A Predictive Anti-Correlated Virtual Machine Placement Algorithm for Green Cloud Computing

Conference Paper · December 2018

DOI: 10.1109/UCC.2018.00035

3 authors:

Rachael Shaw
Galway-Mayo Institute of Technology
9 PUBLICATIONS **262** CITATIONS
SEE PROFILE

Enda Howley
University of Galway
113 PUBLICATIONS **2,913** CITATIONS
SEE PROFILE

Enda Barrett
71 PUBLICATIONS **2,160** CITATIONS
SEE PROFILE

A Predictive Anti-Correlated Virtual Machine Placement Algorithm for Green Cloud Computing

Rachael Shaw
College of Engineering and Informatics,
National University of Ireland,
Galway, Ireland
r.shaw4@nuigalway.ie

Enda Howley
College of Engineering and Informatics,
National University of Ireland,
Galway, Ireland
ehowley@nuigalway.ie

Enda Barrett
College of Engineering and Informatics,
National University of Ireland,
Galway, Ireland
enda.barrett@nuigalway.ie

***Abstract*—Energy related costs and environmental sustainability present a significant challenge for cloud computing practitioners and the development of next generation data centers. Inefficient resource management is one of the greatest causes of high energy consumption in the operation of data centers today. Virtual Machine (VM) placement is a promising technique to save energy and improve resource management. A key challenge for VM placement algorithms is the ability to accurately forecast future resource demands due to the dynamic nature of cloud applications. Furthermore, the literature rarely considers placement strategies based on co-located resource consumption which has the potential to improve allocation decisions. Using real workload traces this work presents a comparative study of the most widely used prediction models and introduces a novel predictive anti-correlated VM placement approach. Our empirical results demonstrate how the proposed approach reduces energy by 18% while also reducing service violations by over 47% compared to some of the most commonly used placement policies.**

***Index Terms*—energy efficiency, virtual machine placement, machine learning, cloud data centers**

I. Introduction

Cloud computing is a popular service provisioning model which offers users access to on-demand resources accessible over the internet and charged on a pay-as-you-go basis [1]. Today, the use of virtualization technology such as Xen and VmWare [21] serves as a standard mechanism to improve resource management in data center operations. Virtualisation technology enables better utilisation of available resources through the execution of multiple workloads on the same physical machine. This is achieved by partitioning the resources of large physical servers into multiple smaller independent machines each of which is known as a Virtual Machine. Each VM executes in apparent isolation running its own Operating System (OS) and applications [2].

However, despite the benefits gained through the use of virtualisation technology energy efficiency and environmental sustainability have been identified as critical issues in the operation of data centers today. The formidable problem of energy conservation in data centers challenges the research community and practitioners alike to introduce and adopt more efficient, energy driven resource management strategies. Research has revealed that in 2013, U.S data centers consumed a staggering 91 billion kilowatt-hours of electricity. By 2020 the level of consumption is projected to increase to approximately 140 billion kilowatt-hours per annum, costing 13 billion dollars per year and generating carbon emissions of 150 million metric tons [3]. In particular, inefficient resource utilisation has been identified as one of the greatest causes of energy consumption in data center deployments [4]. VM placement is one approach that has been studied extensively over the years in an attempt to remediate this persisting issue. The VM placement problem can be formulated as a bin-packing problem, the objective is to allocate as many VMs as possible on to a reduced number of servers using live migration in an effort to optimize resource usage while also satisfying user specified Service Level Agreements (SLA) [5]. One key factor in the design of an effective VM placement algorithm is the ability to manage the energy-performance tradeoff [5]. Consolidating a large number of VMs on to a reduced number of servers can cause resource contention leading to performance degradation and the need to migrate VMs to additional hosts in the data center. Conversely, placing a VM on an underutilized server promotes the continuation of poor resource utilisation and prevents underutilized hosts from being powered down to conserve energy.

Currently, the allocation approach adopted by many solutions considers the VM placement problem from the perspective of allocating a single VM to a host at any given time [6]–[8]. This type of approach provisions the resource requirements of VMs independently ensuring each host has sufficient capacity to execute the workload. However, this method lends itself to less efficient resource utilisation [11]. Application workloads often exhibit time varying demand patterns which can be anti-correlated in that resource requirements at a given time complement one another [12]. By coupling complementary VMs together and allocating them to a host based on their aggregated resource requirements, re-

This work is supported by the Irish Research Council through the Government of Ireland Postgraduate Scholarship Scheme.

source utilisation can be improved while reducing energy and enhancing the ability of service providers to adhere to SLA. Another important consideration is that many conventional VM placement solutions employ a placement strategy which seeks to optimize resource usage based on current resource demands [13]–[15]. However, application workloads invariably fluctuate over time posing a significant challenge to such approaches. Failing to consider future demand can quickly result in redundant placement decisions having a negative impact on energy and performance. As a result, VM placement solutions need to be able to forecast future resource demand to optimize limited resource availability.

In this paper we address the problem by presenting a novel Predictive Anti-Correlated Placement Algorithm (PACPA) which we expect to impact the state-of-the-art as follows:

- Unlike the vast majority of existing solutions our work focuses on improving the VM placement problem by considering the relationship between migrating VMs prior to placement. Our solution attempts to combine VMs for placement when their predictive CPU resource demands complement each other in order to reduce the amount of resources required to execute the workloads. As a result improving both energy efficiency and performance.
- PACPA incorporates both current and future CPU demands of migrating VMs to inform its placement strategy. Given the recent growth in the application of Machine Learning (ML) approaches we conduct a comparative study to evaluate the accuracy of some of the most popular predictive algorithms as no single technique is widely agreed to be the best one.
- Our approach is implemented in a large scale simulated data center using real workload traces. We compare our solution to some of the most widely used placement heuristics which are known to deliver good results. We demonstrate the benefits of incorporating predictive modeling while also considering the relationship among VMs prior to placement.

The remainder of this paper is structured as follows: Section II discusses related work and background material. Section III formulates the research problem and the proposed solution. In Section IV we evaluate the accuracy of some of the most popular predictive techniques and select an appropriate model to incorporate into our placement strategy. Section V presents our experimental results. Lastly, Section VI concludes the paper and discusses future work.

II. Related Work & Background

In this section we discuss related research while also providing the necessary background material on the forecasting techniques that will be evaluated as part of this work.

A. Resource management in the Cloud

In recent years VM placement optimization has gained tremendous attention in the literature as it proves to be a promising solution having far-reaching effects on data center resource management policies. Heuristic based resource management algorithms are the most commonly used methodologies to optimize resource usage in the cloud due to their simplicity. Lee et al. [4] proposed two task consolidation heuristics which seek to maximize resource utilisation while considering active and idle energy consumption in placement decisions. Verma et al. [13] presented pMapper, a power and migration cost aware placement controller. Cardosa et al. [14] explored the impact of min, max and shares parameters. The authors used these parameters to inform their placement decisions while also providing a mechanism for managing the power-performance tradeoff in modern data centers. Son et al. [10] introduced a dynamic resource overbooking strategy based on historical resource utilisation data to improve both energy efficiency and performance. One of the most highly cited works in the area is presented by Beloglazov et al. [16]. They proposed the Power Aware Best Fit Decreasing (PABFD) heuristic. This heuristic sorts VMs in decreasing order based on current CPU utilisation and allocates each VM to a host that will result in the least power increase. Unlike our work, these approaches allocate VMs independently without considering the relationship between VMs prior to making allocation decisions. This is an important factor to consider given that the peak demands in one workload may not necessarily coincide with that of another. In this case coupling both VMs and placing them on the same host might be more optimal resulting in less fragmentation of limited available resources.

It is inherently difficult to make informed placement decisions in the absence of a relatively accurate estimate of future resource requirements. Recently, more efforts have been made to develop placement policies capable of predicting future resource demands using sophisticated ML algorithms. Farahnakian et al. [8] introduced a predictive VM consolidation approach which employs a K-nearest neighbours forecasting methodology to improve energy, SLA and also the number of migrations required in the data center to optimize resource utilisation. Haehnel et al. [9] employed a probability density function to estimate stochastic workloads to enable more efficient consolidation. Nguyen et al. [7] employed Multiple Linear Regression (MLR) to predict over and under-loaded server capacity and showed how it can be integrated with existing placement solutions to improve energy consumption. Bobroff et al. [6] proposed the use of an Autoregressive (AR) forecasting model to adapt placement decisions based on predicted resource demand. Their results show that by leveraging the predictive capabilities of an AR model they could improve the delivery of SLA guarantees and promote greater resource usage efficiency.

A limited number of works in the literature consider joint VM allocation strategies in the placement problem [11], [18]. While these studies advocate the use of VM resource demands that are temporally unaligned to improve placement decisions they are not future facing solutions as they only consider current resource demands. Unlike the work of Chen et al. [12] we use real workload traces to improve energy efficiency and performance holistically. Recently, Affinity based VM placement was introduced by Fu et al. [19]. Their solution employed an Auto-Regressive Integrated Moving Average (ARIMA) model to forecast resource demand. Our work differs in that we conduct a comparative study exploring the application of linear and non linear prediction algorithms including ARIMA and Artificial Neural Network (ANN) methodologies. Over the years, several forecasting models have been proposed in the literature, of which ARIMA and ANN methodologies have become widely popular. However, there have been mixed conclusions regarding the superiority in performance of one forecasting model over the other [20].

B. Forecasting

Recently, there has been a growing interest in the application of statistical methods and ML techniques to improve the overall efficiency of cloud data centers. Several works demonstrate the compelling benefits of such approaches which seek to provide cloud infrastructure with the means to better adapt to dynamic changes in resource utilisation in order to optimize resource allocation, scheduling and migration [1], [2], [21], [22]. However, the ability to accurately predict future resource demand is one of the most significant challenges facing cloud resource management strategies due to the growing complexity of modern data centers [23]. In the forecasting literature ARIMA and ANN methodologies have been the main focus of research. These models are widely recognized as the most popular and commonly used forecasting models among others such as Moving Average (MA) and Random Walk (RW) models. In particular, both models are often compared and have been extensively studied with mixed conclusions in terms of superiority in forecasting performance [20].

In this work we provide a competitive analysis of the above models in terms of their predictive accuracy on the time series data used in this study. Based on our empirical evaluation the best performing model is selected and implemented as part of the proposed PACPA algorithm. Below is an outline of the models in more detail.

- **ARIMA** modelling is one of the most frequently used methodologies for time series forecasting [24]. ARIMA models are defined by three fundamental components denoted as (p, d, q). Identifying a valid model is the process of finding suitable values for (p, d, q) which capture the systematic patterns in the time series data. The autoregressive (AR) component (p) represents the influence of preceding observations on current values in the series. For example, an $AR(1)$ model estimates future values based on the value of the previous observation y_{t-1} as defined below in (1).

$$\hat{y}_t = \phi(y_{t-1}) + \varepsilon_t \ . \tag{1}$$

where ϕ is a parameter of the model, ε_t is the random variation at time t and $\hat{y}_t$ is the predicted value. The moving average term (q) represents the effects of previous random variation on the current values random error. For example, an $MA(1)$ model predicts values based on a combination of the current random variation and previous error as given in (2).

$$\hat{y}_t = \theta(\varepsilon_{t-1}) + \varepsilon_t \ . \tag{2}$$

where ε_{t-1} is the value of the previous random shock and θ is a parameter of the model. Lastly, one of the fundamental requirements in the application of ARIMA modelling is a non-stationary series must be transformed to a stationary series, that is the series statistical properties such as mean and variance must remain constant over time. The integrated component of the model (d) is the order of differencing applied to the series in order to transform a non-stationary series into a stationary series. The combined model assuming the data has already been differenced is denoted in (3) where c is defined as a constant.

$$\begin{aligned} \hat{y}_t = c + \phi_1(y_{t-1}) +, ..., + \phi_p(y_{t-p}) + \\ \theta_1(\varepsilon_{t-1}) +, ..., + \theta_q(\varepsilon_{t-q}) + \varepsilon_t \end{aligned} \tag{3}$$

- **Artificial Neural Networks** are a state-of-the-art technique for many different learning problems including classification, control and online and offline learning [25]. ANNs are inspired by the human brain which is a highly complex, non-linear information processing system consisting of billions of connected neurons [26]. A standard feedforward neural network (also known as a multi-layer perceptron) consists of a network of interconnected computational nodes organized into an input layer, one or more hidden layers and an output layer. During training the network receives data through the input layer. In our problem we define the input vector as a normalized sequence of CPU utilisation values over a given time frame. These values are fed to the neurons in the hidden layer each of which is characterized by a set of weighted connections. The network calculates the sum of the weighted signals for each neuron u_k as given in (4).

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{4}$$

where $\{x_1, x_2, ..., x_m\}$ are the input signals of neuron k and $\{w_{k1}, w_{k2}, ..., w_{km}\}$ are the connection weights. An activation function is applied to the output signal of each neuron which limits the range of the signal to a finite value often between 0 and 1. The most commonly used activation function in the construction of a neural network is the sigmoid also known as the logistic function as defined in (5).

$$\varphi(u_k) = \frac{1}{1 + \exp(-a u_k)} \tag{5}$$

where a is the slope, u_k is the activation value for neuron k and $exp()$ is the exponential function. Once the signal has been propagated through the network it generates a predicted output value $\hat{y}_t$ for a future time step which is compared to the target output to generate the error term δ_k. The popular backpropagation algorithm [25] is used to propagate the error back through the network in order to update the weighted connections.

Feedforward neural networks are capable of modelling more complex non-linear functions as a result they are widely used time series forecasters. In the literature there are a variety of ANN types with many studies focused on evaluating their performance on a range of different problems. Chen at al. demonstrate the usefulness of a feedforward neural network for predicting workloads on the cloud to improve VM consolidation [12]. Hicham et al. [30] recently demonstrated the benefits of applying ANN algorithms to improve CPU scheduling in the cloud. In their study they compared several types of ANNs which have been successfully applied across multiple fields, including multi-layer perceptrons and recurrent neural networks. The results showed that the multi-layer perceptron model performed best for the CPU scheduling problem. Gers et al. also demonstrated that a sliding window multi-layer perceptron could produce more accurate results than a Long Short-Term Memory (LTSM) neural network approach [31]. For these reasons we focus on a multi-layer perceptron model using the sliding window technique and compare its performance to the other predictive models used in this study.

- **Moving Average** is a classical and commonly implemented forecasting approach [24]. The basic intuition behind the MA model lies in the assumption that the best predictor of future values is the average of the previous m values. A future value $\hat{y}_t$ can be computed using (6) below.

$$\hat{y}_t = \frac{y_{t-1} + y_{t-2}, ..., y_{t-m+1}}{m} \tag{6}$$

- **Random Walk** is the most simplistic forecasting technique implemented in our analysis. The predicted value in the next time step is calculated as the current value y_t with some random variation ε_t.

III. Anti-Correlated VM Placement

In this work we consider the relationship between migrating VMs as an important factor in optimizing the usage of available resources. It is widely accepted knowledge that VM workloads exhibit dynamically changing resource requirements over time due to changes in the number of user requests [6], [13]. This presents us with two different aspects of the placement problem to consider. Firstly, by placing highly complementary VMs together based on their aggregated CPU requirements, resource utilisation can potentially be improved as CPU is one of the most dominant factors in energy consumption and performance reliability [17]. Illustrated in Fig.1 is an example of using anti-correlated VM placement. As shown each VM displays time dependent demand patterns with a capacity bound based on the VMs maximum demand. However, the demand of each VM is anti-correlated showing high points of utilisation at different times while the overall range of utilisation for both workloads differs i.e 30-60% and 10-20%. In this example placing these VMs separately requires an allocation of 80% capacity across the two hosts. As shown, by placing two VMs together on a single host according to their aggregated resource demand it is possible to allocate less resources to accommodate the same workload. This results in saving 10% capacity in this placement decision alone. Furthermore, co-locating more than one VM on a host deemed to be the most optimal choice for placement also has the potential to reduce resource fragmentation across the data center. By considering the possible combinations from the VMs ready to be reallocated we can make better placement decisions to optimize the usage of limited available resources. This can improve usage of the data centers overall capacity resulting in greater efficiency. The second aspect of the placement problem which is also a growing challenge for cloud applications is the ability to accurately predict future resource demand due to the the dynamic nature of application workloads. A more efficient VM placement strategy should have the capacity to forecast future demand in order to support and prevent redundant placement decisions. Our solution uses the predictive capabilities of an ANN to help improve the likelihood of placement decisions remaining advantageous over short term future time steps.

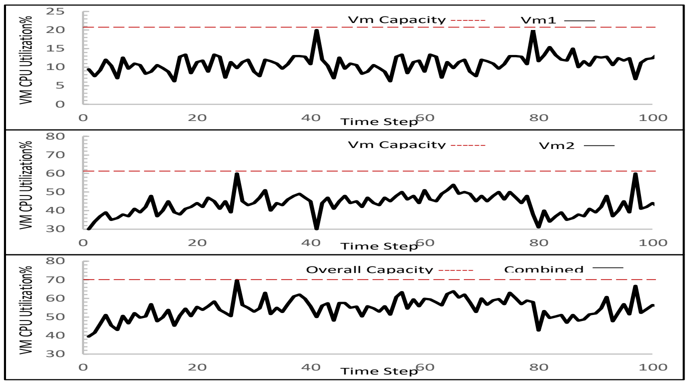

Fig. 1. Example of combining complimentary VM's

A. System Description

We model the problem using a large scale data center consisting of heterogeneous resources, including m Physical Machines (PMs) and n VMs, where $PM = \{pm_1, pm_2, ..., pm_m\}$ and $VM = \{vm_1, vm_2, ..., vm_n\}$. Fig. 2 illustrates the system architecture. In our system architecture application workloads must be processed which are dynamic by nature. The VMs in our data center are initially provisioned on to a suitable PM based on the requested resources. The requested CPU

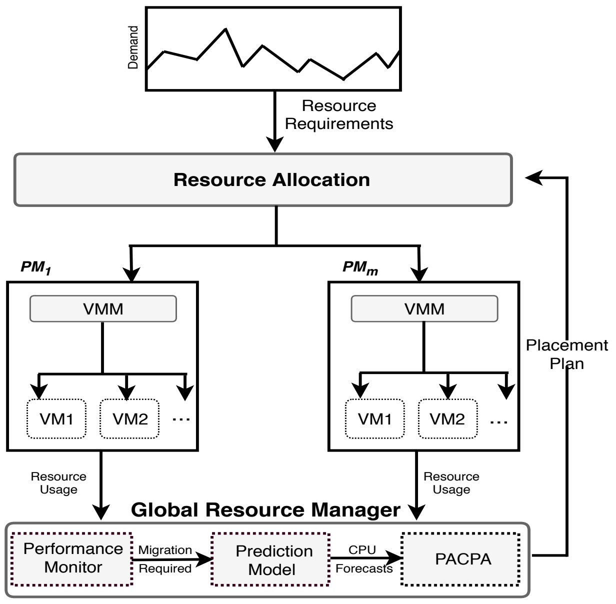

Fig. 2. System Architecture

resources of each VM dynamically change over time, as a result the system must monitor resource usage as hosts become loaded. To reduce energy and service violations it is critical to optimize the distribution of VMs using live migration and reallocate VMs to other PMs in the data center. VM resource optimization is often considered a three stage process. All three stages occur in a cyclical manner of firstly detecting hosts that are likely to become overloaded, selecting VMs to be migrated and finally placing VMs on to a more suitable host that will drive energy efficiency. In our system architecture the global resource manager is responsible for managing resource allocation and live migration. It consists of three key components: the performance monitor, prediction model and the proposed PACPA algorithm which provides decision support for VM placement optimization. The performance monitor continuously monitors resource usage on each host at each time step and stores this as historical resource usage data. Next the popular Local Regression Minimum Migration Time (LR-MMT) algorithm [16] is used to infer the probability of a host becoming loaded and selects which VMs to migrate. The prediction model is then used to generate the CPU forecasts for the VMs in the migration list. Based on these forecasts the PACPA algorithm makes placement decisions considering the anti-correlated relationship that often exists among migrating VMs. In particular, PACPA decides which VMs from the migration list should be placed together and to which PM they should be allocated.

B. Proposed PACPA Algorithm

Our proposed PACPA algorithm denoted below (Algorithm 1) predicts short-term CPU demand of VMs requiring reallocation due to the dynamic nature of application workloads. As mentioned previously, the system leverages the well known LR-MMT algorithm introduced by Beloglazov et al. [16]. This algorithm manages host overloaded detection and VM selection, two fundamental aspects of dynamic VM resource optimization. The selected VMs are used as input to the PACPA algorithm. Our proposed solution attempts to pair two VMs at any one time. We justify our approach based on previous studies in the literature and the inherently dynamic nature of application workloads. According to Kim et al. [18] aggregating the workloads of multiple VMs at any one time would be applicable only when future host utilisation is perfectly known. Although predictive techniques can be relatively accurate, future values can never be predicted with absolute certainty. By aggregating multiple vms based on predicted values it could result in a higher risk of performance degradation. PACPA calculates the predicted host resource usage for all possible combinations of VMS across all suitable hosts. In order to do so the algorithm selects the first VM in the migration list. To assess the anti-correlated property among the remaining VMs in the migration list we select the next VM in the list and simply calculate the variance between the two workloads at each time step. If the sum of the variance is high the VM is deemed to be anti-correlated with the selected VM and can then be considered for placement. The outcome of this task generates a refined migration list of VMs that are anti-correlated with the current VM selected from the original migration list. Next the predicted demand for the next T time steps is obtained using (7) based on predictions generated from the best performing forecasting model in our study.

$$Vm_i^p = (vm_i^d(t+1), vm_i^d(t+2), ..., vm_i^d(t+T)) \qquad (7)$$

where Vm_i^p contains a vector of predicted CPU values, one for each time step. Specifically, i denotes the VM index, p

Algorithm 1: *PACPA Algorithm*

```
Input: VM migration list and vector of CPU predictions
foreach $Vm_i$ in migrationlist do
    sum ← 0
    if $Vm_i$ allocated = False then
        //calculate anti-correlated property
        foreach $Vm_j$ in migrationlist do
            if sum of variance = high then
                new migration list ← $Vm_j$
            end
        end
        //calculate predicted demand over next T time steps
        $Vm_i^p \leftarrow (vm_i^d(t+1), vm_i^d(t+2), ..., vm_i^d(t+T))$
        $sum \leftarrow sum + Vm_i^p$
        foreach candidate $Vm_j$ in new migration list do
            $Vm_j^p \leftarrow (vm_j^d(t+1), vm_j^d(t+2), ..., vm_j^d(t+T))$
            $sum \leftarrow sum + Vm_j^p$
            //calculate total average predicted utilisation
            $U_{i,j}^{avg} = \frac{sum}{T} + \overline{o}$
            foreach suitable $pm_i$ in host list do
                if $R_{capacity} > U_{i,j}^{avg}$ then
                    //calculate host util and size of combined VMs
                    $pm_i^u = \frac{\sum_{k=1}^{n} vm_i^d}{pm_i^c} \times 100 + \frac{U_{i,j}^{avg}}{pm_i^c} \times 100$
                end
            end
        end
        Select optimal combination
    end
end
```

indicates that it is a predicted value, d represents VM demand and t denotes a particular time step in the future. A candidate VM is also selected from the new migration list to potentially combine with Vm_i^p. The predicted demand for the candidate VM Vm_j^p is also calculated using (7). We then calculate the total average predicted usage U_{ij}^{avg} over T time steps using (8). It is important to acknowledge that although predictive models may have good prediction accuracy they inevitably produce prediction errors. To help alleviate this problem a safety margin $\overline{o}$ is added to the total average predicted CPU usage. The $\overline{o}$ value is estimated based on the average error generated from the selected predictive model over the next T time steps. Using the average error the predicted value generated from the model can be adjusted in an attempt to accommodate these prediction errors.

$$U_{i,j}^{avg} = \frac{Vm_i^p + Vm_j^p}{T} + \overline{o} \tag{8}$$

where Vm_i^p and Vm_j^p are the vectors of predicted CPU values for the VMs that could potentially be placed together. T denotes the number of time steps and $\overline{o}$ is the estimated safety margin.

The selection of an appropriate host is also an integral factor in the placement problem in order to manage the energy-performance tradeoff. Placing VMs on to the least amount of hosts can cause resource contention resulting in service violations. Inversely, selecting an underutilized host promotes the continuation of inefficient resource utilisation preventing such hosts from being powered down to conserve energy. Ideally, an effective placement algorithm should strike a balance in the dispersal of VMs across the data center so as to prevent hosts becoming overloaded too quickly but also ensuring that resources are used efficiently. In light of this, to find the most optimal host to potentially place a candidate set of VMs PACPA calculates a value for each suitable host pm_i in the data center whose estimated resource capacity $R_{capacity}$ is greater than the average predicted usage U_{ij}^{avg} for Vm_i and the candidate VM Vm_j. In particular, as denoted in (9) we define the value given to each potential host as the estimated host utilisation rate. We calculate it as the estimated CPU resource demand for each VM already allocated to the host as a percentage of the hosts overall capacity pm_i^c, coupled with the predicted size of the combined set of VMs which could potentially be placed together. We estimate the CPU resource demand on any given host by keeping track of the VMs currently allocated to the host and also their predicted future demand. Based on this, PACPA generates a mapping of anti-correlated VMs on to suitable hosts and selects a host using a best fit approach which results in improved resource usage.

$$pm_i^u = \frac{\sum_{k=1}^{n} vm_i^d}{pm_i^c} \times 100 + \frac{U_{i,j}^{avg}}{pm_i^c} \times 100 \tag{9}$$

where i indicates the index of the PM, u represents the estimated resource usage, c denotes the overall capacity of the PM and vm_i^d is the estimated demand for a VM currently running on the host.

IV. Forecasting Experimental Details and Results

The experimental analysis and results of the selected forecasting models are presented below.

A. Forecasting Experimental Setup

A key limitation of neural networks for forecasting time series data is the temporal dependency of the data must be specified. In light of this, experiments were conducted implementing the sliding window method [27] in order to find the temporal dependency which would allow the network to learn an improved mapping function from a set of input sequences to output values. A sliding window maps an input sequence of size k to an output value $\hat{y}_t$, where the input sequence x is a vector of CPU values over k time intervals and the predicted value $\hat{y}_t$ is a single time step ahead of the input sequence. By conducting experimental parameter sweeps sliding windows in increments of 10 provided good performance for this problem.

Furthermore, linear time series models such as ARIMA are generally fit to a specific series of observations composed of a distinct set of characteristics such as a possible long term trend, seasonal fluctuations and correlations between sequential observations. The goal of such models is to identify and describe the underlying components and systematic variations in the specific time series data in order to forecast future values. Our study focuses on the placement of anti-correlated VMs with discrete characteristics. As a result two models were fitted for each time series in our study in order to provide the best fit for each trace type. In our analysis the combined errors are reported to provide a fair comparison of performance between linear and non-linear models.

B. Training and Test Data

To validate our approach the workload used in this study was generated based on a large real-world data set provided by the CoMon project, a monitoring infrastructure for PlanetLab [16]. The traces provide the CPU demand collected between March and April 2011 from more than a thousand VMs running on servers situated in more than 500 locations globally. The CPU usage is measured in 5 minute intervals resulting in 288 intervals in a given trace and measurements are provided for 10 days generating traces from over 10,000 VMs in total. In order to construct the training data to evaluate the performance of the selected prediction models we firstly filter the original data. Where the utilisation of some VMs in the data set are severely underutilized we replace usage rates of 0% with 5% for the purposes of our evaluation. Given that this work is primarily focused on predicting the CPU demand of anti-correlated VMs we select data from a subset of the traces which the VMs appear to display complimentary utilisation. To generate enough samples to train and validate our models the selected traces were sampled at each time interval and the corresponding value was used as input into a Gaussian distribution in order to produce valid traces over the period of 10 days. This served to inject a level of random variability into the CPU demand of each VM.

TABLE I
Summary of Experimental Parameters

Parameter	Value
Learning Rate α	0.05
Momentum γ	0.01
Training Epochs	3000
Number of Layers	2
Neurons per Layer	50

C. Parameter Selection and Training

The selection of a good network topology for a ANN is often determined using trial and error. In general, the number of layers and neurons in the network depends on the complexity of the problem and the number of training examples available. The inputs to the model are the current and previous CPU values from the training data. In this work the architecture of the ANN is selected using a parameter sweep. The model is configured with two hidden layers with 50 neurons each which provided good performance for this problem. The network is also configured with one output neuron that corresponds to the network's prediction of future CPU demand in the next time step. The backpropagation algorithm with gradient descent was used to train the neural network models. Furthermore, an additional parameter known as momentum was used with the gradient descent algorithm. A common problem when training neural networks with gradient descent is that the algorithm can inadvertently get stuck in a local minimum. To overcome this we use momentum which takes into account the gradient in past time steps when updating the weights. The networks were trained over 3000 epochs and their accuracy was evaluated using an unseen test set. The models were re run 10 times to ensure reliability as ANNs can be sensitive to the initial values assigned to the weights in the network. Table I displays the parameters used for training the ANNs.

D. Error Metrics

We evaluate the accuracy of the trained models using the Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Squared Error (MSE) and Mean Absolute Percentage Error (MAPE) statistics. These statistics are defined as follows:

$$RMSE = \sqrt{\frac{\sum\limits_{t=1}^{n} (\hat{y}_t - y_t)^2}{n}} \tag{10}$$

$$MAE = \frac{1}{n}\sum_{t=1}^{n} |\hat{y}_t - y_t| \tag{11}$$

$$MSE = \frac{\sum_{t=1}^{n} (\hat{y}_t - y_t)^2}{n} \tag{12}$$

$$MAPE = \frac{100}{n}\sum_{t=1}^{n} \left| \frac{\hat{y}_t - y_t}{y_t} \right| \tag{13}$$

where $\hat{y}_t$ and y_t denote the forecasted and actual values respectively. While *n* is the number of values in the sequence.

E. Results

A comparison of the accuracy of each of the forecasting models is presented below in Table II. Based on the characteristics of the data the results show that the non-linear models perform best overall demonstrating their ability to model more complex underlying correlations with the capacity to generalize well to unseen data. In particular, the ANN with a sliding window of 50 emerged as the best performer overall. This model generated the smallest error across three out of the four performance metrics with a RMSE of 3.6696, MAE of 2.7003 and MSE of 13.4661. Our empirical evaluation found that by adjusting the size of the input sequence the neural network models could discover a more optimal mapping of inputs to output values. As shown in Fig.3 by expanding the sliding window from 2 to 50 values performance improves as the errors begin to reduce before discovering a global minimum. However, as the sliding window continues to expand the accuracy of the ANN begins to decline showing a significant increase in the error rate.

In terms of the more simplistic models such as MA and RW they performed least best. In particular, they demonstrate their inability to model the fundamental characteristics of the data. One reason for this may be due to when the fitted model attempts to extrapolate over more than a single time step it begins to flatline falling to the mean of the series resulting in significant errors. Overall the results of the experiments show that the ANN (Sliding window 50) has the capabilities to improve upon the performance of popular prediction methods such as RW, MA and ARIMA to predict CPU utilisation with a high degree of accuracy as shown in Fig.4.

TABLE II
TEST DATA PREDICTION ACCURACY

Algorithm	RMSE	MAE	MSE	MAPE
Random Walk	18.9169	15.6907	179.3378	22.0358
Moving Average	16.821	13.6824	145.766	18.4646
ARIMA	5.7475	4.4444	25.2992	8.9955
ANN (Sliding Window 2)	4.3501	3.3336	18.9236	13.6542
ANN (Sliding Window 10)	4.0845	3.0903	16.6833	8.6692
ANN (Sliding Window 20)	4.0285	3.0697	16.2290	12.8590
ANN (Sliding Window 30)	3.9175	2.9860	15.3466	12.6014
ANN (Sliding Window 40)	3.8671	2.9582	14.9543	12.3946
ANN (Sliding Window 50)	3.6696	2.7003	13.4661	10.0038
ANN (Sliding Window 60)	4.1517	2.8221	17.2369	10.6164
ANN (Sliding Window 70)	4.3286	2.9674	18.7365	10.3189
ANN (Sliding Window 80)	4.9667	3.0802	24.6678	11.0306

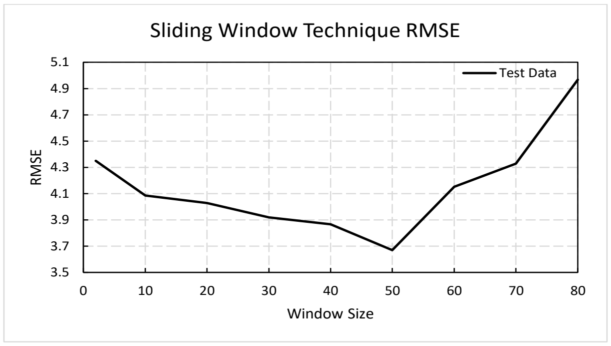

Fig. 3. RMSE of all sliding windows

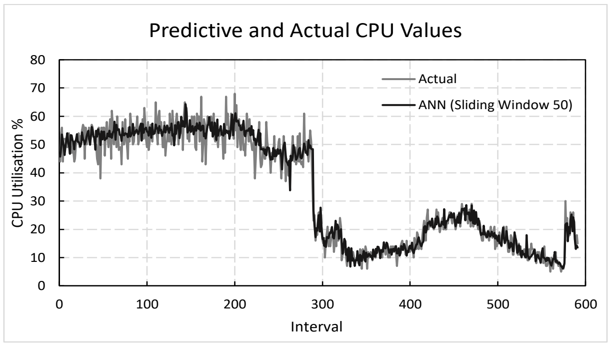

Fig. 4. Predictive performance of ANN (Sliding window 50)

In particular, our evaluation suggests that ANNs are more superior than ARIMA models which is consistent with the findings presented by Camara et al. [29].

V. CLOUD EXPERIMENTAL DETAILS AND RESULTS

In this section we discuss the experimental analysis and cloud simulation results using the proposed PACPA algorithm. To evaluate the efficiency of our PACPA algorithm we use the CloudSim toolkit [16] which is a widely used simulation framework for conducting cloud computing research. In CloudSim we model the problem using a data center consisting of 800 physical machines. In particular, the setup consists of two types of physical hosts modelled as HP ProLiant ML110G4(Intel Xeon 3075, 2 cores 1860 MHz, 4GB) and HP ProLiant ML110G5(Intel Xeon 3075, 2 cores 2660 MHz, 4GB). Furthermore, we consider four types of VMs with configurations similar to those offered by Amazon web services. These consist of High-CPU Medium Instances (2500 MIPS, 0.85 GB), Extra Large Instances (2000 MIPS, 3.75 GB), Small Instances (1000 MIPS, 1.7 GB) and also Micro Instances (500 MIPS, 613 MB).

A. Comparative Placement Algorithms

We compare the proposed PACPA approach to some of the most widely used placement heuristics which are known to deliver good results. These heuristics include First Fit (FF) [28] which places a VM on the first available PM with sufficient capacity. Best Fit (BF) [8] selects the PM with the minimum remaining resource that can adequately fulfill the resources requested by the VM. We also compare our work to the Power Aware Best Fit Decreasing (PABFD) algorithm proposed by Beloglazov et al. in one of their most highly cited works [5]. PABFD considers the heterogeneity of cloud resources by selecting the most energy efficient hosts first in order to allocate VMs.

B. Cloud Performance Metrics

The main goal of our proposed PACPA algorithm is while considering the relationship between VMs and their future resource demand generate a placement policy that reduces energy consumption while also improving the ability of service providers to delivery performance guarantees. As a result the key cloud performance metrics used to evaluate the effectiveness of the proposed algorithm are:

- **Energy Consumption**: energy is defined as the total energy consumed by the data centers computational resources caused by application workloads. In our work the energy consumption metric is computed by CloudSim where power consumption of a PM is represented by its CPU utilisation and reported as energy consumption over time (kWh) [16].
- **SLA Violations**: the ability of cloud providers to deliver SLA is critical. A service violation is measured as the performance degradation experienced by each VM due to both hosts becoming overloaded and also the number of VM migrations incurred by the cloud system.

C. Simulation Results

We evaluate the performance of the proposed PACPA algorithm over a 10 day workload. As shown in Fig.5, PACPA generates an improvement in energy consumption resulting in a reduction of 18% at best. Unlike the other approaches PACPA leverages the predictive capabilities of an ANN providing the algorithm with the ability to reason over the impact of dynamically changing CPU resource demands in order to inform its placement strategy.

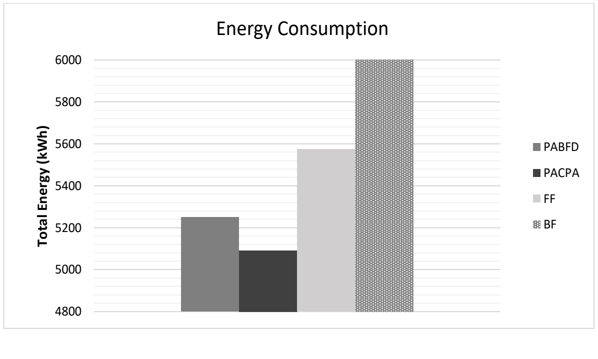

Fig. 5. Energy for PABFD, PACPA, FF and BF algorithms

Overall, the forecasting component of our PACPA algorithm aids in the prevention of redundant decisions by improving the likelihood of placement decisions remaining advantageous in future time steps. Furthermore, PACPA also attempts to foster placement strategies based on co-located resource consumption by combining compatible VMs in order to improve resource utilisation, which is a key factor in causing excessively large energy rates in data centers today. The PABFD algorithm closely followed demonstrating its ability to effectively reduce energy consumption in the data center. Our PACPA algorithm outperformed PABFD by 3% equating to a savings of 183 kWh over 10 days. The BF algorithm results in the highest amount of energy of 6185 kWh.

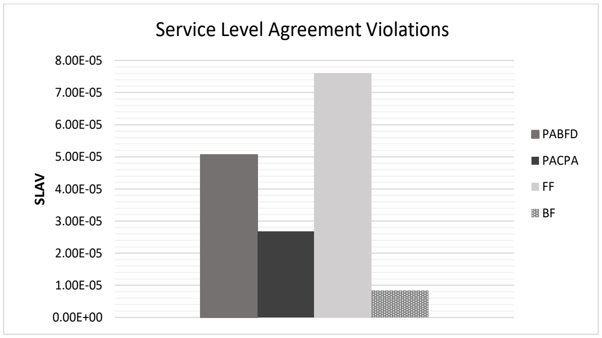

Fig. 6. SLA violations for PABFD, PACPA, FF and BF algorithms

The performance of each algorithm in terms of its ability to deliver reliable cloud services is illustrated in Fig.6. An important aspect in achieving greater energy efficiency by optimizing VM placement strategies is managing the fundamental tradeoff between energy and performance. The results show an interesting observation in that the BF algorithm produces the least amount of SLA violations followed by our proposed PACPA algorithm. However, for the BF algorithm this is achieved at a large cost in terms of high energy consumption as shown in Fig.5, thus demonstrating its inability to efficiently manage this tradeoff, a similar observation can also be seen with the PABFD algorithm. In contrast, PACPA strikes a more optimal balance between both energy and SLA violations. The SLA violations generated are relatively low with over a 47% improvement on the PABFD and FF algorithms, while resulting in the least amount of energy consumption overall. Furthermore, a paired t-test confirmed that the results achieved by the PACPA algorithm are also statistically significant. The test resulted in p-values less that 0.0001 with a 95% confidence interval of -16.494054517 to -14.931428903 (energy) and -0.00000271 to -0.00000208 (SLA violations). These results indicate the improved efficiency achievable through the implementation of the proposed PACPA algorithm. In particular, we demonstrate the benefits of a more proactive joint provisioning placement algorithm to support the movement of VMs between hosts in the data center.

VI. Conclusion

Energy related costs and environmental sustainability of modern data centers have become major concerns for cloud computing practitioners and the development of next generation data centers. In this paper we present PACPA, a novel predictive anti-correlated VM placement algorithm. Unlike current approaches, we provide a comparative study of the performance of popular ML methodologies and select the best predictive algorithm to incorporate into our proposed PACPA algorithm. In our work we consider both current and predicted future resource demands to improve our placement strategy. In addition, we also consider the relationship between migrating VMs as an important factor in optimizing the usage of available resources. Our experimental results show that PACPA accurately predicts future CPU resource utilisation which is a key challenge for VM placement algorithms. PACPA is also capable of achieving improved energy efficiency of at best 18%. Furthermore, reducing service violations by over 47% compared to some widely known VM placement algorithms, thus, enhancing the ability of practitioners to achieve signicant improvements in the quality of the service provided.

Acknowledgment

The authors would like to thank the Irish Research Council for funding this work.

References

[1] R. Shaw, E. Howley, and E. Barrett, ”Predicting the Available Bandwidth on Intra Cloud Network Links for Deadline Constrained Workflow Scheduling in Public Clouds,” Proc. International Conference on Service-Oriented Computing, Springer, Cham, Nov. 2017, pp. 221-228.

[2] R. Shaw, E. Howley, and E. Barrett, ”An Advanced Reinforcement Learning Approach for Energy-Aware Virtual Machine Consolidation in Cloud Data Centers,” Proc. 12th International Conference for Internet Technology and Secured Transactions, IEEE, Dec. 2017, pp. 61-66.

[3] J. Whitney, and P. Delforge, ”Scaling up energy efficiency across the data center industry:evaluating key drivers and barriers,” technical report, Natural Resources Defense Council, 2014.

[4] Y.C. Lee, and A.Y. Zomaya, ”Energy efficient utilisation of resources in cloud computing systems,” The Journal of Supercomputing, vol. 60, May. 2012, pp. 268-280.

[5] A. Beloglazov, J. Abawajy, and R. Buyya, ”Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing,” Future generation computer systems, vol. 28, May. 2012, pp. 755-768.

[6] N. Bobroff, A. Kochut, and K. Beaty, ”Dynamic placement of virtual machines for managing sla violations,” Proc. IFIP/IEEE International Symposium on Integrated Network Management (IM), May. 2007, pp. 119-128.

[7] T.H. Nguyen, M. Di Francesco, and A. Yla-Jaaski, ”Virtual machine consolidation with multiple usage prediction for energy-efficient cloud data centers,” IEEE Transactions on Services Computing, Jan. 2017.

[8] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, N.T. Hieu, and H. Tenhunen, ”Energy-aware vm consolidation in cloud data centers using utilisation prediction model,” IEEE Transactions on Cloud Computing, Oct. 2016.

[9] M. Haehnel, J. Martinovic, G. Scheithauer, A. Fischer, A. Schill, and W. Dargie, ”Extending the Cutting Stock Problem for Consolidating Services with Stochastic Workloads,” IEEE Transactions on Parallel and Distributed Systems, Apr. 2018.

[10] J. Son, A.V. Dastjerdi, R.N. Calheiros, and R. Buyya, ”SLA-aware and energy-efficient dynamic overbooking in SDN-based cloud data centers,” IEEE Transactions on Sustainable Computing, vol. 2, Apr. 2017, pp. 76-89.

[11] X. Meng, C. Isci, J. Kephart, L. Zhang, E. Bouillet, and D. Pendarakis, ”Efficient resource provisioning in compute clouds via vm multiplexing,” Proc. the 7th International Conference on Autonomic Computing, ACM, Jun. 2010, pp. 11-20.

[12] T. Chen, Y. Zhu, X. Gao, L. Kong, G. Chen, and Y. Wang, ”Improving Resource utilisation via Virtual Machine Placement in Data Center Networks,” Mobile Networks and Applications, vol. 23, Apr. 2017, pp. 227-238, doi:10.1007/s11036-017-0925-7.

[13] A. Verma, P. Ahuja, and A. Neogi, ”pMapper: power and migration cost aware application placement in virtualized systems,” Proc. the 9th ACM/IFIP/USENIX International Conference on Middleware, Dec. 2008, pp. 243-264.

[14] M. Cardosa, M.R. Korupolu, and A. Singh, ”Shares and utilities based power consolidation in virtualized host environments,” Proc. the 11th IFIP/IEEE International Conference on Symposium on Integrated Network Management, IEEE Press, Jun. 2008, pp. 327-334.

[15] N. Khalilzad, H.R. Faragardi, and T. Nolte, ”Towards energy-aware placement of real-time virtual machines in a cloud data center,” Proc. High Performance Computing and Communications (HPCC), 2015 IEEE 7th International Symposium on Cyberspace Safety and Security (CSS), 2015 IEEE 12th International Conference on Embedded Software and Systems (ICESS), 2015 IEEE 17th International Conference on, Aug. 2015, pp. 1657-1662.

[16] 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. vol. 24, Sep. 2012, pp. 1397-1420.

[17] J. Wang, C. Huang, K. He, X. Wang, X. Chen, and K. Qin, ”An energy-aware resource allocation heuristics for VM scheduling in cloud,” Proc. High Performance Computing and Communications & IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC), Nov. 2013, pp. 587-594.

[18] J. Kim, M. Ruggiero, D. Atienza, and M. Lederberger, ”Correlation-aware virtual machine allocation for energy-efficient datacenters,” Proc. Conference on Design, Automation and Test in Europe, EDA Consortium, Mar. 2013, pp. 1345-1350.

[19] X. Fu, and C. Zhou, ”Predicted Affinity Based Virtual Machine Placement in Cloud Computing Environments,” IEEE Transactions on Cloud Computing, vol. 9, Aug. 2017, pp-1-1, doi:10.1109/TCC.2017.2737624.

[20] G.P. Zhang, ”Time series forecasting using a hybrid ARIMA and neural network model,” Neurocomputing, vol. 50, Jan. 2003, pp. 159-175.

[21] E. Barrett, E. Howley, and J. Duggan, ”Applying reinforcement learning towards automating resource allocation and application scalability in the cloud,” Concurrency and Computation: Practice and Experience, vol. 25, Aug. 2013, pp. 1656-1674.

[22] E. Barrett, E. Howley, and J. Duggan, ”A learning architecture for scheduling workflow applications in the cloud,” Proc. Ninth IEEE European Conference on Web Services (ECOWS), IEEE, Sep. 2014, pp. 83-90.

[23] M. Duggan, K. Mason, J. Duggan, E. Howley, and E. Barrett, ”Predicting Host CPU utilisation in Cloud Computing using Recurrent Neural Networks,” Proc. 12th International Conference for Internet Technology and Secured Transactions, IEEE, Dec. 2017, pp. 67-72.

[24] R.j. Hyndman, and G. Athanasopoulos, ”Forecasting: principles and practice,” OTexts, 2013, Melbourne, Australia.

[25] K. Mason, J. Duggan, and E. Howley, ”Evolving multi-objective neural networks using differential evolution for dynamic economic emission dispatch,” Proc. the Genetic and Evolutionary Computation Conference Companion, ACM, Jul. 2017, pp. 1287-1294.

[26] S.S. Haykin, ”Neural networks and learning machines,” vol. 3, Pearson, Nov. 2009, Upper Saddle River, NJ, USA.

[27] T.G. Dietterich, ”Machine learning for sequential data: A review,” Proc. Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR), Springer, Aug. 2002, pp. 15-30.

[28] G. Keller, M. Tighe, H. Lutfiyya, and M. Bauer, ”An analysis of first fit heuristics for the virtual machine relocation problem,” Proc. Network and service management (cnsm), 2012 8th international conference and 2012 workshop on systems virtualiztion management (svm), IEEE, Oct. 2012, pp. 406-413.

[29] A. Camara, W. Feixing, and L. Xiuqin, ”Energy consumption forecasting using seasonal ARIMA with artificial neural networks models,” International Journal of Business and Management, vol. 11, Apr. 2016, pp. 231-243.

[30] G.T. Hicham, E. Chaker, and E. Lotfi, ”Comparative Study of Neural Networks Algorithms for Cloud Computing CPU Scheduling,” International Journal of Electrical and Computer Engineering (IJECE), vol. 7, Dec. 2017, pp. 3570-3577.

[31] F.A. Gers, D. Eck, and J. Schmidhuber, ”Applying LSTM to time series predictable through time-window approaches,” Neural Nets WIRN Vietri-01, 2002, pp. 193-200, Springer, London.