Nguyen, T. H.; Francesco, M. Di; Yla-Jaaski, A.

# Virtual Machine Consolidation with Multiple Usage Prediction for Energy-Efficient Cloud Data Centers

*Published in:*
IEEE Transactions on Services Computing

*DOI:*
10.1109/TSC.2017.2648791

Published: 01/01/2020

*Document Version*
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

*Please cite the original version:*
Nguyen, T. H., Francesco, M. D., & Yla-Jaaski, A. (2020). Virtual Machine Consolidation with Multiple Usage Prediction for Energy-Efficient Cloud Data Centers. *IEEE Transactions on Services Computing*, *13*(1), 186-199. Article 7807360. https://doi.org/10.1109/TSC.2017.2648791

# Virtual Machine Consolidation with Multiple Usage Prediction for Energy-Efficient Cloud Data Centers

Nguyen Trung Hieu, Mario Di Francesco, *Member, IEEE,* and Antti Ylä-Jääski, *Member, IEEE*

**Abstract**—Virtual machine consolidation aims at reducing the number of active physical servers in a data center so as to decrease the total power consumption. In this context, most of the existing solutions rely on aggressive virtual machine migration, thus resulting in unnecessary overhead and energy wastage. Besides, virtual machine consolidation should take into account multiple resource types at the same time, since CPU is not the only critical resource in cloud data centers. In fact, also memory and network bandwidth can become a bottleneck, possibly causing violations in the service level agreement. This article presents a virtual machine consolidation algorithm with multiple usage prediction (VMCUP-M) to improve the energy efficiency of cloud data centers. In this context, multiple usage refers to both resource types and the horizon employed to predict future utilization. Our algorithm is executed during the virtual machine consolidation process to estimate the long-term utilization of multiple resource types based on the local history of the considered servers. The joint use of current and predicted resource utilization allows for a reliable characterization of overloaded and underloaded servers, thereby reducing both the load and the power consumption after consolidation. We evaluate our solution through simulations on both synthetic and real-world workloads. The obtained results show that consolidation with multiple usage prediction reduces the number of migrations and the power consumption of the servers while complying with the service level agreement.

**Index Terms**—Virtual machine consolidation, virtual machine migration, multiple resource prediction, cloud computing, data centers.

## 1 Introduction

Commercial cloud Infrastructure-as-a-Service (IaaS) providers, such as Amazon EC2, offer several types of virtual machines (VMs) that differ in their amount of resources based on the pay-as-you-go model [1]. This allows cloud users to run their applications on the most appropriate virtual machine instances and pay for the actual resources that are used [2, 3]. However, the resources supplied by cloud providers can vary over time due to highly dynamic workloads that require resizing, creating and (or) terminating VMs. Furthermore, such resources consist of multiple types (or *dimensions*) including CPU, memory, disk and network bandwidth. As a consequence, if the owners of cloud data centers cannot effectively schedule and reallocate heterogeneous VM instances and resource types, some hosts might become overloaded while other hosts may be underutilized. Eventually, such an unbalanced use of hosts results in unnecessary activation of servers, thus increasing the actual costs [4–6]. Conversely, increasing the workload of some VMs may cause the corresponding physical servers to be overloaded, possibly affecting the quality of service (QoS) experienced by the hosted applications. In fact, the QoS level offered to cloud users needs to fulfill the service level agreement (SLA) of the cloud provider [7, 8].

The live VM migration technology enables the consolidation of VMs, thus allowing cloud providers to reallocate VMs into a few physical servers and switch off unused machines [9–11]. This approach helps improve the resource utilization and allows energy savings while keeping a satisfactory level of QoS [12]. VM migration is closely related to the problem of determining when a server is overloaded (i.e., a *hot spot*) or underloaded (i.e., a *cold spot*), which has been studied in the literature [9, 10, 13]. The main challenge is to decide whether a host is overloaded or underutilized due to the diverse set of user applications and the variability of the VM workloads with time, especially in a cloud data center with thousands of heterogeneous machines. In this context, several VM consolidation schemes have simply taken the current utilization of a single resource (i.e., CPU) into account while deciding whether a physical server is overloaded or underutilized [4, 10]. Other schemes consider the current CPU, memory, storage and (or) network usage, then transform them into a single metric [9, 13]. In any case, as they are purely based on the last observed utilization for decision making, existing solutions may cause unnecessary migrations and eventually increase overheads: the energy for VM migration, the performance degradation of hosted applications, and the extra network communication [14, 15]. Consequently, more efficient schemes are needed to correctly take decisions on VM migration. In other words, hot and cold spots should be reliably determined across multiple resources to limit the frequency of VM migrations.

When a server is overloaded, it is challenging to determine which and how many VMs should be selected for migration to suitable hosts. As migration is expensive, VM selection plays an important role to limit the number of VMs migrations. Additionally, the target physical server should also be correctly selected for placing a VM under migration. For instance, the target host should not be overloaded in both the current and the future period of time after allocating the migrated VM. During the migration process, if there is no active physical server with sufficient resources available, an inactive server is started and the selected VMs are allocated on such a machine. On the other hand, when a host is underutilized, all VMs from such a host are selected for migration if they can be consolidated into other hosts without causing overutilization. Idle servers are then switched to a low-power state to save energy [16, 17]. However, switching the power state of a host from idle

• *T. H. Nguyen, M. Di Francesco and A. Ylä-Jääski are with the Department of Computer Science, Aalto University, Finland.*
*E-mail: {nguyen.hieu, mario.di.francesco, antti.yla-jaaski}@aalto.fi.*

to a low-power state and vice versa consumes additional energy [15]. Besides, migration decisions do not only affect the performance of the hosted applications but also that of the data center as a whole. Therefore, as VM migrations and server switches are essential for power reduction, it is even more important to avoid massive migrations and limit power state switches.

This article presents a VM consolidation algorithm called VMCUP-M that embeds both multiple resource and multiple step usage prediction. In particular, VMCUP-M consolidates VMs according to the usage of multiple resources and a tunable horizon to predict future utilization. This article extends our previous work [18, 19] as follows. We first present an efficient multiple usage prediction (MUP) approach to estimate the long-term utilization of each resource type based on the local history of the considered servers. The joint use of current and predicted utilization allows for a reliable characterization of overloaded and underutilized servers, thus enabling cloud providers to increase their compliance with the SLA. We also propose an efficient algorithm, called VM consolidation with multiple usage prediction (VMCUP-M), for energy-efficient cloud data centers. VMCUP-M considers VMs by using two independent procedures, namely, overloaded server migration and underloaded server migration, which have a polynomial time complexity on the number of the VMs to be allocated in the data center. Through extensive simulations on both synthetic and real-world workloads, we show that the proposed MUP scheme can easily be integrated into existing VM selection and placement algorithms to increase the performance of a data center. Furthermore, we also show that the proposed VMCUP-M algorithm reduces the energy consumption while limiting the number of active servers, VM migrations and power state changes, thus achieving a better compliance with the SLA than the state of the art.

The rest of this article is organized as follows. Section 2 discusses the related work and highlights the related limitations. Section 3 introduces the multiple usage prediction scheme for VM consolidation and evaluates the effectiveness of the proposed prediction approach. Section 4 describes and analyzes the VMCUP-M algorithm. Section 5 presents the experimental setup and compares the results obtained by our proposed scheme with existing solutions for VM consolidation. Finally, Section 6 concludes the article.

## 2 Related Work

In this section, we review relevant approaches proposed in the literature on cloud and distributed computing.

Some works addressed VM consolidation through migration to optimize power consumption [4, 9, 10, 20]. In such cases, static hot and cold thresholds were used to determine whether a host is overloaded or underutilized, respectively. As a consequence, these approaches keep the current (CPU) utilization of a server between the two thresholds. However, setting static thresholds and using the current utilization of a single resource are not effective measures for environments with dynamic workloads, in which the utilization of VMs running on a server continuously changes over multiple resource dimensions. The work in [10] proposed a set of metrics to rank servers by considering an adaptive upper bound based on a statistical analysis of historical CPU data. Even though the used thresholds are not static, these approaches only use the current CPU utilization as the main criterion to decide on VM migrations. Thus, they do not allow for a reliable characterization of overloaded and underloaded servers, eventually resulting in unnecessary migrations and energy wastage. The impact of multiple types of resources on the detection of hot and cold servers was not considered in that work either. The work in [4] presented an energy–aware task consolidation (ETC) technique. In particular, ETC minimizes energy consumption by restricting only single resource (CPU) usage below a specified threshold. The main limitations of this work are that it is restricted to CPU utilization only and does not explicitly support overloaded and underloaded host management mechanisms.

A linear regression-based CPU usage prediction (LiRCUP) was presented in [21] for VM migration. Specifically, future CPU usage is estimated to predict overloaded and underloaded hosts based on historical data of each server. Based on that, some of VMs are migrated to other hosts before a SLA violation occurs. Consequently, such a solution relies on early migration of VMs even when the current resource usage of the considered hosts is still acceptable, thus resulting in unnecessary migrations. Furthermore, LiRCUP considers only a single type of resource and applies aggressive VM migration, while our scheme supports multiple types of resources and allows for a reliable characterization of overloaded and underloaded servers in the long-term future. The work in [22] used two learning algorithms (i.e., neural networks and linear regression) to predict future resource requirements in the cloud with respect to time. Their study showed that models based on neural networks obtain superior prediction accuracy than linear regression. However, the training of neural network models takes significant time which depends on the size of the input as well as the frequency of prediction. Therefore, it is important to determine effective learning algorithms for consolidating VMs in cloud data centers with thousands of heterogeneous machines and diverse resource types. The work in [23] predicted the number of VM requests along with their amounts of CPU and memory resources. Based on these metrics, the proposed framework provides an accurate estimation of the number of needed physical machines, thus reducing energy consumption by putting to sleep unneeded machines. However, while providing a solution to predict the number of VM requests and the number of activated servers are important starting from the VM submission phase, overloaded and underloaded host detection algorithms are even more important to support continuous consolidation of already–placed VMs on the least number of physical servers.

Sandpiper [13] combined three dimensions into a single *volume* metric as the product of CPU, memory and network utilization. The same work [13] also introduced a black-box and gray-box (BG) strategy, based on the volume criterion, for VM consolidation in large data centers. BG sorts overloaded servers based on their volume metric and the VMs in each server based on their volume-to-size ratio (VSR). BG then considers the server with the highest volume first; the VM to be migrated is then the one with the highest VSR. The BG scheme also adopts the volume metric to select target servers, i.e., they are sorted by increasing volume to allocate

**Algorithm 1:** UP($p, d, m$)

Set $\mathbf{X} \leftarrow \mathbf{0}$, $\mathbf{y} \leftarrow \mathbf{0}$, $\boldsymbol{\beta} \leftarrow \mathbf{0}$;
// Training dataset: $\mathbf{X}$ (input) and $\mathbf{y}$ (output)
**for** $t = 0$ *to* $n - m$ **do**
  $\mathrm{X}_{t,0} = 1$;
  **for** $i = 0$ *to* $m$ **do**
    $\mathrm{X}_{t,i+1} = U_i^d(p)$;
  $y_t = U_t^d(p)$; //$y_t \in \mathbf{y}$
// Estimate the regression coefficients $\boldsymbol{\beta}$ with OLS
$\boldsymbol{\beta} \leftarrow (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$;
// Estimate the future resource usage
$U_{t+1}^d(p) = \mathbf{U}^d(p) \cdot \boldsymbol{\beta}$;
**return** $U_{t+1}^d(p)$.

the VMs under migration. The major limitation of the BG scheme is that it does not enable a reliable characterization of overloaded and underloaded servers. Moreover, the BG scheme does not support migrating underloaded servers and is limited to homogeneous physical machines.

Other works have explicitly addressed energy-efficient cloud data centers. The solution in [24] modeled VM placement under network-aware SLA metrics as an integer programming model, then derived approximation algorithms with a low time complexity. The work in [25] proposed scheduling and runtime adaptation mechanisms that are eco-friendly by explicitly taking into account the carbon footprint associated with operating a cloud data center. The approach in [26] explicitly considered SLAs for allocating physical resources (i.e., CPU cores) to VMs, with focus on resource isolation. All these solutions, however, do not consider VM machine migration as a method to increase energy efficiency of data centers. The work in [27], instead, targeted distributed load management through software agents and live VM migration. However, the proposed solution was not evaluated in terms of impact on SLAs.

## 3 Multiple Usage Prediction

In this section, we first formally define the VM consolidation problem. We then present the proposed multiple usage prediction (MUP) scheme and its performance for forecasting future resource utilization.

### 3.1 Notations and Metrics

Let us consider a cloud data center that provides computing resources in the form of VM instances according to the Infrastructure as a Service (IaaS) delivery model. We specifically consider the VM consolidation problem as consisting of the following phases: determining when a server is overloaded, then migrating the potential VMs from such a server to maintain a certain QoS; determining when a server is underloaded, then migrating all VMs from such a server to minimize energy consumption; selecting the potential VM that should be migrated from an overloaded server to limit the number of VMs migrations; finally, finding a new placement for the VMs under migration.

We denote a set of $M$ heterogeneous physical servers in a cloud data center as $P = \langle p_1, p_2, \ldots, p_M \rangle$. Each server is uniquely identified in the form $p = \langle pi, vm, \hat{r}^d \rangle$ with multiple types of resources, $d \in \{1, ..., D\}, D \in \mathbb{N}$, where: *pi* is the unique identifier of a server; *vm* is a set of VM instances that are allocated to *p*; and $\hat{r}^d = \{\hat{r}^1, \hat{r}^2, ..., \hat{r}^D\}$ describes the type and the amount of the *d*-th resource consumed, where each dimension corresponds to one type of physical resource (e.g., CPU, memory, storage and network bandwidth).

TABLE 1. Multiple resource and multiple step usage prediction (UP($p, d, m + k - 1$), $m = 1$, $p \in P$, $d \in D$ and $k \in K$).

| Step ($k$) | Inputs of MUP | Output | Usage prediction |
|---|---|---|---|
| 1 | $U_t^d(p)$ | $U_{t+1}^d(p)$ | UP($p, d, 1$) |
| 2 | $U_t^d(p), U_{t+1}^d(p)$ | $U_{t+2}^d(p)$ | UP($p, d, 2$) |
| 3 | $U_t^d(p), U_{t+1}^d(p), U_{t+2}^d(p)$ | $U_{t+3}^d(p)$ | UP($p, d, 3$) |

A VM instance can be represented similarly to the resource dimensions of physical servers. In detail, we use an instance in the form $v = \langle vi, r^d \rangle$ to uniquely identify a VM. To this end, we denote a set of $N$ VMs to be allocated to the system as $V = \langle v_1, v_2, \ldots, v_N \rangle$.

Let $U_t^d(p)$ be the utilization of resource $d \in \{1, ..., D\}$ of a server $p$ at time $t$. Then $U_t^d(p)$ of type $d$ is defined as the total resource usage of all running VMs in $p$ divided by the total resource capacity of the considered server

$$U_t^d(p) = \frac{u_t^d(p) + w^d(p)}{\hat{r}^d(p)}, \tag{1}$$

where $u_t^d(p)$ is the total resource usage of the *d*-th dimension of an already placed set *vm* of VMs that are allocated to *p* at time *t*, $u_t^d(p) = \sum_{v \in vm(p)} r^d(v)$; $w^d(p)$ is the initial load of the *d*-th resource of *p*, and $\hat{r}^d(p)$ is the amount of the *d*-th resource consumed[1] by the considered server *p*.

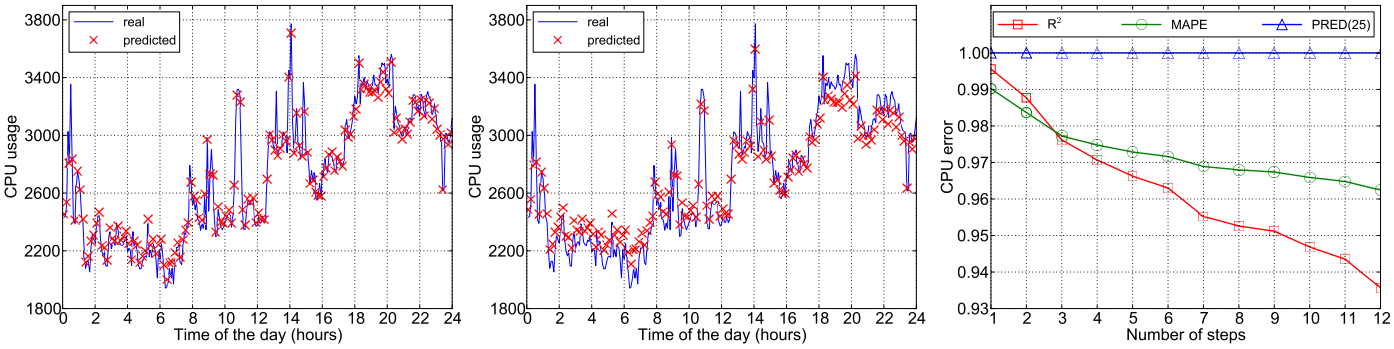
### 3.2 Usage Prediction

Let us assume that the last $n$ observed utilizations for resource type *d* of a server *p* – namely $U_{(t+1)-n}^d(p), ..., U_t^d(p)$ – are known. The goal of prediction is to estimate the future resource usage $U_{t+1}^d(p)$. To this regard, we used multiple linear regression to estimate the relationship between the input variables and the output [28]. Such a method is especially attractive for consolidating VMs in cloud data centers with millions of heterogeneous machines and resource types due to its time complexity, especially if compared to other solutions in the state of the art [22]. In our usage prediction scheme, UP (Algorithm 1), the predicted resource usage $U_{t+1}^d(p)$ of a server *p* is defined as the linear prediction function according to *m* independent utilizations, i.e., $\mathbf{U}^d(p) = [1, U_{(t+1)-m}^d(p), \ldots, U_t^d(p)]^T$, $m < n$, by a straight line as follows:

$$U_{t+1}^d(p) = \beta_0 + \sum_{i=1}^{m} \beta_i \cdot U_i^d(p), \tag{2}$$

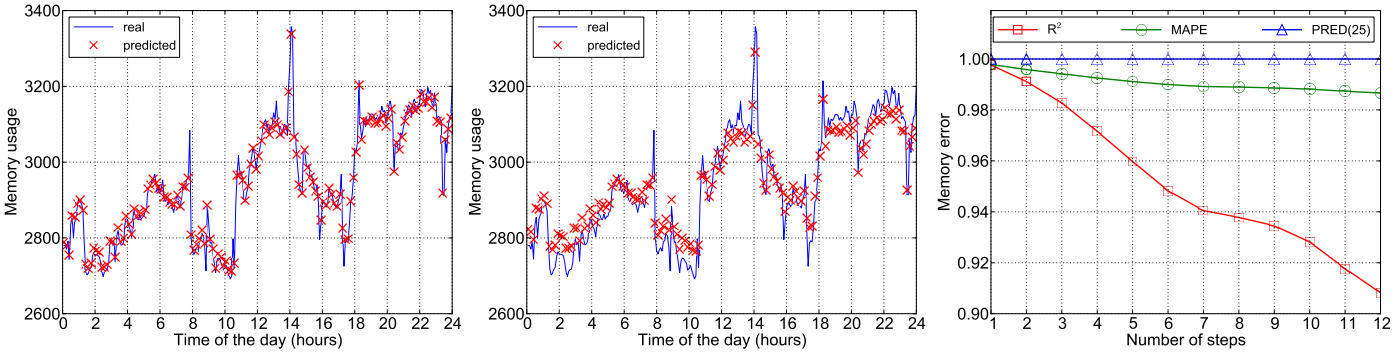
where $\beta_i, i \in \{0, \ldots, m\}$ are the regression coefficients estimated according to the $n$ last observations and $m + 1$ is the regressor size of the prediction model. The regression model in Eq. (2) is obtained by determining the coefficient parameters $\boldsymbol{\beta} = [\beta_0, \ldots, \beta_m]^T$ so that the regression line has

1. For instance, *d*=1 refers to CPU and $\hat{r}^1(p)$=0.8 indicates the amount of the CPU consumed (i.e., 80% of CPU utilization).




(a) (b) (c)

Fig. 1. Prediction of CPU resource usage in the Google cluster traces: (a) one-step prediction; (b) six-step prediction and (c) impact of the number of steps on usage prediction.




(a) (b) (c)

Fig. 2. Prediction of memory resource usage in the Google cluster traces: (a) one-step prediction; (b) six-step prediction and (c) impact of the number of steps on usage prediction.

the best fit for the training data. The original least squares (OLS) [28] is a popular method that estimates the $(m+1)$-dimensional vector of $\boldsymbol{\beta}$ as follows:

$$\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}, \tag{3}$$

where $\mathbf{X}$ is the $(n-m) \times (m+1)$ matrix of input variables and $\mathbf{y}$ is the $(n-m) \times 1$ vector of output variables, respectively. In Eq. (3), $\mathbf{X}^T$ is the transpose of $\mathbf{X}$.

### 3.3 Multiple Usage Prediction

In our multiple usage prediction scheme (MUP) we aim at predicting the long-term usage of multiple resource types $d \in \{1, ..., D\}$ of a server $p$ over a time period $K \in \mathbb{N}$. This requires predicting the usage of the resource type $d$ at the $k$-th step ahead in time, where $k \in \{1, \ldots, K\}$. In other words, we have to estimate $U^d_{t+k}(p)$ from the current resource utilization $U^d_t(p)$. Specifically, we predict the multiple resource and multiple step usage, i.e., $U^d_{t+1}(p), ..., U^d_{t+k}(p)$, by iterating the usage prediction UP corresponding to the regressor size $(m+k-1)$, as follows:

$$\begin{aligned}
U^d_{t+1}(p) &= f_1(U^d_{(t+1)-m}(p), ..., U^d_t(p)) = \mathrm{UP}(p, d, m),\\
U^d_{t+2}(p) &= f_2(U^d_{(t+1)-m}(p), ..., U^d_t(p), U^d_{t+1}(p))\\
&= \mathrm{UP}(p, d, m+1),\\
U^d_{t+k}(p) &= f_k(U^d_{(t+1)-m}(p), ..., U^d_t(p), U^d_{t+1}(p), ..., U^d_{(t+k)-1}(p))\\
&= \mathrm{UP}(p, d, m+k-1).
\end{aligned}$$

Table 1 illustrates how the usage prediction (UP) scheme is iterated to obtain the parameters in the multiple usage prediction (MUP) algorithm. Note that the predicted output at each step is fed back as an input to the next prediction step. This helps minimize the least square error while updating the coefficient parameters at each prediction step.

### 3.4 Performance of Multiple Usage Prediction

In the following, we first evaluate the performance of our proposed MUP prediction scheme by using the real workload traces in the Google Cluster Data dataset [29], consisting of approximately 12,000 machines. In detail, the Google trace provides the resource usage about CPU, memory and disk for each task. The related values are collected every five minutes over a time span of 29 days (May 2011). Based on that, we have derived the total resource usage over a duration of 24 hours by summing up the CPU and memory usage of all the running tasks in the system. This resource utilization reported by the Google cluster indicates the actual resource consumption of CPU and memory. In the considered dataset, both the CPU and memory resource utilization increase linearly with time.

We applied the Leave-One-Out cross-validation technique [30] for estimating the accuracy of the MUP scheme. In such a strategy, each observation $[\mathbf{x}(i), \mathbf{y}(i)], i \in \{1, \ldots, n\}$ in the sample dataset of size $n$ is successively taken out and the remaining $n-1$ observations of the

set are used to train the prediction model to estimate the predicted resource usage $\hat{y}(i)$. The actual output $y(i) \in \mathbf{y}$ is then used to validate the predicted output $\hat{y}(i)$ inferred by the fitted model. We evaluated the accuracy of the MUP scheme under the following metrics (the higher the better): the $R^2$ prediction accuracy, the mean absolute percentage error (MAPE), and the $PRED(25)$ measure [22, 31].

Figure 1 and Figure 2 show the performance of the proposed prediction approach for CPU and memory compared to the real resource usage. The results show that the values predicted by MUP are always close to the real ones even during peaks. Specifically, the results show that the MUP prediction scheme obtains the best performance for a one-step CPU and memory usage prediction (Figure 1a and Figure 2a) and then gets worse as the number of steps increases, i.e., when the number of steps is equal to six (Figure 1b and Figure 2b). This happens since the distribution of the actual resource utilization in the considered dataset is linear over time. Figure 1c and Figure 2c compare the responsiveness of the prediction accuracy for different values of the steps ahead, up to $k = 12$ (corresponding to one hour). According to the $R^2$ prediction accuracy, the measured values decrease from 0.995 to 0.935 for CPU and from 0.997 to 0.908 for memory as the number of steps $k \in \{1, ..., 12\}$. In particular, we found that $R^2$ = 0.963 for CPU and $R^2$ = 0.948 for memory usage are achieved with setting $K$ = 6 corresponding to a future period of 30 minutes. Additionally, the $MAPE$ values also decrease with increasing the number of steps $k$ for both the CPU and memory resource types. Besides, the $PRED(25)$ has correct prediction rate of one for all the prediction steps, thus indicating that the proposed MUP is a perfect fit. In the rest of this article, we set the number of prediction steps to $K$ = 6. This ensures that the $R^2$ prediction accuracy is larger than 0.948 for both the CPU and memory resource types.

## 4 VM Consolidation with MUP

In this section, we present our VM consolidation with multiple usage prediction (VMCUP-M) algorithm to reduce the energy consumption of a cloud data center. The VMCUP-M algorithm itself makes use of two major functions: overloaded host detection with multiple usage prediction (OHD-MUP) and underloaded host detection with multiple usage prediction (UHD-MUP). In the following, we first describe these functions then analyze the time complexity of VMCUP-M.

### 4.1 Overutilized Host Detection

The proposed overloaded server detection OHD-MUP is presented in Algorithm 2 based on the prediction scheme introduced in Section 3.3. Accordingly, a server is considered *overloaded* in any resources $d \in \{1, ..., D\}$ if the following conditions are satisfied ($h$ is referred to as a *hot threshold*, $k \in \{1, ..., K\}$):

- the server is overloaded in both the current and the future period of time, i.e., $U_t^d(p) > h^d$ and $U_{t+k}^d(p) > h^d$;
- the server is currently operating normally and is overloaded in the future period of time, i.e., $U_t^d(p) \leq h^d$ and $U_{t+k}^d(p) > h^d$.

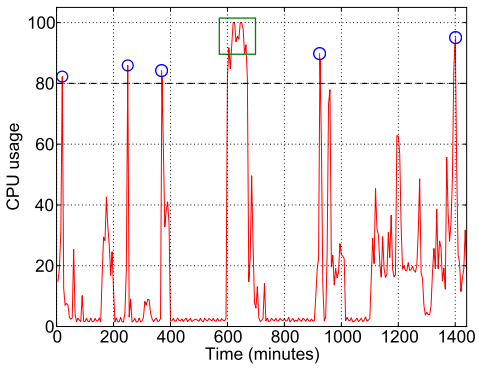



Fig. 3. CPU resource usage of a cloud server in our university measured every five minutes over 24 hours.

This indicates that the server is a potential candidate for migration because it is overloaded in both the current and the future time period or overloaded in the near future. Thus, migrating the VMs from this server increases the compliance with the SLA.

As an example, Figure 3 shows the CPU utilization of a cloud server in our university measured every five minutes over 24 hours. The figure illustrates how existing algorithms based on the current CPU utilization as the main criterion to detect an overloaded server may cause an unreliable overloaded host detection. Specifically, we have considered the following criterion: a host is considered overloaded when the current CPU usage exceeds the threshold $h^{cpu} = 0.8$ (i.e., $80\%$ of load). In Figure 3, the small circles on the top of the trace denote false hot detection points because the load of the considered host will rapidly decrease in the short-term future. In such a situation, the VMs allocated to a server do not need to be migrated to reduce the resource load. For the same trace, the OHD-MUP algorithm reported only one point as a hot spot, i.e., the one marked with the rectangle in the period of time from 600 to 670 minutes. In this period, some VMs need to be migrated to avoid SLA violations. The example shows how MUP plays an important role in decision making on overloaded servers and how OHD-MUP avoids unnecessary VM migrations due to varying resource demands. In a cloud data center with thousands of machines and a high variability of the VM workloads with time, determining VM migration simply based on a static threshold may result in hundreds of hot spots or even worse. Instead, our proposed OHD-MUP leverages the current and multiple predicted utilization so as to limit the number of hot spots to the few ones that are really necessary.

### 4.2 Underutilized Host Detection

Algorithm 3 (UHD-MUP) describes the underloaded server detection with multiple usage prediction. When no host is overutilized, the one with the lowest value of maximum resource utilization in the data center is considered. We define such a server $p$ as *underutilized* if its multiple predicted resource is equal or below the current resource utilization. This indicates that the server is currently underutilized and its load will decrease in the considered time period, thus the host is a potential candidate to be switched to a low-power

**Algorithm 2:** OHD-MUP($p, D, m, K$)

```
for ∀d in D do
    for ∀k in K do
        U^d_{t+k}(p) ← UP(p, d, m + k − 1);
        if U^d_{t+k}(p) ≤ h^d then return false ;
return true
```

**Algorithm 3:** UHD-MUP($p, D, m, K$)

```
for ∀d in D do
    for ∀k in K do
        U^d_{t+k}(p) ← UP(p, d, m + k − 1);
        if U^d_{t+k}(p) > U^d_t(p) then return false ;
return true
```

mode for energy saving. The joint use of both current and predicted utilization metrics (over diverse resources and multiple steps) allows to correctly identify underutilized servers. If the load of the considered server will increase above the current usage in any time instant during the considered period, the algorithm takes no action. After migrating all VMs on underloaded server, the idle server is switched to a low-power mode.

It is worth emphasizing that we do not employ any static cold thresholds for underutilized host detection. Instead, our UHD-MUP algorithm considers as potential candidates for migration, at each evaluation step, all the VMs from the server with the lowest value of maximum resource utilization. However, migration is only carried out if there is a target server with enough resources to accommodate the migration of the candidate VMs. This allows for additional flexibility as no cold threshold needs to be explicitly defined.

### 4.3 Virtual Machine Selection and Placement

Once an overloaded server $p$ is considered, the most utilized resource of a server $p$ is the critical resource which drives the consolidation process, because it becomes a bottleneck for the server. To this regard, the next step is to select a potential VM running on $p$ for migration to reduce the resource load. We thus define the type (or dimension) of the hottest resource $\hat{d} \in \{1, \ldots, D\}$ and the resource temperature ratio $RT(p)$ of an overutilized server $p$ at time $t$ as:

$$\hat{d} = \underset{d \in \{1,\ldots,D\}}{\operatorname{argmax}} \; U_t^d(p), \tag{4}$$

$$RT(p) = (U_t^{\hat{d}}(p) - h^{\hat{d}})^2. \tag{5}$$

We then introduce a VM selection policy – namely, the minimum resource temperature (MRT) – that migrates a VM $v$ to reduce the resource temperature of a given server $p$ the most. As migration is expensive, our goal is to select only the VMs that contribute the most to the host load. Let $vm(p)$ be a set of VMs currently allocated to the host $p$. The MRT policy finds a VM $v \in vm(p)$ such that $\forall a \in vm(p)$ the following conditions hold:

$$\frac{RT(p|vm(p) \setminus v)}{RAM_u(v)} \geq \frac{RT(p|vm(p) \setminus a)}{RAM_u(a)}. \tag{6}$$

**Algorithm 4:** PABFD-MUP($P, v, D, m, K$)

```
Set p ← ∅; minPower ← MAX;
for ∀p_i in P do
    if ∀d ∈ D; r^d(v) + u^d_t(p_i) + w^d(p_i) ≤ r̂^d(p_i) then
        oldPower ← getPower(p_i);
        Place v on p_i, update U^d_t(p_i);
        newPower ← getPower(p_i);
        incPower ← newPower − oldPower;
        if incPower < minPower AND
          OHD-MUP(p_i, D, m, K) = false then
            minPower ← incPower; p ← p_i;
        else Release v from p_i, update U^d_t(p_i);
return p
```

where $RAM_u(v)$ is the amount of memory currently utilized by the VM $v$ and is updated at each consolidation interval. Recall that the cost of migrating a VM is mostly determined by its memory size. Eq. (6) ensures that the memory used by the selected VM is small enough to effectively limit the migration overhead [9, 13]. Furthermore, the VM $v$ selected by Eq. (6) is the most appropriate to reduce the resource temperature, i.e., $RT(p|vm(p) \setminus v)$, with respect to other VMs allocated to a given server $p$, thus reducing the number of migrations.

We also extend the power-aware best fit decreasing (PABFD) algorithm introduced in [10] for multiple resource VM placement, with focus on energy efficiency. In particular, we embed the MUP scheme into PABFD through a new VM placement algorithm called PABFD-MUP (Algorithm 4). Accordingly, PABFD-MUP selects a target physical server based not only on the least increased power consumption but also on its utilization stability, which can be predicted by using the MUP scheme. Importantly, PABFD-MUP decreases the chance of the target host being overloaded in the future period of time after placing the migrating VM.

### 4.4 The VMCUP-M Algorithm

VMCUP-M is detailed by Algorithm 5. It executes periodically to evaluate the VM consolidation process based on the future prediction resource usage of the considered servers.

The overloaded server migration procedure takes place until all servers in $P$ have been considered (line 2). If a sever $p_i$ is overloaded (line 3), the next step is to select the VMs to be migrated. In our VMCUP-M algorithm, the VM $v$ is selected according to the minimum resource temperature (MRT) policy (line 4). After selecting the VM $v$ to be migrated, the appropriate server for placing the migrating VM $v$ is obtained by the power aware best fit decreasing with multiple usage prediction (PABFD-MUP) algorithm (line 5). If server $p$ exists, then $v$ is placed to $p$ and the server $p$ is updated with new values for $U_t^d$ (line 6). Otherwise, if all the servers are not already active, an inactive server $p_{inact}$ is switched to an active state for allocating the selected VM (lines 7-10). The VM placement should be rejected if a server $p$ does not satisfy the demands of all the resources in $v$ or is overutilized after accepting $v$ in the current and future period of time.

**Algorithm 5:** VMCUP-M($P, D, m, K$)

```
1  // Overloaded server migration with MUP;
2  for ∀p_i in P do
3      while OHD-MUP(p_i, D, m, K) = true do
4          v ← MRT(p_i);
5          p ← PABFD-MUP(P \ {p_i}, v, D, m, K);
6          if p ≠ ∅ then Place v on p, update U_t^d(p) ;
7          else if ∃p_inact then
8              Switch p_inact to an idle mode;
9              Place v on p_inact, update U_t^d(p_inact);
10             P ← P ∪ {p_inact};
11         else break;
12 // Underloaded server migration with MUP;
13 Set status ← true;
14 while status = true do
15     Set p ← p_0;
16     for ∀p_i in P do
17         if max_{d∈D} U_t^d(p) > max_{d∈D} U_t^d(p_i) then
18             p ← p_i;
19     if UHD-MUP(p, D, m, K) = true then
20         Set W ← ∅;
21         for ∀v_i in p do
22             s ← PABFD-MUP(P \ {p}, v_i, D, m, K);
23             if s = ∅ then status ← false; break;
24             else W ← W ∪ {s};
25         if status = true then
26             for ∀v_i in p do
27                 Remove server s from W in FIFO order;
28                 Place v_i on s, update U_t^d(s);
29             Switch p to a low-power mode;
               P ← P \ {p};
```

The underloaded server migration procedure starts when no host is overutilized and the host with the lowest value of maximum resource utilization is considered (lines 15-18). Recall that our model defines a server as cold when its resource usage over a given time period is equal or below the current utilization. If a server $p$ is underutilized, all VMs placed in $p$ need to be migrated before switching $p$ to a low-power mode. For each VM $v_i$ in $p$, if a set of potential servers $W$ is found by the PABFD-MUP algorithm, then all placed VMs in $p$ are migrated in sequence to a physical server $s$ in $W$ (lines 21-28). To this end, the cold server $p$ is switched to a low-power state. If at least one VM in a cold spot $p$ cannot find a new placement, the underloaded server migration procedure does not migrate the VMs and $p$ is kept active. The underloaded server migration procedure continues until a server with the lowest utilization has not been considered as a cold spot.

### 4.5 Complexity Analysis

In this section, we analyze the time complexity of VMCUP-M that consists of the two independent phases in Algorithm 5. Let us define: $M$ as the number of active servers in the system; $N$ as the number of VMs to be allocated in the data center; $N_{vm}$ as the average number of VMs hosted on a physical machine; and $n$ as the history size of each resource utilization. We then detail our analysis as follows.

#### 4.5.1 Complexity of Overloaded Server Migration

We calculate the time complexity of the for loop in Algorithm 5. At each step, exactly one server is considered and $M$ is decreased by one. After $M$ iterations, $P$ will be empty. Therefore, the time complexity of the for loop is the same as the number of hosts $M$.

We then calculate the time complexity inside the for loop that consists of three phases. In the first phase, the time complexity of the OHD-MUP algorithm is mainly based on the complexity of the UP algorithm, i.e., it takes $O(D{\cdot}n{\cdot}(m+K))$ time to check if a server is overutilized. In the second phase (i.e., the MRT function), a total of $O(N_{vm})$ time is needed to find an appropriate candidate. The third phase (i.e., the PABFD-MUP function) takes a total of $O(M \cdot D \cdot n \cdot (m+K))$ time. Thus, the time complexity inside the for loop is $O(D{\cdot}n{\cdot}(m+K){\cdot}N_{vm}){+}O(D^2{\cdot}n^2{\cdot}(m+K)^2{\cdot}M)$. Clearly, in the multiple resource model, the number of resources $D$ that need to be considered is usually a small constant when adopting the $D$-dimensional usage prediction (e.g., 2, 3 or 4). Furthermore, $n$, $m$ and $K$ are typically small numbers. Therefore, the time complexity becomes $O(N_{vm}) + O(M)$.

By combining the complexity of the for loop and of its inner statements together, the time complexity becomes $O(M \cdot N_{vm}) + O(M^2)$. Indeed, the total number of active physical machines in the data center can be approximated as $M \approx \frac{N}{N_{vm}}$. Hence, the complexity of overloaded server migration procedure is $O(N^2)$ in the worst-case scenario, in which each VM is allocated on exactly one server.

#### 4.5.2 Complexity of Underloaded Server Migration

A total of $O(M \cdot D)$ time is needed to find a server $p$ with the lowest value of maximum resource utilization. UHD-MUP then takes $O(D \cdot n \cdot (m + K))$ time to check if a server $p$ is underutilized. For each VM in a considered cold server $p$, it takes a total of $O(M \cdot D \cdot n \cdot (m + K))$ time to find an appropriate candidate server, since there are $N_{vm}$ placed VMs to be considered for migration. Thus, the time complexity becomes $O(N_{vm} \cdot M \cdot D \cdot n \cdot (m + K))$, hence, $O(M \cdot D) + O(D^2 \cdot n^2 \cdot (m + K)^2 \cdot N_{vm} \cdot M)$.

Again, $M \approx \frac{N}{N_{vm}}$ and in the worst-case scenario all servers in a data center are underutilized; therefore, the total time complexity of underloaded server migration procedure is $O(N^2)$.

#### 4.5.3 Complexity of the VMCUP-M Algorithm

In the VMCUP-M algorithm, the overloaded server migration and underloaded server migration procedures take place independently from each other. Therefore, the overall time complexity of VMCUP-M is $O(N^2)$.

## 5 Performance Evaluation

To evaluate the effectiveness of our proposed scheme in a practical cloud scenario, we used the CloudSim simulation toolkit [32] and the same experimental setup as [10] for comparison purposes. We extended CloudSim to handle multiple types of resources. We then implemented VMCUP-M on top of such an extended version of CloudSim.

### 5.1 Experimental Setup

Before proceeding further, let us introduce the experimental setup used in the rest of this article. We considered a data center comprising 800 heterogeneous hosts: half of the hosts are HP ProLiant ML110 G4 servers with 1,860 MIPS per core, and the other half are HP ProLiant ML110 G5 servers with 2,660 MIPS per core. Each server has 2 cores, 4 GB of memory and 1 GB/s of network bandwidth. The power consumption of active[2] servers in the simulation is derived from the corresponding figures in the Standard Performance Evaluation Corporation (SPEC) [33] power benchmark results as a function of the utilization level, similarly to [10].

We considered four different types of VMs whose CPU and memory correspond to the following Amazon EC2 [34] instances: High-CPU Medium Instance (2,500 MIPS, 0.87 GB); Extra Large Instance (2,000 MIPS, 1.74 GB); Small Instance (1,000 MIPS, 1.74 GB); and Micro Instance (500 MIPS, 613 MB). Initially, VMs are allocated according to the resource requirements defined by the VMs.

The utilization of the VMs follows the traces from two real-world workloads. The first publicly available workload consists of the resource usage from the Google Cluster Data (GCD) dataset which provides traces over a 29 days period in May 2011 [29]. The GCD workload comprises 672,003 jobs, each with one or more tasks, and contains the normalized value of the average number of used cores and the utilized memory. To create the CPU and the memory utilization of VMs, the tasks of each job were aggregated by summing their CPU and memory consumption every five minutes in a period of 24 hours. We then extracted the VM workloads over the first ten days period by filtering the utilization of CPU and memory from 5% to 90%, resulting in a total of 1,600 VMs. The second real-world workload we considered consists of the CPU and the memory utilization of 11,746 PlanetLab VMs measured every five minutes [10]. In particular, we extracted the CPU and memory usage of VMs from ten days of the PlanetLab VMs workload traces collected in March and April 2011. We then filtered the utilization of CPU and memory from 2% to 100%, resulting in a total of 1,473 VMs. The characteristics of the VMs and their resource utilization in the GCD and PlanetLab traces are presented in Table 2.

In the synthetic workload, we assumed that the CPU and the memory utilization of the VMs followed a uniform distribution as in [10] for comparison purposes. Moreover, we generated the CPU and memory utilization of 800 VMs, each allocated to exactly one server. According to the resource dimensions of the considered[3] workloads, we set $D$ = 2 and adopted a bi-dimensional VM consolidation with prediction. At the beginning of the simulation, VMs were randomly assigned a workload trace from one of the VMs in the traces.

### 5.2 Evaluation Metrics and Comparison Benchmarks

We evaluated VMCUP-M over a time span of 24 hours for the all workload traces in our experiments. We set the initial number of input variables to $m$ = 1 according to [22] and the number of steps to $K$ = 6 according to our preliminary investigation detailed in Section 3.4. We measured the resource usage every five minutes. After one hour from the beginning of the simulation (i.e., at $n$ = 12), the VMCUP-M consolidation process started and it was applied to every active host in the cloud data center every five minutes. We then evaluated the following metrics:

- energy, as the total consumption of all the physical machines in a data center during their lifetime;
- number of active servers;
- number of migrations per VM;
- number of power state changes per server;
- SLA compliance, in terms of the average number of SLA violations[4] (the lower the better).

We compared our proposed method with the static and dynamic hot thresholds for overutilized host detection approaches in [10]:

- Static threshold (THR): the hot CPU and memory threshold is set to 80% of load, i.e., $h^{cpu}$ = 0.8 and $h^{mem}$ = 0.8. The algorithm considers a host as overloaded if the current utilization of any resource exceeds the corresponding hot threshold.
- A dynamic threshold based on local regression (LR). The algorithm detects overloaded hosts based on the estimated CPU and memory utilization thresholds.

We performed VM consolidation with two different strategies: one with MUP and the other without MUP. We compared our results to the approaches proposed in [10]: algorithms based on static (THR) and dynamic (LR) hot threshold for overutilized host detection; the minimum migration time (MMT), maximum correlation (MC), minimum utilization (MU) and random selection (RS) schemes for VM selection; and the power-aware best fit decreasing (PABFD) algorithm for VM placement. Furthermore, we compared VMCUP-M to the multiple resource black-box and gray-box (BG) scheme introduced in [13]. For a fair comparison, we extended the volume metric originally defined in [13] to consider two resources, i.e., CPU and memory. We also adopted the underloaded server migration with and without MUP in the considered BG algorithm.

TABLE 2. Characteristics of the considered workload traces.

| Workload | Date | VMs | Res. | Mean (%) | St. dev (%) | Median |
|---|---|---|---|---|---|---|
| GCD | 10 days May 2011 | 1,600 | CPU | 21.84 | 13.62 | 18 |
| | | | Mem | 19.55 | 16.66 | 12 |
| PlanetLab | 10 days Mar. - Apr. 2011 | 1,473 | CPU | 19.77 | 14.55 | 15 |
| | | | Mem | 6.27 | 6.01 | 5 |

### 5.3 Impact of Multiple Usage Prediction

We first evaluate the impact of MUP on the average number of hot spots, cold spots and active physical machines per data center by varying the data center size between 200 and 1,400 hosts. The ratio of VMs to physical servers is 1:1 in the initial layout. The MRT VM selection policy and the PABFD-MUP VM placement algorithm are used while consolidating VMs. In the plots, we reported the average

2. The power consumption of inactive servers is assumed negligible.

3. We would like to recall that our proposed MUP scheme also supports additional resource types, such as disk and network bandwidth.

4. Specifically, we consider SLA violations due to both overutilization (i.e., the percentage of time during which active servers have experienced 100% utilization of any resource) and migration (i.e, the overall performance degradation while migrating VMs) as defined in [10].

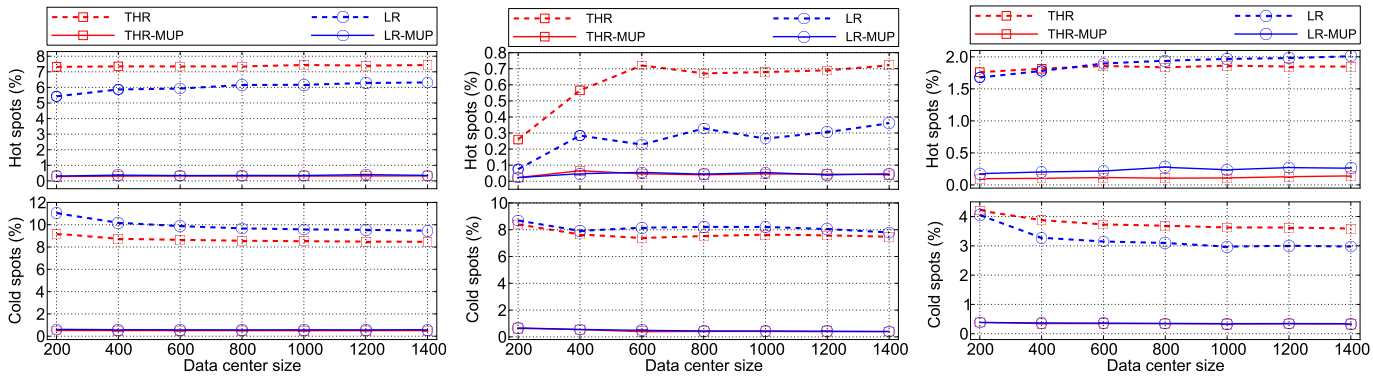



Fig. 4. Impact of the MUP scheme on the average number of hot and cold spots per data center for the: (a) random; (b) GCD and (c) PlanetLab workloads.

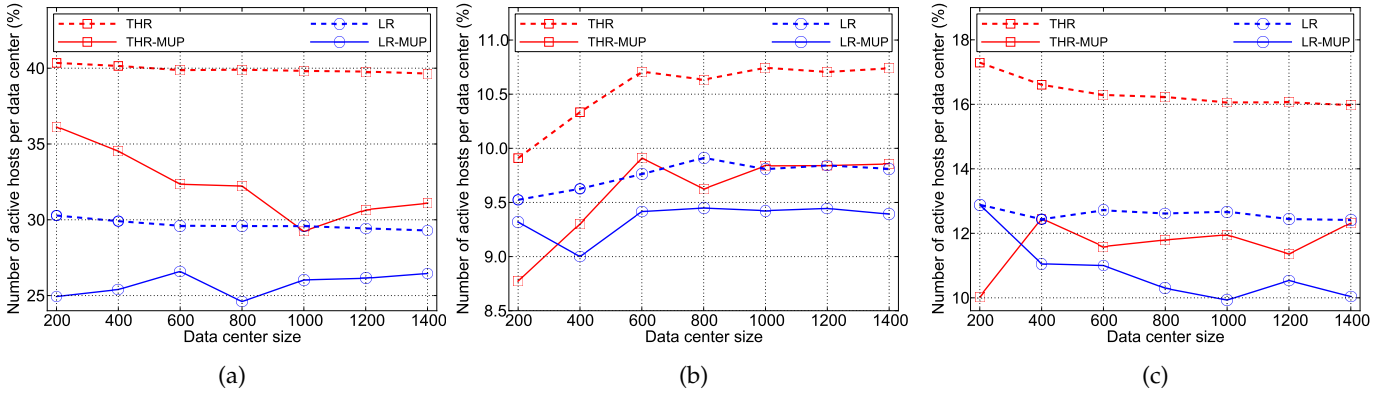



Fig. 5. Impact of the MUP scheme on the average number of active machines per data center for the: (a) random; (b) GCD and (c) PlanetLab workloads.

values over a time span of 24 hours for the random, the GCD and the PlanetLab workload traces. In particular, the average number of hot and cold spots per data center are shown in Figure 4 for two different overloaded host detection approaches, i.e, THR and LR. The results show that VM consolidation with MUP significantly reduces the number of real hot and cold spots in the system. Specifically, for the random workload trace (Figure 4a), MUP obtains a 93% reduction in both hot and cold spots compared to the algorithms without prediction for a cloud data center with 1,400 of machines. Additionally, for the GCD workload trace (Figure 4b), MUP reduces the number of hot and cold spots of more than 87% for the considered thresholds. The results are similar for the PlanetLab workload trace (Figure 4c). As a consequence, the algorithms correctly identify overloaded and underutilized servers in the system when using MUP.

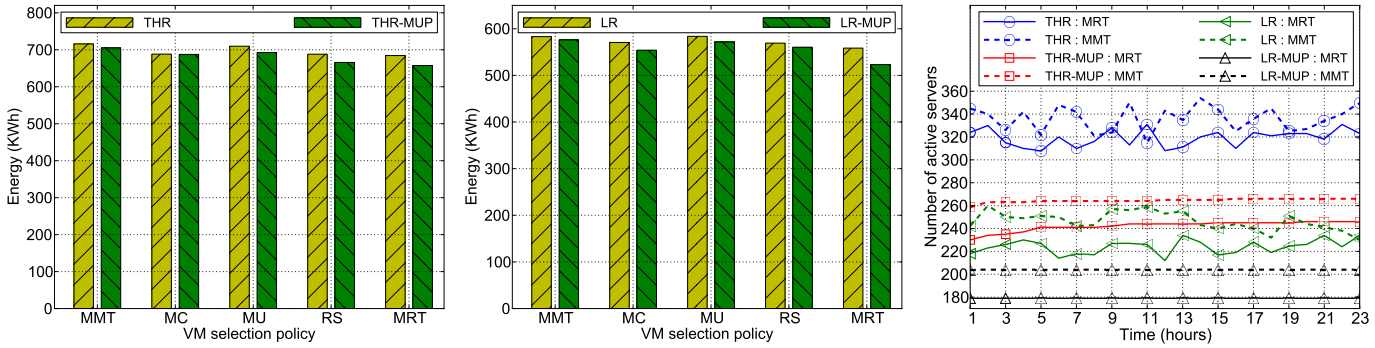
Next, we study the impact of MUP on the average number of active machines per data center for the random (Figure 5a), GCD (Figure 5b) and PlanetLab (Figure 5c) workload traces. The obtained results show that the overloaded and underloaded host detection approaches with MUP need the smallest number of active servers, even when the size of data center is high. In contrast, THR and LR without MUP result in the highest number of servers used. In fact, these algorithms take only current resource utilization as the main criterion for hot and cold spots detection, thus fail to reliably detect hot and cold spots due to the varying resource demands of VMs. As a result, they incur in unnecessary migrations, eventually resulting in powering on additional servers for placing the migrating VMs.

### 5.4 Experimental Results

In the following, we compare the performance of VMCUP-M against the other considered schemes for VM consolidation over the random and two real-world workloads.

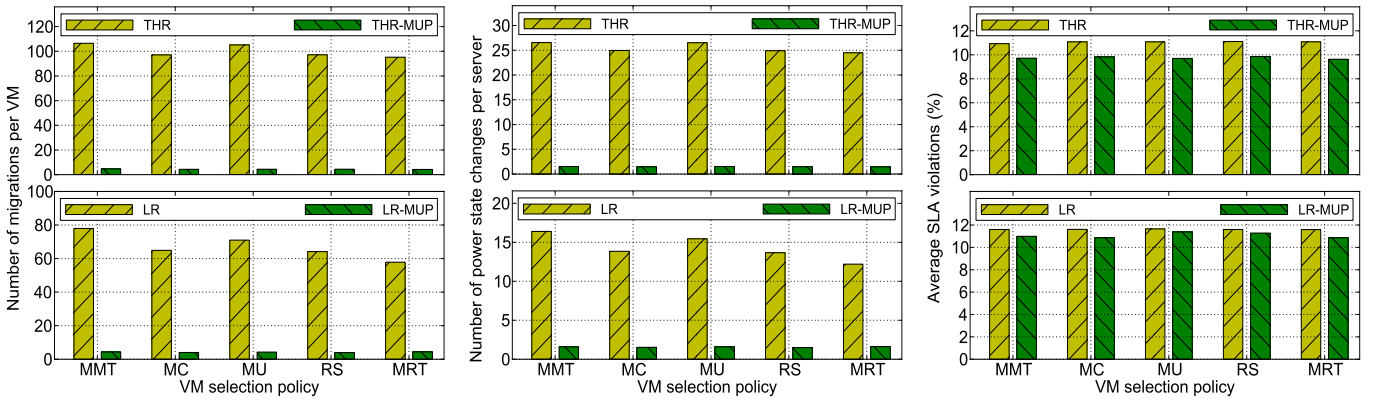
#### *5.4.1 Random Workload*

We first compare the energy consumption of VMCUP-M with the different VM selection policies for the THR (Figure 6a) and LR (Figure 6b) algorithms. In this case, VMCUP-M clearly obtains the best performance compared to THR and LR without prediction for all the considered VM selection policies. In particular, the results show that VM consolidation with MUP can decrease the energy consumption by more than 4.4% (THR) and 8% (LR). This indicates that VMCUP-M can correctly predict underutilized servers and minimizes the energy costs by switching idle hosts to a low-power state. Specifically, the proposed MRT selection scheme has the smallest energy consumption among the considered schemes. Without MUP, the decrease in the energy consumption with MRT is apparent with respect to MMT (more than 4.22%) and MU (3.56%), while it is limited if compared to MC and RS. Overall, the obtained results




(a) (b) (c)

Fig. 6. Energy consumption of VMCUP-M for the random workload trace with: (a) the THR and (b) the LR overloaded host detection schemes. Number of active servers for MRT and MMT as a function of time (c).




(a) (b) (c)

Fig. 7. Performance of VMCUP-M under the THR and LR algorithms with different VM selection policies for the random workload trace: (a) number of migrations per VM; (b) number of power state changes per server and (c) SLA compliance.

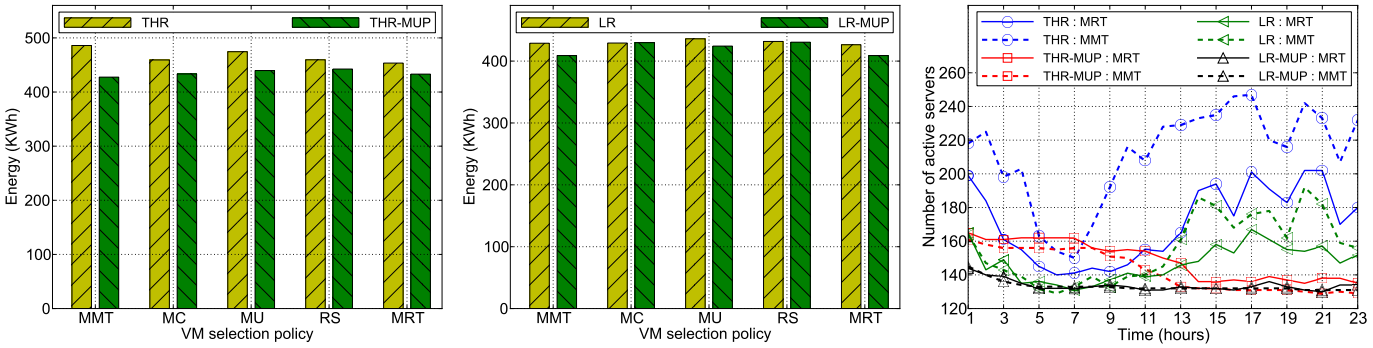
show that VMCUP-M combined with MRT significantly reduces the energy consumption by avoiding unnecessary migrations and server switches while adapting to the varying resource needs of VMs.

Figure 6c compares the performance of MRT with the MMT VM selection scheme in terms of the number of powered-on servers over time. In this case, MRT clearly obtains the best performance for both the THR and LR overutilized host detection approaches. Without MUP, the algorithms simply use the current resource load for making decisions. Therefore, it may incorrectly forecast overloaded and underloaded servers, thus resulting in unnecessary migrations, eventually increasing the number of active physical machines. When MUP is employed, the VM consolidation procedure decreases the number of active servers to a smaller number which is almost constant over time. As a consequence, MUP helps avoid rapid changes in the number of active machines in a cloud data center.

We now focus on the number of migrations per VM, switches per server, and SLA violations, shown in Figure 7 with THR (top part of the figures) and LR (bottom part of the figures). VMCUP-M obtains the best performance, compared to all other VM selection policies. Figure 7a shows the number of migrations per VM in the system with multiple usage prediction is the smallest one. The figure shows that VMCUP-M reduces the number of migrations of more than 93% compared to all other schemes due to temporary resource load. Figure 7b shows that the number of power state changes per server with MUP is much smaller for all considered VM selection policies. The reduction in the server switches is more than 94% (THR) and 88% (LR) for all considered VM selection schemes because VMCUP-M correctly forecasts underutilized servers. This helps to limit the frequency of server switches from idle to a low-power state and vice versa. It is important to remember that hosts cannot perform any useful processing while changing their power states. Figure 7c shows that VMCUP-M significantly reduces the number of SLA violations compared to the considered VM selection schemes. As a consequence, the reported results demonstrate that the proposed approach is effective while consolidating VMs.

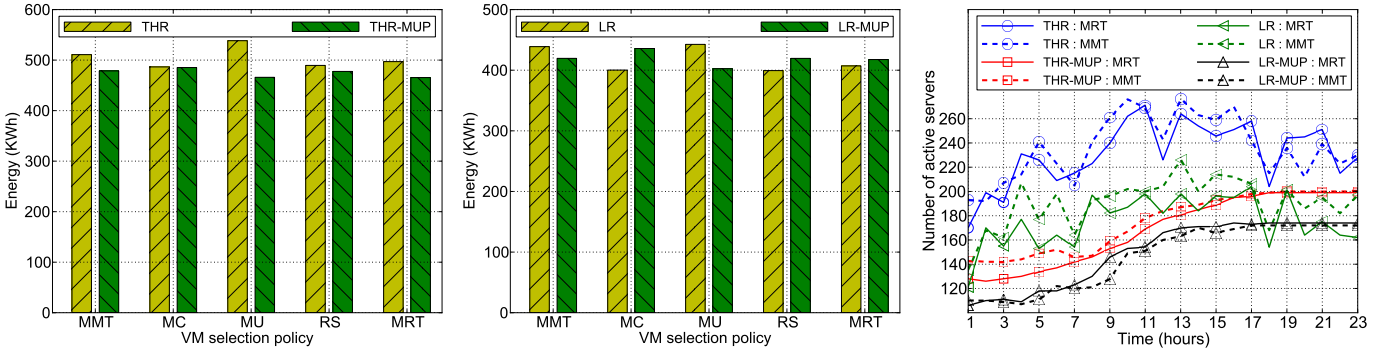
### 5.4.2 *Real Workloads*

We now compare the energy consumption of VMCUP-M under different VM selection policies for the GCD workload trace with THR (Figure 8a) and LR (Figure 8b). In this case, VMCUP-M reduces the power consumption of 5.6% or more (THR) and 4.6% or more (LR) with respect to algorithms without MUP. The results indicate that consolidating VMs with MUP plays an important role in minimizing energy costs. Moreover, MRT also achieves the lowest energy consumption compared to all other VM selection policies, even




(a) (b) (c)

Fig. 8. Energy consumption of VMCUP-M for the GCD workload trace with: (a) the THR and (b) the LR overloaded host detection schemes. Number of active servers for MRT and MMT as a function of time (c).




(a) (b) (c)

Fig. 9. Energy consumption of VMCUP-M for the PlanetLab workload trace with: (a) the THR and (b) the LR overloaded host detection schemes. Number of active servers for MRT and MMT as a function of time (c).

without prediction. Figure 8c shows that VM consolidation with multiple usage prediction clearly obtains the smaller number of active hosts in the system. This is because VMCUP-M not only takes into account the current state of resources but also future usage.

For the PlanetLab workload trace, VMCUP-M significantly reduces the energy consumption while consolidating VMs for most of the VM selection policies under the THR (Figure 9a) and the LR (Figure 9b) hot thresholds. It reduces the energy of 4.4% than MMT and 9.07% than MU; however, it has a slightly higher energy consumption than the MC and RS selection schemes. Thus happens as the PlanetLab VMs traces are mainly CPU-bound. Besides, the memory utilization of VMs in PlanetLab is almost constant over time. In terms of the number of active servers, Figure 9c shows that the algorithms with MUP achieve the smaller number of active hosts in the system.

We then evaluate the performance of VMCUP-M in terms of the number of migrations per VM, switches per server, and average SLA violations, shown in Figure 10 and Figure 11. We separate the THR (top part of the figures) and the LR (bottom part of the figures) hot threshold algorithms into sub-figures. VMCUP-M reduces the number of migrations per VM by more than 65% (Figure 10a) for GCD; and by more than 92% for PlanetLab (Figure 11a) for both the THR and LR algorithms, respectively. The results are similar for the number of power state changes per server (Figure 10b and Figure 11b). These results are due to the fact that VMCUP-M correctly predicts overutilized and underutilized servers based on current as well as future load. Furthermore, VMCUP-M also effectively finds the suitable destination hosts while evaluating VM migration, thus avoiding unnecessary migrations from the selected target hosts in near future. VMCUP-M reduces the average SLA violation percentage, i.e., more than 16% (with the GCD workload, shown in Figure 10c); and more than 22% (with the PlanetLab workload, shown in Figure 11c) for both THR and LR, compared to the other approaches by using a MUP of the overloaded hosts. This ensures that the destination servers do not become overutilized while migrating VMs.

Finally, we observe that the proposed MUP scheme is able to cooperate with the existing VM selection policies (i.e, MMT, MC, MU and RS) to lower the energy consumption, limit the frequency of VM migrations and server switches while getting a better compliance with the SLA. Furthermore, MUP can jointly applied with existing VM placement algorithms (i.e., PABFD) to correctly select the target host that does not become a hot spot in the long-term future.

#### 5.4.3 *Comparison Between* VMCUP-M *and* BG

In this section, we compare our proposed VMCUP-M approach against the BG algorithm. In detail, Figure 12 shows the performance of VMCUP-M in terms of energy consumption. We can see that VMCUP-M consumes less energy than

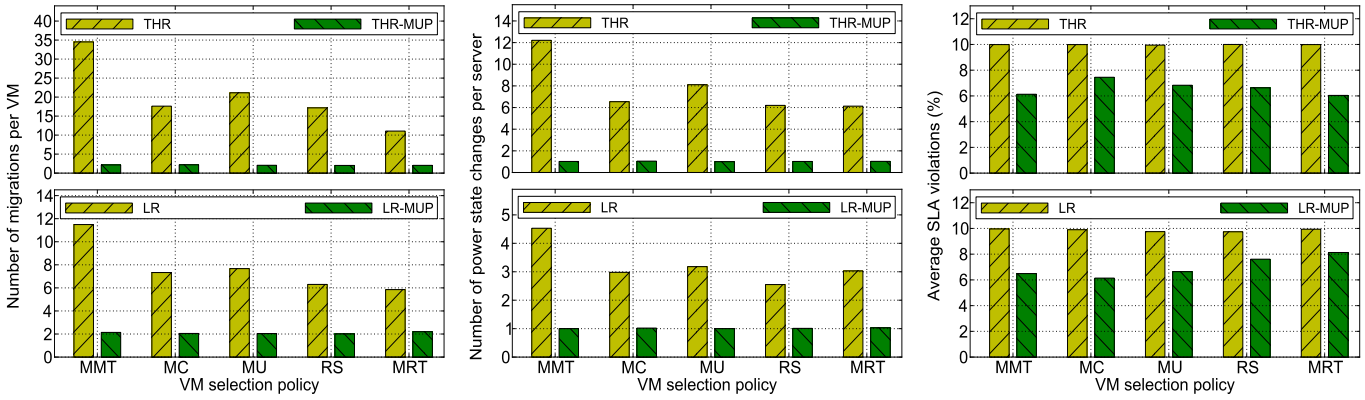



Fig. 10. Performance of VMCUP-M under the THR and LR algorithms with different VM selection policies for the GCD workload trace: (a) number of migrations per VM; (b) number of power state changes per server and (c) SLA compliance.

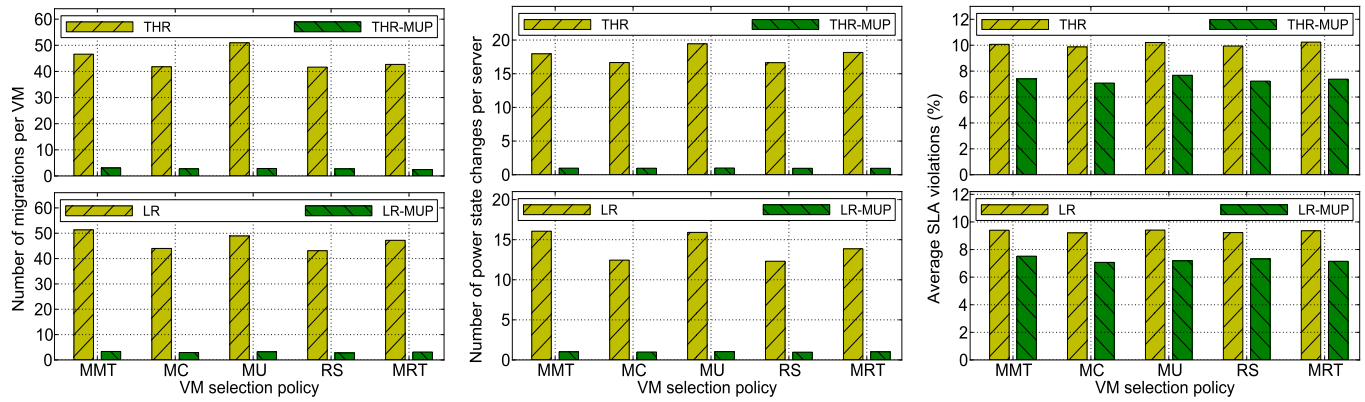



Fig. 11. Performance of VMCUP-M under the THR and LR algorithms with different VM selection policies for the PlanetLab workload trace: (a) number of migrations per VM; (b) number of power state changes per server and (c) SLA compliance.

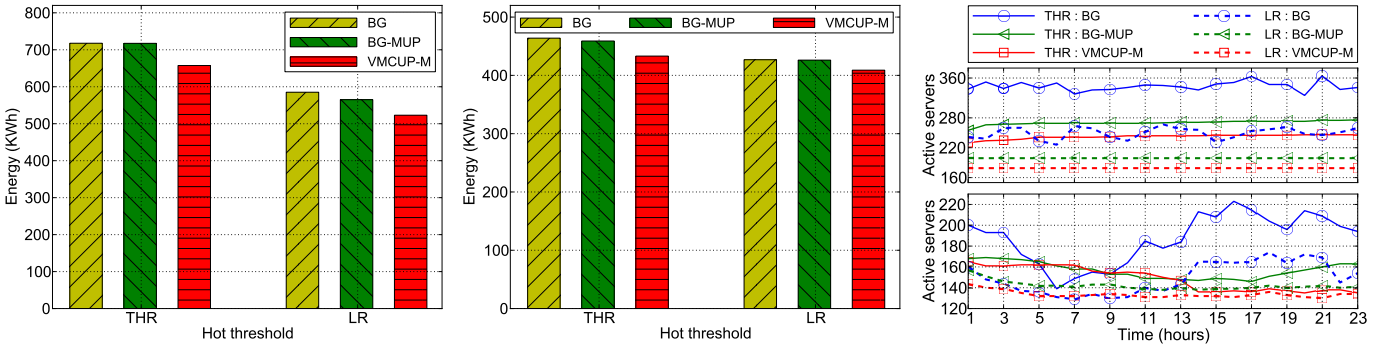
both BG and BG-MUP over the THR and LR overutilized host detection approaches for both the random (Figure 12a) and the GCD (Figure 12b) workloads. These results are due to the fact that BG allocates the migrating VMs onto idle servers first because they have the lowest volume metric. Furthermore, Figure 12c shows the number of active hosts over time under the random (top part of the figure) and the GCD (bottom part of the figure) workloads. As a result, VM consolidation with MUP plays an important role in reducing the energy consumption and the number of active servers in a data center.

Figure 13 compares the performance of VMCUP-M against the BG algorithms for the random workload trace. In particular, VMCUP-M significantly reduces both the number of migrations (Figure 13a) and the number of power state changes per server (Figure 13b). These results are due to the fact that VMCUP-M only selects the VMs that contribute more to the considered overloaded server according to its resource temperature. This helps avoid unnecessary migrations and power state changes. VMCUP-M also reduces the average SLA violation percentage (Figure 13c), compared to the BG algorithm by using MUP for the overloaded and underloaded hosts. The reason is that the destination servers do not become overutilized in neither the current nor the future time period while migrating VMs.

To complete our analysis, we evaluate the performance of VMCUP-M compared to BG for the GCD trace (Figure 14). Again, the improvement over THR and LR is apparent. In particular, Figure 14a shows that VMCUP-M results in the smallest number of migrations; especially VMCUP-M reduces about 78.26% of the migrations. Furthermore, VMCUP-M obtains a better performance than BG in terms of the number of server switches (Figure 14b) and SLA compliance (Figure 14c). Consequently, combining current and future prediction of multiple resource usage results in a reduction of both energy consumption and SLA violations.

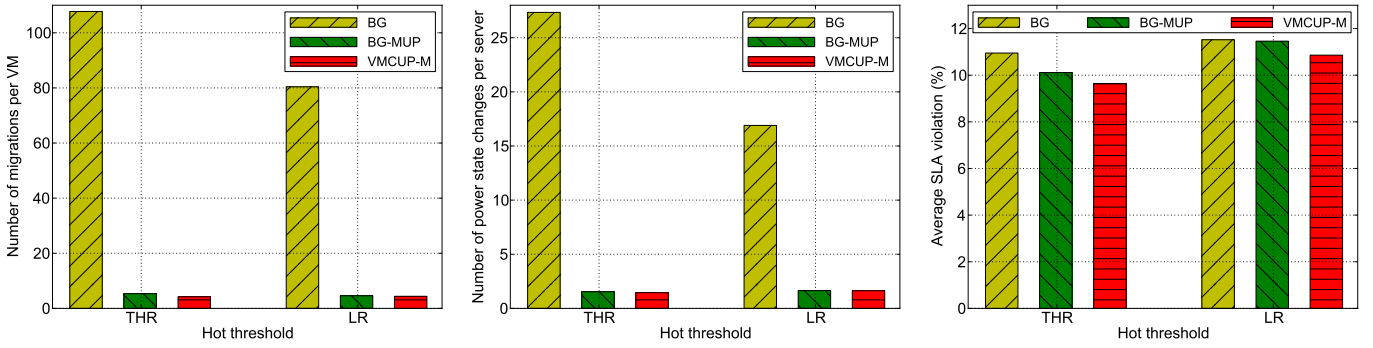
## 6 Conclusion and Discussion

In this article, we addressed the VM consolidation problem by adopting multiple usage prediction. Our aim was to reduce the frequency of VM migrations and server switches to save energy. To this end, we proposed a consolidation algorithm with multiple usage prediction for energy-efficient cloud data centers. The proposed algorithm effectively reduces the number of active servers, migrations, power state changes and the energy consumption of the servers. Simulation results on both synthetic and real-world workload traces confirmed that our approach can significantly decrease the energy consumption generated by active physical machines, VM migrations and host switches with a better compliance with the SLA.




(a) (b) (c)

Fig. 12. Performance of VMCUP-M and BG with the THR and LR overutilized host detection approaches. Energy consumption for the: (a) random workload trace and (b) GCD workload trace. Number of active servers for both the random (top) and GCD (bottom) workload traces (c).




(a) (b) (c)

Fig. 13. Performance of VMCUP-M and BG with THR and LR overutilized host detection approaches for the random workload trace: (a) number of migrations per VM; (b) number of power state changes per server and (c) SLA compliance.

As a future work, we seek to evaluate the performance of the proposed algorithm in real data centers. Furthermore, we intend to characterize the impact of the network load on the energy consumption of cloud data centers.

## ACKNOWLEDGMENTS

The authors are grateful to the editor and to the anonymous reviewers for their insightful comments and constructive suggestions. The authors would also like to thank Anton Beloglazov for providing the PlanetLab dataset used in the evaluation. This work was partially supported by the Academy of Finland under grant numbers 278207 and 299222.

## REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Journal of Future Generation Computer Systems*, vol. 25, pp. 599–616, 2009.

[2] S. Bhardwaj, L. Jain, and S. Jain, "Cloud computing: A study of Infrastructure as a Service (IaaS)," *International Journal of Engineering and Information Technology*, vol. 2, pp. 60–63, 2010.

[3] J. Zhu, *Cloud Computing Technologies and Application*. Hanbook of Cloud Computing, 2010, pp. 21–45.

[4] C.-H. Hsu, K. D. Slagter, S.-C. Chen, and Y.-C. Chung, "Optimizing energy consumption with task consolidation in clouds," *Information Sciences*, vol. 258, pp. 452–462, 2014.

[5] S. Chaisiri, B.-S. Lee, and D. Niyato, "Optimization of resource provisioning cost in cloud computing," *IEEE Transactions on Services Computing*, vol. 5, pp. 164–177, 2012.

[6] R. W. Ahmad, A. Gani, S. H. A. Hamid, M. Shiraz, A. Yousafzai, and F. Xia, "A survey on virtual machine migration and server consolidation frameworks for cloud data centers," *Journal of Network and Computer Applications*, vol. 52, pp. 11–25, 2015.

[7] A. Beloglazov and R. Buyya, "Managing overload hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints," *IEEE Transactions of Parallel and Distributed Systems*, vol. 24, pp. 1366–1379, 2012.

[8] S. K. Garg, A. N. Toosi, S. K. Gopalaiyengar, and R. Buyya, "SLA-based virtual machine management for heterogeneous workloads in a cloud datacenter," *Journal of Network and Computer Applications*, vol. 45, pp. 108–120, 2014.

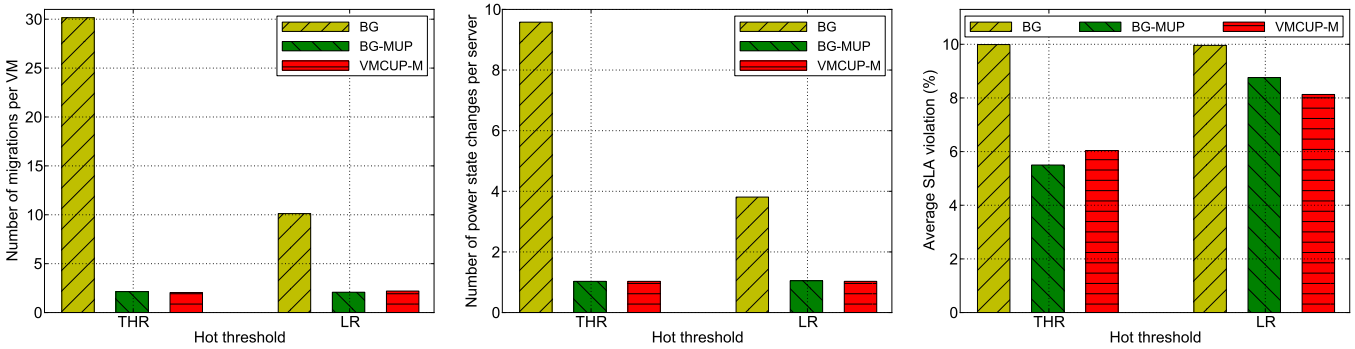
[9] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *IEEE Transactions of Parallel and Distributed Systems*, vol. 24, pp. 1107–1116, 2013.

[10] 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*, pp. 1397–1420, 2012.

[11] A.-C. Orgerie, M. D. D. Assuncao, and L. Lefevre, "A survey on techniques for improving the energy efficiency of large–scale distributed systems," *ACM Computing Surveys*, vol. 46, pp. 1–31, 2014.

[12] T. Mastelic, A. Oleksiak, H. Claussen, I. Brandic, J.-M. Pierson, and A. V. Vasilakos, "Cloud computing: Survey on energy efficiency," *ACM Computing Surveys*, vol. 47, pp. 1–36, 2014.

[13] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Black-box and gray-box strategies for virtual machine migration," in *The 4th*




(a) (b) (c)

Fig. 14. Performance of VMCUP-M and BG with THR and LR overutilized host detection approaches for the GCD workload trace: (a) number of migrations per VM; (b) number of power state changes per server and (c) SLA compliance.

*USENIX conference on Networked systems design and implementation*, 2007, pp. 229–242.

[14] T. C. Ferreto, M. A. S. Netto, R. N. Calheiros, and C. A. F. De Rose, "Server consolidation with migration control for virtualized data centers," *Future Generation Computer System*, vol. 27, pp. 1027–1034, 2011.

[15] I. Takouna, E. Alzaghoul, and C. Meinel, "Robust virtual machine consolidation for efficient energy and performance in virtualized data centers," in *The IEEE International Conference on Green Computing and Communications (GreenCom)*, September 2014, pp. 470–477.

[16] M. Blackburn, "Five ways to reduce data center server power consumption," The Green Grid, Tech. Rep., 2008. [Online]. Available: http://www.thegreengrid.org/~/media/WhitePapers/White_Paper_7_-_Five_Ways_to_Save_Power.pdf?lang=en

[17] V. Mathew, R. K. Sitaraman, and P. Shenoy, "Energy-aware load balancing in content delivery networks," in *2012 IEEE INFOCOM*, 2012, pp. 954–962.

[18] T. H. Nguyen, M. Di Francesco, and A. Ylä-Jääski, "A multi-resource selection scheme for virtual machine consolidation in cloud data centers," in *The 6th IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, 2014, pp. 234–239.

[19] ——, "Virtual machine consolidation with usage prediction for energy-efficient in cloud data centers," in *The 8th IEEE International Conference on Cloud Computing (CLOUD)*, 2015, pp. 750–757.

[20] C. Mastroianni, M. Meo, and G. Papuzzo, "Probabilistic consolidation of virtual machines in self-organizing cloud data centers," *IEEE Transactions on Cloud Computing*, vol. 1, pp. 125–228, 2013.

[21] F. Farahnakian, P. Liljeberg, and J. Plosila, "LiRCUP: Linear regression based CPU usage prediction algorithm for live migration of virtual machines in data centers," in *The 39th Euromicro Conference Series on Software Engineering and Advanced Applications*, September 2013.

[22] S. Islam, J. Keung, K. Lee, and A. Liu, "Empirical prediction models for adaptive resource provisioning in the cloud," *Future Generation Computer Systems*, vol. 28, pp. 155–162, 2012.

[23] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Energy-efficient cloud resource management," in *2014 IEEE INFOCOM*, 2014, pp. 386–391.

[24] X. Dai, J. M. Wang, and B. Bensaou, "Energy–efficient virtual machines scheduling in multi–tenant data centers," *IEEE Transaction on Cloud Computing*, vol. 4, pp. 210–221, 2016.

[25] U. Wajid et al., "On achieving energy efficiency and reducing $co_2$ footprint in cloud computing," *IEEE Transaction on Cloud Computing*, vol. 4, pp. 138–151, 2016.

[26] L. Tomás, E. B. Lakew, and E. Elmroth, "Service level and performance aware dynamic resource allocation in overbooked data centers," in *The 16th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid'16)*, 2016, pp. 42–51.

[27] J. O. Gutierrez-Garcia and A. Ramirez-Nafarrate, "Collaborative agents for distributed load management in cloud data centers using live migration of virtual machines," *IEEE Transaction on Services Computing*, vol. 6, pp. 916–929, 2015.

[28] S. Weisberg, *Applied Linear Regression, 3rd edition*. Wiley Series in Probability and Statistics, 2005.

[29] Google Cluster Workloads, https://github.com/google/cluster-data, 2016.

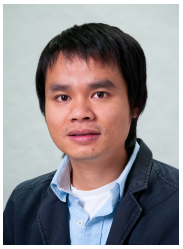
[30] S. Arlot and A. Celisse, "A survey of cross-validation procedures for model selection," *Statistics Surveys*, vol. 4, pp. 40–79, 2010.

[31] M. Jorgensen, "Experience with the accuracy of software maintenance task effort prediction model," *IEEE Transactions on Software Engineering*, vol. 21, pp. 674–681, 1995.

[32] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software Practice and Experiment*, vol. 41, pp. 23–50, 2011.

[33] D. Rice, J. Glick, D. Cercy, C. Sandifer, and B. Cramblitt, "Standard Performance Evaluation Corporation (SPEC)," https://www.spec.org/power_ssj2008/results/, 2015.

[34] Amazon EC2, http://aws.amazon.com/ec2/, 2016.



**Nguyen Trung Hieu** received his Ph.D. degree in Computer Science and Engineering from Aalto University, Finland, in 2016. He is currently a postdoctoral researcher with the School of Information Sciences of the University of Tampere, Finland. He has been a visiting researcher at the Next Generation Networks Group with the University of Colorado at Boulder between November 2015 and March 2016. His current research interests include distributed processing, cloud computing, energy–efficient cloud data centers, and big data analytics.

**Mario Di Francesco** received his Ph.D. degree in Information Engineering from the University of Pisa, Italy, in 2009. He is an Assistant Professor with the Department of Computer Science at Aalto University, Finland. He received a best paper award at the UbiComp 2014 conference. He is an area editor of the Pervasive and Mobile Computing Journal and he was co-chair for the third workshop on the IoT: Smart Objects and Services in 2014. His research interests include the Internet of things, pervasive computing, mobile networking, and performance evaluation.

**Antti Ylä-Jääski** received his Ph.D. degree in ETH Zuerich 1993. He worked with Nokia between 1994 and 2009 in several research and research management positions with focus on future Internet, mobile networks, applications, services and service architectures. He has been a professor of Telecommunications Software at the Department of Computer Science of Aalto University since 2004. He supervised over 200 master's thesis and 14 doctoral dissertations during his professorship. He has published more than 60 international peer–reviewed journal and conference articles.