2013 39th Euromicro Conference Series on Software Engineering and Advanced Applications

LiRCUP: Linear Regression based CPU Usage Prediction Algorithm for Live Migration of Virtual Machines in Data Centers

Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila
Department of Information Technology
University of Turku
Turku, Finland
{ fahfar, pakrli, juplos}@utu.fi

***Abstract*— Virtualization is a vital technology of cloud computing which enables the partition of a physical host into several Virtual Machines (VMs). The number of active hosts can be reduced according to the resources requirements using live migration in order to minimize the power consumption in this technology. However, the Service Level Agreement (SLA) is essential for maintaining reliable quality of service between data centers and their users in the cloud environment. Therefore, reduction of the SLA violation level and power costs are considered as two objectives in this paper. We present a CPU usage prediction method based on the linear regression technique. The proposed approach approximates the short-time future CPU utilization based on the history of usage in each host. It is employed in the live migration process to predict over-loaded and under-loaded hosts. When a host becomes over-loaded, some VMs migrate to other hosts to avoid SLA violation. Moreover, first all VMs migrate from a host while it becomes under-loaded. Then, the host switches to the sleep mode for reducing power consumption. Experimental results on the real workload traces from more than a thousand PlanetLab VMs show that the proposed technique can significantly reduce the energy consumption and SLA violation rates.**

Keywords: cloud computing; virtulization; live migration; dynamic consolidation; regression; green IT

I. INTRODUCTION

As the computational world has become very large and complex, cloud computing is a popular computing model to provide large data centers for external users. Cloud data centers are provided customers in a flexible and service-oriented manner on a pay-as-you-go basis [1]. We consider the cloud architecture with three layers depends on the different types of services offered (Figure 1). Software-as-a-service (Saas) provides standard application software functionality as service on demand. Platform-as-a-service (Paas) offers computational resources via a platform upon which applications and services can be developed. Infrastructure-as-a-service (Iaas) delivers basic infrastructure support services. This layer can use virtualization capabilities for enhancing accessibility of cloud data center infrastructure.

Virtualization [2] technology addresses the efficiency of resource utilization by sharing a physical server (host) among multiple performance-isolated platforms called virtual machines (VMs). Each VM can run multiple application requests on a host simultaneously. The live migration of VMs is an extremely powerful feature of virtualization that is used for load balancing, online infrastructure maintenance and proactive fault tolerance [3][4][5]. Moreover, the VMs live migration dynamically consolidates VMs on the minimal number of active hosts according to their current resource requirements in order to minimize the energy cost [6]. Since there is a trade-off between power consumption and the Quality of Service (QoS), the live migration must consider maximization of the QoS beside the minimization of power consumption. The QoS requirements are formalized via Service Level Agreement (SLA) that describes such characteristics as minimal throughput, maximal response time or latency delivered by the deployed system.

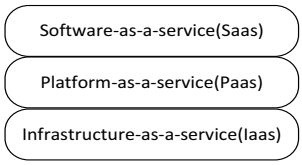

Figure 1. Cloud layered architecture

A live migration approach is presented in [6] to guarantee high-level of SLA and the efficient power management. In this approach, when a host is under-loaded (with 0% CPU utilization), all VMs migrate to other hosts. Then the host switches to a sleep mode for reducing the power costs. Moreover, if the requested CPU demand exceeds the available CPU capacity on a host, the host is over-loaded. Therefore, some VMs should be migrated to other hosts to minimize the SLA violation rate. In general, four challenges addressed in the approach:

1. How to determine a host is over-loaded?
2. How to determine a host is under-loaded?
3. How to determine which VMs should be migrated from an under-loaded or over-loaded host?
4. How to determine which host is selected as the new placement for selected VM?

978-0-7695-5091-6/13 $26.00 © 2013 IEEE
DOI 10.1109/SEAA.2013.23

This paper focuses on the first two critical problems that considerably influence the decision making of other problems. We are interested in the problem of estimating the expected CPU requirements of a host, when being determined an over-loaded or under-loaded host. So, a prediction method is proposed to forecasts the future CPU utilization and named the Linear Regression based CPU Utilization Prediction (LiRCUP). The main objective of LiRCUP is to minimize the power consumption and SLA violation level. Since the CPU parameter consumes the major partition of the host power, the proposed method predicts the CPU utilization of each host based on the linear regression technique. The linear regression is a strong statistical method that used in machine learning schemes to estimate a prediction function. In LiRCUP, the linear regression creates the function according to the past utilization values in a host.

The remainder of this paper is organized as follows. In Section II, the related work is discussed. Some background information on linear regression techniques is given in Section III. In Section IV, the main motivation of the proposed method is described. The proposed algorithm is explained in Section V. The simulation results are reported in Section VI, while the conclusion is given in the last section.

II. Related Work

There is an extensive amount of literature on virtualization technology that relates to dynamic VMs consolidation. An efficient consolidation requires that VMs are migrated commensurate with the current application requests. A dynamic server migration algorithm is implemented in [7] to improve the amount of required capacity and the rate of SLA violations. It can predict variable workloads over intervals shorter than the time scale of demand variability. The VirtualPower architecture [3] utilizes a power management system based on local and global policies. On the local level, the system leverages guest operating system's power management strategies. Global policy applies VMs live migration to reallocate the VMs. The pMapper [8] presents a power-aware application placement controller in virtualized heterogeneous systems. The placement of VMs has been optimized to minimize power consumption and migration cost at each time frame. The PADD [9] uses an adaptive buffering scheme to determine how much reserve capacity is required. It gets the best combination of energy saving and performance for consolidating VMs in a virtualized environment. Experiments in this work show the reduction energy when the number of VMs increase. In [10] VMs are consolidated in the data center taking advantage of min, max and shares features inherent in virtualization technologies. It provides a smooth mechanism for power-performance tradeoffs in cloud data centers. The Sandpiper [11] implements heuristic algorithms to control VMs migration. It determines which VM to migrate from an overloaded server, where to migrate it, and a resource allocation for the virtual machine on the target server.

In some approaches, the VM consolidation has formulated as an optimization problem [11][13][14]. Although an optimization problem is associated with constraints like data center capacity and SLA. Therefore, these works utilize a heuristic method for the multi-dimensional bin packing problem as an algorithm for the workload consolidation. Data centers are bins and VMs are objects, with each data center being one dimension of the size. Algorithms solve this problem of minimizing the number of bins while packing all the objects. So, they achieve the minimum power consumption in data centers.

In [15] a static CPU utilization threshold is utilized to decide when a host is over-loaded. If the utilization exceeds 85% of total capacity, the host is assumed to be over-loaded. Static threshold heuristics do not adapt to workload changes, as they are not suitable for a system with unknown and dynamic workloads. Therefore, more recent works have focused on decision-making based on statistical analysis of historical data. A server over-load forecasting technique based on time-series analysis of historical data is proposed in [7]. In another work [16], a trace-based workload migration controller is employed that uses a combination of both periodic and reactive thresholds. It detects the server load and maintains a balance between supply and demand for capacity while minimizing power usage. In [6] the statistical analysis methods are studied for determining an upper threshold. These methods adjust the value of the upper utilization threshold depending on the strength of the deviation of the CPU utilization. Moreover, their previous work [17] utilizes two static lower and upper thresholds to determine the host status. If the CPU utilization of a host falls below the lower threshold, the host is under-loaded. If the utilization exceeds the upper threshold in a host, the host is over-loaded.

III. Linear Regression

Regression is a popular statistical approach to estimate the relationship between one or more input variables and the output variable. The case of one input variable is called simple regression. More than one input variable is multiple regression. In all cases, regression approximates a function (regression function) that it can be considered as linear or non-linear. The linear regression models the relationship between an input variable x and output variable y by a straight line.

$$y = \beta_0 + \beta_1 x \qquad (1)$$

The parameters β0 and β1 are regression coefficients. A measure of goodness of fit, that is, how well it predicts the output variable y is the magnitude of the residual ε_i at each of the n data points.

$$\varepsilon_i = y_i - \hat{y}_i \qquad (2)$$

ε_i is the difference between the prediction output ($\hat{y}$) and the real output (y) in data point i.

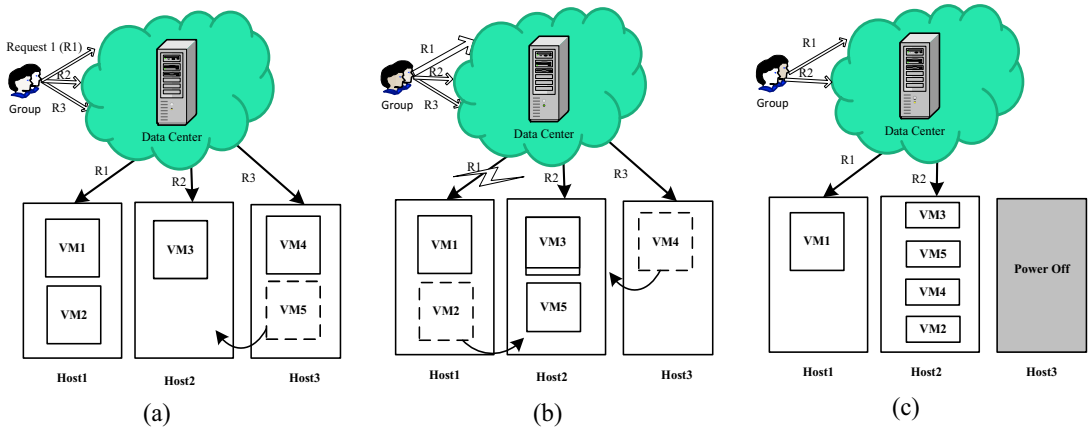

Figure 2. VMs live migration

Ideally, if all the residuals ε_i are zero, one may have found an equation in which all the points lie on the model. Thus, minimization of the residual is an objective for obtaining regression coefficients. The most popular method to minimize the residual is the least squares method [19][21], where the coefficient parameters of the models are chosen such that the sum of the squared residuals over all data points is minimized, that is minimize

$$S(\beta_0, \beta_1) = \sum_{i=1}^{n} {\varepsilon_i}^2 = \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i)^2 \quad (3)$$

Suppose the least-squares estimators of β0 and β1 are $\hat{\beta}_0$ and $\hat{\beta}_1$, must satisfy

$$\frac{\partial S}{\partial \beta_0} = -2 \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i) = 0 \quad (4)$$

$$\frac{\partial S}{\partial \beta_1} = -2 \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i) = 0 \quad (5)$$

Simplifying these two equations yields

$$\hat{\beta}_0 = \frac{1}{n} \sum_{i=1}^{n} y_i - \frac{\hat{\beta}_1}{n} \sum_{i=1}^{n} x_i = \bar{Y} - \hat{\beta}_1 \bar{X} \quad (6)$$

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} y_i x_i - \frac{1}{n} (\sum_{i=1}^{n} y_i)(\sum_{i=1}^{n} x_i)}{\sum_{i=1}^{n} {x_i}^2 - \frac{1}{n} (\sum_{i=1}^{n} x_i)^2}$$

$$= \frac{\sum_{i=1}^{n} (X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n} (X_i - X)^2} \quad (7)$$

where $\bar{X}$ and $\bar{Y}$ are the means of the X_i and Y_i observations, respectively.

IV. MOTIVATION

We consider a scenario in which a large-scale data center as a resource provider consisting of *M* heterogeneous hosts. Several users submit the requests for provisioning of *N* VMs which are allocated *M* hosts. Initially, the VMs are allocated according to the requested characteristics assuming 100% CPU utilization. The VMs experience dynamic workloads, which mean that the CPU usage by a VM arbitrarily varies over time. The power consumption is mostly calculated based on the CPU, memory, disk storage and network interfaces in the data center. Since the CPU consumes the main part of energy, in this study the power measured based on the CPU utilization. Moreover, Recent studies [6][22] [23] shows the CPU utilization has an impact on power consumption, the impact is linear when dynamic voltage and frequency scaling is applied. Therefore, the resource capacities of the host and resource usage by VMs are characterized by a single parameter, the CPU performance. The CPU performance defined in Millions Instructions Per Second (MIPS).

The VMs live migration optimizes the VMs placement in order to reduce the number of hosts with maximum expected benefit. This benefit is the combination of two factors: the rate of SLA violations and power consumption in the resulting reallocation. SLA shows the level by which the QoS requirements negotiated between the data center and costumers. The data center pays a penalty to the client in case of an SLA violation. An SLA violation occurs when a given VM cannot get the amount of MIPS that are requested. This can happen in cases when VMs sharing the same host require a CPU performance that cannot be provided due to the consolidation. The VMs live migration process can be divided four steps [6]:

1) **Over-loading detection*:*** detecting when a host is over-loaded.

2) **Under-loading detection:** detecting when a host is under-loaded.

3) **VM selection:** choosing which VMs should be migrated from the under-loaded or over-loaded host.

4) **VM allocation*:*** selecting a host for migration the selected VMs from the under-loaded or over-loaded host.

We proposed a CPU usage prediction method (LiRCUP) to optimize two first steps in the live migration process. The LiRCUP forecast an over-loaded host in the first step and all VMs can migrate to other hosts before a SLA violation happens. Furthermore, if LiRCUP predicts an under-loaded host in the second step, some VMs migrate to other hosts and it switches to sleep mode as soon as possible. We clearly describe LiRCUP effects on the quality of the live migration process using an example.

There are three hosts and five VMs for running three groups of request (*Ri*) from users. LiRCUP predicts Host 3 will be under-loaded soon and all VMs should be migrated to other host. The VMs selection method decides to migrate VM5 from Host 3. The VMs allocation method selects Host 2 for allocating the VM5 (Figure 2(a)). As illustrated in Figure 2(b), the number of the first group of requests (R1) increase and LiRCUP forecasts Host 1 becomes over-loaded and a SLA violation occurs. So the VMs selection chooses VM2 for migration. Moreover, VM4 on Host 3 move to Host 2, because LiRCUP predicted the Host 3 becomes under-loaded. Figure 2(c) shows the R3 is completed and Host 3 to be inactive in order to eliminate the idle power consumption. In addition, the Host1 is not overloaded because the VM2 moves to Host 2 before a violation happens. Therefore, LiRCUP can reduce the SLA violation rate and power cost.

V. Linear Regression based CPU Utilization Prediction Algorithm

Predicting the future resource demand is critical for efficient resources management in cloud data centers. As the power consumption is calculated based on the CPU utilization in this paper, the proposed method predicts CPU usage in each host.

LiRCUP employs the linear regression to estimate a prediction function according to the history of past CPU utilization over one hour ago. In this section, we first explain how LiRCUP algorithm estimates the expected CPU requirements in a host. Then, we describe how LiRCUP is applied in over-loading detection and under-loading detection steps of the live migration process.

LiRCUP algorithm approximates a prediction function based on the linear regression method. The function shows the linear relationship between the future and current CPU utilization in each host as follows:

$$y = \beta_0 + \beta_1 x \qquad (8)$$

where β_0 and β_1 are regression coefficient parameters that estimate according to the *k* last CPU utilization in a host. The *y* and x are the expected and current utilization values, respectively. The value of *k* is set to 12 in our simulation, because the interval of utilization measurements is five minutes. The history of CPU utilization over one hour ago is significant for forecasting the short-time future utilization. The pseudo-code of LiRCUP algorithm is:

UPLiR

Input: *h*
Output: *predictUtil*

1. // Approximate the prediction function based on *k* past utilization
2. *Initialize the* β_0 *and* β_1 *to a random small value;*
3. ***For*** *i=1 to k* ***do***
4. $x_i \leftarrow$ *UtilHistory (i);*
5. $y_i \leftarrow$ *UtilHistory (i+1);*
6. $\hat{y}_i = \beta_0 + \beta_1 * x_i$;
7. // Calculate the loss
8. $E = \sum_{j=1}^{i} (y_j - \hat{y}_j)^2$;
9. *Update the* β_0 *value using Equation (6) over i samples of k.*
10. *Update the* β_1 *using Equation (7) over i samples of k.*
11. ***end for***
12. // Use the regression function
13. $predictUtil = \beta_0 + \beta_1 * CurrTotalUtil(h)$;
14. ***return*** *predictUtil;*

Initially, the coefficient parameters are set in a random small value (line 2). Then, the prediction function is calculated by updating β_0 and β_1 based on the *K* past utilization in host *h* (line 3-11). The linear regression calculates the difference between the real and the predicted utilization values in each data point. For this purpose, each next utilization value is the real utilization for the previous data point (line 4 and 5). Also, the predicted utilization estimates based on coefficient parameters and the current utilization values (line 6). The sum of the squared residuals over all *i* past data points is measured (line7). Then, β_0 and β_1 values update to minimize the difference between the real and the predicted utilization (line 9 and 10). Finally, the function is used for forecasting the utilization based on the current total utilization of all VMs on the host (line 13).

In addition, LiRCUP is applied in the overloading detection and under-loading detection methods in order to minimize the power consumption and SLA violation. These methods are an essential part of the VM migration process as follows:

VMs Live Migration

1. ***For*** *each host in hostList* ***do***
2. ***while****(OverloadeingDetection (host)=true* ***OR*** *UnderloadeingDetection (host)=true)* ***then***
3. *VM←VMSelection(host);*
4. *otherHosts←hostList-host;*
5. *h←VMsAllocation(otherHosts, VM);*
6. *migrate(VM,h);*
7. ***end while***
8. ***end for***

The *while* loop (line 2 -7) repeats for each over-loaded or under-loaded host. First the VM selection method selects a VM on the over-loaded or under-loaded host for migration (line 3). Then, VM allocation method finds a host to migrate the selected VM by the VM selection method (line 4 and 5). Finally the selected VM migrates to the new place on another host (line 6). The pseudo-code of over-loading detection method is as follows:

Over–Loading Detection

```
Input: host
Output: Boolean
1.  if(UtilHistory.length <12)
2.      if(CurrTotalUtil(host)>0.85*(TotalUtil(host))
3.          return true;
4.      else
5.          return false;
6.      end if
7.  end if
8.  else
9.      predictionUtili←LiRCUP(host);
10.     if(predictionUtil>CurrTotalUtil(host))
11.         return true;
12.     else
13.         return false;
14.     end if
15. end else
```

We assume a host is over-loaded if the current utilization is greater than 85% of the total utilization in first one hour (line1-8). After collecting the history of 12 past utilization, LiRCUP algorithm determines the future CPU usage in a host (line 9). The host becomes over-loaded while the prediction utilization value is larger than the available CPU capacity (line 10 and 11). In contrast, the host is not over-loaded soon (line12-15). Therefore, LiRCUP can avoid a SLA violation by movement of the VMs once a host becomes over-loaded. Moreover, we use LiRCUP in the under-loading detection method for forecasting an under-loaded host**.**

Under_Loading Detection

```
Input: host
Output: Boolean
1.  if(UtilHistory.length <12)
2.      if(CurrTotalUtil(host)=0)
3.          return true;
4.      else
5.          return false;
6.      end if
7.  end if
8.  else
9.      predictionUtil←LiRCUP (host);
10.     if(predictionUtilization<0.1*( TotalUtil(host));
11.         return true;
12.     else
13.         return false;
14.     end if
15. end else
```

If the CPU utilization of a host equals zero, the host is under-loaded (line1-8). After one hour, we predict an under-loaded host with LiRCUP, when the prediction utilization will be less than 10% of total host utilization (line 9-15). In order to minimize the power consumption all VMs on the host should be migrated to other hosts. Then the host switches to the sleep mode after placing all VMs.

We compared LiRCUP method with four algorithms in CloudSim [6] which are implemented the statistical analysis methods to detect an over-loaded host. Therefore, we supposed the same VMs selection and VMs allocation polices in these algorithms. The VMs selection policy selects a VM for migration that requires the minimum migration time than other VMs on the host. So this policy is named Minimum Migration Time (MMT) and the migration time is calculated by dividing the amount of RAM utilized by the VM on the available network bandwidth for the current allocated host. The VMs allocation policy allocates the selected VM to a host that provides the least increase of power consumption and the host is not overloaded after allocation.

VI. Simulation Results and Analysis

To evaluate the efficiency of our approach, implementations have been performed on the CloudSim toolkit [24]. CloudSim is becoming increasingly popular in the cloud computing community due to their support for flexible, scalable, efficient, and repeatable evaluation of provisioning policies for different applications. Our proposed method is compared with four benchmark algorithms that are presented in CloudSim. The first algorithm monitors the CPU utilization and migrates a VM when the current utilization exceeds a static threshold (THR). The threshold sets to 80% for detecting an over-loaded host. The next two algorithms adapt the utilization threshold dynamically based on the median absolute deviation (MAD) and the interquartile range (IQR). The fourth method utilizes Loess method [26], a local regression (LR) approach, to estimate the CPU utilization. We used two metrics to evaluate their performance:

1. Average SLA violation percentage: represents the percentage of average CPU performance that has not been allocated to an application when requested, resulting in performance degradation. It is calculated by Equation (9) as a fraction of the difference between the requested by all VMs ($U_{rj}(t)$) and the actually allocated MIPS ($U_{aj}(t)$) relatively to the total requested MIPS over the life-time of the VMs ,where M is the number of VMs.

Table 1.The power consumption at different load levels in Watts

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

$$SLA = \frac{\sum_{j=1}^{M} \int U_{rj}(t) - U_{aj}(t)dt}{\sum_{j=1}^{M} \int U_{rj}(t)dt} \qquad (9)$$

2. Energy consumption: In order to compare the efficiency of the algorithms, we obtained the results for random and real workloads. In these workloads, a data center comprising 800 heterogeneous hosts is simulated. Half of hosts are HP ProLiant ML110 G4 servers, and the other half consists of HP ProLiant ML110 G5 servers.
Each server has 1 GB/s network bandwidth. The frequency of the servers' CPUs is mapped onto MIPS ratings: 1860 MIPS each core of the HP ProLiant ML110 G5 server and 2660 MIPS each core of the HP ProLiant ML110 G5 server. In [6] utilizes the real data on power consumption instead of using an analytical model. Table 1 illustrates the power consumption characteristics of the selected servers in the simulator.

A. Random workload

The users submit requests for provisioning of 800 heterogeneous VMs that fill the full capacity of the simulated data center. Each VM runs an application with the variable workload, which is modeled to generate the utilization of CPU according to a uniformly distributed random variable. Each application has a length that determines the number of instructions with MI. The application runs for 150,000 MI that is equal to 10 minutes of the execution on 250 MIPS CPU with 100% utilization. Table 2 illustrates the SLA violation levels caused by LiRCUP, THR, MAD, IQR and LR methods in the random workload. LiRCUP minimizes the percentage of SLA violation less than other methods. The obtained results can be explained by the fact that LiRCUP can predict the load in a host before it will overloaded and a SLA violation happens.

Table 2. Average SLA violation percentage in the random workload

LiRCUP	THR	MAD	IQR	LR
10,26 %	12,75%	11,28 %	10,70 %	14,89%

Figure 3 shows LiRCUP consumes less power than other benchmarks algorithms. The proposed method can predict an under-loaded host and moves all VMs to other hosts earlier, for switching it to sleep mode.

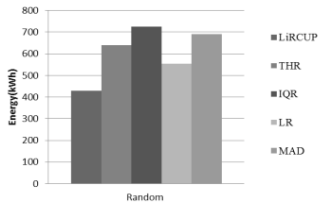

Figure 3. Energy consumption by LiRCUP and benchmark methods in random workload

B. Real workload

Real workload data that is provided as a part of the CoMon project, a monitoring infrastructure for PlanetLab[25]. In this project, the CPU utilization data is obtained from more than a thousand VMs from servers located at more than 500 places around the world. Data is collected every five minutes and is stored in a variety of files. We selected five days from the workload traces collected during April 2011 of the project. During the simulations, each VM is randomly assigned a workload trace from one of the VMs from the corresponding day. Table 3 shows the number of VMs that are considered for each day.

Table 3. The number of VMs in the real workload

Date	Number of VMs
3 April	1463
9 April	1358
11 April	1233
12 April	1054
20 April	1033

Table 4 shows LiRCUP approach reduced the average SLA violations than other four benchmark algorithms due to forecasting the SLA violation.

Table 4. Average SLA violation percentage in the real workload

Date	LiRCUP	THR	MAD	IQR	LR
3 April	7,97 %	10,07 %	10,11 %	10,11 %	10,05 %
9 April	8,34 %	10,25 %	10,16 %	10,04 %	10,16 %
11 April	8,45 %	10,08 %	10,38 %	10,23 %	10,41 %
12 April	8,55 %	10,27 %	10,23 %	10,11 %	10,24 %
20 April	9,94 %	10,75 %	10,56 %	10,39 %	11,28 %

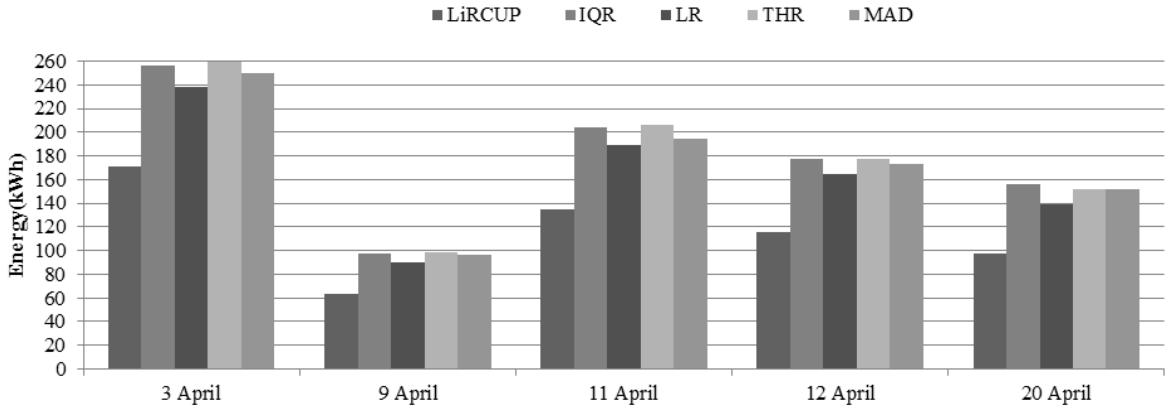

Figure 4. Energy consumption by LiRCUP and benchmark methods in real workload

Moreover, it has the lowest energy consumption as observed from the results in Figure 4. For example, LiRCUP reduces the energy about 51%, 46%, 50% and 49% compared with THR, MAD, IQR and LR on 3 April, respectively. This is because it can predict the under-loaded host and minimize the energy cost by changing the idle mode to sleep mode in the host.

VII. Conclusion

We presented a load prediction method (LiRCUP) for energy-efficient consolidation of virtual machines in cloud data centers. Since CPU consumes the main part of energy, the power is measured based on the CPU utilization in this paper. The proposed method predicted the CPU utilization in each host based on the linear regression technique. This technique used the last utilization over one hour ago and approximated a function. The function can forecast the short-term future utilization based on the current requested utilization in each host. If the prediction utilization is greater than the current utilization capacity, then the host will be over-loaded. Therefore, some VMs should migrate from the almost over-loaded host to maintain the high SLA level. Moreover, LiRCUP is used to predict an under-loaded host and migrate all VMs to other hosts. Due to the host switching to a sleep mode, the power consumption is minimized. The experiments show that LiRCUP method is able to minimize energy cost and SLA violation rate more efficiently than THR, MAD, IQR and LR.

References

[1] BP. Rimal, E. Choi, I. Lumb, "A Taxonomy and, Survey of Cloud Computing Systems", Proceedings of the Fifth International Joint Conference on INC, IMS and IDC, pp.44–51, 2009.

[2] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the Art of Virtualization", Proceedings of the nineteenth ACM symposium on Operating Systems Principles (SOSP'03), pp.164-177, 2003.

[3] R. Nathuji and K. Schwan, "Virtual Power: Coordinated Power Management in Virtualized Enterprise Systems", Proceedings of the 22st ACM Symposium on Operating Systems Principles (SOSP'07),pp. 265–278 , 2007.

[4] H. Liu, H. Jin, X. Liao, L. Hu, and C. Yu, "VMs live migration of Virtual Machine Based on Full System Trace and Replay", Proceedings of the 18th International Symposium on High Performance Distributed Computing (HPDC'09), pp.101-110, 2009.

[5] A. B. Nagarajan, F. Mueller, Ch. Engelmann and S. L. Scott, "Proactive Fault Tolerance for HPC with Xen Virtualization", Proceedings of the 21st ACM International Conference on Supercomputing (ICS'07), pp.23-32, 2007.

[6] A. Beloglazov and R. Buyya, "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 (CCPE), Vol.24, pp.1397-1420, 2012.

[7] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic placement of virtual machines for managing SLA violations", Proceedings of the 10th IFIP/IEEE Intl. Symp. on Integrated Network Management (IM), pp.119–128, 2007.

[8] A. Verma, P. Ahuja, and A. Neogi, "pMapper: power and migration cost aware application placement in virtualized systems", Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware, p.p 243–264, 2008.

[9] M. Y. Lim, F. Rawson, T. K. Bletsch, V. W. Freeh. "PADD: Power-Aware Domain Distribution", Proceedings of the 29th International Conference on Distributed Computing Systems (ICDCS), pp. 239-147, 2009.

[10] M. Cardosa, M. R. Korupolu, A. Singh. "Shares and Utilities based Power Consolidation in Virtualized Server Environments", Proceedings of the 11th IFIP/IEEE international conference on Symposium on Integrated Network Management (IM), pp. 327-334, 2009.

[11] T. Wood, P. J. Shenoy, A. Venkataramani, M. S. Yousif, "Sandpiper: Black-box and gray-box resource management for virtual machines", Journal of Computer Networks, vol. 53, pp. 2923–2938, 2009.

[12] S. Mehta and A. Neogi, "Recon: a tool to recommend dynamic server consolidation in multi-cluster data centers", Proceedings of the NOMS, pp.363-370, 2008.

[13] Y. Ajiro and A. Tanaka, "Improving packing algorithms for server consolidation," Proceedings of the International Conference for the Computer Measurement Group (CMG), pp. 399–407, 2007.

[14] M. Wang, X. Meng, L. Zhang, "Consolidating Virtual Machines with Dynamic Bandwidth Demand in Data Centers", Proceedings of IEEE INFOCOM 2011 MINI-CONFERENCE, pp. 71-75, 2011.

[15] X. Zhu, D. Young, B. J. Watson, Z. Wang, J. Rolia, S. Singhal, B. McKee, C. Hyser et al., "1000 Islands: Integrated capacity and workload management for the next generation data center," Proceedings of the 5th Intl. Conf. on Autonomic Computing (ICAC), pp. 172–181, 2008.

[16] D. Gmach, J. Rolia, L. Cherkasova, and A. Kemper, "Resource pool management: Reactive versus proactive or lets be friends", Journal of Networks, vol. 53, pp. 2905–2922, 2009.

[17] A. Beloglazov, J. Abawajy, R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing", Journal of the Future Generation Computer Systems, pp. 755-768, 2011.

[18] J. O. Kephart, H. Chan, R. Das, D. Levine, G. Tesauro, F. Rawson, and C. Lefurgy, "Coordinating multiple autonomic managers to achieve specified power-performance tradeoffs", Proceedings of the International Conference on Autonomic Computing, pages 24-24, 2007.

[19] R. Das, J. O. Kephart, C. Lefurgy, G. Tesauro, D. W. Levine and H. Chan, "Autonomic multi-agent management of power and performance in data centers," Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems: Industrial Track, pp. 107-114, 2008.

[20] T. Hastie, R. Tibshirani, and J. Friedman, "The Elements of Statistical Learning; Data Mining, Inference and Prediction", Springer, 2001.

[21] G.A.F. Seber, "Linear Regression Analysis", Wiley-Interscience, 2003.

[22] X. Fan, WD. Weber, LA. Barroso, "Power provisioning for a warehouse-sized computer", Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA 2007), pp.13– 23, 2007.

[23] D. Kusic, JO. Kephart, JE. Hanson, N. Kandasamy, G. Jiang, " Power and performance management of virtualized computing environments via lookahead control", In Proceedings of the International Conference on Autonomic Computing (ICAC), pp.3-12, 2008.

[24] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. Rose, R. Buyya, "CloudSim: a toolkit for modeling and simulation of Cloud computing environments and evaluation of resource provisioning algorithms". Journal of Software: Practice and Experience, vol. 41, pp.23–50, 2011.

[25] K.S. Park, V.S. Pai, "CoMon: a mostly-scalable monitoring system for PlanetLab", ACM SIGOPS Operating Systems Review, pp.65-47 , 2006.

[26] WS. Cleveland, "Robust locally weighted regression and smoothing scatterplots", Journal of the American statistical association, pp.829–836, 1979.