for the Cloud Service providers (CSPs). The vast energy consumption is not due to high computing resources but the inefficient use of the available resources. The energy efficiency factor is greatly impacted by the server utilization. The more the server utilization, higher is the energy efficiency and decreases with a decrease in the utilization [6]. Therefore, there exists a trade-off between utilization efficiency and the performance. Many studies have been put forward to deal with the issue of resource management.

Dynamic VM consolidation is one effective way of improving resource utilization and optimizing energy efficiency while reducing SLAVs. The method of dynamic VM consolidation helps in minimizing the energy requirements of a data centre [7]. In dynamic consolidation the VMs migrate from overloaded and underloaded hosts to normally loaded hosts while ensuring that the migration does not violate the set threshold values. The goal of the CSPs is to leverage consolidation scheme so that the under-utilized can be turned off after migration and by distributing the load of the overloaded ones. Many researchers set a static threshold value and keep the server utilization to maximum level. However, such static thresholds may lead to inefficient resource utilization in light workload and may lead to SLAV with heavy workloads. Such a situation may not be appropriate for varying workloads. The dynamic nature of the workloads makes resource management a challenging task and can be dealt with dynamic thresholds. The objective of the research is to enhance the performance of the cloud environment by minimizing the amount of consumed energy. Main contributions of the paper are:

- To propose an adaptive P_n estimator-based host overload detection policy predicting future threshold that considers past CPU utilization.
- Performance evaluation of the proposed scheme against various baseline Dynamic Virtual Machine Consolidation (DVMC) schemes using PlanetLab workload.
- Performance evaluation of the proposed scheme against S_n based Host Overload Detection (S_nBODA) [8] using PlanetLab workload.

The rest of the paper is organized as follows: Section 2 gives a detail of literature carried out for host overload detection. The proposed host overload detection scheme, P_n estimator based Adaptive Threshold policy ($P_n\_AT_HP$) has been explained in Section 3. The experimental setup and the simulation results are shown in Section 4. Section 5 presents the discussions and limitations of the proposed work. Section 6 concludes the paper.

2.Literature review

In literature, many researchers have focussed on VM consolidation process to improve the energy efficiency of the cloud system. *Figure 1* shows the process of consolidation involving overloaded host detection, wherein the time for consolidation to start is selected, VM selection and VM placement. The very first step is the detection of overloaded or the underload hosts. Once such hosts are identified, the next step is the selection of the VMs for migration from these hosts. And lastly, the final step constitutes of the placement of these migrated VMs to other hosts.

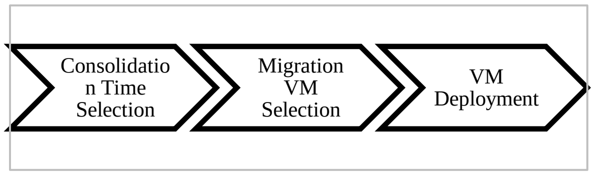

Figure 1 Process of consolidation

The threshold of the host is set according to the utilization of the resources and the overload of the host is checked against this threshold. The upper and the lower thresholds help in deciding upon which hosts are underloaded and how many are overloaded. The techniques suggested by various researchers for host detection can be broadly divided into two categories: static threshold and dynamic threshold. In case of static threshold, a fixed value is set for both upper limit and lower limit. Whereas, in dynamic threshold past utilizations are taken into account and a threshold value is set dynamically.

According to Zhu et al. [9], the system is considered to be overloaded if the utilization exceeds the set value of 85%.

Gmach et al. [10] has also followed the scheme proposed by Zhu et al. [9]. Beloglazov et al. [11] has also proposed a static threshold technique based upon double threshold. The upper threshold for finding the overloaded hosts and the lower threshold to look for the under loaded ones. The resource under consideration for evaluating the utilization is the CPU. Carrying forward the work of [11], the authors [12] have suggested a technique considering both CPU and disk utilization for setting of the threshold.

Fard et al. [13] has focussed on the temperature ratio for fixing the upper limit of utilization.

The researchers Xiao et al. [7] has put forward a double threshold scheme where the upper threshold is set at 80% whereas the lower has been set at 40%. Though static thresholds contribute to better energy efficiency, but are not suitable for dynamic and varying workloads. Various dynamic threshold-based load detection techniques have been proposed by numerous researchers.

Buyya et al.[14] have formulated a scheme which takes into consideration the median of deviations from the median of the past CPU utilization. The technique was termed as Median Absolute Deviation (MAD). Otherwise good, but MAD does not well for non-Gaussian distributions. Based upon the values of the mean and the standard deviation of the resource utilization.

Monil and Rahman [15] proposed an energy efficient load detection scheme. Various authors [16, 17] have deployed probabilistic approaches for the load computation and threshold calculation. Many authors proposed regression-based techniques, some used the concept of machine learning and Markov chains [18–20] while few proposed fuzzy learning-based techniques [21, 22].

Farahnakian et al. [23] proposed a scheme Linear Regression Based CPU Usage Prediction (LiRCUP) based on Linear Regression (LR) and used the regression technique to predict the short term future requirements. To predict the future load in the system, the authors [24] have proposed a scheme by integrating LR and back propagation. The proposed algorithm is self-adaptive in nature and experimental results prove its superiority over back propagation and LR.

With the aim of minimizing the median value, the authors Yadav and Zhang [25] concentrated on the value of the squared residuals.

Host Overloading Prediction Based on Logistic Regression (HOPBLR) proposed by Jararweh et al. [26] combines the properties of logistic regression and MAD to compute the upper threshold.

Whereas Mapetu et al. [27] suggested a technique, Pearson Host Overload Detection, that associates Pearson Correlation with logistic regression. The authors have presented a load detection algorithm based on MAD using Markov model. To achieve the desired goal and perform the statistical analysis, first order Markov chains have been used by the authors.
In an attempt to provide a dynamic threshold value, Bala and Padha [8] have modified MAD and suggested a location-free estimator that computes the median of absolute deviations. However, the scheme doesn’t perform well in terms of QoS and leads to higher SLAVs.

Xie et al. [28] has proposed a LR-based predictive scheme that considers multiple resources for the computation of the upper threshold. The problem with the issues depending on LR is that LR looks only at the mean of the dependent variable. However, the threshold needs to consider the extreme values as well.

Zhou et al. [29] used the ascending and descending trends of forecasting techniques to set an upper threshold value. Though dynamic in nature but the threshold varied only between 85%-90%. Many researchers have worked on minimizing the energy consumed by a data center, but energy efficiency remains an open challenge to be addressed. Overload situations in the computing environment cause high energy consumption.

In an attempt to deal with the dynamic workload, Sharma and Saini [30] has suggested a median based adaptive threshold scheme. Though the scheme achieved reduction in the violations of SLA but the issue of energy consumption by the data center remained unattended.

Farahnakian et al. [31] proposed a self-adaptive scheme for host overload detection. However, the disadvantage of the scheme lies in the use of reinforcement learning. The use of reinforcement learning makes the proposed scheme complex as it requires large amount of data and computation. Too much reinforcement learning can lead to the overload states which can diminish the results.

For host overload detection, Minarolli et al. [16] tried to incorporate the long term prediction methods. However, having a large value of time interval for the prediction may lead to skipping some overload states that do not last long. Thus, increasing the number of prediction time intervals does not increase the stability and performance of the approach.

Significant loss of energy has been observed with varying workloads in the work presented by Dambreville et al. [32]. Overload and underload prediction estimates lead to compromising the energy consumption.

Li et al.[12] tried to reduce the number of VM migrations but could not help reduce energy consumption. The possible reason can be the use of Bayesian networks that are highly sensitive to the probabilities priory chosen.

Saadi and El [33] proposed an overload detection scheme based upon the ratio of performance to power. The scheme exhibited high-performance rate in terms of energy efficiency and QoS but the ratio is calculated at 90% CPU utilization, which is static in nature. Based upon the study of the host overload detection schemes, the latter can be categorised as shown in *Figure 2*. The overload detection schemes have been classified as static or dynamic depending upon the type of threshold used. The dynamic threshold based host overload detection schemes are further divided as schemes based on regression, machine learning, statistical method based and fuzzy logic based.

On the contrary to prior work in the literature, the proposed scheme considers the dynamic and asymmetric behaviour of the data center workload while deciding on the value of the upper threshold. One of the ways to deal with the fluctuating host load is to provide an adaptive threshold policy that can efficiently handle the spikes in utilization. The current research proposes a scheme based on P_n estimator. The scale estimator makes use of pairwise means to add dynamicity to the nature of the threshold. The advantage of the estimator lies in its high efficiency of 86% with Gaussian distribution. The workload traces in a CDC are highly asymmetric, and the estimator works very well with such distributions. Depending upon the dispersion of the past utilization, a new threshold is set to deal with the dynamic workload.

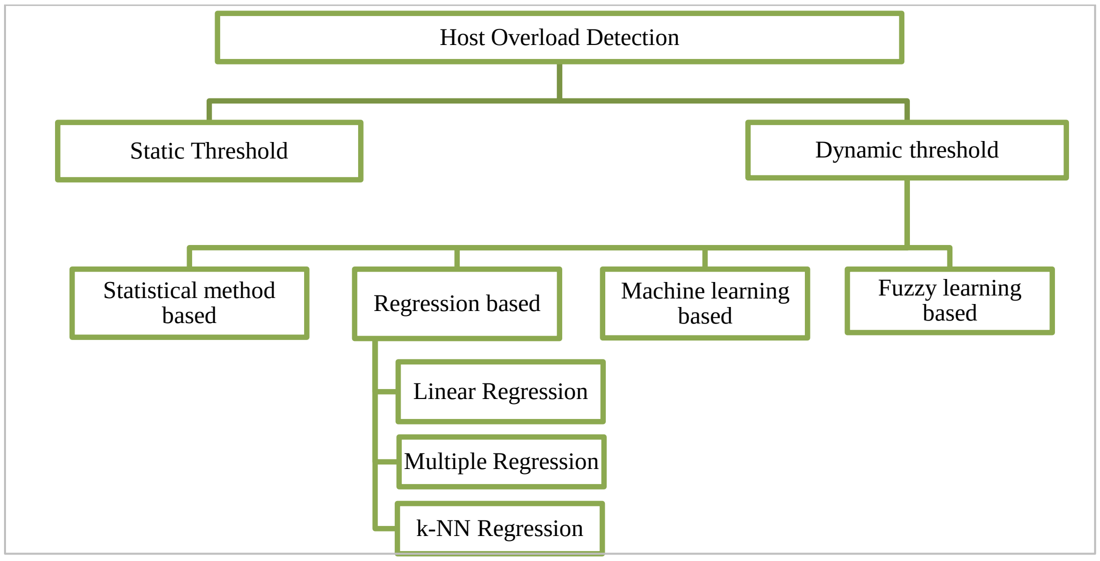

Figure 2 Classification of host overload detection techniques

3.P_n estimator based adaptive threshold policy ($P_n\_AT_HP$)

The $P_n\_AT_HP$ is based on the analysis of data for CPU utilization, collected in near past. Workload handled by a data center are dynamic in nature and follow an asymmetric dispersed pattern. These dispersions can be studied for past n time frames to adjust the upper threshold of the host for near future. Dispersion of CPU utilizations in the past has been used to set the upper threshold. If the deviation of CPU utilization comes less, then set threshold is low and vice versa. Most of the researchers have utilized the concept of traditional estimators for finding the deviations. However, those classical estimators are not robust to outliers. To have a dynamic upper threshold, the research study implements an estimator P_n , based upon the pairwise means. The motivation behind using P_n estimator is that it does not require a location estimate and is highly efficient with wide range of distributions [34].

3.1 $P_n\_AT_HP$ framework

Figure 3 presents the proposed P_n estimator-based framework, $P_n\_AT_HP$. To check whether a host is overloaded or not, $P_n\_AT_HP$ comes into play. The past CPU utilization of the host is analysed using the scale estimator. The value of the future upper threshold has been set with the help of the scale estimator. Thus, the upper threshold remains dynamic in nature, depending upon the utilization of CPU in the past time frames. A host exceeding its upper threshold is considered an overloaded one, and the migration process is initiated upon its detection. The reason behind the migration is to avoid degradation of system performance owing to the load imbalance. Few VMs are migrated to another suitable host, and the load is managed.

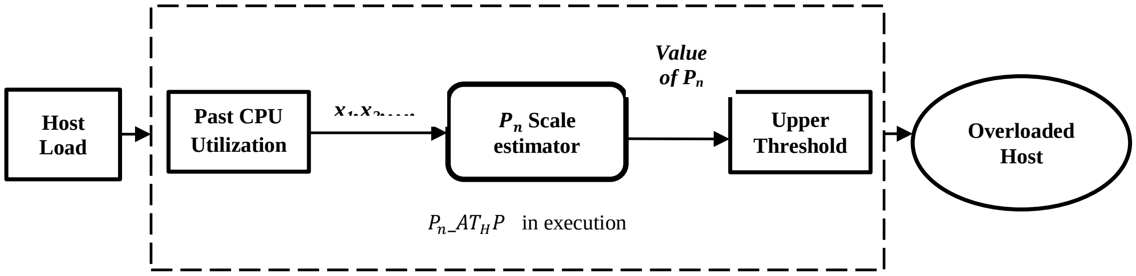

Figure 3 $P_n\_AT_HP$ framework

3.1.1Description of each component used in the $P_n\_AT_HP$ Framework

a) **Host Load:** This is the first component of the diagram and specifies the load of the host which is dynamic in nature. The execution of the proposed scheme begins by considering the dynamic host load.

b) **Past CPU utilization:** This component is responsible for retrieving the CPU utilization for past *"n"* time frames.

c) **P_n Scale estimator**: This component is used to statistically analyze the past CPU utilization. As proposed by Tarr et al. [34] in 2012, the P_n estimator is analogous to Inter Quartile Range (IQR) and in its most basic form is calculated as the interquartile range of the pairwise means.

According to Tarr et al. [34], P_n is defined as follows (Equation 1-3).

$$\boldsymbol{P_n(\tau) = c_\tau\left[H_n^{-1}\left(\frac{1+\tau}{2}\right) - H_n^{-1}\left(\frac{1-\tau}{2}\right)\right]} \qquad (1)$$

$$\boldsymbol{H_n(t) := \frac{2}{n(n-1)} \sum_{i<j} II\left\{h(x_i, x_j) \le t\right\}} \qquad (2)$$

$$\boldsymbol{H_n^{-1}(p) := inf\{t : H_n(t) \ge p\}} \qquad (3)$$

$H_n(t)$ is the empirical distribution of the pairwise means given$(x_1, x_2, x_3, \ldots\ldots\ldots, x_n)$ a set of independent values with $\{h(x_i, x_j), 1 \le i < j \le n\}$, a set of $\binom{n}{2}$ pairwise means, such that $h(x_1, x_2) = (x_1 + x_2)/2$.
According to Tarr and Muller, $\tau = 0.5$, works extremely well for a wide range of distributions. And, therefore, Equation 1 can be reduced to Equation 4.

$$\mathrm{P_n(0.5) = c_\tau[H_n^{-1}(0.75) - H_n^{-1}(0.25)]} \qquad (4)$$

d) Upper threshold: The input to this component is the value evaluated by P_n scale estimator. The value so obtained is used for computing the upper threshold using Equation 5.

$$Thr_{upp} \leftarrow 1 - s \times P_n(\tau) \qquad (5)$$

Where s is the safety parameter and the value is set to 1.5 for experimental purposes. Thus, upper threshold is calculated for each host dynamically depending upon its past utilizations.

e) Overloaded host: This component takes as input the computed upper threshold Thr_{upp} calculated in Equation 5 and checks the host load with the input value of upper threshold. If the host load exceeds the threshold value, it is considered as an overloaded host.

3.1.2$P_n_AT_HP$ estimator workflow for host overload detection

Let $X = (x_1, x_2, x_3, \ldots\ldots\ldots, x_n)$ be the set of CPU utilization of a host h taken over past n frames. P_n estimator has been used to add dynamicity to the upper threshold value. Algorithm 1-4 has been used for evaluating the value of the robust estimator P_n, which in turn helps in finding the upper threshold. This flow of execution is shown in *Figure 4*. The process begins with the generation of a set of pairwise means from past CPU utilization. Next, the empirical and inverse empirical distribution functions are computed. This helps in calculating the value of the scale estimator that provides with the value of the upper threshold.

Algorithm 1. $\boldsymbol{Computepairmeans}$ ()

Algorithm 1 is used for computing the pairwise means. The input to the algorithm is the set of the past CPU utilizations $(x_1, x_2, x_3, \ldots\ldots\ldots, x_n)$ taken over past n frames. The means, $(x_i + x_j)/2$, are calculated using these utilization values such that for each mean $1 \leq i < j \leq n$ generating a set of $\binom{n}{2}$ pairwise means.

```
INPUT: $x[i]$ , An array of $n$ elements consisting data of a hosts CPU utilization in past $n$ time frames
{
Initialize $i, j, z = 0$ ;
for  $(i = 1 : n)$
        for $(j = 1 : n)$
                if  $(i < j)$ then
                        $pairwise\_means[z] \leftarrow (x[i] + x[j])/2$ ;
                        z ← z + 1 ;
                end if
        end
end
        $spairwise\_means[\,] \leftarrow sort(pairwise\_means[z])$
}
return $spairwise\_means[\,]$
    }
```

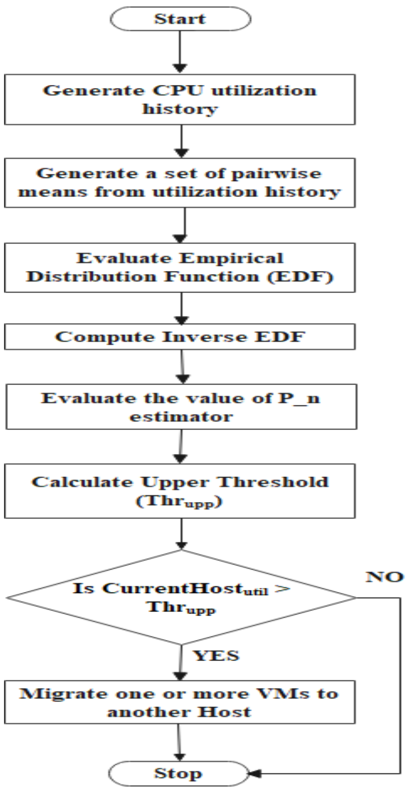
Start
Generate CPU utilization history
Generate a set of pairwise means from utilization history
Evaluate Empirical Distribution Function (EDF)
Compute Inverse EDF
Evaluate the value of P_n estimator
Calculate Upper Threshold (Thr_upp)
Is CurrentHost_util > Thr_upp
NO
YES
Migrate one or more VMs to another Host
Stop

Figure 4 $P_n\_AT_HP$ framework flow

Algorithm 2. $\boldsymbol{ComputeHn\_t}$ ()

The Algorithm 2 is used for calculating the empirical distribution function. The input to $ComputeHn\_t$ () is the set of pairwise means obtained so far using Algorithm 1. This set of pairwise means contains $\binom{n}{2}$ elements each of the form $h(x_i, x_j)$, such that $1 \leq i < j \leq n$. Algorithm 2 evaluates the empirical distribution function, $H_n(t)$, for the set of pairwise means.

```
INPUT: Set of pairwise means
{
Initialize $k, l, count = 0$ ;
for  $(k = 1 : \binom{n}{2})$
        for $(l = 1 : \binom{n}{2})$
                if$(spairwise\_means[l] \leq spairwise\_means[k])$ then
                $count++$ ;
                end if
        end
        $Hn\_t[k] \leftarrow \binom{n}{2}^{-1} \times count$ ;
        $count \leftarrow 0$ ;
end
return $Hn\_t[\,]$;
}
```

Algorithm 3. $\boldsymbol{ComputeHn\_inverse\ (p)}$

This algorithm helps in computing the inverse empirical distribution. In order to evaluate the estimator, the inverse of the distribution function needs to be calculated. The inverse depends upon the comparison of the $H_n(t)$ and (3) helps in finding the inverse function. Inverse of the distribution function equals the minimum value of t such that $(H_n(t) \geq p)$.

```
INPUT: $Hn\_t[\,]$ and $spairwise\_means[\,]$
{
Initialize $h, min\_t = spairwise\_means[1]$ ;

for  $(h = 1 : \binom{n}{2})$
    if  $(Hn\_t[h] \geq p)$ then

        If$(min\_t < spairwise\_means[h])$
            $(min\_t \leftarrow spairwise\_means[h])$
        end if
    end if
end
return $min\_t$
}
```

Algorithm 4. $\boldsymbol{ComputeUpperThreshold}$ ()

For setting the upper threshold, means is calculated followed by the evaluation of the empirical distribution function of means and its inverse. The inverse is evaluated for first and third quartile. Once the inverse has been found out, this helps in computing the value of the P_n estimator. (4) has been used for evaluating the value of the estimator.

```
{
Call $Computepairmeans$ ();
Call $ComputeHn\_t$ ();
$Hn\_inv1 \leftarrow ComputeHn\_inverse$ (0.75) ;
$Hn\_inv2 \leftarrow ComputeHn\_inverse$ (0.25) ;
$Pn\_t \leftarrow c_\tau(Hn\_inv1 - Hn\_inv2)$ ;
$Thr_{upp} \leftarrow 1 - s \times Pn\_t$ ;
return $Thr_{upp}$
}
```

4.Experimentation and results

This section details about the setup used for the experimentation of the proposed scheme and the results obtained on comparing the proposed scheme with the existing algorithms.

4.1Simulation test bench

The performance of the proposed $P_n\_AT_HP$ has been analysed using CloudSim toolkit [35]. The simulator has been developed at Clouds Laboratory in Melbourne University. The algorithm is implemented using Java programming language and is run on a windows based PC with Inter Core(TM) i5-8250U CPU @1.60 GHz and 8 GB of RAM. The hosts and VMs used for the simulation purpose are heterogeneous in nature. Two types of hosts have been used with the characteristics as shown in *Table 1* while VMs used are divided into four categories depending upon their characteristics as shown in *Table 2*. Planetlab workload has been used to evaluate the performance of the proposed scheme in contrast to S_nBODA [8] and baseline algorithms [36, 37]. The Planetlab workload used for the study involves the data collected for 10 days as on 3 March 2011, 6 March 2011, 9 March 2011, 22 March 2011, 3 April 2011, 9 April 2011, 11 April 2011, 12 April 2011 and 20 April 2011. The dataset included is a part of CoMon [38] project which is a monitoring infrastructure for PlanetLab. The data encompasses CPU utilizations of various VMs of hosts located on more than 500 locations across the world, taken at regular 5-minute interval. The characteristics of these workloads are shown in *Table 3.*

Table 1 Host characteristics

Server type	**Frequency of Core (MIPs)**	**Number of cores**	**Memory (GB)**	**Bandwidth (Gb/s)**	**Storage (GB)**
HPProLiantML110G4	1860	2	4	1	1000
HPProLiantML110G4	2660	2	4	1	1000

Table 2 VM characteristics

VM type	**Frequency of core (MIPs)**	**Number of cores**	**Memory (MB)**	**Bandwidth (Mb/s)**	**Storage (MB)**
Type #1	2500	1	870	100	2.5
Type #2	2000	1	1740	100	2.5
Type #3	1000	1	1740	100	2.5
Type #4	500	1	613	100	2.5

Table 3 Workload characteristics

Workload types	**Date**	**No. of VMs**	**No. of Hosts**
Real (PlanetLab)	03-03-2011	1052	800
	06-03-2011	898	800
	09-03-2011	1061	800
	22-03-2011	1516	800
	25-03-2011	1078	800
	03-04-2011	1463	800
	09-04-2011	1358	800
	11-04-2011	1233	800
	12-04-2011	1054	800
	20-04-2011	1033	800

4.2Performance metrics

It is the duty of every service provider to provide good quality of service to its users which is defined in terms of SLAs. CSPs work to achieve Minimum violation of SLAs and maximum quality levels. The major metrics for evaluating the performance of the consolidation schemes are: Energy Consumption, Energy performance metric, SLA time per active host, SLA performance degradation due to migration and SLAV.

Energy consumption (KWh)

In a data center, energy consumption depends upon the utilization of the underlying resources like CPU, memory, disk and also, on the functional capacities of the cooling elements. The energy consumption and the resource utilization share a linear relationship. More is the utilization of resources; more is the load on the server and higher is the energy consumption. Better resource utilization leads to improved energy efficiency. Being multi-core in nature, it becomes difficult to speculate the consumed energy levels of the virtualized environment. Therefore, the study uses the benchmark framework as specified by SPECPower [39] for energy consumption at different levels of utilization.

SLA time per active host (SLATAH)

SLATAH is the average of the ratio of time for which the hosts experience full utilization ($\mathrm{T}_{\mathrm{f}_\mathrm{i}}$) to the total active duration of the host ($\mathrm{T}_{\mathrm{a}_\mathrm{i}}$).

$$SLATAH = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{f_i}}{T_{a_i}} \qquad (6)$$

Where N is the total number of hosts in active mode.

SLA performance degradation due to migration (SLAPDM): During the process of VM migration, CPU, memory and bandwidth are involved for the migration of a VM from one host to another. This migration may cause some degradation in the system's performance which is expressed in terms of PDM i.e., performance degradation due to migration.

$$PDM = \frac{1}{M}\sum_{i=1}^{M}\frac{C_{d_j}}{C_{r_j}} \qquad (7)$$

Where C_{d_j} denotes total degradation due to migration and C_{r_j} represent the total resource request with M being the total number of VMs involved in the migration process.

• **SLA violation (SLAV):** SLAV specifies the time for which the desired QoS is not provided by the CSP. It is expressed in terms of the product between SLATAH and PDM.

$$SLAV = SLATAH \times PDM \qquad (8)$$

• **Energy and SLAV (ESV)**: ESV is the product of EC and SLAV. This metric is used to express the energy efficiency of the system. There exists a trade-off between the two. An efficient algorithm tries to maintain a balance these two metrics.

$$ESV = EC \times SLAV \qquad (9)$$

• **Energy, SLAV and VM migrations (ESM)**: It is a combined metric of ESV and the number of VM migrations. The objective is to minimal SLAVs with minimized energy consumption and least migrations.

$$ESM = EC \times SLAV \times No.of\ VM\ migrations \qquad (10)$$

4.3Comparison of $P_{n\_}AT_HP$ with the existing DVMC schemes

The simulation runs compare energy aware host overload detection policies in combination with VM selection policies. The research study includes comparison of five host overload detection schemes, i.e., MAD, IQR, LR, Static Threshold (THR) and $\mathrm{S_nBODA}$ in combination with four VM selection policies that include Minimum Migration Time (MMT), Maximum Correlation (MC), Minimum Utilization (MU) and Random Selection (RS). The placement policy used is Power Aware Best Fit Decreasing (PABFD). *Figure 5* presents the taxonomy of the existing DVMC Schemes along with the proposed PN host overload detection scheme.

Table 4 represents different DVMC plans used in simulations. Each scheme has an entry corresponding to Host Overload Detection Policy, Migration VM selection Policy, and Migration VM Selection Policy. These entries denote the policies used for detecting an overloaded host, the selection of VMs from the overloaded host, and the placement of the migrated VMs, corresponding to each DVMC scheme. For example, R21 in *Table 4* denotes PNMMT that utilizes the proposed $\boldsymbol{P_{n\_}AT_HP}$ as the overloaded host detection policy, MMT as the VM selection policy and PABFD as the VM placement policy.

4.3.1Comparison using PlanetLab workload

The performance of $P_{n\_}AT_HP$ has been compared with the baseline schemes in this section. For making the evaluation applicable, the real workload traces have been used. The section involves ten experiments for each DVMC scheme using ten-day PlanetLab real workload between March 2011 and April 2011. The results of the simulation are shown in *Table 5* and *Table 6*. Each DVMC scheme has been compared against the value of energy consumption, number of VM migrations, SLAPDM, SLATAH, ESV, ESM and the number of host shutdown. Each row entry of the *Table 5* corresponds to median of values obtained in those ten experiments while each row entry of the

Table 6 corresponds to average of values obtained in those ten experiments.

Figure 5 and *Table 6* represent different combinations of schemes used for comparison. For example, PNMMT has been compared with MADMMT, IQRMMT, LRMMT and THRMMT, shown by R1, R5, R9 and R13 respectively. Similarly, R22 presents the PNMC scheme that has been compared with the corresponding MADMC, IQRMC, LRMC and THRMC given in R2, R6, R10, R14 respectively. Likewise, PNMU (23) has been compared with MADMU (R3), IQRMU (R7), LRMU (R11) and THRMU (R15). In the same manner, the performance of PNRS (R24) has been compared with MADRS (R4), IQRRS (R8), LRRS (R12) and THRRS (R16). A complete list of abbreviations is shown in *Appendix I*.

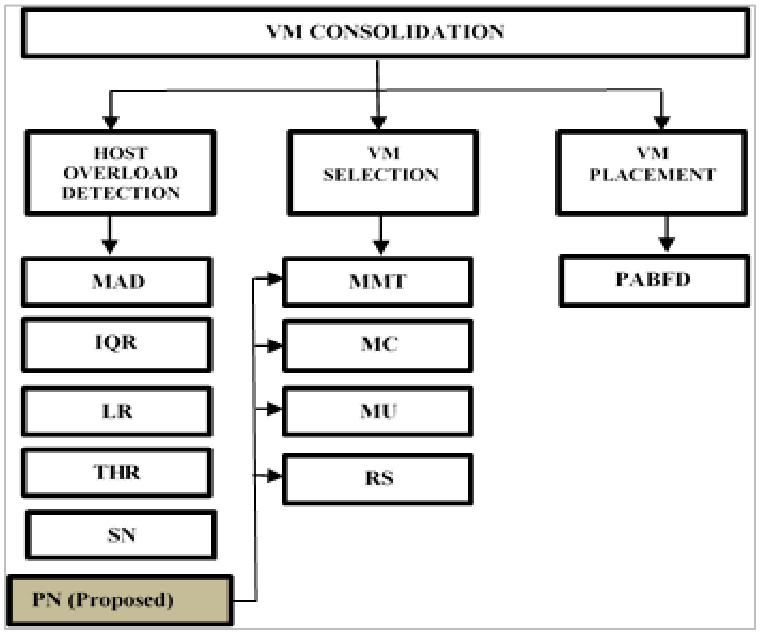

Figure 5 VM consolidation taxonomy

Table 4 Dynamic VM consolidation (DVMC) schemes

S. No.	DVMC scheme	Host overload detection policy	Migration VM selection policy	Migration VM placement policy
R1	MADMMT	MAD	MMT	PABFD
R2	MADMC	MAD	MC	PABFD
R3	MADMU	MAD	MU	PABFD
R4	MADRS	MAD	RS	PABFD
R5	IQRMMT	IQR	MMT	PABFD
R6	IQRMC	IQR	MC	PABFD
R7	IQRMU	IQR	MU	PABFD
R8	IQRRS	IQR	RS	PABFD
R9	LRMMT	LR	MMT	PABFD
R10	LRMC	LR	MC	PABFD
R11	LRMU	LR	MU	PABFD
R12	LRRS	LR	RS	PABFD
R13	THRMMT	THR	MMT	PABFD
R14	THRMC	THR	MC	PABFD
R15	THRMU	THR	MU	PABFD
R16	THRRS	THR	RS	PABFD
R17	SNMMT	S_nBODA	MMT	PABFD
R18	SNMC	S_nBODA	MC	PABFD
R19	SNMU	S_nBODA	MU	PABFD

S. No.	DVMC scheme	Host overload detection policy	Migration VM selection policy	Migration VM placement policy
R20	SNRS	S_nBODA	RS	PABFD
R21	PNMMT	P_n_AT$_H$P	MMT	PABFD
R22	PNMC	P_n_AT$_H$P	MC	PABFD
R23	PNMU	P_n_AT$_H$P	MU	PABFD
R24	PNRS	P_n_AT$_H$P	RS	PABFD

Table 5 Median values of simulation runs

S. No.	DVMC Scheme	Energy consumption (KWh)	Number of VM Migrations	SLAPDM	SLATAH (10^-2)	SLAV (10^-5)	ESV (10^-3)	ESM	No. of Hosts Shutdown
1	MADMMT	179.20	25449.5	0.07	5.04	3.27	5.96	157.67	5586.5
2	IQRMMT	183.34	25724.5	0.06	5.00	3.00	6.01	153.59	5652.5
3	PNMMT	177.99	25170.5	0.06	4.97	2.98	5.67	150.07	5540
4	LRMMT	158.27	27418	0.08	6.13	4.81	8.01	212.86	4985.5
5	THRMMT	184.84	25851	0.07	5.02	3.46	6.23	168.26	5682.5
6	MADMC	169.92	22729.5	0.10	7.05	7.02	12.01	273.70	5210.5
7	IQRMC	172.71	22495.5	0.10	6.92	6.88	11.94	268.68	5285.5
8	PNMC	169.39	22975	0.10	7.00	6.98	11.65	270.24	5227
9	LRMC	145.28	22880.5	0.10	7.36	7.33	11.03	256.87	3984.5
10	THRMC	175.93	23407.5	0.10	6.91	6.87	12.18	283.25	5328.5
11	MADMU	193.60	28768	0.07	7.58	5.26	9.78	289.96	6130.5
12	IQRMU	197.75	28963.5	0.06	7.48	4.55	9.75	276.26	6249.5
13	PNMU	192.61	28242	0.06	7.52	4.54	9.29	262.46	6042.5
14	LRMU	168.92	28718.5	0.07	8.50	5.95	10.97	288.94	5351
15	THRMU	199.99	29500.5	0.07	7.47	5.16	10.04	293.78	6299.5
16	MADRS	171.01	23046	0.10	7.12	7.09	12.12	284.92	5248.5
17	IQRRS	174.57	23012.5	0.10	7.02	6.63	12.12	279.27	5339
18	PNRS	169.16	22594.5	0.09	7.07	6.38	11.84	268.64	5191
19	LRRS	145.43	22978	0.10	7.47	7.47	11.62	267.86	3990
20	THRRS	176.91	23621	0.10	6.94	6.90	12.10	288.63	5345

Table 6 Average values of simulation runs

S. No	DVMC Scheme	Energy consumption (KWH)	Number of VM migrations	SLAPDM	SLATAH (10^-2)	SLAV (10^-5)	ESV (10^-3)	ESM	No. of hosts shutdown
1	MADMMT	183.49	26305.1	0.07	5.05	3.29	5.96	158.79	5665.10
2	IQRMMT	187.53	26496.9	0.07	5.02	3.28	6.04	161.68	5749.40
3	PNMMT	181.91	25907.1	0.06	5.00	3.21	5.75	150.63	5622.60
4	LRMMT	161.87	28174.7	0.08	6.21	4.99	8.01	230.34	5056.40
5	THRMMT	188.50	26601.9	0.07	5.04	3.44	6.36	170.46	5742.90
6	MADMC	173.79	23419.6	0.10	7.04	7.11	12.29	293.90	5266.60
7	IQRMC	177.73	23410.4	0.10	6.93	6.79	11.93	283.34	5330.00
8	PNMC	172.05	23182.6	0.10	6.98	6.99	11.88	279.04	5234.60
9	LRMC	148.51	23931.2	0.10	7.47	7.66	11.27	276.05	4116.60
10	THRMC	179.38	23961.8	0.10	6.91	6.92	12.28	299.10	5373.00
11	MADMU	198.07	29615.6	0.07	7.56	5.14	10.04	300.58	6230.80
12	IQRMU	202.35	30018.5	0.07	7.48	4.86	9.73	296.27	6335.30
13	PNMU	196.18	29188.4	0.06	7.51	4.81	9.35	276.60	6188.50
14	LRMU	173.06	29419.2	0.07	8.65	6.26	10.74	322.67	5426.70
15	THRMU	204.09	30384.5	0.07	7.48	5.09	10.24	314.98	6395.30
16	MADRS	174.86	23697.3	0.10	7.09	7.16	12.38	297.05	5300.50

S. No	DVMC Scheme	Energy consumption (KWH)	Number of VM migrations	SLAPDM	SLATAH (10^-2)	SLAV (10^-5)	ESV (10^-3)	ESM	No. of hosts shutdown
17	IQRRS	178.46	23671.6	0.10	6.99	6.78	11.98	287.61	5348.50
18	PNRS	172.97	23338.5	0.10	7.04	6.76	11.59	274.87	5254.10
19	LRRS	148.07	23760.6	0.10	7.63	7.90	11.61	281.88	4077.60
20	THRRS	180.31	24023.9	0.10	6.95	6.81	12.17	297.40	5387.50

- **Analysis of results**

The following observations have been deduced from the *Table 6* of observations.

Case (i) Reduced Migrations with reduced energy consumption (PNMMT vs IQRMMT)

The number of migrations in the case of PNMMT is lesser than that of IQRMMT. The reason behind fewer migrations is the better performance of the P_n scale estimator for estimating the deviations compared to other scale measures. Better performance yields better threshold values. This helps reduce the number of migrations because the migration process is initiated once the overload situation has occurred. This further minimizes energy consumption. Reduced number of migrations and energy consumption helps achieve better ESM values. As shown in *Table 6* (the highlighted rows), the ESM for PNMMT is 150.63 whereas for IQRMMT, the value is 161.68. In the same manner, PNMMT shows improvement over MADMMT and THRMMT.

Case (ii) Reduced Migrations with more energy consumption (PNMMT vs LRMMT)

On comparing the results of PNMMT with LRMMT, it has been observed that though the number of migrations has reduced with PNMMT, the energy consumption of PNMMT has not reduced and is still more than LRMMT. But, the overall performance of PNMMT in terms of ESM remains better than LRMMT. For PNMMT, ESM is 150.63, and for LRMMT, the value of ESM is 230.34. The reason behind this improved performance is the trade-off existing between the energy consumption and the SLAV. There is an increase in SLAV. In the case of LRMMT, the SLAV are much more than PNMMT. This led to an overall improvement in the performance of PNMMT.

The performance metrics for different DVMC schemes are shown in the form of a boxplot in *Figure 6 to Figure 13*. *Figure 6* shows the graphical representation of energy consumption for all the compared DVMC schemes. The X-axis shows the different DVMC schemes whereas the Y-axis shows the amount of energy consumed in KWh. Considering MMT as VM selection policy, PNMMT shows an improvement of 0.86%, 2.99%, and 3.49% with MAD, IQR, and THR. Similarly, with MC, there is an improvement of 1 %, 3.19%, and 4.08 % against MAD, IQR, and THR. Taking MU as selection policy, PNMMT performs 0.95%, 3.04%, and 3.87% better than MAD, IQR, and THR. Taking all the dynamic host overload detection policies, the average energy consumption of PNMC is 14.97% of IQRMU.

Figure 7 shows the graphical representation of the number of VM migrations taking place after the detection of host overload for all the compared DVMC schemes. DVMC schemes are shown across the X-axis while the number of VM migration are shown in the Y-axis. Number of migrations show improvements with maximum of 8.04% when compared with LRMMT, 3.25% when compared with THRMC, 3.93% when compared with THRMU and 2.85% when compared with THRS.

The SLAV depend upon SLAPDM and SLATAH. *Figure 8* and *Figure 9* graphically represent the result obtained for SLAPDM and SLATAH, respectively. The results for SLAV are shown in the form of a boxplot in *Figure 10.* The X-axis show the DVMC scheme while SLAV incurred for each scheme are depicted by the Y-axis. According to results obtained, the SLAV show maximum improvement over LR host detection scheme. With MMT as the VM selection scheme, there is an improvement of 38.04% and with MC improvement of 4.77% is seen. An improvement of 23.69% is observed with MU selection policy while with RS, the proposed scheme shows an improvement of 14.59%.

ESV is a combined metric that shows the performance of a scheme in terms of both energy consumption and SLAV. *Figure 11* shows the results achieved for the proposed scheme against different schemes concerning ESV. The X-axis shows the DVMC scheme and the Y-axis shows the corresponding ESV obtained. Maximum improvement is shown when PNMMT is compared with LRMMT. There is an average improvement of 29.21%.

ESM is a combined metric of energy, SLAV and migrations. The results show that PNMMT shows the maximum improvement of 41.83% compared to LRMMT. However, the estimator P_n does not work well with LRRS and LRMC schemes. For MMT, there is an improvement of 5.06% with MAD, 2.34% with IQR and 12.12% with THR. Similarly, for MC, there is an improvement of 1.27% with MAD and 4.81% with THR. Taking MU as the selection policy, PNMU performs 10.48% better than MADMU, 5.26% better than IQRMU, 10.09% better than LRMU and 11.93% better than THRMU. Whereas, PNRS shows an improvement of 6.06%, 3.95% and 7.43% when compared with MADRS, IQRRS and THRRS, respectively. The results for ESM obtained in all the DVMC schemes is shown in *Figure 12*. Values of achieved ESM are represented by the Y-axis for corresponding DVMC schemes on the X-axis.

Number of host shutdowns shows an improvement over MAD, IQR, LR and THR. Maximum improvement is observed against THR. With MMT as the VM selection policy, there is an improvement of 2.5% against THR. PNMC shows an improvement of 1.9% for host shutdowns when compared with THRMC. While the figure is 2.88% when PNRS is compared with THRRS. Whereas PNMU exhibits a maximum improvement of 4.07% on being compared with THRMU.

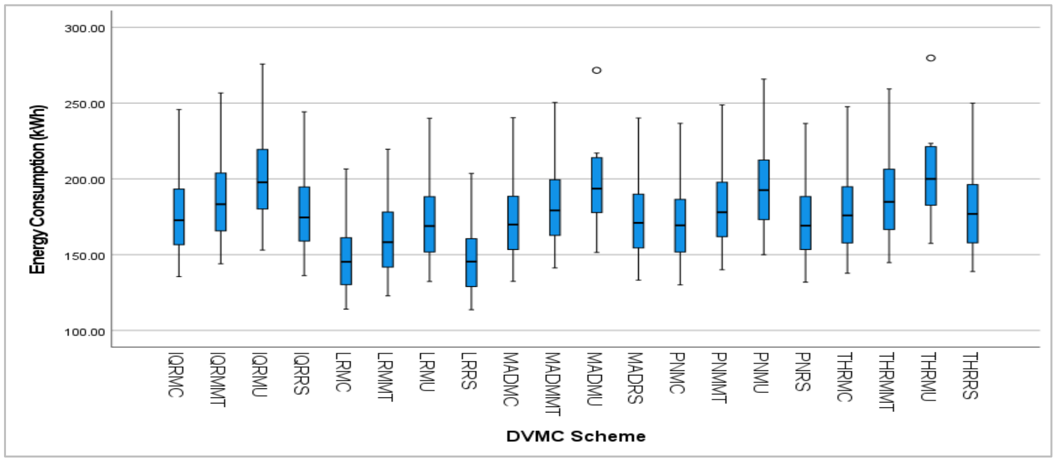

Figure 6 Energy consumption vs DVMC scheme

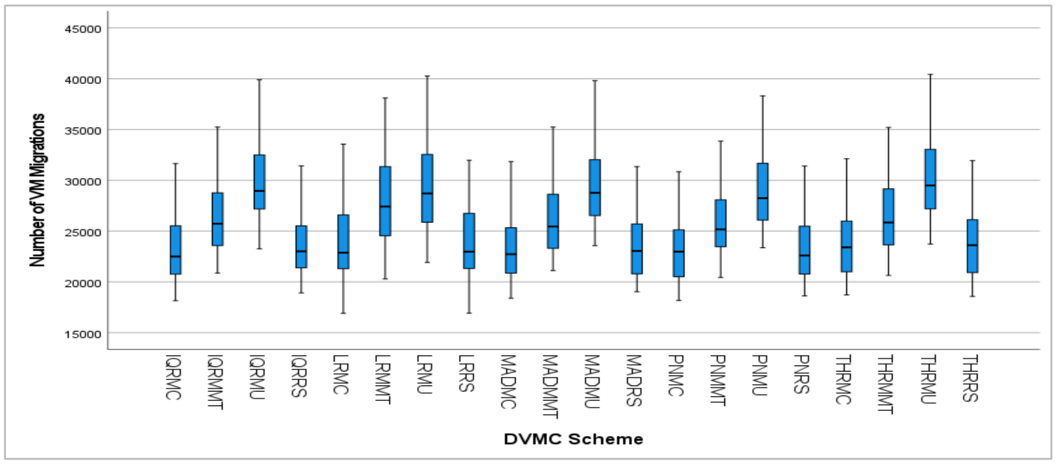

Figure 7 Number of migrations vs DVMC scheme

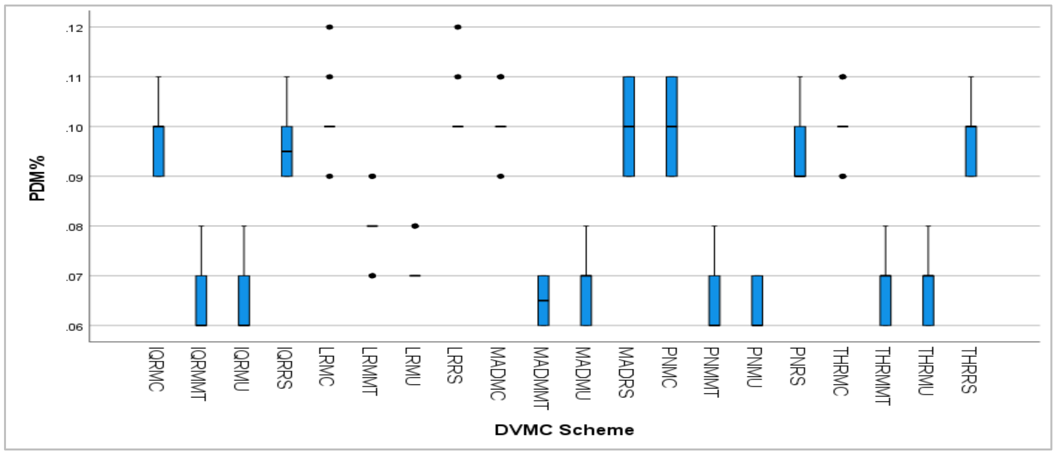

Figure 8 SLAPDM vs DVMC scheme

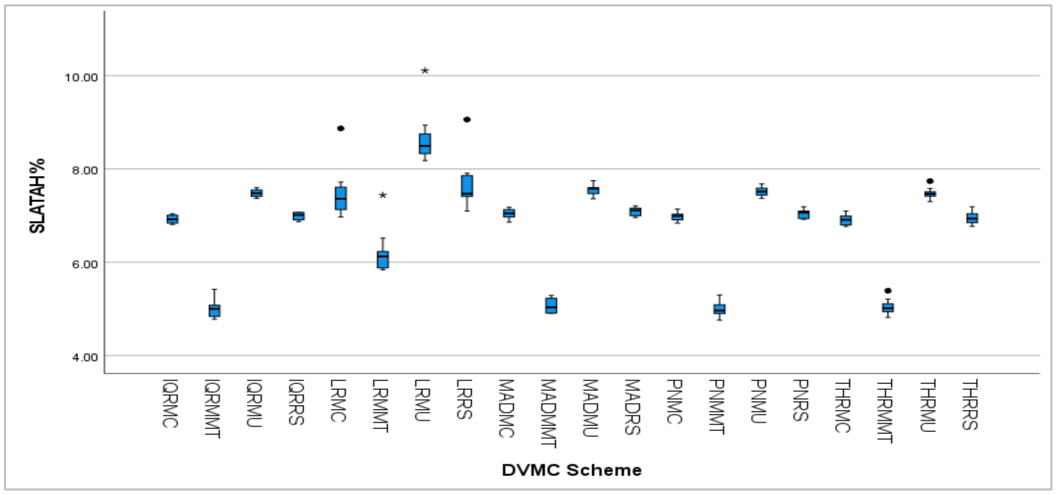

Figure 9 SLATAH vs DVMC scheme

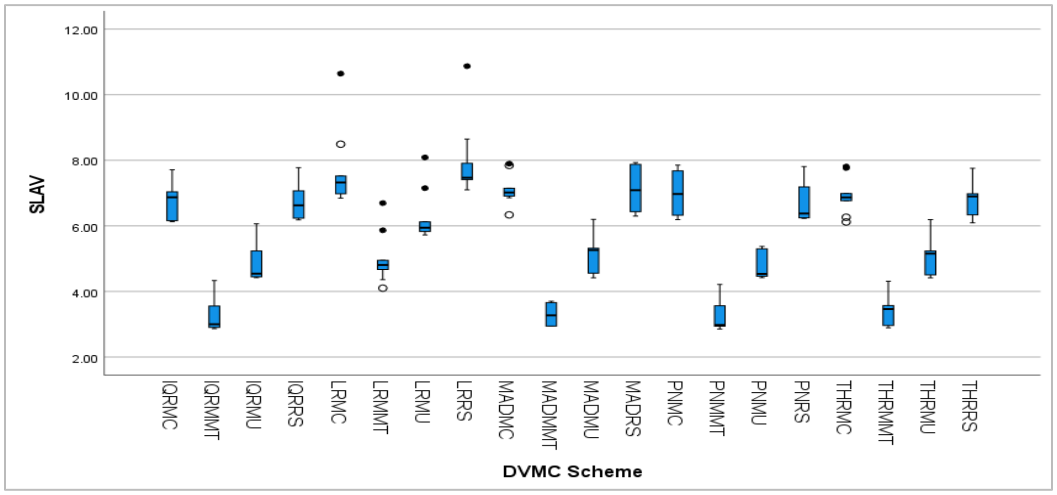

Figure 10 SLAV vs DVMC scheme

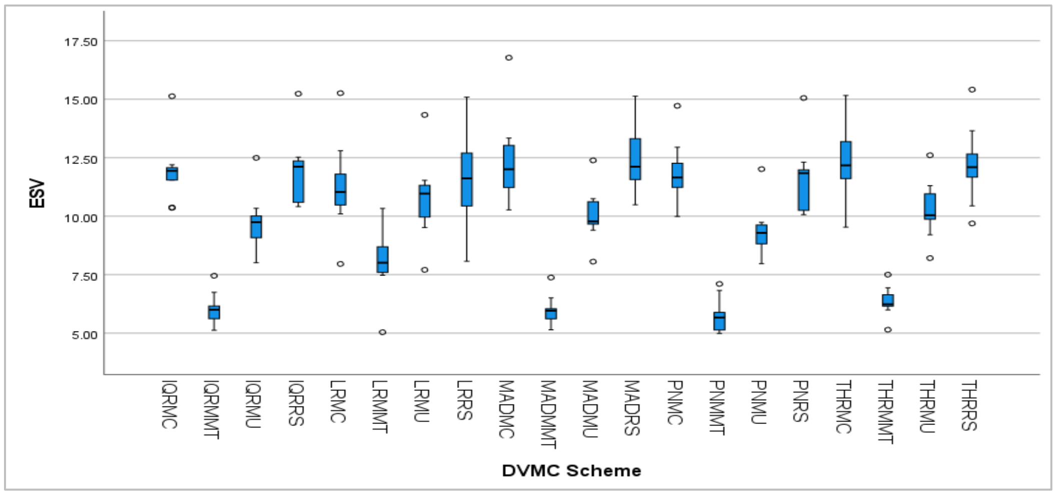

Figure 11 ESV vs DVMC scheme

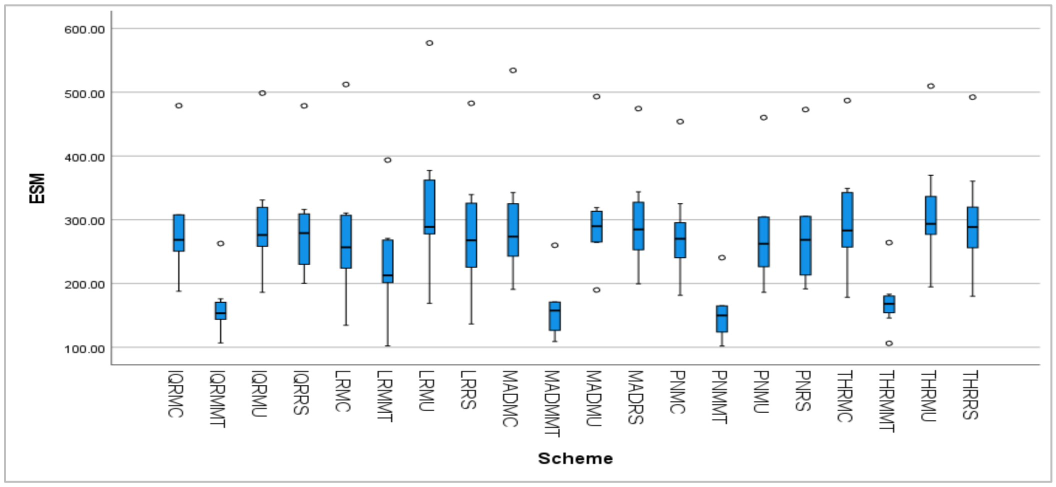

Figure 12 ESM vs DVMC scheme

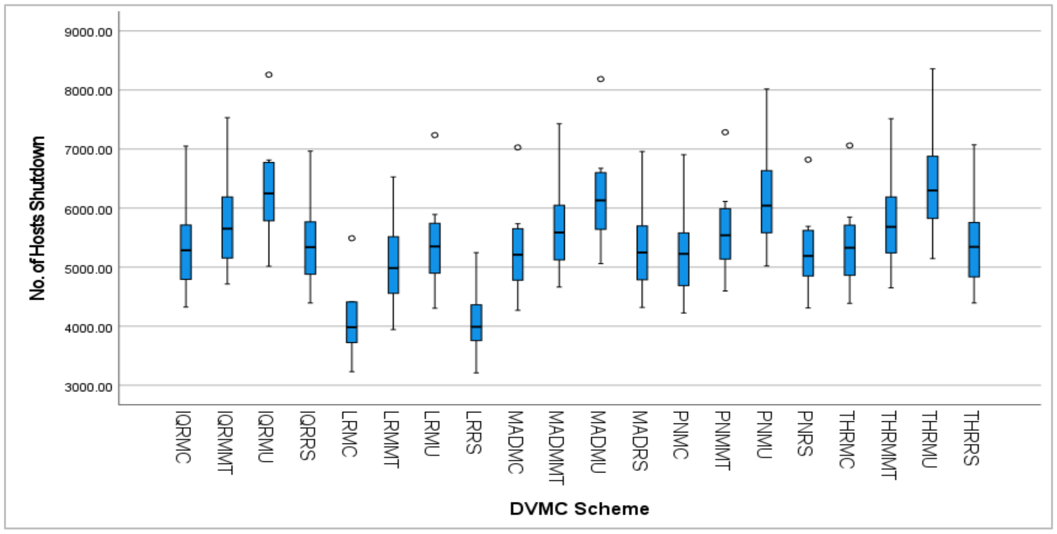

Figure 13 Number of hosts shutdown vs DVMC scheme

4.3.2 Comparison with S_nBODA with varying Selection Schemes and single PlanetLab workload

This section presents the comparison of $P_n\_AT_HP$ with S_nBODA, proposed by Bala and Padha [8], and traditional schemes including MAD and IQR. *Table 7* show the performance of MAD, IQR, SN, and PN along with MMT, MU, MC and RS against different performance metrics. The simulation results correspond to PlanetLab workload for 22 March 2011 with varying host overload detection and VM selection policies.

Comparing $P_n\_AT_HP$ against S_nBODA for different selection policies various results have been obtained. The comparison of metrics between $P_n\_AT_HP$ and S_nBODA with various selection schemes is shown in *Figure 14 to Figure 20.* The X-axis shows the different selection policies that have been used for comparison whereas the Y-axis shows the corresponding value of the performance metric.

Figure 14 shows the graphical representation of the results obtained for energy consumption when $P_n\_AT_HP$ is compared against S_nBODA. For energy consumption, PNMMT shows an improvement of 0.54% and 2.77% with respect to MAD and IQR. PNMU being 0.78 % and 3.25% better than MADMU and IQRMU respectively. An improvement of 0.52% and 2.23 % is observed with respect to MADMC and IQRMC. However, compared to S_nBODA, $P_n\_AT_HP$ shows an average increase of 3.51% consumed energy.

For migrations, PNMMT shows an improvement of 8.81% over SNMMT, PNMU at 5.5% over SNMU, PNMC with 11.34% improvement against SNMC while migrations are 9.05% less with PNRS as compared to SNRS. The graphical representation of number of migrations for $P_n\_AT_HP$ and S_nBODA is shown in *Figure 15.*

$P_n\_AT_HP$ shows an average improvement of 20.26% in SLAV, 6.78% in SLATAH and 17.79% in ESV when compared with S_nBODA. *Figure 16, Figure 17 and Figure 18* show the SLAPDM, SLATAH and SLAV achieved in case of $P_n\_AT_HP$ and S_nBODA in graphical form.

The improvement in terms of ESV is shown in *Figure 19*. PNMMT shows an improvement of 27.5%, PNMU shows an improvement of 14.91%, PNMC shows an improvement of 15.71%, whereas a minimum improvement of 13.7% is seen when PNRS is compared with SNRS in terms of ESV.

PNMMT shows an ESM improvement of 5.79%, 8.99% and 51.20% over MADMMT, IQRMMT and SNMMT. With MU as the selection scheme, PNMU performs 2.82%, 8.62% and 25.56% better than MAD, IQR and SN. Similarly, the percentage improvement of PN is 2.05, 8.62 and 34.56 for MU selection policy against MAD, IQR and SN respectively. The results for ESM are shown in *Figure 20.*

4.3.3 Comparison with S_nBODA with varying Planetlab workloads and single VM selection Scheme

This section compares $P_n\_AT_HP$ with S_nBODA for four days of PlanetLab workload, with MMT as the VM selection policy and PABFD as the VM placement policy. *Table 8* shows the performance metrics obtained of MAD, IQR, PN and SN as host selection policies.

Table 7 Comparative results of DVMC schemes

S. No.	DVMC Scheme	Energy consumption	Number of VM migrations	SLAPDM	SLATAH (10^-2)	SLAV (10^-5)	ESV (10^-3)	ESM
1	MADMMT	197.33	28628	0.06	5.04	3.00	6.00	170.83
2	MADMU	214.02	32035	0.06	7.62	4.60	9.80	313.46
3	MADMC	187.87	25337	0.09	7.04	6.30	11.90	301.60
4	MADRS	188.26	25711	0.09	7.15	6.40	12.10	311.48
5	IQRMMT	201.86	28948	0.06	5.02	3.00	6.10	176.00
6	IQRMU	219.49	33084	0.06	7.6	4.60	10.00	331.13
7	IQRMC	191.16	25566	0.09	7	6.30	12.00	307.89
8	IQRRS	193.11	25732	0.09	7.07	6.40	12.30	316.18
9	SNMMT	190.66	30600	0.08	5.58	4.20	8.00	244.16
10	SNMU	206.77	33518	0.07	7.89	5.50	11.40	382.77
11	SNMC	180.11	28345	0.1	7.49	7.80	14.00	397.68
12	SNRS	180.17	28023	0.1	7.53	7.70	13.80	387.79
13	PNMMT	196.25	27903	0.06	4.94	3.00	5.80	161.48
14	PNMU	212.35	31672	0.06	7.59	4.60	9.70	304.84
15	PNMC	186.88	25128	0.09	7.03	6.30	11.80	295.52
16	PNRS	188.55	25485	0.09	7.1	6.40	12.00	305.42

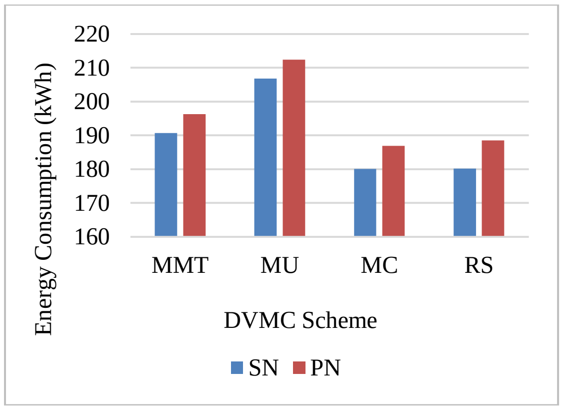

Figure 14 Energy consumption VS host overload detection scheme

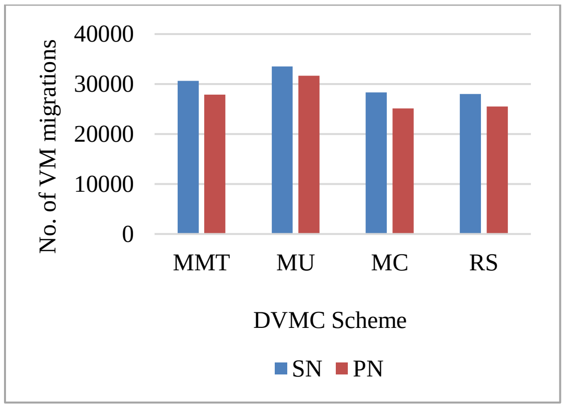

Figure 15 Number of VM migrations VS host overload detection scheme

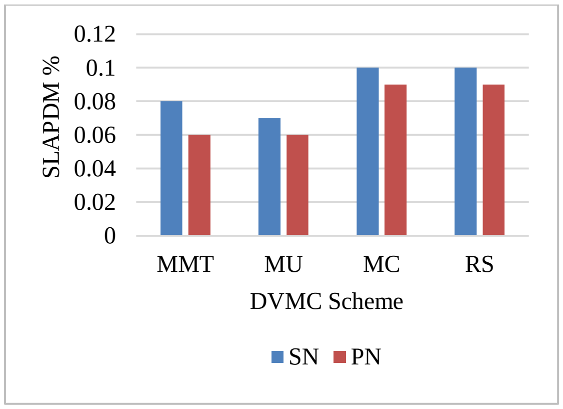

Figure 16 SLAPDM vs host overload detection scheme

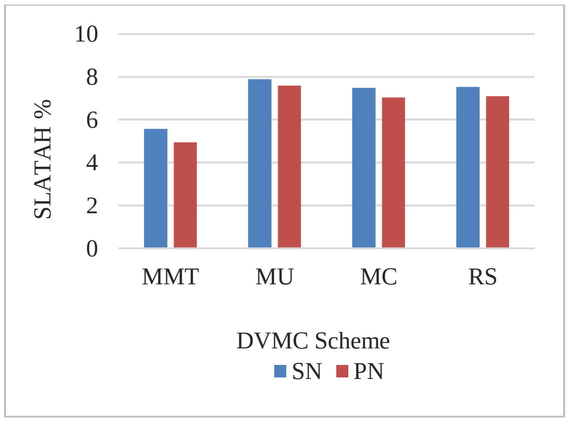

Figure 17 SLATAH vs host overload scheme

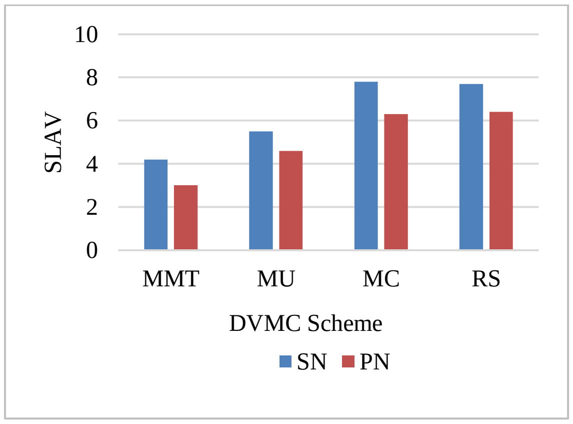

Figure 18 SLAV vs host overload detection scheme

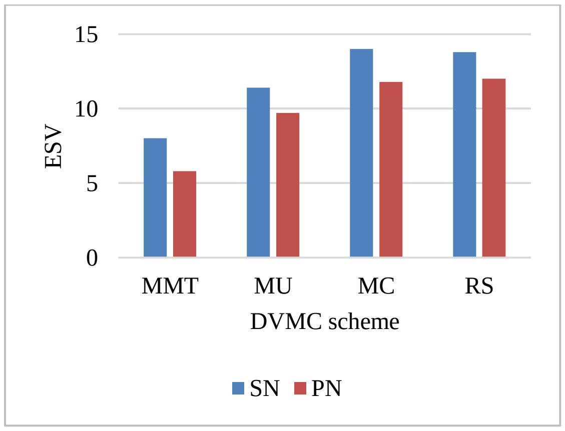

Figure 19 ESV vs host overload detection scheme

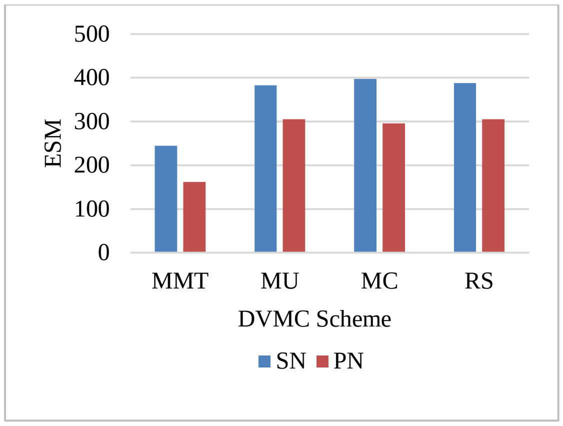

Figure 20 ESM vs host overload detection scheme

Figure 21 to *Figure 27* shows the graphical representation of PNMMT and SNMMT for different performance metrics. The X-axis shows the different PlanetLab workloads for which the corresponding values of Y-axis represents the values of performance metrics that have been obtained.

The comparison of SNMMT and PNMMT based on energy consumption is shown in *Figure 21*. Out of all DVMC schemes, SNMMT consumes minimum energy on average and is 3.65% less than MADMMT, 5.78% less than IQRMMT and 2.56% less than PNMMT.

Table 8 Comparative results of DVMC schemes for PlanetLab workload using MMT selection policy

Date	**DVMC scheme**	**Energy consumption**	**Number of VM migrations**	**SLAPDM**	**SLATAH 10^-2)**	**SLAV (10^-5)**	**ESV (10^-3)**	**ESM**
03-03-2011 (20110303)	MADMMT	184.88	26292	0.07	5.03	3.52	6.5	171.15
	IQRMMT	188.86	26476	0.06	4.96	2.98	5.6	148.81
	SNMMT	179.58	31495	0.1	5.94	5.74	10.3	324.39
	PNMMT	183.77	25895	0.06	4.9	2.94	5.4	139.91
22-03-2011 (20110322)	MADMMT	197.33	28628	0.06	5.04	3.02	6	170.83
	IQRMMT	201.86	28948	0.06	5.02	3.01	6.1	176.00
	SNMMT	190.66	30600	0.08	5.58	4.19	8	244.16
	PNMMT	196.25	27903	0.06	4.94	2.96	5.8	162.31
03-04-2011 (20110403)	MADMMT	250.48	35240	0.06	4.91	2.95	7.4	260.04
	IQRMMT	256.75	35245	0.06	4.84	2.9	7.5	262.79
	SNMMT	242.66	40591	0.09	5.75	5.07	12.3	498.96
	PNMMT	249.79	33863	0.06	4.76	2.86	7.1	241.58
20-04-2011 (20110420)	MADMMT	151.59	22236	0.07	5.29	3.7	5.6	124.82
	IQRMMT	155.61	22903	0.08	5.42	4.34	6.7	154.53
	SNMMT	143.73	21788	0.08	5.54	4.53	6.5	141.74
	PNMMT	150.76	22326	0.07	5.3	3.71	5.6	124.87

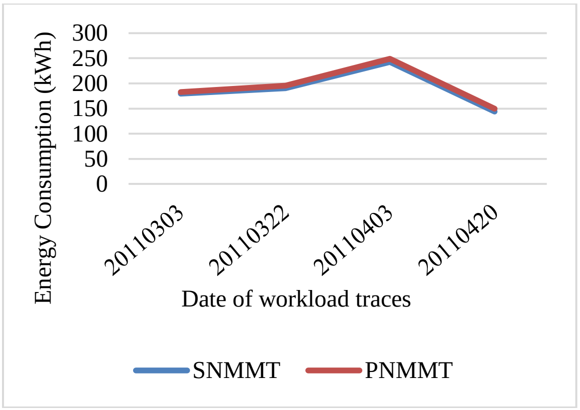

Figure 21 Energy consumption vs host overload detection scheme

Figure 22 shows the graphical representation of compared schemes on the basis of number of VM migrations. Considering the average number of migrations, PNMMT shows the least number and is 2.14%, 3.15%, and 11.63% better than MADMMT, IQRMMT and SNMMT. *Figure 23* shows the SLAPDM observed in four days for SNMMT and PNMMT and *Figure 24* shows the graphical representation of SLATAH when PNMMT and SNMMT are executed for four days PlanetLab workload. The SLAV incurred in the case of both SNMMT and PNMMT are graphically represented in *Figure 25*. PNMMT shows an average improvement of 34.95 % over SNMMT. Maximum improvement of 48.78% is observed on 03/03/2011. *Figure 26* shows the graphical representation of ESV for four days' workload of both SNMMT and PNMMT. There is an average improvement of 32.79% when PNMMT is compared with SNMMT. Maximum improvement of 47.57% is observed on 03/03/2011 whereas a minimum of 13.84% is seen on 20/04/2011.

ESM, a combined energy, SLAV and migration metric, shows an average improvement of 8.70% of PNMMT over MADMMT, 10.98% over IQRMMT and 80.93% against SNMMT. The results for ESM are shown graphically in *Figure 27*.

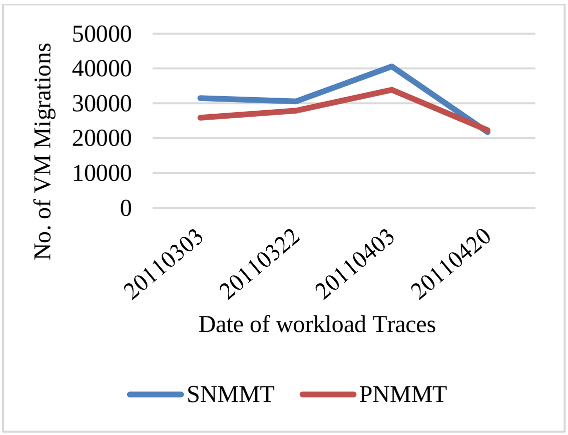

Figure 22 Number of VM migrations vs host overload detection scheme

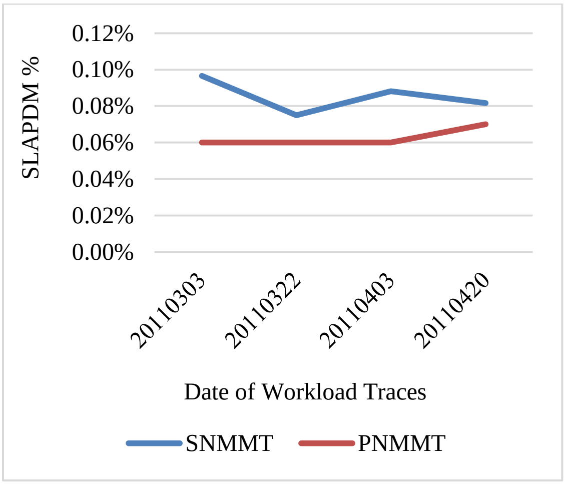

Figure 23 SLAPDM vs host overload detection scheme

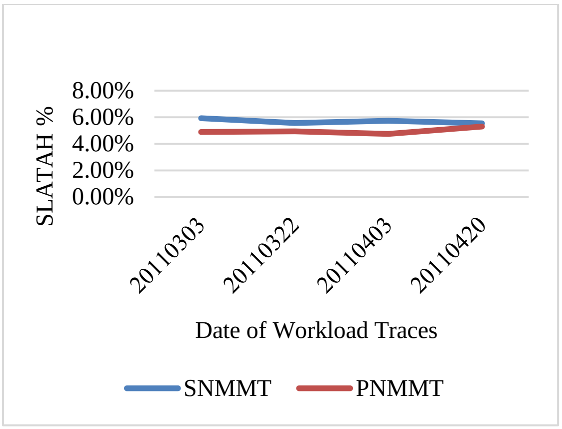

Figure 24 SLATAH vs host overload detection scheme

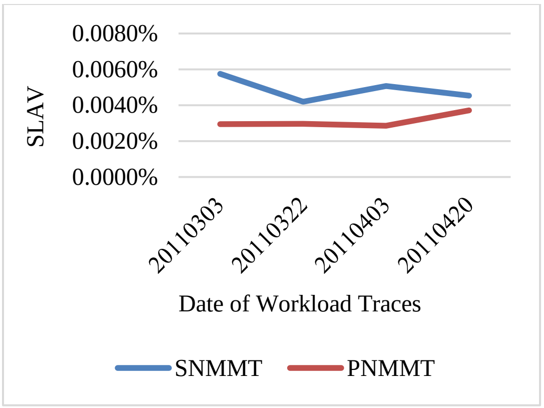

Figure 25 SLAV vs host overload detection scheme

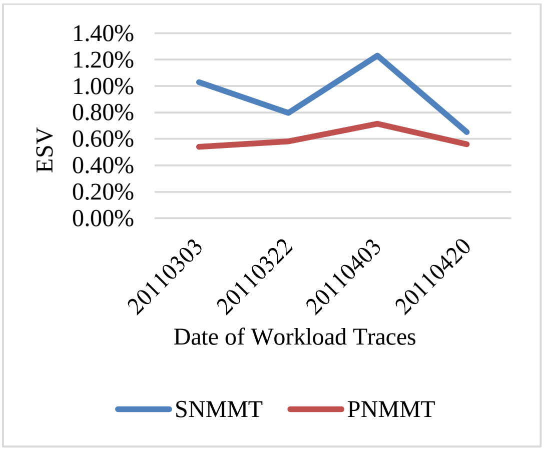

Figure 26 ESV vs host overload detection scheme

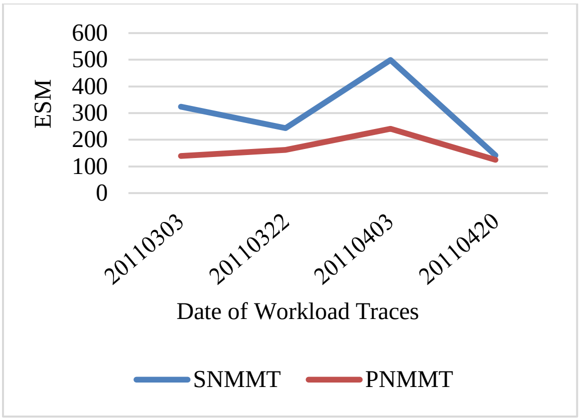

Figure 27 ESM vs host overload detection scheme

5.Discussion

The static threshold does not hold well in the case of highly dynamic workloads. While handling dynamic workloads, it has been observed from the utilization history that the utilization shows high deviations. This increases the chances of the fluctuating server utilization to reach 100% and causes more migration overhead. Dynamic threshold deals with the issue of frequent migrations and helps reduce the number of VM migrations. The energy consumption of a data center and the SLAVs incurred are greatly affected by the number of live migrations while consolidating the VMs. Thus, a reduced number of VM migrations helps in minimizing the overall load of the data center. As seen from the observations in *Table 5* and *Table 6*, our proposed scheme shows the minimum number of VM migrations on average for all the ten days' workload compared with the benchmark algorithms. This has also helped reduce energy consumption and improve the overall performance in terms of the ESM metric.

5.1Limitations

In the cloud environment, the consolidation schemes exploit the thresholds for the detection of overloaded and underloaded host machines that helps in vacating the VMs to maintain a balance between the energy consumption and the utilization of the server. The proposed scheme adaptively adjusts the upper threshold considering the past CPU utilization of the host with the help of a scale estimatorP_n. However, in the workload consolidation process of the CDC, detection of the underloaded hosts is also a critical phase. In the proposed scheme, the value of the lower threshold for the detection of the underloaded hosts has not been taken into consideration. Thus, the lower threshold remains static and does not vary with the dynamic workload.

6.Conclusion and future work

To guarantee QoS while reducing the consumed energy aimed at avoiding SLAVs is a challenging task for the Cloud Service providers. The vast energy consumption is just not due to high computing resources but due to the inefficient use of the available resources. To enhance the performance of the cloud environment, efficient resource provisioning techniques are required to minimize the energy consumption and to avoid SLAVs. The aim of the research is to minimize energy consumption by considering dynamic threshold policy for host overload detection. The proposed scheme, $P_n\_AT_HP$, adaptively adjusts the upper threshold by considering the past CPU utilization of the host with the help of a scale estimator P_n. The proposed scheme shows better performance when compared with the baseline algorithms MAD, IQR, LR, THR and SnBODA. Simulation results show the reduction in SLAVs. In addition, the energy consumption has also been reduced by minimizing the number of virtual machine migrations. As a future research direction, the proposed work can be carried forward with different types of workloads, considering more parameters like RAM, bandwidth, GPU to calculate the upper threshold and to implement the $P_n\_AT_HP$ in real cloud environment like OpenStack.

Acknowledgment

None.

Conflicts of interest

The authors have no conflicts of interest to declare.

References

[1] https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html. Accessed: 10 October 2021.

[2] Koot M, Wijnhoven F. Usage impact on data center electricity needs: a system dynamic forecasting model. Applied Energy. 2021.

[3] Hintemann R, Hinterholzer S. Energy consumption of data centers worldwide-how will the internet become green?. In ICT4S 2019.

[4] Mastroianni C, Meo M, Papuzzo G. Probabilistic consolidation of virtual machines in self-organizing cloud data centers. IEEE Transactions on Cloud Computing. 2013; 1(2):215-28.

[5] Chen YW, Chang JM. EMaaS: cloud-based energy management service for distributed renewable energy integration. IEEE Transactions on Smart Grid. 2015; 6(6):2816-24.

[6] Zhou Z, Abawajy JH, Li F, Hu Z, Chowdhury MU, Alelaiwi A, et al. Fine-grained energy consumption model of servers based on task characteristics in cloud data center. IEEE Access. 2017; 6:27080-90.

[7] Xiao H, Hu Z, Li K. Multi-objective VM consolidation based on thresholds and ant colony system in cloud computing. IEEE Access. 2019; 7:53441-53.
[8] Bala M, Padha D. An adaptive overload detection policy based on the estimator sn in cloud environment. International Journal of Service Science, Management, Engineering, and Technology. 2017; 8(3):93-107.
[9] Zhu X, Young D, Watson BJ, Wang Z, Rolia J, Singhal S, et al. 1000 Islands: integrated capacity and workload management for the next generation data center. In international conference on autonomic computing 2008 (pp. 172-81). IEEE.
[10] Gmach D, Rolia J, Cherkasova L, Belrose G, Turicchi T, Kemper A. An integrated approach to resource pool management: policies, efficiency and quality metrics. In international conference on dependable systems and networks with FTCS and DCC (DSN) 2008 (pp. 326-35). IEEE.
[11] Beloglazov A, Abawajy J, Buyya R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Generation Computer Systems. 2012; 28(5):755-68.
[12] Li H, Zhu G, Cui C, Tang H, Dou Y, He C. Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing. Computing. 2016; 98(3):303-17.
[13] Fard SY, Ahmadi MR, Adabi S. A dynamic VM consolidation technique for QoS and energy consumption in cloud environment. The Journal of Supercomputing. 2017; 73(10):4347-68.
[14] Buyya R, Beloglazov A, Abawajy J. Energy-efficient management of data center resources for cloud computing: a vision, architectural elements, and open challenges. arXiv preprint arXiv:1006.0308. 2010.
[15] Monil MA, Rahman RM. Implementation of modified overload detection technique with VM selection strategies based on heuristics and migration control. In international conference on computer and information science 2015 (pp. 223-7). IEEE.
[16] Minarolli D, Mazrekaj A, Freisleben B. Tackling uncertainty in long-term predictions for host overload and underload detection in cloud computing. Journal of Cloud Computing. 2017; 6:1-18.
[17] Li Z, Yan C, Yu X, Yu N. Bayesian network-based virtual machines consolidation method. Future Generation Computer Systems. 2017; 69:75-87.
[18] Melhem SB, Agarwal A, Goel N, Zaman M. A markov-based prediction model for host load detection in live VM migration. In 5th international conference on future internet of things and cloud 2017 (pp. 32-8). IEEE.
[19] Li Z. An adaptive overload threshold selection process using markov decision processes of virtual machine in cloud data center. Cluster Computing. 2019; 22(2):3821-33.
[20] Hsieh SY, Liu CS, Buyya R, Zomaya AY. Utilization-prediction-aware virtual machine consolidation approach for energy-efficient cloud data centers. Journal of Parallel and Distributed Computing. 2020; 139:99-109.
[21] Masoumzadeh SS, Hlavacs H. An intelligent and adaptive threshold-based schema for energy and performance efficient dynamic VM consolidation. In European conference on energy efficiency in large scale distributed systems 2013 (pp. 85-97). Springer, Berlin, Heidelberg.
[22] Salimian L, Esfahani FS, Nadimi-shahraki MH. An adaptive fuzzy threshold-based approach for energy and performance efficient consolidation of virtual machines. Computing. 2016; 98(6):641-60.
[23] Farahnakian F, Liljeberg P, Plosila J. LiRCUP: linear regression based CPU usage prediction algorithm for live migration of virtual machines in data centers. In Euromicro conference on software engineering and advanced applications 2013 (pp. 357-64). IEEE.
[24] Mao L, Qi D, Lin W, Zhu C. A self-adaptive prediction algorithm for cloud workloads. International Journal of Grid and High Performance Computing. 2015; 7(2):65-76.
[25] Yadav R, Zhang W. MeReg: managing energy-SLA tradeoff for green mobile cloud computing. Wireless Communications and Mobile Computing. 2017.
[26] Jararweh Y, Issa MB, Daraghmeh M, Al-ayyoub M, Alsmirat MA. Energy efficient dynamic resource management in cloud computing based on logistic regression model and median absolute deviation. Sustainable Computing: Informatics and Systems. 2018; 19:262-74.
[27] Mapetu JP, Kong L, Chen Z. A dynamic VM consolidation approach based on load balancing using pearson correlation in cloud computing. The Journal of Supercomputing. 2021; 77(6):5840-81.
[28] Xie L, Chen S, Shen W, Miao H. A novel self-adaptive VM consolidation strategy using dynamic multi-thresholds in IAAS clouds. Future Internet. 2018; 10(6):1-18.
[29] Zhou H, Li Q, Choo KK, Zhu H. DADTA: a novel adaptive strategy for energy and performance efficient virtual machine consolidation. Journal of Parallel and Distributed Computing. 2018; 121:15-26.
[30] Sharma O, Saini H. VM consolidation for cloud data center using median based threshold approach. Procedia Computer Science. 2016; 89:27-33.
[31] Farahnakian F, Bahsoon R, Liljeberg P, Pahikkala T. Self-adaptive resource management system in IAAS clouds. In international conference on cloud computing 2016 (pp. 553-60). IEEE.
[32] Dambreville A, Tomasik J, Cohen J, Dufoulon F. Load prediction for energy-aware scheduling for cloud computing platforms. In international conference on distributed computing systems 2017 (pp. 2604-7). IEEE.
[33] Saadi Y, El KS. Energy-efficient strategy for virtual machine consolidation in cloud environment. Soft Computing. 2020; 24(19):14845-59.
[34] Tarr G, Müller S, Weber N. A robust scale estimator based on pairwise means. Journal of Nonparametric Statistics. 2012; 24(1):187-99.

[35] Calheiros RN, Ranjan R, Beloglazov A, De RCA, Buyya R. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience. 2011; 41(1):23-50.
[36] Beloglazov A, Buyya R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. Concurrency and Computation: Practice and Experience. 2012; 24(13):1397-420.
[37] Beloglazov A. Energy-efficient management of virtual machines in data centers for cloud computing (Doctoral dissertation). The University of Melbourne, 2013.
[38] Park K, Pai VS. CoMon: a mostly-scalable monitoring system for PlanetLab. ACM SIGOPS Operating Systems Review. 2006; 40(1):65-74.
[39] Lange KD. Identifying shades of green: the sPECpower benchmarks. Computer. 2009; 42(3):95-7.

Ms. Bhagyalakshmi pursed Bachelor of Computer Science from University of Jammu, India in 2010 and Master of Computer Science and Engineering from Shri Mata Vaishno Devi University, India in the year 2013. She is currently pursuing Ph.D. from the Department of Computer Science & Information Technology, School of Applied Sciences since 2016. Her main research work focuses on routing in Computer Networks and Cloud Computing.
Email: bhagyalakshmi.magotra@gmail.com

Dr. Deepti Malhotra pursed Bachelor of Computer Science from University of Jammu, India in 2005, M.E. Computer Science from Thapar University, India in the year 2007 and completed her Ph.D. from University of Jammu in 2013. She is currently working as Assistant Professor in Department of CS&IT in Central University of Jammu, J&K, India. Having teaching and research experience of 10 years, she has published more than 37 research papers in reputed international journals and conferences. She is a Member of the Institution of Engineers (India) and a Life Time Member of All India Science Congress. In 2019, she has also organized an AICTE training and learning academy (ATAL) Faculty Development Programme (FDP) on the topic "Artificial Intelligence". Her main research work focuses on Grid Computing, Cloud Computing, Big Data Analytics, Data Mining, Artificial Intelligence and Natural Language Processing.
Email: deepti.csit@cujammu.ac.in

Appendix I

S. No.	Abbreviation	Description
1	CDC	Cloud Data Center
2	CPU	Central Processing Unit
3	CSP	Cloud Service Provider
4	DVMC	Dynamic Virtual Machine Consolidation
5	ESM	Energy, SLAV and number of Migrations
6	ESV	Energy and SLAV
7	GB	Giga Bytes
8	IQR	Inter Quartile Range
9	IQRMC	Inter Quartile Range Minimum Correlation
10	IQRMMT	Inter Quartile Range Minimum Migration Time
11	IQRMU	Inter Quartile Range Maximum Utilization
12	IQRRS	Inter Quartile Range Random Selection
13	LiRCUP	Linear Regression based CPU Usage Prediction
14	LR	Linear Regression
15	LRMC	Linear Regression Minimum Correlation
16	LRMMT	Linear Regression Minimum Migration Time
17	LRMU	Linear Regression Maximum Utilization
18	LRRS	Linear Regression Random Selection
19	MAD	Median Absolute Deviation
20	MADMMT	Median Absolute Deviation Minimum Migration Time
21	MADMC	Median Absolute Deviation Minimum Correlation
22	MADMU	Median Absolute Deviation Maximum Utilization
23	MADRS	Median Absolute Deviation Random Selection
24	MC	Maximum Correlation
25	MMT	Minimum Migration Time
26	MU	Minimum Utilization
27	PABFD	Power Aware Best Fit Decreasing
28	PC	Personal Computer
29	PNMC	P_n Estimator based Adaptive Threshold Policy Minimum Correlation
30	PNMMT	P_n Estimator based Adaptive Threshold Policy Minimum Migration Time
31	PNMU	P_n Estimator based Adaptive Threshold Maximum Utilization
32	PNRS	P_n Estimator based Adaptive Threshold Policy Random Selection
33	P_n	P_n Estimator based Adaptive Threshold Policy
34	P_n_AT$_H$P	P_n Estimator based Adaptive Threshold Policy
35	QoS	Quality of Service
36	RAM	Random Access Memory
37	RS	Random Selection
38	SLA	Service Level Agreement
39	SLATAH	SLA Time per Active Host
40	SLAPDM	SLA Performance Degradation due to Migration
41	SLAV	Service Level Agreement Violation
42	Sn	Sn based overload detection algorithm
43	S_nBODA	S_n based Host Overload Detection
44	THR	Static Threshold
45	THRMC	Static Threshold Minimum Correlation
46	THRMMT	Static Threshold Minimmum Migration Time
47	THRMU	Static Threshold Maximum Utilization
48	THRRS	Static Threshold Random Selection
49	VM	Virtual Machine