Egyptian Informatics Journal 18 (2017) 161–170

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

Full length article

Virtual machine consolidation enhancement using hybrid regression algorithms

Amany Abdelsamea [a,*], Ali A. El-Moursy [b,a], Elsayed E. Hemayed [c], Hesham Eldeeb [a]

[a] *Computers and Systems Department, Electronics Research Institute, Giza, Egypt*
[b] *Electrical and Computer Engineering Department, University of Sharjah, Sharjah, UAE*
[c] *Computer Engineering Department, Cairo University Faculty of Engineering, Giza, Egypt*

ARTICLE INFO

Article history:
Received 5 April 2016
Revised 2 October 2016
Accepted 19 December 2016
Available online 9 January 2017

Keywords:
VM consolidation
Host overload detection
Multiple regression
Single factors
Multiple factors

ABSTRACT

Cloud computing data centers are growing rapidly in both number and capacity to meet the increasing demands for highly-responsive computing and massive storage. Such data centers consume enormous amounts of electrical energy resulting in high operating costs and carbon dioxide emissions. The reason for this extremely high energy consumption is not just the quantity of computing resources and the power inefficiency of hardware, but rather lies in the inefficient usage of these resources. VM consolidation involves live migration of VMs hence the capability of transferring a VM between physical servers with a close to zero down time. It is an effective way to improve the utilization of resources and increase energy efficiency in cloud data centers. VM consolidation consists of host overload/underload detection, VM selection and VM placement. Most of the current VM consolidation approaches apply either heuristic-based techniques, such as static utilization thresholds, decision-making based on statistical analysis of historical data; or simply periodic adaptation of the VM allocation. Most of those algorithms rely on CPU utilization only for host overload detection. In this paper we propose using hybrid factors to enhance VM consolidation. Specifically we developed a multiple regression algorithm that uses CPU utilization, memory utilization and bandwidth utilization for host overload detection. The proposed algorithm, Multiple Regression Host Overload Detection (MRHOD), significantly reduces energy consumption while ensuring a high level of adherence to Service Level Agreements (SLA) since it gives a real indication of host utilization based on three parameters (CPU, Memory, Bandwidth) utilizations instead of one parameter only (CPU utilization). Through simulations we show that our approach reduces power consumption by 6 times compared to single factor algorithms using random workload. Also using PlanetLab workload traces we show that MRHOD improves the ESV metric by about 24% better than other single factor regression algorithms (LR and LRR). Also we developed Hybrid Local Regression Host Overload Detection algorithm (HLRHOD) that is based on local regression using hybrid factors. It outperforms the single factor algorithms.

1. Introduction

Cloud computing can be considered as a computing paradigm with many exciting features like on-demand computing resources, elastic scaling, elimination of up-front capital and operational expenses, and establishing a pay-as-you-go business model for computing and information technology services [1]. Cloud computing data centers consume enormous amounts of electrical energy resulting in high operating costs and carbon dioxide emissions. The reason for high energy consumption is not just the quantity of computing resources and the power inefficiency of hardware, but rather lies in the inefficient usage of these resources. One way to address the energy inefficiency is to leverage the capabilities of the virtualization technology. The reduction in energy consumption can be achieved by switching idle nodes to low-power modes (i.e., sleep, hibernation), thus eliminating the idle power consumption. Moreover, by using live migration the VMs can be dynamically consolidated to the minimal

[*] Corresponding author.
E-mail addresses: amany@eri.sci.eg (A. Abdelsamea), aelmoursy@sharjah.ac.ae (A.A. El-Moursy), hemayed@ieee.org (E.E. Hemayed), heldeeb@eri.sci.eg (H. Eldeeb).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.eij.2016.12.002

number of physical nodes according to their current resource requirements.

Most of current researches migrates VMs based on CPU utilization since there is a relationship between the total power consumption by a server and its CPU utilization [2]. Basically their model proposes that power consumption by a server grows linearly with the growth of the CPU utilization. CPU utilization based models are able to provide an accurate prediction for CPU-intensive applications; however they tend to be inaccurate for other types of applications like network, I/O and memory intensive applications [2]. The purpose of this work is:

- Develop Multiple Regression Host Overload Detection algorithm (MRHOD).
- Develop Hybrid Local Regression Host Overload Detection algorithm (HLRHOD).
- Compare different host overload detection algorithms.

The paper is organized as follows: Section 2 explains dynamic virtual machine consolidation. Section 3 presents the virtual machine selection. Section 4 discusses the related work. Section 5 discusses the Multiple Regression Host Overload Detection (MRHOD) algorithm. Section 6 discusses the Hybrid Local Regression Host Overload Detection. Section 7 discusses the evaluation methodology. Section 8 simulation results and analysis. Finally, the conclusion and future work is discussed in Section 9.

2. Dynamic virtual machine consolidation

All Dynamic Virtual Machine (VM) consolidation is a promising approach for reducing energy consumption by dynamically adjusting the number of active machines to match resource demands. To address this problem, most of the current approaches apply Regression based algorithms that is based on estimation of future CPU utilization. The limitation of these approaches is that they lead to sub-optimal results and do not allow the administrator to explicitly set a QoS goal. Comparison between types of Host Overload Detection Algorithms [3] is shown in Table 1.

The static utilization threshold is a simple method since it is based on a fixed CPU utilization threshold but it is unsuitable for dynamic environment. Adaptive utilization based algorithms are suitable for dynamic environment but give poor prediction of host overloading. Regression based algorithms give better predictions of host overloading since they are based on estimation of future CPU utilization but they are complex. Once a host overload is detected, the next step is to select VMs to offload the host to avoid performance degradation. Once a host overload is detected, the next step is to select VMs to offload the host to avoid performance degradation.

3. Virtual machine selection

Once a host overload is detected, the next step is to select VMs to offload the host to avoid performance degradation. After a selection of a VM to migrate, the host is checked again for being overloaded. If it is still considered as being overloaded, the VM selection policy is applied again to select another VM to migrate from the host. This is repeated until the host is considered as being not overloaded. This section presents three policies for VM selection.

3.1. Minimum migration time (MMT)

The Minimum Migration Time policy migrates a VM that requires the minimum time to complete a migration relative to the other VMs allocated to the same host. The migration time is estimated as the amount of RAM utilized by the VM divided by the spare network bandwidth available for the host [6].

3.2. Random choice (RC)

Random choice policy is another simple method to select VMs from overloading hosts. It randomly selects a VM to be migrated from the host according to a uniformly distributed discrete random variable [6]. If it is still overloaded, repeat the step until the host considered being not overloaded.

3.3. Maximum correlation

The idea behind the Maximum Correlation (MC) policy is that the higher the correlation between the resource usages by applications running on an oversubscribed server is the higher the probability of the server overloading will be. According to this idea, we select those VMs to be migrated that have the highest correlation of the CPU utilization with other VMs. To estimate the correlation between CPU utilizations multiple correlation coefficients is applied. It is used in multiple regression analysis to assess the quality of the prediction of the dependent variable. The multiple correlation coefficients correspond to the squared correlation between the predicted and the actual values of the dependent variable [6].

4. Related work

Prior approaches to energy efficient dynamic VM consolidation can be broadly divided into three categories: periodic adaptation of the VM placement (no overload detection), threshold-based heuristics, and decision-making based on statistical analysis of historical data. All categories enjoyed significant attention from the research community, so we focus here only on a certain subset of the most relevant work.

4.1. Periodic adaptation of VM placement

Lots of work has been proposed for energy efficiency and management on cloud data centers. In some approaches, VM consolidation has been formulated as an optimization problem with the objective to find a near optimal solution since an optimization problem is associated with constraints, such as data center capacity and SLA. Farahnakian et al. [8] presented distributed system architecture to perform dynamic VM consolidation to improve

Table 1
Types of host overload detection algorithms.

Type	Static utilization threshold based algorithms	Adaptive utilization based algorithms	Regression based algorithms
Explanation	Based on fixed CPU utilization threshold	Based on statistical analysis of historical data of VM	Based on estimation of future CPU utilization
Pros	Simple	Suitable for dynamic environment (robust)	Better predictions of host overloading
Cons	Unsuitable for dynamic environment	Poor prediction of host overloading	Complex
Examples	THR (Averaging threshold-based algorithm) [4]	MAD (Median Absolute Deviation) [5], IQR (Inter Quartile Range) [6]	LR (Local Regression) [7], LRR (Robust local Regression) [7]

resource utilizations of PMs and to reduce their energy consumption. The authors proposed a dynamic VM consolidation approach that uses a highly adaptive online optimization meta-heuristic algorithm called Ant Colony System (ACS) to optimize VM placement. The proposed ACS-based VM Consolidation (ACS-VMC) approach uses artificial ants to consolidate VMs into a reduced number of active PMs according to the current resource requirements. These ants work in parallel to build VM migration plans based on a specified objective function. The authors plan to further improve the proposed system model by clustering PMs and assigning them to the respective consolidation managers and also they intend to evaluate the performance of other heuristic methods for VM consolidation.

Ghribi et al. [9] presented two algorithms for energy efficient scheduling of virtual machines (VMs) in cloud data centers. Modeling of energy aware allocation and consolidation to minimize overall energy consumption leads us to the combination of an optimal allocation algorithm with a consolidation algorithm relying on migration of VMs at service departures. The optimal allocation algorithm is solved as a bin packing problem with a minimum power consumption objective. It is compared with an energy aware best fit algorithm. The exact migration algorithm results from a linear and integer formulation of VM migration to adapt placement when resources are released. The results show the benefits of combining the allocation and migration algorithms and demonstrate their ability to achieve significant energy savings while maintaining feasible convergence times when compared with the best fit heuristic. This approach is achieved at the virtual machine (VM) level (or IaaS level) and hence it is better to be achieved at the task level to be able to fit the Platform or Software as a Service (PaaS, SaaS) levels.

4.2. Threshold-based heuristics

Zhu et al. [10] studied the dynamic VM consolidation problem and applied a heuristic of setting a static CPU utilization threshold of 85% to determine when a host is overloaded. The host is assumed to be overloaded when the threshold is exceeded. However, this approach is not suitable for an IaaS environment serving different kinds of applications, as the threshold values have to be tuned for each workload type to allow the consolidation controller to perform efficiently.

Zhou et al. [11] proposed a virtual machine deployment algorithm called Three-threshold Energy Saving Algorithm (TESA), which is based on the linear relation between the energy consumption and (processor) resource utilization. In TESA, according to load, hosts in data centers are divided into four classes, host with light load, host with proper load, host with middle load and host with heavy load by setting three thresholds. Based on TESA, five kinds of VM selection policies (minimization of migrations policy based on TESA (MIMT), maximization of migrations policy based on (MAMT), highest potential growth policy based on TESA (HPGT), lowest potential growth policy based on TESA (LPGT) and random choice policy based on TESA (RCT) are presented. In real workload it is not practical to determine the actual value of the three thresholds. Fortunately, it is likely to obtain optimal intervals between each two thresholds. Its disadvantage is that the host overload detection is based on setting three fixed thresholds which is not suitable for the dynamic nature of cloud. Also it doesn't consider multiple factors when determining the host utilization.

4.3. Decision making based on statistical analysis of historical data

Fixed values of utilization thresholds are unsuitable for an environment with dynamic and unpredictable workloads, in which different types of applications can share a physical resource. The system should be able to automatically adjust its behavior depending on the workload patterns exhibited by the applications.

Beloglazov and Buyya [6] presented a heuristics for dynamic consolidation of VMs based on an analysis of historical data from the resource usage by VMs. To calculate the upper CPU utilization threshold a statistical methods (Median absolute deviation and Interquartile range) are used. Also Regression based algorithms (Local regression and Local robust regression) that are based on estimation of future CPU utilization are used. These statistical methods and policies to select a VM to be migrated are combined to form various strategies. These approaches do not consider hybrid parameters for host utilization calculation on contrary they depend on CPU only.

Monil and Rahman [12] proposed a new host overload detection algorithm based on mean, median and standard deviation (MMSD) of utilization of VMs. Also a fuzzy VM selection method is proposed which takes intelligent decision to select a VM to be migrated from one host to the other. The disadvantage of the host overload detection algorithm is that they still use mean and standard deviation that are very much influenced by terminal/outlier values.

Shidik et al. [13] presented a VM selection model in dynamic VM consolidation to improve the energy efficiency in cloud data center based on Fuzzy Markov Normal Algorithm. Fuzzy logic has been used for categorizing the attributes of VM candidates then deciding to which category VM should be migrated. The proposed VM selection model has been evaluated using various VM instance conditions (homogeneous or heterogeneous). Its disadvantage is that it does not consider hybrid factors.

Beloglazov and Buyya [4] presented OpenStack Neat to provide an extensible framework for dynamic consolidation of VMs based on the OpenStack platform. The functionality covered by this project will be implemented in the form of services separate from the core OpenStack services. The services of this project will interact with the core OpenStack services using their public APIs. Now OpenStack Neat is used as the Terracotta [14] code base at the very early stage. Terracotta is an extension to OpenStack implementing dynamic consolidation of resources, e.g. Virtual Machines (VMs) using live migration. Since Terracotta is at its early stage and the devices to measure power are not on our premise we use an authenticated simulator.

Most of the above work relies on one parameter only for calculating the host utilization while this paper proposes the enhancement of host overload detection based on multiple regression and hybrid factors (CPU utilization, memory utilization and bandwidth utilization). None of the previous works use multiple regression technique for enhancement of host overload detection.

5. Multiple regression host overload detection (MRHOD)

5.1. Multiple regression

Multiple regression [15] is an extension of simple linear regression. It is used to assist the prediction of the value of a variable based on the value of two or more other variables. The variable we want to predict is called the dependent variable (or sometimes, the outcome, target or criterion variable). The variables we are using to predict the value of the dependent variable are called the independent variables (or sometimes, the predictor, explanatory or regressor variables). The design requirements of multiple regression are [16]:

- one dependent variable (criterion),
- Two or more independent variables (predictor variables),
- Sample data: at least 10 times as many cases as independent variables.

5.1.1. Multiple regression assumptions

Multiple regression is based on independence, normality, homoscedasticity and linearity assumptions [17].

5.1.2. Multiple regression model

A regression model that contains more than one regressor variable is called a multiple regression model [16] as shown in Eq. (1).

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \ldots + \beta_k X_{ki} + \varepsilon \tag{1}$$

where

Y_i is the dependent variable,
X_{ij} are the independent variables,
B_i is the slope (regression) coefficients which are partial derivatives of Y with respect to X variable,
ε is that nonsystematic part of Y not linearly related to any of the X's.

The key assumptions in this model are that the dependent variable Y is linearly related to the X's also there are no exact linear dependencies among the regressors. The independence assumption is shown in Eq. (2):

$$E(\varepsilon | X_1, X_2 \ldots X_k) = 0 \tag{2}$$

This assumption of a zero conditional mean for the error process implies that it does not systematically vary with the X's nor with any linear combination of the X's. The coefficients of the multiple regression models are estimated using sample data with k independent variables. To calculate these coefficients we must create k normal equations then solve these equations together to obtain the multiple regression coefficients. Eq. (3) consider the k-variable model. The estimated Ordinary Least Square (OLS) equation contains the parameters of interest:

$$\hat{y} = b_0 + b_1 x_1 + b_2 x_2 + \ldots + b_k x_k \tag{3}$$

The ordinary least squares criterion can be defined in terms of the OLS residuals, calculated from a sample of size n, from Eq. (4):

$$\min S = \sum_{i=1}^{n} (y_i - b_0 - b_1 x_{i1} - b_2 x_{i2} - \ldots - b_k x_{ik})^2 \tag{4}$$

The minimization of this expression is performed with respect to each of the k parameters b_0, b_1, b_2... b_k. We have a sample larger than the number of parameters to be estimated. The minimization is carried out by differentiating the scalar S with respect to each of the b's in turn, and setting the resulting first order condition to zero. This gives rise to (k + 1) simultaneous equations in (k + 1) unknowns, the regression parameters, that are known as the least squares normal equations. For the "k-variable" regression model, we can write out the normal equations as shown in Eqs. (5)–(8):

$$\sum y = nb_0 + b_1 \sum x_1 + b_2 \sum x_2 + b_3 \sum x_3 \tag{5}$$

$$\sum x_1 y = b_0 \sum x_1 + b_1 \sum x_1^2 + b_2 \sum x_1 x_2 + b_3 \sum x_1 x_3 \tag{6}$$

$$\sum x_2 y = b_0 \sum x_2 + b_1 \sum x_1 x_2 + b_2 \sum x_2^2 + b_3 \sum x_2 x_3 \tag{7}$$

$$\vdots$$

$$\sum x_k y = b_0 \sum x_k + b_1 \sum x_1 x_k + b_2 \sum x_2 x_k + \ldots + b_k \sum x_k^2 \tag{8}$$

These equations may be uniquely solved, by normal algebraic techniques or linear algebra, for the estimated least square parameters. The solution to the normal Equations are the least squares estimators of the regression coefficient. In next section we discuss how multiple regression is used in host overload detection.

5.2. Host overload detection using multiple regression

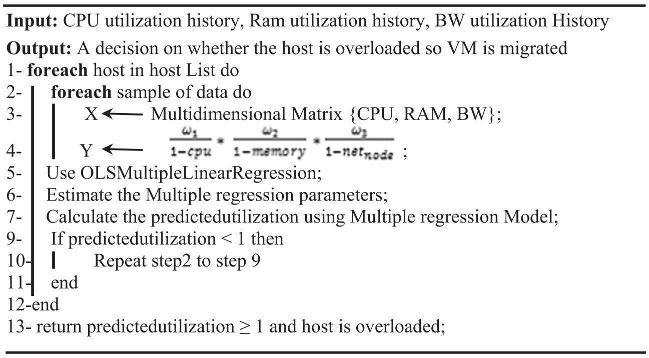
In order to host a VM, a physical machine must provide all resources the VM requires, including CPU, memory, storage and network bandwidth. It is obvious that different goals (CPU, RAM and BW) may have different scales or measures. It is not possible to calculate the host utilization based on (CPU, RAM and BW) by summed them up directly. Therefore, there must be an existing formula to calculate the host utilization. In this paper we use the formula developed in [18] to calculate the host utilization based on hybrid factors by using a metric that captures the combined CPU-network-memory load of virtual and physical servers. The volume of a physical or virtual server is defined as the product of its CPU, network and memory loads as shown in Eq. (9):

$$Vol_{node} = \frac{\omega_1}{1 - cpu} * \frac{\omega_2}{1 - memory} * \frac{\omega_3}{1 - net_{node}} \tag{9}$$

where
ω_i: The weight of CPU, memory and network load,
cpu: The physical host CPU utilization,
memory: The physical host memory utilization,
net_{node}: The physical host network port utilization.

In this paper we propose Multiple Regression Host Overload Detection algorithm (MRHOD) to enhance VM consolidation using hybrid factors. The pseudo code for the Multiple Regression Host Overload Detection (MRHOD) algorithm is presented in Algorithm 1.

Algorithm 1. multiple regression host overload detection algorithm (MRHOD)



```
Input: CPU utilization history, Ram utilization history, BW utilization History
Output: A decision on whether the host is overloaded so VM is migrated
1- foreach host in host List do
2-     foreach sample of data do
3-         X ← Multidimensional Matrix {CPU, RAM, BW};
4-         Y ← $\frac{\omega_1}{1-cpu} * \frac{\omega_2}{1-memory} * \frac{\omega_3}{1-net_{node}}$ ;
5-     Use OLSMultipleLinearRegression;
6-     Estimate the Multiple regression parameters;
7-     Calculate the predictedutilization using Multiple regression Model;
9-     If predictedutilization < 1 then
10-        Repeat step2 to step 9
11-    end
12-end
13- return predictedutilization ≥ 1 and host is overloaded;
```

First, the host CPU utilization, RAM utilization and BW utilization for each host are calculated as the average utilization of the VMs in a host divided by the maximum host utilization. Then the inputs to the multiple regression algorithm are two matrices:

- The first one is two dimensional array named X with size of rows equal to the length of Data and column size equal to the number of independent parameters which are three in this algorithm (CPU utilization, RAM utilization and BW utilization).
- The second input is one dimension array named Y which includes a sample of data of host utilization calculated from Equation (9).

The multiple regression is complicated since we need to obtain the coefficients of the regression model by solving the normal equations. We use Ordinary Least Square Multiple Regression function (OLSMultipleRegression function) [16,17] to calculate the regression coefficients (parameters) of the multiple regression model. The regression coefficients are calculated once per host. Once the regression coefficients are calculated the predictedhostutilization equation can be formed as shown in Eq. (10):

$$\mathrm{predictedUtilization} = \mathrm{b}_0 + (\mathrm{b}_1 * \mathrm{CPUutilization}) + (\mathrm{b}_2 * \mathrm{Ramutilization}) + (\mathrm{b}_3 * \mathrm{Bwutilization}) \quad (10)$$

By substituting values of CPU utilization and RAM utilization and BW utilization in the equation we obtain the predictedUtilization. The triggering point of the algorithm is that if predictedUtilization is greater than or equal 1 then the host is considered to be overloaded so we select a VM to be migrated from the overloaded host.

6. Hybrid local regression host overload detection method (HLRHOD)

Local regression [7] is used to model a relation between a predictor variable and response variable. Local regression model is based on the form:

$$Y_i = f(x_i) + \epsilon_i \quad (11)$$

where

f(x) is an unknown function,

ϵ_i is an error term, representing random errors in the observations or variability from sources not included in the x_i.

The main idea of the method of local regression is fitting simple models to localized subsets of data to build up a curve that approximates the original data. The proposed method is based on the Loess's method [7] proposed by Cleveland [19]. Local regression host overload detection algorithm presented in [6] is based on the estimation of the future CPU utilization. They outperform the threshold-based and adaptive-threshold based algorithms because of their better prediction of host overloading but it depends on singe factor (CPU utilization) for host overload detection.

We use the formula developed in [18] to calculate the host utilization based on hybrid factors by using a metric that captures the combined CPU-network-memory load of virtual and physical servers. After calculating the host utilization from the above formula, we use the Local Regression (LR) host overload detection algorithm proposed in [6].

7. Evaluation methodology

7.1. Experimental setup

Since cloud computing is purely an internet based technology it suffers from a major problem which is the monetary cost involved in ensuring that the internet is always accessible. Additionally it is exceedingly hard to direct repeatable large scale experiments on a real system, which is needed to make an evaluation and comparison to the recommended heuristics. Additionally for assurance of the repeatability of experiments, simulations are picked as an approach to highlight the enhancement of recommended algorithms. Simulation tools have important benefits: no capital cost presented, provide enhanced results as using such tools help to change inputs and other parameters also in an easier way which result in better and efficient outputs. Simulation tools are also easy to learn since while working with such simulation tools user needs to have only programming abilities [20]. Simulation-based techniques allow the test of experiments with various workload blend and resource performance scenarios on simulated systems for elaborating, examining versatile application provisioning methods.

The CloudSim toolkit [20] has been chosen as a simulation platform, as it is a modern simulation framework aimed at Cloud computing environments. We use CloudSim toolkit to simulate all combinations of the following host overload detection algorithms [6] (THR, IQR, MAD, LR, LLR, MMSD and MRHOD and HLRHOD) with three VM selection algorithms (MMT, RC, MC) to detect the host overload based on hybrid factors.

7.2. Power model

Power consumption by computing nodes in data centers is mostly determined by the CPU, memory, disk storage, power supplies and cooling systems [21]. This fact combined with the difficulty of modeling power consumption by modern multi-core CPUs makes building precise analytical models a complex research problem. Therefore, we utilize real data on power consumption provided by the results of the SPECpower benchmark [6] instead of using an analytical model of power consumption by a server. We have simulated a data center that comprises 50 hosts and 50 VMs using random workload traces. The host overload is frequently evaluated according to the scheduling interval which is set to 300 s [5]. The host types are:

HP ProLiant ML110 G4 (Intel Xeon 3040, 2 cores 1860 MHz, 4 GB), and HP ProLiant ML110G5 (Intel Xeon 3075, (2 cores 2660 MHz, 4 GB). The configuration and power consumption characteristics [5] of the selected servers are shown in Table 2.

7.3. Performance metrics

In order to compare the efficiency of the algorithms we use several metrics to evaluate their performance. The following metrics are used:

Total energy consumption (E) is defined as the sum of energy consumed by the physical resources of a data center as a result of application workloads. Energy consumption is calculated according to the model defined in [10].

Number of VM migrations: For dynamic VM consolidation once the overloaded or under-loaded hosts are found, the VMs are then selected for migration. The minimization of the VM migration time is the most important constraint in migration step and it is achieved by the reduction of the total number of VM migrations.

SLA (Service Level Agreement): It can be determined in terms of such characteristics as minimum throughput or maximum response time delivered by the deployed system. As these characteristics can vary for different applications, it is necessary to use workload independent metric that can be used to evaluate SLA delivered to any VM deployed in an IaaS. SLAs are delivered when 100\% of the performance requested by applications in a VM is provided at any time bounded only by the parameters of the VM [10].

We use two metrics for measuring the level of SLA violations [9] in an IaaS environment which are defined as: The first metric is the

Table 2
Power consumption by the selected servers at different load levels in Watts [5].

Server	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP ProLiant G4	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP ProLiant G5	93.7	97	101	105	110	116	121	125	129	133	135

percentage of time, during which active hosts have experienced the CPU utilization of 100\%, SLA violation Time per Active Host (SLATAH) as shown in Eq. (12):

The percentage of time, during which active hosts have experienced the CPU utilization of 100%, SLA violation Time per Active Host (SLATAH)

$$\mathrm{SLATAH} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}}\frac{T_{si}}{T_{ai}} \tag{12}$$

where

N is the number of hosts,
T_{si} is the total time during which the host i has, experienced the utilization of 100% leading to an SLA violation,
T_{ai} is the total of the host i being in the active state (serving VMs).

The second metric is the overall performance degradation by VMs due to migrations, Performance Degradation due to Migrations (PDM) as shown in Eq. (13):

$$\mathrm{PDM} = \frac{1}{\mathrm{M}}\sum_{\mathrm{j}=1}^{\mathrm{M}}\frac{C_{dj}}{C_{rj}} \tag{13}$$

where

M is the number of VMs,
C_{dj} is the estimate of the performance degradation of the VM j caused by migrations,
C_{rj} is the total CPU capacity requested by the VM j during its lifetime.

Both the SLATAH and PDM metrics are independently and with equal importance characterize the level of SLA violations by the infrastructure. A combined metric that encompasses both performance degradation due to host overloading and performance degradation due to VM migrations is shown in Eq. (14). They denote the combined metric SLA Violation (SLAV) [6] which takes place when a VM cannot get the promised Quality of Service (QoS).

$$\mathrm{SLAV} = \mathrm{SLATAH} * PDM \tag{14}$$

Energy consumption by physical nodes and SLAV are negatively correlated as energy can usually be decreased by the cost of the increased level of SLA violations. The objective of the resource management system is to minimize both energy and SLA violations. Therefore, a combined metric denoted by Energy and SLA Violations (ESV) captures both energy consumption and the level of SLA violations is proposed in [6]. For the ESV metric lower is better since it indicates that energy saving is higher than SLA violations.

$$\mathrm{ESV} = \mathrm{E} * \mathrm{SLAV} \tag{15}$$

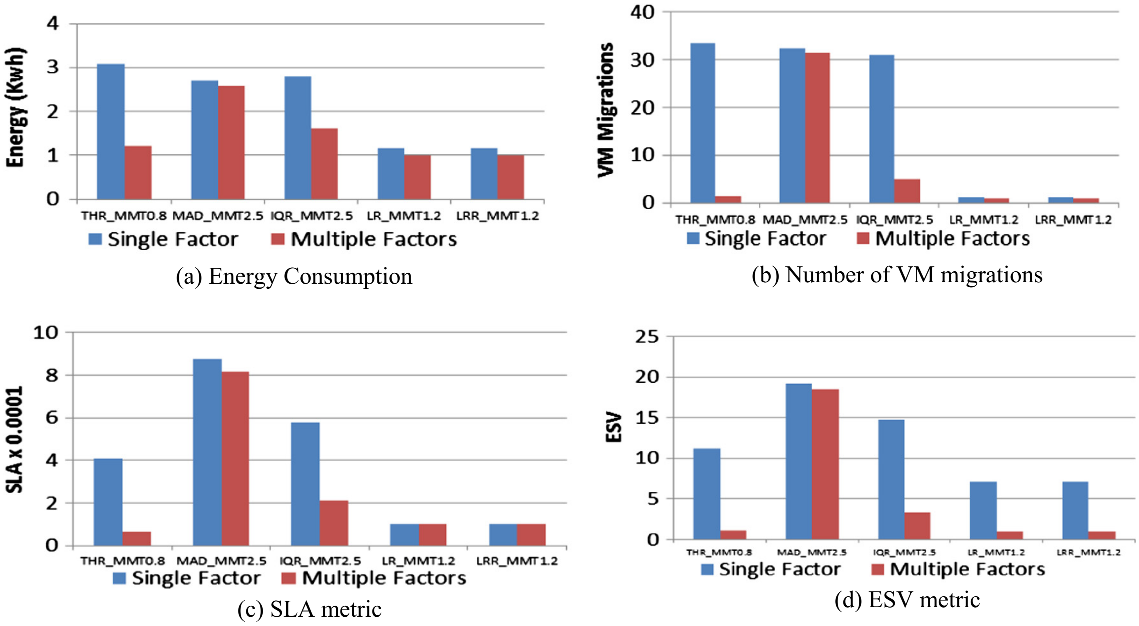
8. Simulation results and analysis

We have simulated all combinations of the following host overload detection algorithms (THR, IQR, MAD, MMSD, LR and LLR and MRHOD and HLRHOD) with three VM selection algorithms (MMT, RC and MC) to detect the host overload based on hybrid parameters. For host overload detection based on three factors (CPU, RAM and BW) utilizations we use Multiple Regression Host Overload Detection algorithm (MRHOD) and (HLRHOD).

8.1. VM selection policy evaluation

Initially we start by evaluating MMT across different algorithms (THR, MAD, IQR, LR, and LRR) using random workload traces since it is the easiest type of workload to start our experiments with it since there are no large real world data files, no sensitive data and easily modified without affecting operation. There is more saving in energy consumption, Number of VM migrations and SLA metric when using hybrid factors compared to the results obtained from single factor host utilization overload detection as shown in Fig. 1.

Energy consumption is reduced by about 15 times in case of hybrid factors (THR and IQR) than single factor and by about 5 times in case of hybrid factors (MAD, LR and LRR) than single factor. Number of VM ESV is increased by about 10 times in case of hybrid factors (THR, IQR, LR and LRR) than single factors. Since



(a) Energy Consumption

(b) Number of VM migrations

(c) SLA metric

(d) ESV metric

Figure 1. Algorithm combinations comparing multiple factors and single factors.

the results obtained from hybrid factors is better than the results from single factor so the VM behavior is rarely a function in one variable it should be a function of multiple factors. Regression based algorithms outperform the threshold-based and adaptive-threshold based algorithms using hybrid factors as well as using single factors since the Energy Service level agreement Violation metric (ESV) in case of LR-MMT and LRR-MMT is reduced compared to MAD-MMT, IQR-MMT and THR-MMT in both cases.

Next we perform similar evaluation for Random choice VM selection across the same set of algorithms (THR, MAD, IQR, LR and LRR). Energy consumption by physical nodes and SLAV are negatively correlated. As the energy consumption increase the SLAV decrease and vice versa. ESV metric is used to compare between algorithms as shown in Fig. 2. MAD_RS is the highest algorithm in term of ESV. LR_RS is the lowest algorithm in term of ESV so it is preferred to be used.

In Fig. 3 we compare between MMT, RS and MC VM selection algorithms. MMT gives poor results in term of ESV in adaptive-threshold based algorithms (MAD, IQR) compared to RS and MC. MC gives high value of ESV in case of MAD compared to MMC and MC. In case of regression algorithms they nearly give the same performance as shown in Fig. 3.

8.2. Sensitivity analysis

8.2.1. Safety parameter

In this experiment we study the effect of changing the safety parameter on MRHOD to determine the best value of safety parameter that gives best performance. MRHOD gives the best values in term of energy consumption, ESV metric when safety parameter is 1.5 as shown in Table 3.

We study also the effect of changing the safety parameter on HLRHOD algorithm. HLRHOD gives the best values in term of energy consumption and ESV metric when safety parameter is 1.4 as shown in Table 4.

8.2.2. Number of VMs

Our sensitivity analysis is based on changing the number of virtual machines while keeping the number of hosts constant (100 host) and scheduling interval equals to 300 using random workload traces. We study the effect of varying the number of VMs on energy consumption and ESV metric for different algorithms and the proposed HLRHOD algorithm and MRHOD algorithm as shown in Tables 5 and 6 respectively.

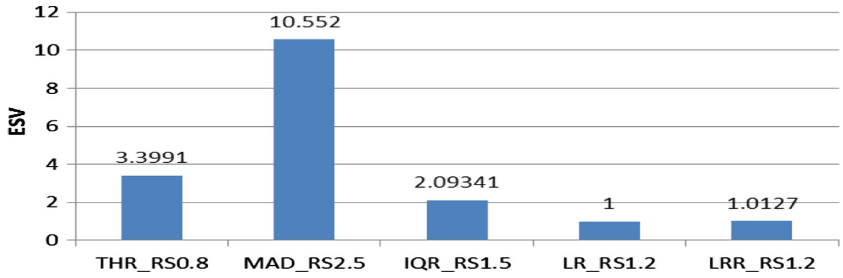


Figure 2. ESV for RS with different host overload detection algorithms.

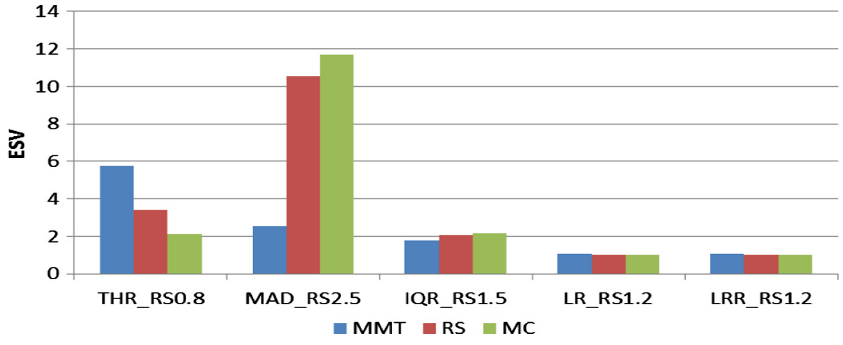


Figure 3. ESV comparison for RS vs. MMT vs. MC.

Table 3
Safety parameter for MRHOD.

Safety parameter	0.8	0.9	1	1.1	1.2	1.3	1.4	1.5	1.6	1.7
Energy (kW h)	16.06	16.01	15.23	15.58	15.56	15.46	13.53	13.48	15.72	15.74
VM Migrations (total no. of VM migrations)	349	386	348	282	274	216	145	120	119	116
SLA (100\% of the application perf. done at any time)	0.011	0.014	0.016	0.01	0.0125	0.009	0.007	0.006	0.005	0.004
PDM (estimated Perf. degradation/requested capacity)	0.02	0.02	0.02	0.02	0.02	0.01	0.01	0.01	0.01	0.01
SLATAH (Ratio)	75.14	75.61	80.93	81.93	80.52	83.31	82.05	80.42	83.27	83.71
SLAV	1.502	1.51	1.618	1.63	1.6104	0.833	0.8205	0.8042	0.8327	0.8371
ESV	24.13	24.21	24.65	25.52	25.058	12.87	11.1	**10.841**	13.09	13.17

Table 4
Safety parameter for HLRHOD.

Safety parameter	**0.8**	**0.9**	**1**	**1.1**	**1.2**	**1.3**	**1.4**	**1.5**	**1.6**	**1.7**
Energy (kW h)	16.06	16.22	15.65	15.56	15.55	15.46	13.53	15.66	15.72	15.74
VM Migrations (total no. of VM migration)	337	351	317	260	270	216	145	120	119	116
SLA (100\ % of the application perf. done at any time)	0.011	0.013	0.014	0.011	0.0126	0.00962	0.00744	0.00479	0.00539	0.004
PDM (estimated Perf. degradation/requested capacity)	0.01	0.02	0.02	0.01	0.02	0.01	0.01	0.01	0.01	0.01
SLATAH (ratio)	75.71	75.22	79.6	78.74	81.13	83.31	82.05	83.84	83.27	83.71
SLAV	0.757	1.504	1.592	0.7874	1.6226	0.8331	0.8205	0.8384	0.8327	0.837
ESV	12.1	24.401	24.915	12.252	25.231	12.878	**11.101**	13.129	13.09	13.17

Table 5
Energy consumption vs. number of VMs.

Algorithm	30	50	70	90	110	130	150	170	190
THR_MMT-0.8	2.759	3.139	2.953	2.693	2.477	2.433	2.346	2.208	2.344
IQR-MMT-1.5	2.406	2.6579	2.41	2.251	2.06	2.025	2.104	1.958	2.109
MAD-MMT-2.5	2.371	2.632	2.409	2.184	2.042	1.998	2.055	1.959	2.096
MMSD-MMT-2.5	**2.301**	**2.554**	**2.38**	**2.148**	**1.962**	**1.935**	**1.981**	**1.912**	**2.031**
LRR-MMT-1.2	1.367	1.458	1.328	1.147	1.101	1.062	1.154	1.139	1.1717
LR-MMT-1.2	1.367	1.458	1.328	1.147	1.101	1.062	1.154	1.139	1.1717
HLRHOD-1.4	1.009	1.172	1.103	1.022	1.015	1.035	1.014	1.004	1.049

Table 6
ESV metric vs. number of VMs.

Algorithm	30	50	70	90	110	130	150	170	190
THR_MMT-0.8	9.5	10.822	9.717	8.916	8.137	7.837	9.73	12.065	15.22
IQR-MMT-1.5	15.292	18.977	19.797	18.277	18.255	20.286	21.093	21.756	24.82
MAD-MMT-2.5	**14.491**	**19.259**	**18.662**	**19.275**	**19.324**	**21.398**	**22.224**	**22.665**	**24.811**
MMSD-MMT-2.5	15.023	20.807	21.764	20.599	20.172	22.289	22.82	23.683	27.109
LRR-MMT-1.2	4.321	6.008	6.12	5.793	5.549	5.537	6.71	6.51	6.66
LR-MMT-1.2	4.321	6.008	6.12	5.793	5.549	5.537	6.71	6.51	6.66
HLRHOD-MMT-1.4	0.976	1.183	1.108	1.029	1.022	1.016	0.999	1.002	1.008

We normalize the results with respect to MRHOD-MMT1.5. MRHOD and HLRHOD algorithms save more energy than all algorithms when we run random workload traces. MRHOD gives better results than MMSD since it is a regression based algorithm which is based on estimation of future CPU utilization which represents a better prediction of host overloading but MMSD is based on statistical analysis of historical data using mean, median and standard deviations which are sensitive to outliers. As number of VMs (30, 50, and 70) increases, the energy saving of MRHOD and HLRHOD algorithms gets higher than LR and LRR algorithms. For high number of VMs, the improvement in MRHOD and HLRHOD is reduced compared to LR and LRR however they are still the best marginally.

8.2.3. Scheduling interval

In this experiment we study the effect of changing the scheduling interval on the energy consumption and ESV metric for different algorithms beside the proposed MRHOD algorithm. We change the scheduling interval while keep the number of hosts constant (50 host) and the number of VMs constant (50) as shown in Table 7. We normalize the results with respect to MRHOD-MMT1.5. As the scheduling interval increases, the ESV decreases across all algorithms.

8.3. Algorithms comparative analysis

8.3.1. Random workload

Comparison between different single factor algorithms and Multiple Regression Host Overload Detection (MRHOD) is shown in Table 8. In Table 8, each algorithm name is followed by a number represents the value of the safety factor used while running the algorithm. Safety factor is a parameter of the method that indicates how aggressively the system consolidates VMs. It allows the adjustment of the safety of the method, the lower the safety parameter, the less the energy consumption, but the higher the level of SLA violations caused by the consolidation. For Table 8 we have selected for each algorithm a different safety parameter since not all of them perform the best with the same safety parameter hence the results shown are for the best performing safety

Table 7
ESV metric vs. scheduling interval.

Algorithm	300	400	500	600	700	800	900
THR_MMT	11.523	7.874	5.273	4.174	3.218	2.673	2.338
IQR-MMT	19.169	13.078	9.735	7.583	6.751	5.397	4.838
MAD-MMT	18.95	21.133	10.179	7.834	6.374	5.389	4.909
MMSD-MMT	**20.817**	**14.42**	**10.699**	**8.441**	**6.947**	**6.068**	**4.996**
LRR-MMT	5.845	5.296	4.796	4.369	4.008	3.656	3.415
LR-MMT	5.845	5.296	4.796	4.369	4.008	3.656	3.415
HLRHOD-MMT	1.156	1.112	1.068	1.025	0.986	0.948	0.942

Table 8
Simulation result of host overload detection algorithms.

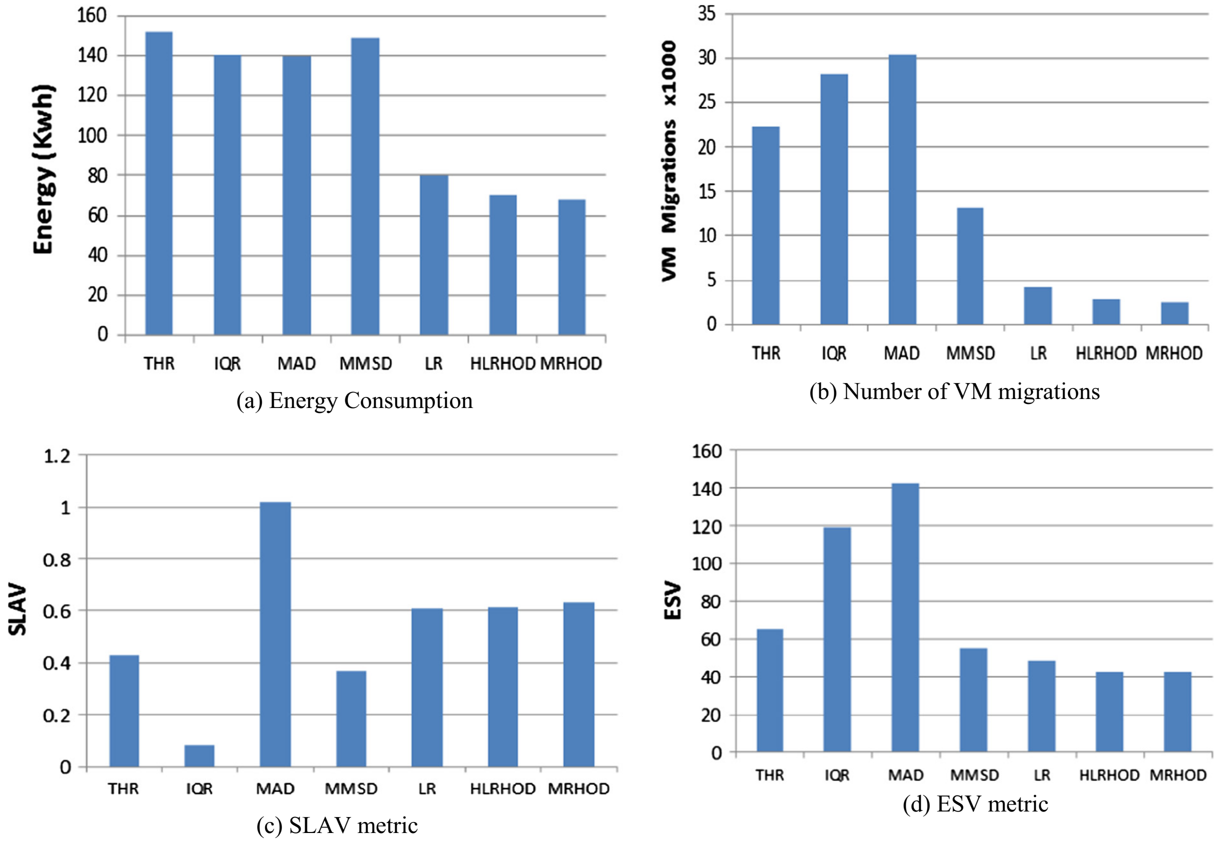
Policy	Energy Consumption	Number Of Vm Migrations	SLA	PDM	SLATAH	SLAV	ESV
THR_MMT0.8	41.81	4839	0.03048	0.23	12.99	2.987	124.917
IQR-MMT1.5	36.4	4685	0.06521	0.27	20.85	5.629	204.914
MAD-MMT2.5	37.84	4490	0.04304	0.25	17.34	4.335	164.036
MMSD_MMT2.5	**34.57**	**1613**	**0.01921**	**0.09**	**20.45**	**1.841**	**63.626**
LRR-MMT1.2	19.7	167	0.00765	0.031	99.12	3.001	59.12
LR-MMT1.2	19.7	167	0.00765	0.031	99.12	3.001	59.12
HLRHOD-1.4	13.53	145	0.00744	0.01	82.05	0.82	11.101
MRHOD-MMT1.5	13.48	120	0.0066	0.01	67.67	0.804	10.8406

parameter for each algorithm. For recently proposed algorithms (THR, IQR, MAD, LRR, and LR) we used the recommended safety parameters provided in [6]. Single factor Local regression based algorithms outperform single factor THR, IQR and MAD and MMSD due to better predictions of host overloading, and therefore decreased SLA violations due to host overloading (SLATAH) and the number of VM migrations. Across different metrics for power/performance and QOS Multiple Regression Host Overload Detection (MRHOD) algorithm outperforms all other algorithms that use single factor in host overload since ESV metric is improved 6 times compared to other single factor regression algorithms (LR, and LRR) and more than 10 times (an order of magnitude) compared to MAD, MMSD, IQR and THR.

8.3.2. PlanetLab workload

In previous section we use random workload traces to compare between the literatures algorithms (THR, MAD, MMSD, IQR, LR, LRR) and the proposed algorithms (HLRHOD and MRHOD). But to make a simulation based evaluation applicable, it is important to conduct experiments using workload traces from a real system. For our experiments we have used data provided as a part of the CoMon project, a monitoring infrastructure for PlanetLab [6]. We use PlanetLab workload (Number of hosts is 800) and (Number of virtual machines is 1033). Comparison between different single factor algorithms and Multiple factor Host Overload Detection algorithms are shown in Fig. 4.

Across different metrics for power/performance and QOS Multiple Regression Host Overload Detection (MRHOD) algorithm outperforms all other techniques that utilize single factor in host overload detection since the energy consumption is minimized by about 20% than LR and LRR algorithms. SLAV is minimized in LRR and LR than other multiple factor algorithms because the energy and SLAV are negatively correlated. So we use ESV metric to decide if the host is the best in predicting of node overloading. For MRHOD the ESV metric is improved by about 24% compared to other single factor regression algorithms (LR, and LRR) and more



(a) Energy Consumption

(b) Number of VM migrations

(c) SLAV metric

(d) ESV metric

Figure 4. Algorithms comparative comparison using PlanetLab workload.

Table 9
Multiple Regression significance test output.

	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%
Intercept	**−0.15522808**	**0.017376925**	**−8.933000704**	**2.10386E−09**	**−0.19094686**	**−0.1195093**
CPU	**0.38366454**	**0.081251539**	**4.721935656**	**7.00234E−05**	**0.216649609**	**0.55067947**
RAM	**0.307722279**	**0.10047659**	**3.062626633**	**0.005052632**	**0.101189692**	**0.514254867**
BW	**−0.061809984**	**0.227545473**	**−0.271637942**	**0.78804571**	**−0.529536402**	**0.405916434**

than 40% compared to THR, MAD, and IQR. Multiple factor regression algorithms give enhanced results than all other single factor algorithms due to improved predictions of host overload detection, and thus minimized the number of migrations of VM which cause a minimization in SLAV. Number of VM migration is very high in MAD and IQR algorithms.

8.4. Multiple regression significance test using Anova

We perform multiple regression significance test using Anova and data analysis tool in excel [22]. R-square is a statistical measure of how close the data are to the fitted regression line. In our experiment adjusted R square is 0.91 which means the model greatly fits the data.

We test the significance of the independent factors (CPU, RAM, BW) on the independent factor (host utilization) using Anova. We obtain the following results presented in Table 9.

From table 9 it is clear that the p value of CPU and RAM is smaller than 0.05 so CPU and RAM values are significant. While the p value for BW is larger than 0.05 so it is insignificant. Although the BW is insignificant but it is very important when considering host utilization because for many applications, the performance does not rely only on CPU utilization. For applications that require communication among services, the communication cost can also influence the overall performance. Furthermore, there are applications require a huge amount of memory hence; memory utilization can also influence the overall performance. So the CPU, RAM and BW are very important parameters for applications so they are used in our proposed multiple regression algorithms.

9. Conclusion

Multiple Regression is an effective way to solve virtual machine consolidation challenges. It is an extension to Linear Regression which produces better results for host overload detection since it is based on the prediction of host utilization which is suitable for the dynamic nature of the cloud. The predicted host utilization in multiple regression depends on multiple factors (CPU, Memory, Bandwidth) so it gives a real indication of host utilization. MRHOD and HLRHOD algorithms outperform the other algorithms that are based on only single factor in term of energy consumption. However MRHOD and HLRHOD give poor results in term of SLAV since energy and SLAV are negatively correlated, it outperforms other algorithms in term of ESV metric by about 6 times compared to other single factor regression algorithms (LR, and LRR). The most effective method for virtual machine consolidation is MRHOD algorithm then HLRHOD algorithm while both are multiple factors host overload detection algorithms. As a future work, we can apply Multiple Regression Host overload Detection algorithm (MRHOD) on real cloud rather than simulation. Studying how to develop a new formula for calculation of host utilization using normalization to achieve better results is another interesting direction for the future work. We plan also to extend the hybrid concept to different management tasks of cloud controllers.

Acknowledgements

The authors send their acknowledge to STDF for their financial grant of the cloud computing center of excellence (number 5220). This research was supported in part by the Distributed and Networked Systems Research Group Operating Grant number 150410, University of Sharjah.

References

[1] Mohan A, Shine. Survey on live vm migration techniques. Int J Adv Res Comput Eng Technol 2013;2:60–74.
[2] Dhiman G, Mihic K, Rosing T. A system for online power prediction in virtualized environments using Gaussian mixture models. In: Proceedings of the 47th ACM/IEEE DAC. p. 807–12.
[3] Abdelsamea A, Hemayed EE, Eldeeb H, Elazhary H. Virtual machine consolidation challenges: a review. Int J Innovation Appl Stud 2014;8:1504–16.
[4] Beloglazov A, Buyya R. Openstack neat: a framework for dynamic and energy-efficient consolidation of virtual machines in openstack clouds. Concurrency Comput: Practice Exp (CCPE) 2015;27:1310–33.
[5] Beloglazov A, Buyya R. Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. IEEE Trans Parallel Distributed Syst (TPDS) 2013;24:1366–79.
[6] 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 Comput: Practice Exp (CCPE) 2012;24:1397–420.
[7] Cleveland W, Loader C. Smoothing by local regression: principles and methods. Stat Theory Comput Aspects Smoothing 1996.
[8] Farahnakian F, Ashraf A, Pahikkala T, Liljeberg P, Plosila J, Porres I, et al. Using ant colony system to consolidate vms for green cloud computing. IEEE Trans Serv Comput 2015;8:187–98.
[9] Ghribi C, Hadji M, Zeghlache D. Energy efficient vm scheduling for cloud data centers: exact allocation and migration algorithms. In: 13th IEEE/ACM international symposium on cluster, cloud and grid computing (CCGrid). p. 671–8.
[10] Zhu F, Li H, Lu J. A service level agreement framework of cloud computing based on the cloud bank model. Comput Sci Automation Eng (CSAE), IEEE 2012;1:255–9.
[11] Zhou Z, Hu Z, Song T, Yu J. A novel virtual machine deployment algorithm with energy efficiency in cloud computing. J Central South Univ 2015;22:94–983.
[12] Monil MAH, Rahman RM. Vm consolidation approach based on heuristics, fuzzy logic, and migration control. J Cloud Comput 2015;5(1):1–18.
[13] Shidik G, Azhari A, Mustofa K. Improvement of energy efficiency at cloud data center based on fuzzy markov normal algorithm vm selection in dynamic vm consolidation. Int Rev Comput Software (IRECOS) 2016;11:511–20.
[14] Terracotta project. Home Page. <http://terracotta.readthedocs.io/en/latest/project_info.html>.
[15] Abdelsamea A, El-Moursy AA, Hemayed EE, Eldeeb H. Multiple regression host overload detection, Sensors to Cloud Architectures Workshop (SCAW2016) held in conjunction with HPCA-22.
[16] Wooldridge MJ. Introductory econometrics: a modern approach. 5th ed. South Western Cengage Learning; 2013.
[17] Bowerman B, O'Connell R, Murphree E, Business statistics in practice, 7th ed.; 2013.
[18] Chen J, Liu W, Song J. Network performance-aware virtual machine migration in data centers. The Third International Conference on Cloud Computing, GRIDs, and Virtualization (CLOUD COMPUTING 2012); 2012. p. 65–78.
[19] Cleveland W. Robust locally weighted regression and smoothing scatterplots. J Am Stat Assoc 1979:829–36.
[20] Rodrigo C, Rajiv R. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. New York, USA: Wiley Press; 2011. p. 23–50.
[21] Beloglazov A, Abawajy J, Buyya R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Generation Comput Syst 2012:755–68.
[22] Anderson M, Braak CT. Permutation tests for multi-factorial analysis of variance. J Stat Comput Simul 2003:85–113.