Time Series Forecasting of Cloud Data Center Workloads for Dynamic Resource Provisioning

Carlos Vazquez, Ram Krishnan*, and Eugene John
The University of Texas at San Antonio
One UTSA Circle, San Antonio, TX 78249, USA
yof042@my.utsa.edu, Ram.Krishnan@utsa.edu, Eugene.John@utsa.edu

Abstract

Cloud computing offers on-demand, elastic resource provisioning that allows an enterprise to provide services to their customers at an acceptable quality while consuming only the requisite computing resources as a utility. Since cloud computing resources scale elastically, utilizing cloud computing reduces the risk of over-provisioning, wasting resources during non-peak hours, and reduces the risk of under-provisioning, missing potential customers. By using an automated resource scaling algorithm, a system implemented using cloud services can fully exploit the benefits of on-demand elasticity. A simple reactive scaling algorithm, resource scaling is triggered after some monitored metric has crossed a threshold, suffers from the fact that cloud computing workloads can varying widely over time and a scalable application needs time to perform the triggered scaling action. Instead, resources can be proactively requested by forecasting future resource demand values based on demand history. Obtaining accurate prediction results is crucial to the efficient operation of an automated resource scaling algorithm. In this work, several forecasting models are evaluated for their applicability in forecasting cloud computing workloads. These forecasting methods were compared for their ability to forecast real cloud computing workloads including Google cluster data and Intel Netbatch logs. Two tests are performed to evaluate the accuracy of each forecasting model: out-of-sample forecasting and rolling forecast origin cross-validation.

Keywords: Cloud Computing, Workload Forecasting, Forecasting Models.

1 Introduction

The increase in popularity of cloud computing in recent years is driven by the advantages offered by the dynamically scalable, pay-as-you-go model. This enables organizations to focus on providing services to their customers while consuming the requisite computing resources as a utility. By eliminating the need for on-premises equipment, organizations avoid large capital expenses and instead focus resources towards faster deployment. The pay-as-you-go model allows an organization to grow naturally with customer demand. Since cloud computing resources scale elastically, utilizing cloud computing reduces the risk of over provisioning, wasting resources during non-peak hours, and reduce the risk of under provisioning, missing potential customers [1]. Success stories of start-ups like Instagram, which built-up a user base of over 150 million users in less than four years using only public cloud solutions [2], exemplify the potential for fast growth that utilizing cloud computing can provide. Cloud computing is a large-scale, distributed computing paradigm which is driven by economies of scale. Providers of cloud computing offer abstracted, virtualized, dynamically scalable, and managed resources on demand to external customers over the Internet [3]. These resources include compute, storage and networking. Cloud computing providers benefit from economies of scale in that they assemble massive datacenters

*Corresponding author: Department of Electrical and Computer Engineering, The University of Texas at San Antonio, One UTSA Circle, San Antonio, TX 78249, USA, Tel: +1-210-458-6293 operating tens of thousands of servers which service a wide customer base. Large-scale operation more effectively absorbs operational costs through the benefits of increasing the utilization of equipment, bulk discounts on purchased equipment, and reducing the cost of cooling and powering equipment [4]. The demand for large-scale computing resources continues to grow as Internet users generate larger sets of data to be processed.

Virtualization is a fundamental component of cloud computing, allowing for pooling and dynamically allocating hardware resources. This technology is used to increase the level of indirection and thus provide the flexibility needed in a scalable cloud computing environment. Since cloud computing resources scale elastically, utilizing cloud computing reduces the risk of over provisioning, wasting resources during non-peak hours, and reduce the risk of under provisioning, missing potential customers [1]. Cloud service users can scale resources dynamically by detecting the need for more resources, acquiring the resources, and distribute the load among the resources with minimal intervention. By using dynamic resource scaling, a cloud applications can best exploit the benefits of on-demand elasticity. A dynamically scalable application needs to make accurate decisions when scaling since reconfiguring a system comes with an overhead. As has been suggested in the related works, a proactive approach to triggering resource requests can be taken by forecasting future resource demand values based on demand history. With accurate predictions, an application can preemptively scale to the correct level and thus alleviate some or all of the scaling overhead. Time series analysis can be used to forecast the future resource demand values. Since cloud computing workloads tend to trace the daily, weekly, monthly, etc. of human web browsing patterns, it is expected that time series forecasting methods could reliably predict resource demand. The goal of this work is to assess the applicability of several time series forecasting models on real cloud workloads to support the development of dynamic resource allocation systems. A number of factors which must be considered in producing accurate forecasts are explored including the frequency at which data is collected, the amount of data used to make a prediction, the model used to make predictions, and the number future values are predicted. The following forecasting models are evaluated for their ability to produce accurate predictions of real cloud workloads:

- First-order autoregressive model (AR(1))
- First-order moving average model (MA(1))
- Simple exponential smoothing
- Double exponential smoothing
- ETS method
- Automated ARIMA method
- Neural network autoregression method

The rest of this work is structured as follows: Section 2 background material on topics including general cloud computing, virtualization, service level agreements, and time series analysis. Section 3 contains a review of related works. Section 4 explains the experimental design including test methods, tested work load traces, the experimental setup used. Section 5 presents results of the tests performed. Section 6 presents concluding remarks.

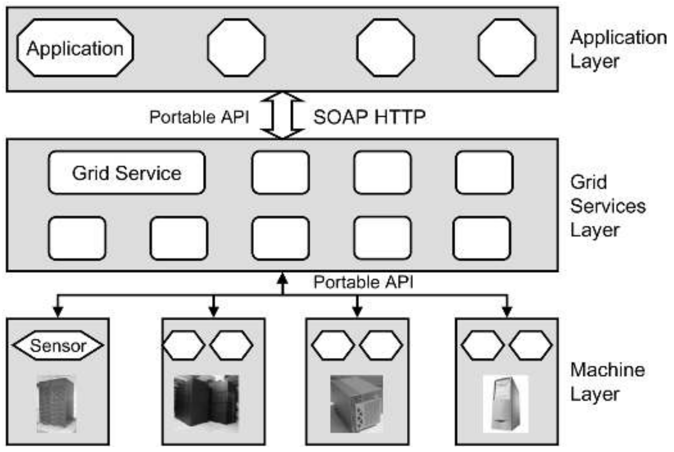


Figure 1: Service layers of a grid computing system [5]

2 Background

2.1 Cloud Computing

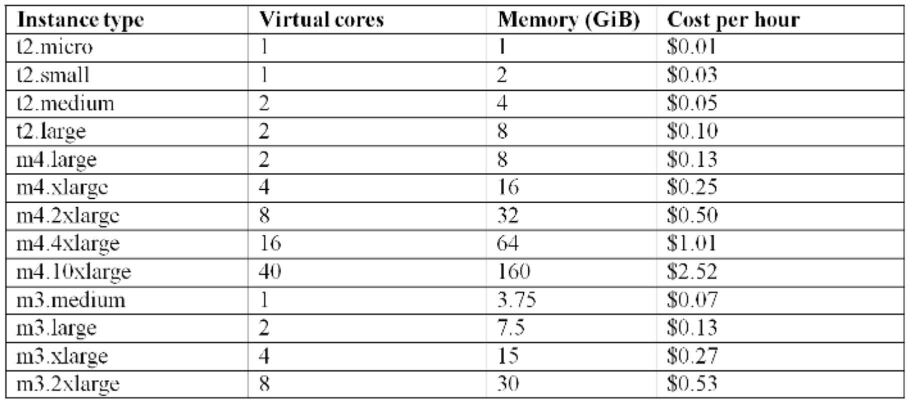
The origins of cloud computing can be traced back to grid computing that was largely useful in high-performance computing. Access to large amounts of widely geographically distributed computing units federated through standardized protocols are offered to grid users. Figure 1 shows the layers of a grid computing system where a user interfaces with grid services through an application layer. The grid services layer coordinates the distributed groups of resources in the machine layer to service the requests issued by the user through the application layer. Heterogeneous computing systems are unified into a single pool of computing resources that is abstracted such that a user can interface with a high-performance uniform computing service without worrying about the complexities involved in coordinating the components of such a large system [3].

The name grid computing is intended to convey the paradigm's ability to offer computing resources as a utility similar to power from the power grid. Previously, access to high-performance computing systems was only available to large enterprises with the capital and time necessary to assemble a datacenter with hundreds of servers. With this new technology, researchers could work on a class of scientific problems requiring large-scale, parallel computing resources that distributed grid computing systems handle exceptionally well. Although grid computing did not achieve long lasting popularity, the concept of computing as a utility, giving users access to nearly unlimited computing power on demand, is the driving force behind the wide spread adoption of cloud computing. Cloud computing is a large-scale, distributed computing paradigm which is driven by economies of scale. Providers of cloud computing offer abstracted, virtualized, dynamically scalable, and managed resources on demand to external customers over the Internet [3]. Access to storage, computing, or networking are offered in abstracted units that can be requested on-demand as the customer needs via a pay-as-you-go business model.

2.2 Essential Characteristics

The essential characteristics of cloud computing as described in the NIST definition of cloud computing [7] are:

- On-demand self-service: The ability to provide computing capabilities as needed automatically. Computing capabilities may include server capacity, network storage, access to applications, or



Instance type	Virtual cores	Memory (GiB)	Cost per hour
t2.micro	1	1	$0.01
t2.small	1	2	$0.03
t2.medium	2	4	$0.05
t2.large	2	8	$0.10
m4.large	2	8	$0.13
m4.xlarge	4	16	$0.25
m4.2xlarge	8	32	$0.50
m4.4xlarge	16	64	$1.01
m4.10xlarge	40	160	$2.52
m3.medium	1	3.75	$0.07
m3.large	2	7.5	$0.13
m3.xlarge	4	15	$0.27
m3.2xlarge	8	30	$0.53

Figure 2: Price and characteristics Amazon EC2 instances with a Linux OS in June 2015 [6]

other services.

- Broad networks access: Cloud services are available over the network, usually through the Internet, and accessed through standard mechanisms including desktop computers, laptops, and mobile devices.

- Resource pooling: Physical and virtual resources are dynamically assigned to serve multiple consumers using a multi-tenant model. The server from which the computing resources are provided to a user may change as consumer demand fluctuates. A goal for a cloud provider is to prevent these fluctuations from imposing performance variations on the requested computing resources.

- Rapid elasticity: Capabilities are elastically provisioned and released quickly without perceived bound. This allows users to scale requested resources appropriately as consumer demand changes.

- Measured service: Cloud services automatically control resource use by leveraging appropriate metering capabilities. Services are typically provided in a pay-per-use model that naturally incentivizes the optimal use of resources.

2.3 Service Layers

The NIST definition of cloud computing [7] categorizes the services that providers offer into three service models: infrastructure-as-a-service (IaaS), platform-as-a-service (PaaS), or a software-as-a-service (SaaS).

- An IaaS provides access to instances of unified resources including computing, storage, and networking. Providers offer flexible computing resources for a usage-based price. These resources are distributed as instances on demand which are treated like physical hardware by the user. This service model offers the most flexibility the user is allowed to run arbitrary software. This also means that the user is left with the responsibility for demanding and initializing new instances when scaling is required.

- A PaaS provides many of the same resources as an IaaS but through an integrated environment which reduces the development burden of using the resources. This comes at the expense of restricting the features available to a user and potentially being locked in to a particular platform. PaaS providers offer a variety of computing and storage resources in a more constrained environment that can be accessed through APIs. By using a pre-built and maintained infrastructure, a

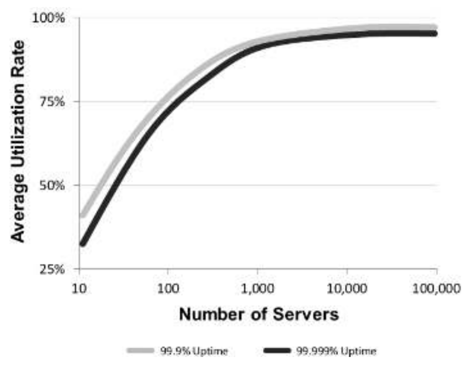


Figure 3: Average utilization rate versus the size of a cloud data center [8]

PaaS user can focus on the higher-level end-user experience. Many application specific tools are pre-built and available to users such as web hosting, data management, business analytics, etc.

- SaaS, such as e-mail and Google Docs, are special-purpose software services that are hosted in a cloud infrastructure and accessed remotely by the end user through a thin client or program interface. They are often built using PaaS and IaaS tools, but their implementation details are hidden from the end-user. The end-user has narrow access to a well-defined set of services that have limited configurability.

2.4 Benefits of Economy of Scale

Cloud computing providers benefit from economies of scale in that they assemble massive datacenters operating hundreds of thousands of servers which service a wide customer base. Large-scale operation more effectively absorbs operational costs through the benefits of increasing the utilization of equipment as resources are made available to the public [4]. Having a larger pool of customers also reduces variability in the aggregate data center workload, allowing for higher resource utilization given a fixed required performance. Assembling and maintaining a cloud data center with 100,000s of servers allows for purchasing equipment at a large volume discount, reducing capital expenses. Since energy consumption accounts for a significant portion of operational costs, engineering efforts to reduce the cost of cooling and powering equipment have allowed cloud data centers to run much more efficiently than smaller-scale operations.

2.5 Virtualization

Virtualization is a fundamental component of cloud computing, allowing for pooling and dynamically allocating hardware resources. A server in a datacenter acting as a host machine is installed with a hypervisor which can simultaneously run instances of virtual machines or guest machines. These virtual machines are operating system instances managed by a separate controlling computer which loads them into respective host machines. With the controlling computer managing the computing resources of many servers, a cloud computing provider thus unifies the datacenter's resources into an encapsulated pool which can be allocated and released according to user demand.

The hardware resources of physical servers are distributed among multiple virtual machines running logically isolated operating system instances. This technology is used to increase the level of indirection and thus provide the flexibility needed in a scalable cloud computing environment. Given the configurability of virtual machine instances, the resources to host applications with diverse storage and compute

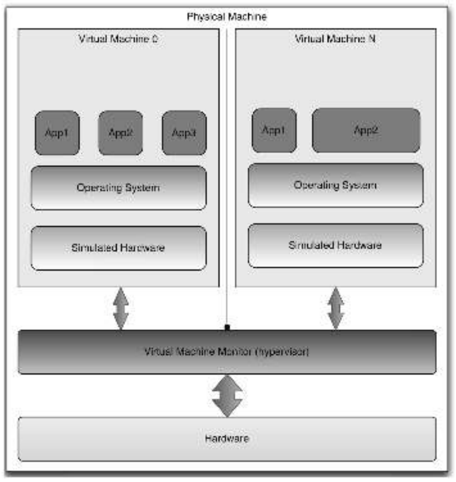


Figure 4: Virtual machine abstraction [9]

requirements can be allocated instance for as long as they are needed and released back into the resource pool [3]. By partitioning resources in this way, servers can achieve higher utilization.

2.6 Cloud Workloads

The demand for large-scale computing resources continues to grow as Internet users generate larger sets of data to be processed. The following workloads are commonly implemented with cloud services [10]:

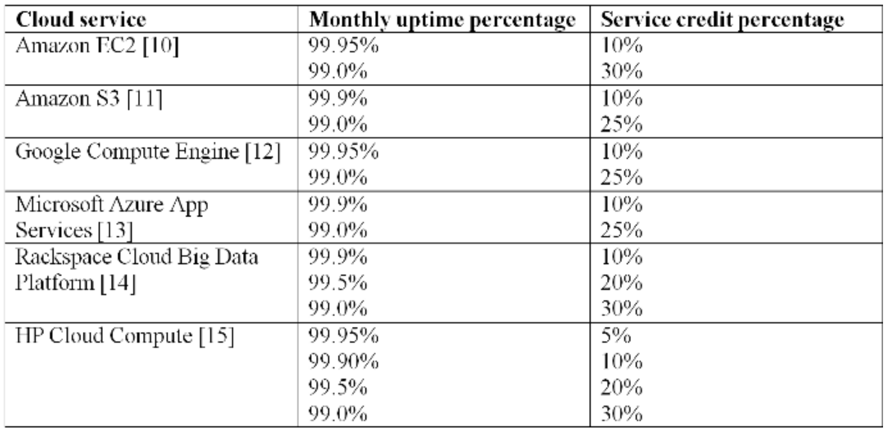
- Data serving: Data stores designed to serve as the backing store for large-scale web applications
- MapReduce [11]: A parallel processing framework used for many cloud-based batch-processing projects.
- Media streaming: Used to packetize and transmit media files in various bit-rates and formats.
- SAT solver: Used to calculate solutions for Boolean satisfiability problems.
- Web frontend: Many instances of stateless web servers are used to host services in scalable, fault-tolerant way.
- Web search: Perform search queries against web indexes.

2.7 Service Level Agreements

As more computing is being done through web-based services and more services become available, cloud computing providers must provide customers of web services a high level of performance consistency to stay competitive. Cloud service providers offer a service level agreement (SLA) to customers to provide a certain quality of service (QoS) or provide compensation, often in the form of a percentage refund of payment. An SLA is composed of service level objectives (SLO) that describe the expected quantitative performance of a service or feature of a service.

A list of common performance SLOs includes [12]:

- Availability: The property of being accessible upon demand by an authorized entity expressed in terms of uptime, percentage of successful requests, or percentage of timely service requests. A list of availability SLOs can be found in figure 2.



Cloud service	Monthly uptime percentage	Service credit percentage
Amazon EC2 [10]	99.95% 99.0%	10% 30%
Amazon S3 [11]	99.9% 99.0%	10% 25%
Google Compute Engine [12]	99.95% 99.0%	10% 25%
Microsoft Azure App Services [13]	99.9% 99.0%	10% 25%
Rackspace Cloud Big Data Platform [14]	99.9% 99.5% 99.0%	10% 20% 30%
HP Cloud Compute [15]	99.95% 99.90% 99.5% 99.0%	5% 10% 20% 30%

Figure 5: Availability SLO offered by popular cloud service providers June 2015

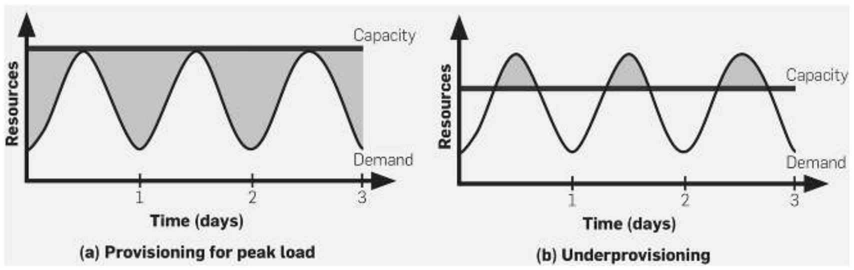


Figure 6: Provisioning resources for a variable workload [1]

- Response time: The time interval between a customer-initiated event and the response to that event. For many applications, such as gaming and SaaS, response time can have a large impact on an end-user's satisfaction with a service.
- Capacity: Maximum amount of some property e.g. data storage, number of CPUs, service throughput, etc. of a cloud service.
- Capability indicators: Guarantee to provide specific functionality that might be essential to cloud service customer. For example, a customer may need a service to be compatible with an external system.

2.8 Resource Allocation

On demand, elastic resource provisioning enables cloud service users to focus on providing services to their customers while consuming the requisite computing resources as a utility. The pay-as-you-go model allows an organization to grow naturally with customer demand. Since cloud computing resources scale elastically, utilizing cloud computing reduces the risk of over provisioning, wasting resources during non-peak hours, and reduce the risk of under provisioning, missing potential customers [1].

Figure 6 illustrates the effects of choosing a static resource capacity when a server is presented with a variable amount of demand. The left figure shows the case where the resource capacity is selected to satisfy the peak load. Although the server is never overloaded, the resource capacity is usually under-utilized as shown in the shaded area. The right figure shows the case where the resource capacity is below the peak load. The resources are more fully utilized in this case but the shaded area above the capacity limit corresponds to customer demand that is not serviced. Instead of selecting a static resource

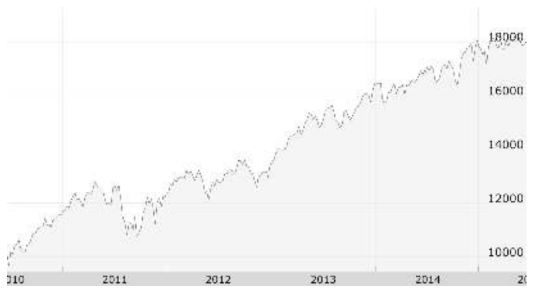


Figure 7: Time series of the Dow Jones Industrial average over 10 year [17]

capacity, cloud service users can scale resources dynamically by detecting the need for more resources, acquiring the resources, and distribute the load among the resources with minimal intervention.

By using dynamic resource scaling, a cloud applications can best exploit the benefits of on-demand elasticity. Scaling a cloud application typically involve one or more of the following concepts:

- Horizontal scaling [13]: Scaling by adding or removing an instance of an application. The application runs on a virtual machine with identical configuration.
- Vertical scaling [13]: Scaling by changing the amount of resources allocated to a virtual machine that is already running.
- Load balancing [14]: Distribute requests to a service evenly among the available virtual machine instances.
- Threshold triggering [15]: Establish a threshold value for a given metric that when crossed will trigger a scaling event.

A dynamically scalable application needs to make accurate decisions when scaling since reconfiguring a system comes with an overhead. Initialization time of a virtual machine instance and migrating virtual machine instances to under-utilized servers can affect the cost of a scaling decision. Making the wrong decision when scaling could lead to over provisioning, wasting resources, or under-provisioning, potential causing SLA violations.

A simple reactive approach to dynamic resource scaling is to monitor a metric, such as CPU utilization, until a threshold is passed. A scaling event is then initiated. This scheme suffers from the fact that cloud computing workloads can varying widely over time and a cloud application needs time to perform the triggered scaling event [16]. An alternative is to use a proactive approach by forecasting future resource demand values based on demand history. With accurate predictions, an application can preemptively scale to the correct level and thus alleviate some or all of the scaling overhead.

2.9 Time Series Forecasting

Time series analysis can be used to forecast the future resource demand values. A time series is a set of observations ordered with respect to time, typically in fixed-length intervals [18]. To forecast future values of a time series, a prediction model is first trained with the time series. Parameters within the model are fitted to extract patterns in the data. With the trained model, a probability distributions for a future value can be calculated. The expected value of the distribution is used as the forecasted value. A series of multiple predicted values are referred to as a forecast horizon.

By aggregating resource demand over a sampling period, a resource demand time series can be extracted and forecasted. Since cloud computing workloads tend to trace the daily, weekly, monthly, etc. of human web browsing patterns, it is expected that time series forecasting methods could reliably predict resource demand.

2.10 Forecasting Methods

The list of time series forecasting methods considered in this work includes:

- Autoregressive (AR(1)) model: A first order autoregressive model is a linear regression of the current value of the series against the prior value of the series [19]. This is expressed by the equation:

  $$X_t = c + \theta X_{t-1} + \varepsilon_t, -1 < \theta < 1 \qquad (2.1)$$

  c = constant
  θ = a model parameter
  ε_t = white noise

- Moving average (MA(1)) model: The first order moving average model can be thought of as a linear regression of the current value of the series against the white noise or random shocks of one or more prior values of the series [19]. A moving average process is expressed by the equation:

  $$X_t = \mu + \varepsilon_t + \theta\varepsilon_{t-1}, -1 < \theta < 1 \qquad (2.2)$$

  μ = mean
  θ = a model parameter
  ε_t = white noise

- Simple exponential smoothing: The equation for simple exponential smoothing is:

  $$\widehat{X}_{t+1} = \alpha X_t + (1-\alpha)\widehat{X}_t \qquad (2.3)$$

  α=smoothing constant

  The forecasted value time t+1 is a weighted combination of the observed value at time t and the forecasted value at time t [20].

- Double exponential smoothing: The equations for double exponential smoothing are given as [21]:

  $$S_t = \alpha X_t + (1-\alpha)(S_{t-1} + b_{t-1}), 0 \le \alpha \le 1 \qquad (2.5)$$

  $$b_t = \gamma(S_t - S_{t-1}) + (1-\gamma)b_{t-1}, 0 \le \gamma \le 1 \qquad (2.6)$$

  α, γ = smoothing constants

  $$X_{t+1} = S_t + b_t \qquad (2.7)$$

  The first smoothing equation (2.5) adjusts for the trend of the previous period, b_t. The second smoothing equation (2.6) then updates the trend, which is expressed as the difference between the last two values. The third equation (2.7) is used to forecast values.

- ETS: The ETS (error, trend, seasonal) [22] method automatically selects an exponential smoothing model. These models, referred to as state space models, are composed of the time series data and a description of how the state transitions over time in terms of level, trend, and seasonal components. Multiple models are evaluated for their goodness of fit to the data using AIC [23] and the best fit model is selected.

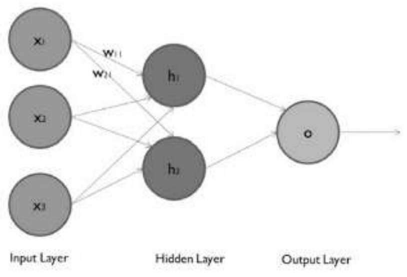


Figure 8: Three layer neural network [27]

- Automated ARIMA: ARIMA models are composed of autoregressive (AR(p)), integrated (I(q)), and moving average (MA(d)) components. The parameters p, q, and d specify the order of each model. An automated ARIMA algorithm [24] discovers appropriate values for p, q, and d by:

  1. The number of differences d is determined using repeated KPSS [25] tests.
  2. The values of p and q are then chosen by minimizing the AICc [23] after differencing the data d times.

     (a) The best model (with smallest AICc) is selected from the following four: ARIMA(2,d,2), ARIMA(0,d,0), ARIMA(1,d,0), ARIMA(0,d,1). If d=0 then the constant c is included; if d=1 then the constant c is set to zero. c affects the long-term forecasts of the model. This is called the “current model”.

     (b) Variations on the current model are considered by varying p and/or q from the current model by ± 1 and including or excluding c from the current model. The best model considered so far becomes the new current model.

     (c) Repeat Step 2(b) until no lower AICc can be found.

- Neural network autoregression: A neural network consists of connected “neurons” that take an input, and produce one or more outputs depending on the number of neurons it is connected to. The neurons are grouped into layers and connected such that the outputs of one layer feed into the inputs of a smaller layer. The connections between each neuron has an associated weight value. At each layer, input values are combined by a node using some function to produce an output that is propagated to the next layer until the output layer is reached. Parameters in the aggregation functions and interconnection weights are learned during a training phase. The neural network autoregressive model [26], can forecast time series values by using a fully connected, three-layer neural network.

2.11 Prediction Accuracy Methods

Several metrics are available to evaluate the ability of a forecasting model to produce accurate predictions from time series data. Given a time series of observed data, , and a time series of predicted values from X, $\widehat{X}$, both of length n, the following accuracy metrics are defined:

- Root mean square error (RMSE) [28]

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(\widehat{X}_t - X_t)^2} \qquad (2.8)$$

- Mean absolute error (MAE) [18]

$$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n}|\widehat{X}_t - X_t| \tag{2.9}$$

- Mean absolute percentage error (MAPE) [18]

$$\text{MAPE} = \frac{1}{n}\sum_{t=1}^{n}\frac{|\widehat{X}_t - X_t|}{X_t} \tag{2.10}$$

- Mean absolute scaled error (MASE) [29]

$$\text{MASE} = \frac{1}{n}\sum_{t=1}^{n}\frac{|\widehat{X}_t - X_t|}{\alpha}, \alpha = \frac{1}{n-1}\sum_{t=2}^{n}|X_t - X_{t-1}| \tag{2.11}$$

3 Related Work

In this section, works related to time series analysis of cloud computing workloads for use in dynamic provisioning algorithms are reviewed. Systems that provide dynamic resource allocation offer cloud service users the ability to scale resources dynamically by detecting the need for more resources, acquiring the resources, and distribute the load among the resources with minimal intervention. To detect the need for scaling events, a proactive approach of forecasting future resource demand values based on demand history allows the system to preemptively scale to the correct level and thus alleviate some or all of the scaling overhead.

In [16], a string matching based scaling algorithm is developed and suggested to predict the aggregate number of CPUs requested in traces of Animoto and IBM cloud workloads. This algorithm is compared to AR(1), linear regression, and the Rightscale democratic voting algorithm [30]. The PRESS [31], predictive elastic resource scaling, system is presented which uses FFT with signal processing for long range prediction and Markov chains for shorter predictions. This system was compared against an autoregressive and an autocorrelation method with three workloads: ClarkNet webserver trace, web access logs from the World Cup '98 official web site, and a hybrid of ClarkNet traces and a synthetic workload. In [32], double exponential smoothing and weighted moving averages are compared for their ability to predict cloud workloads. The CPU utilization values of a simulated cloud system implemented using CloudSim cloud simulator were used as the workload for these methods. A neural network prediction algorithm is developed and compared to linear regression in [27]. The accuracy of these algorithms are assessed by predicting CPU utilization traces of a server running TPC-W benchmarks.

Several full dynamic provisioning algorithms have been proposed. In [33], a system for dynamic resource provisioning that utilizes a server model to relate application server requirements to their dynamic workloads and a predictor, AR(1), to predict future workload values. The workload for two applications, one of which is synthetic and the other is based on the World Cup '98 server logs, were used to evaluate the efficacy the prediction and allocation techniques. In [34], a server provisioning and load dispatching algorithm are used to provision resources for long-lived connections in connection servers. An autoregressive model is suggested for workload prediction. Methods were evaluated in a simulated a cluster of connection servers with data traces of total connected users and login rates obtained from production instant massaging connection servers.

An automated scaling framework is developed in [13] to use vertical and horizontal scaling to meet scaling demands while reducing reconfiguration costs. They suggest using polynomial approximation of future workloads. They tested their system to managing a small virtualized testbed of servers with live migration capability running the Olio cloud benchmark that simulates social networking workloads. In [15], an architecture for monitoring and optimizing resource usage is presented that uses predictions based on autoregressive model of smooth request traces. They tested their implementation with a three-tier, php-based online collaboration system. A self-reconfiguration approach for reallocating VMs in [35] uses genetic algorithms to find optimal reconfiguration policies and Brown's double exponential smoothing [36] to predict load. Their evaluation platform consisted of a large cluster hosting virtual machines deployed with TPC-W benchmark. Activity in the benchmark was driven based on requests in the World Cup '98 web site and ClarkNet web server traces. In [37], a prediction-based method for dynamic resource provisioning and scaling of MMOGs in distributed Grid environments is presented that uses neural networks for prediction. They test several implementations of neural networks with different network structures against predictions based on averages, a moving average, the last value, and exponential smoothing.

Many of the listed related works compare a relatively small number of forecasting methods against each other. Further, a suggested method is often compared to a simple method such as a weighted moving average. Here seven different methods, some of which were suggested in the related work, are evaluated against each other, giving a wider coverage of the relative performances of the forecasting methods than previously available. The workload traces used to access the accuracy of a forecasting method for use in a cloud environment included traces from webservers such as the ClarkNet and World Cup '98 server logs. These traces only include webserver activity which is one of many types of workload (data serving, media streaming, MapReduce, etc.) that can be simultaneously hosted by a cloud data center. This work uses real cloud data center workloads from Google and Intel production cloud environments used to service a diverse variety of requests. Many of the related works that developed full dynamic provisioning algorithms tested the performance of the entire algorithm in terms of some overall cost reduction or efficiency gain. Separating the performance evaluation of the forecasting method from the performance of a larger system allows for a direct comparison of methods that cannot be attained with the current literature.

4 Experimental Design

This section explains the methods used to evaluate the applicability of several time series forecasting models on real cloud job request traces. A list of models tested, tests performed, cloud data center job request traces, and the experimental set up are presented.

The following forecasting models are evaluated for their ability to produce accurate predictions of cloud job request traces:

- First-order autoregressive model (AR(1))
- First-order moving average model (MA(1))
- Simple exponential smoothing
- Double exponential smoothing
- ETS method
- Automated ARIMA method
- Neural network autoregressive method

4.1 Tests Performed

Three tests are performed to evaluate the accuracy of each forecasting model: out-of-sample forecasting, rolling forecast origin cross-validation, training set length analysis.

- Out-of-sample forecasting error [38]: In this test, historical data is used to train a model which is used to make predictions of future values. This test is included to evaluate the forecast model's ability to produce accurate single-point predictions.

  1. In this test, an input time series is first partitioned into two sets: a training set and a test set. This must be done because when evaluating the accuracy of a forecasting model to predict a certain data set, the model can become over-fitted if the same data set is used for the training set and test set, producing overly optimistic accuracy values.
  2. The training set, composed of the first N samples of the data set, is used to train the forecasting model. In order to preserve any seasonal patterns in the data set, the training set should include enough data points to capture several periods.
  3. The trained model is used to perform a single-point forecast for each point in the test set, producing a time series of forecast values. The test set typically starts at time N+1 and includes at least enough points to form one period of the input time series.
  4. Accuracy metrics are calculated between the test set and the forecast time series. Mean absolute percentage error (MAPE) is used as the error metric.

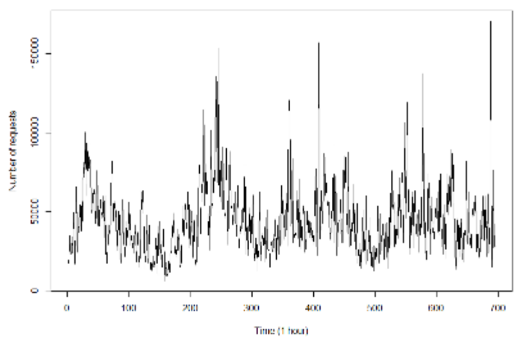
- Rolling forecast origin cross-validation [38]: This test is included to evaluate a forecasting models ability to accurately produce a number of predictions, h.

  1. Select a number of observations required, k, for reliable forecast. The first k observations are used as a training set for the forecasting model.
  2. The following h points starting at time k+1 and ending at time k+h are used as the test set. The trained model is used to perform a multiple-point forecast of the test set, producing a time series of forecast values with length h.
  3. Calculate the mean absolute parentage error (MAPE) for each point in the forecast horizon.
  4. Slide the training set forward and repeat steps 1-3.
  5. Calculate the average of the MAPE for each forecast horizon point

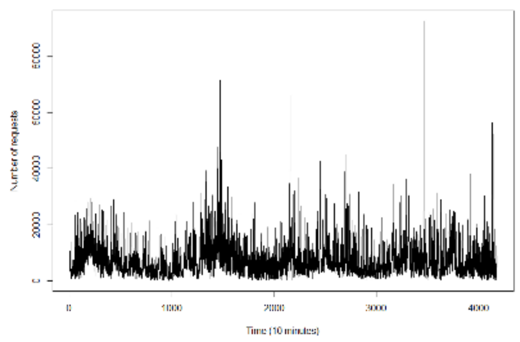
- Training set length analysis - In order to observe the effect of increasing the training set length the following test sequence is performed. This set performs is a series of rolling forecast origin cross-validations with different training set lengths.

  1. Select a training set length, N. For this work, the initial training set length used was 5 points.
  2. Train the model with the training set, which is initially composed of the first N observations in the input time series.
  3. Perform a single point forecast of the next point, at time N+1, and calculate MAPE of the forecast.
  4. Slide the training set forward such that the first point in the previous training set is not included and the point after the previous training set, at time N+1, is included.
  5. Repeat 2-4 over several iterations. In this work, 200 of such iterations are performed for each tested training set length.

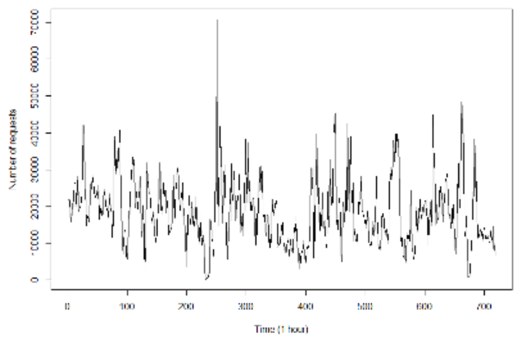
6. Repeat the process with an incremented training set length. Lengths between 5 and 25 are considered.



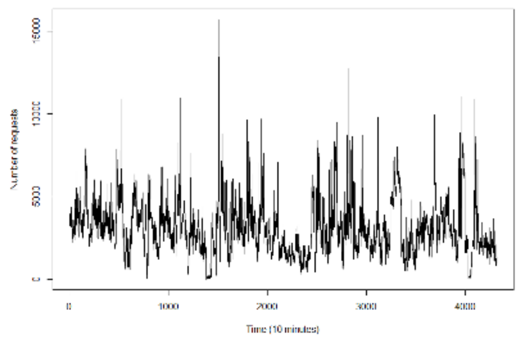
(a) Google cluster data job request time series, 1 hour interval period



(b) Google cluster data job request time series, 10 minute interval period



(c) Intel Netbatch job request time series, 1 hour interval period



(d) Intel Netbatch job request time series, 10 minute interval period

Figure 9: Job request time series.

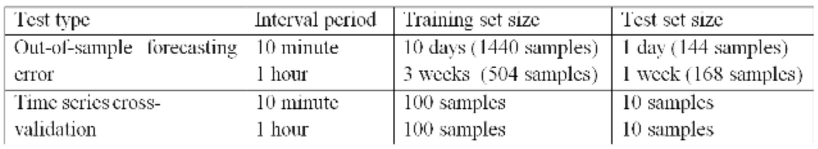
4.2 Workload Traces

Two clouds systems, Intel Netbatch logs [39] and Google cluster data [40], are used to evaluate the forecasting models. Workloads are given as an ordered list of requests to perform a computing job, some process that consumes a share of the cloud system's resources. To form a time series with evenly spaced time intervals, an interval period is selected and the number of jobs per successive interval is used as that interval's value. For each cloud workload, two job request count time series are produced with interval periods of 10 minutes and 1 hour.

4.3 Experimental Setup

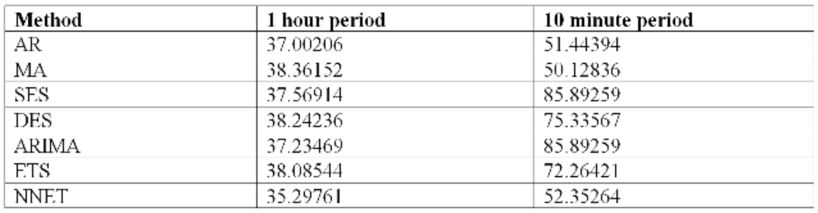
Forecasting models were evaluated by performing an out-of-sample forecasting error test, time series cross-validation, and a training set length analysis for each workload trace. Test set and training set sizes for the out-of-sample forecasting error test and the time series cross-validation can be found in figure 10. Training set lengths ranging from 5 to 25 were considered in the training set length analysis.

The experiments were carried out using R statistical programming [41] version 3.1.3 with the forecast package [26] installed. All calculations were performed on a Lenovo Yoga Pro 2 laptop computer with an Intel Core i7-4500U CPU @ 1.80 GHz-2.40GHz, 8GB of RAM, 512GB SSD, and a Windows 8.1 OS.



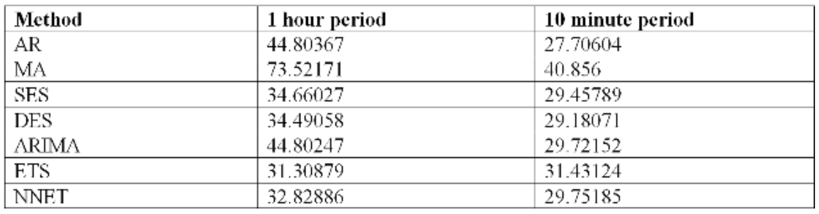
Test type	Interval period	Training set size	Test set size
Out-of-sample forecasting error	10 minute	10 days (1440 samples)	1 day (144 samples)
	1 hour	3 weeks (504 samples)	1 week (168 samples)
Time series cross-validation	10 minute	100 samples	10 samples
	1 hour	100 samples	10 samples

Figure 10: Out-of-sample forecasting error and time series cross-validation parameters



Method	1 hour period	10 minute period
AR	37.00206	51.44394
MA	38.36152	50.12836
SES	37.56914	85.89259
DES	38.24236	75.33567
ARIMA	37.23469	85.89259
ETS	38.08544	72.26421
NNET	35.29761	52.35264

Figure 11: Google cluster out-of-sample forecast error



Method	1 hour period	10 minute period
AR	44.80367	27.70604
MA	73.52171	40.856
SES	34.66027	29.45789
DES	34.49058	29.18071
ARIMA	44.80247	29.72152
ETS	31.30879	31.43124
NNET	32.82886	29.75185

Figure 12: Intel Netbatch out-of-sample forecast error

5 Results

5.1 Out-of-Sample Forecast Error Test

The results of the out-of-sample forecast error test for each of the workload traces are given below in terms of MAPE. In this test, historical data is used to train a model which is used to make predictions of future values. Prediction errors as low as 27.7% and as high as 85.9% were observed. No one forecasting model was superior for all traces-models perform well for some traces and not others. It was observed that the predictability of a trace could change depending on the period it is considered with. The Google cluster data became significantly less predictable when the smaller period of 10 minutes was used as opposed to 1 hour periods. This was not the case for the Intel Netbatch trace which showed similar or reduced forecasting error when a shorter period was considered.

The autoregressive model was calculated using a generic ARIMA function with (p, d, q) at (1, 0, 0). This method performed especially well relative to the other forecasting methods at forecasting the Intel Netbatch 10 minute period trace producing a MAPE of 27.7, the lowest MAPE reported in this study.

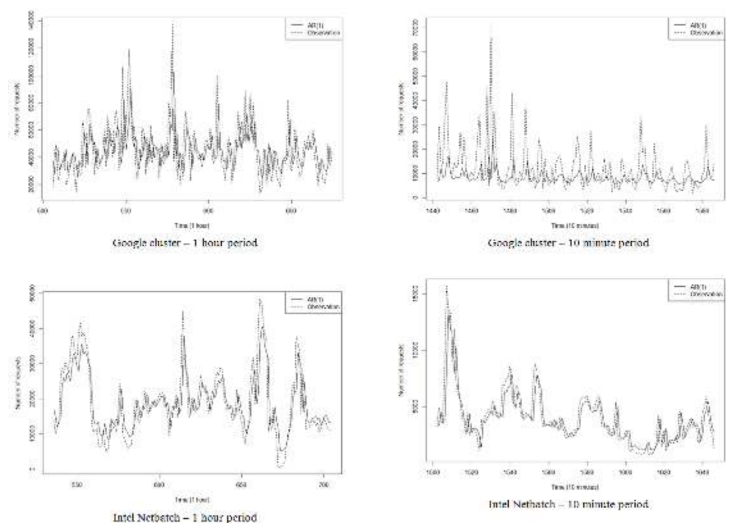


Figure 13: Out-of-sample forecast versus observed value for autoregressive model

The autoregressive model was not able to adapt to the rapid fluctuations in demand in the Google cluster 10 minute period trace, resulting in severe underestimates of the peaks.

The moving average model was calculated using a generic ARIMA function with (p, d, q) at (0, 0, 1). For all of the workload traces, this method produced forecasts that tend to remain near an average value, responding to impulses slowly. The 1 hour Intel Netbatch trace was particularly poorly forecasted, with a MAPE of 73.5. Large lengths of overestimation and underestimation can be observed in figure 14. The 10 minute Google cluster trace had shorter, pointed fluctuations from a relatively constant level. By tracing that constant level, this model produced the best MAPE, 50.1, of all the forecasting methods.

The simple exponential smoothing model was calculated using a generic ARIMA function with (p, d, q) at (0, 1, 1). The smoothing performed by this model has a pronounced effect on rapidly fluctuating data, as can be seem on both of the Google cluster traces. This effect works well (MAPE of 37.7) for the 1 hour period trace, where the predictions fit trends in the data well, and not well (MAPE of 85.9) for the 10 minute period trace, where values are often overestimated. The double exponential smoothing model was calculated using a generic ARIMA function with (p, d, q) at (0, 2, 2). Double exponential smoothing produced similar results to simple exponential smoothing in terms of prediction error and smoothing effects observed in figure 15 although this method tended to overestimate the prediction values than the other methods except ETS. As described in section 4, the automated ARIMA method tests several different models for the most appropriate fit to the training data before fitting training the model's parameters. Although this method attempts to find the best ARIMA model for a data set, it selected a single exponential smoothing model for the 10 minute Google cluster trace which produced the least accurate forecast, MAPE of 85.9. Models produced by this method are closely comparable if not identical to other evaluated forecasting models.

ETS is an automated model selecting method that produces an exponential smoothing model. This method produced the best results for the 1 hour Intel Netbatch trace with a MAPE of 31.3. Similar to previous exponential smoothing models, ETS produced an overly smoothed estimation of the 10 minute Google cluster trace with low forecasting accuracy, MAPE of 72.2.

Neural network autoregression is a method for producing and training a three-layer neural network. For example, the 1 hour Google cluster trace was modeled with a neural networks with a 2-2-1 structure. This model produced best predictions for the trace with a MAPE of 35.2. Other models produced with this method were more accurate than most other methods.

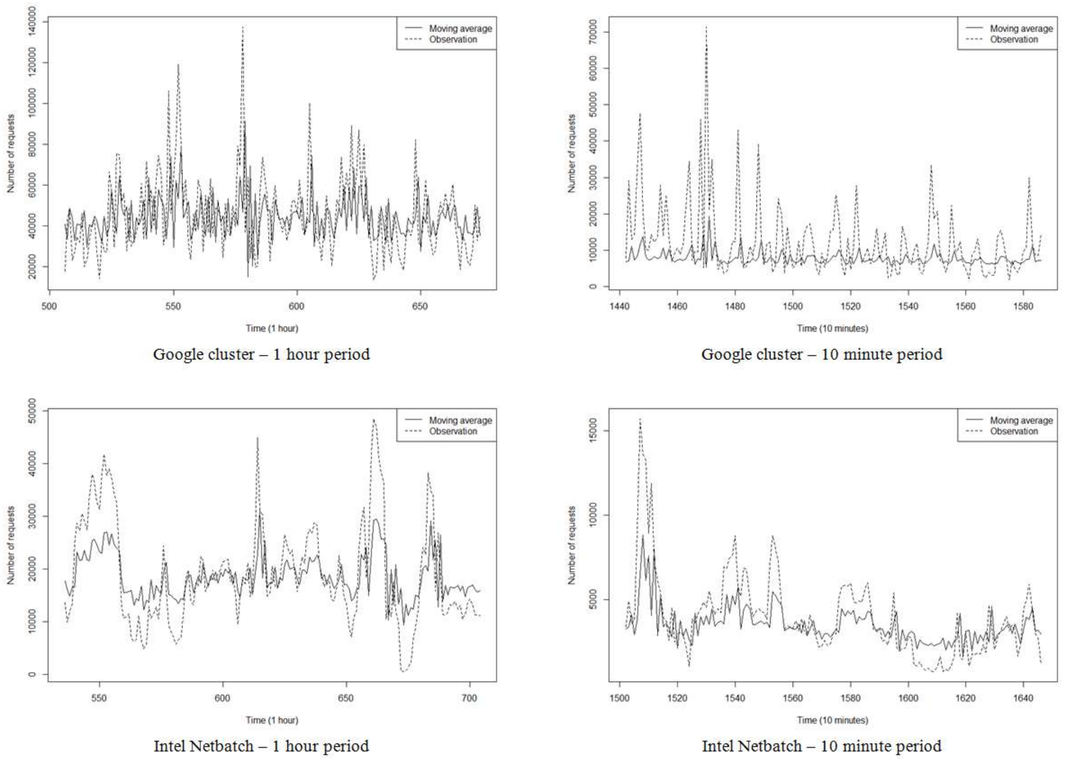


Figure 14: Out-of-sample forecast versus observed value for moving average model

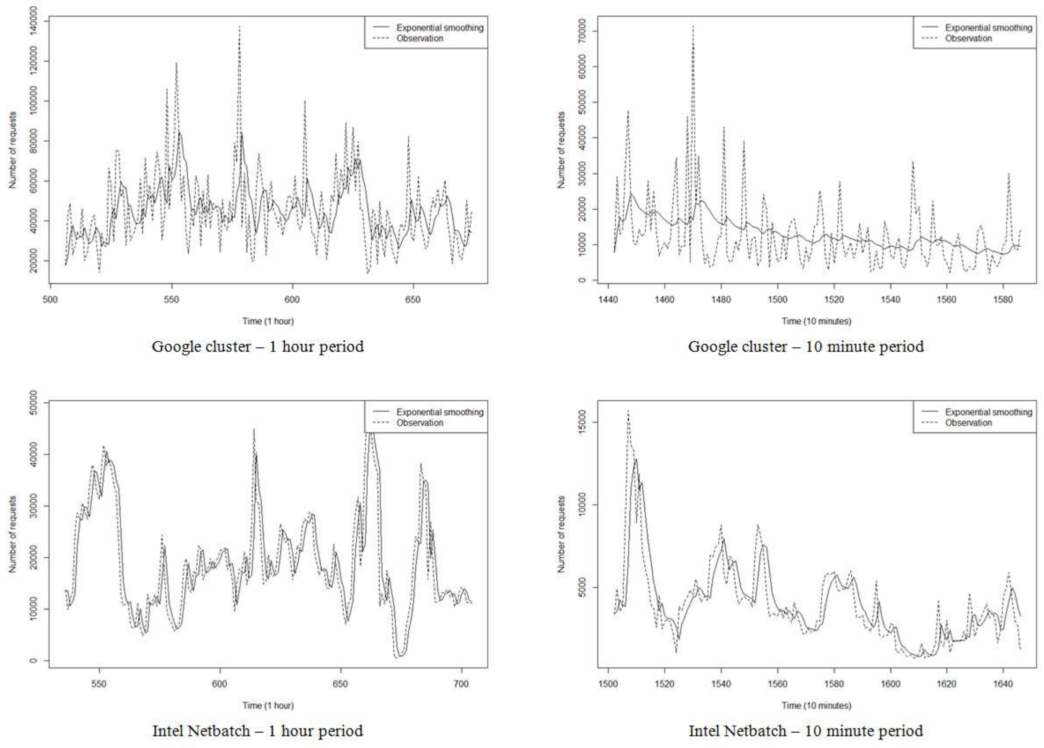


Figure 15: Out-of-sample forecast versus observed value for simple exponential smoothing

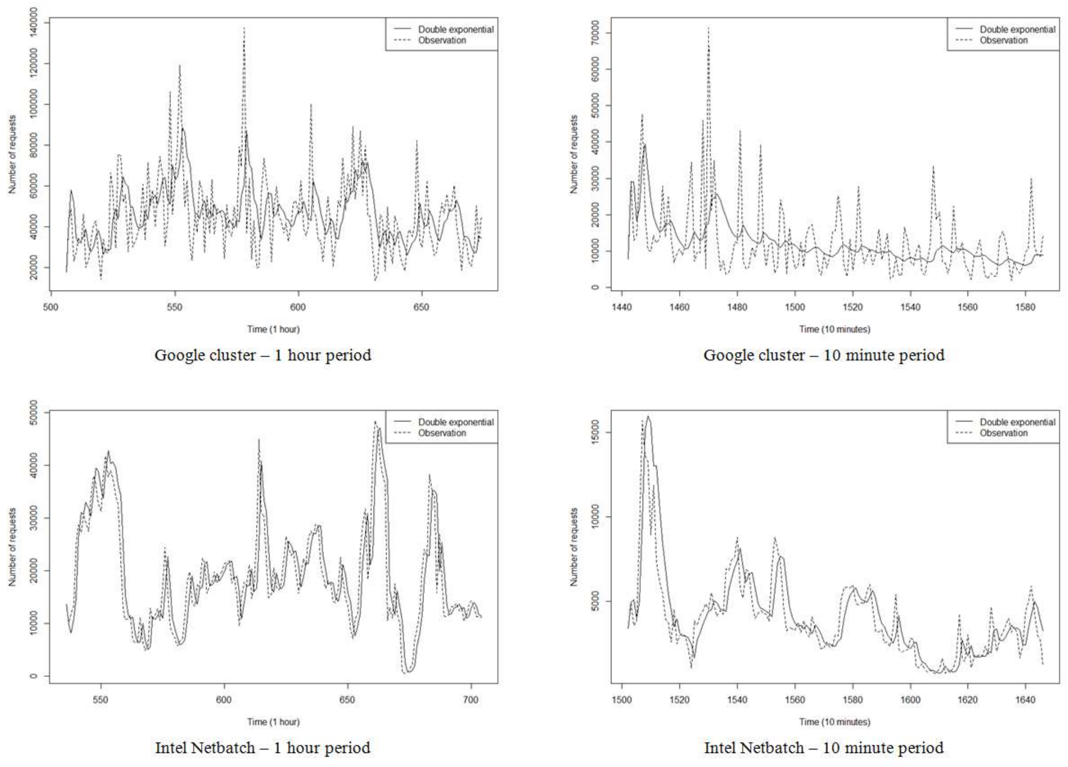


Figure 16: Out-of-sample forecast versus observed value for double exponential smoothing

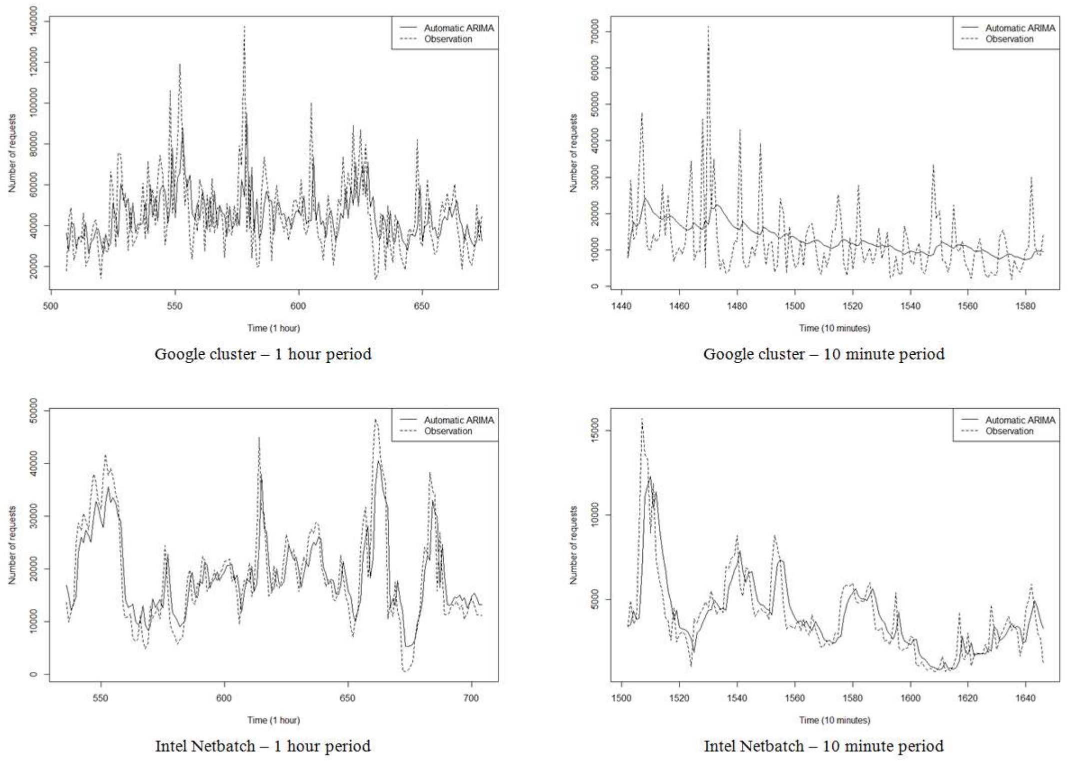


Figure 17: Out-of-sample forecast versus observed value automated ARIMA

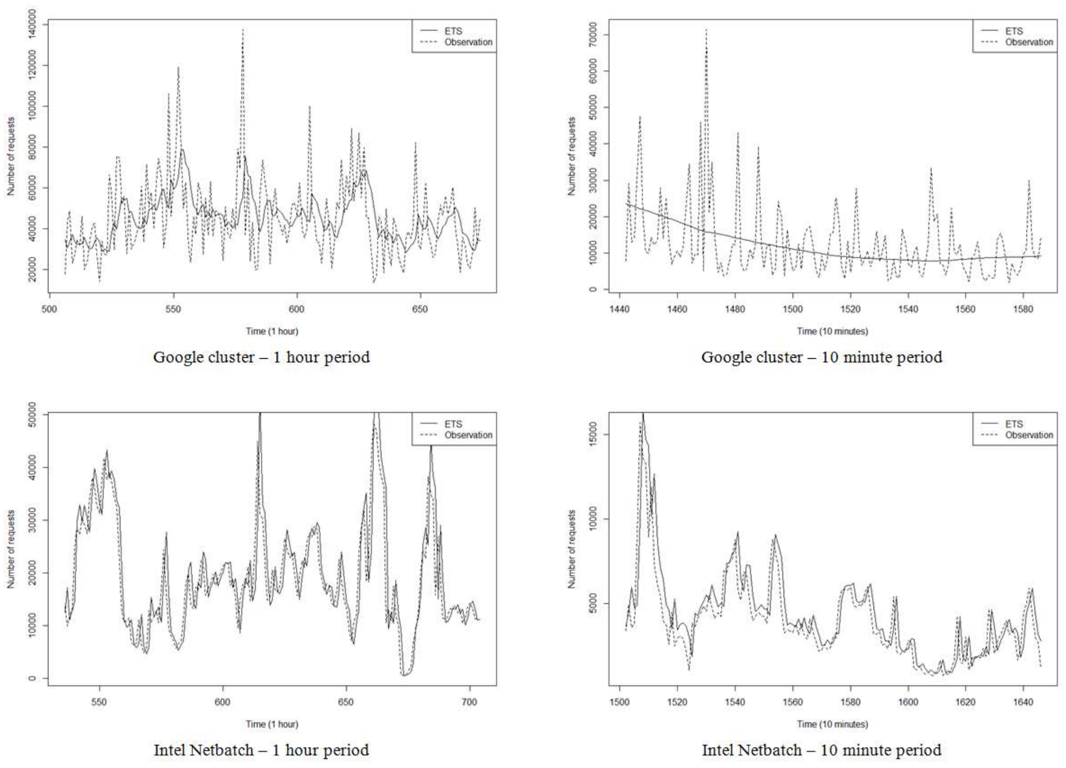


Figure 18: Out-of-sample forecast versus observed value for ETS

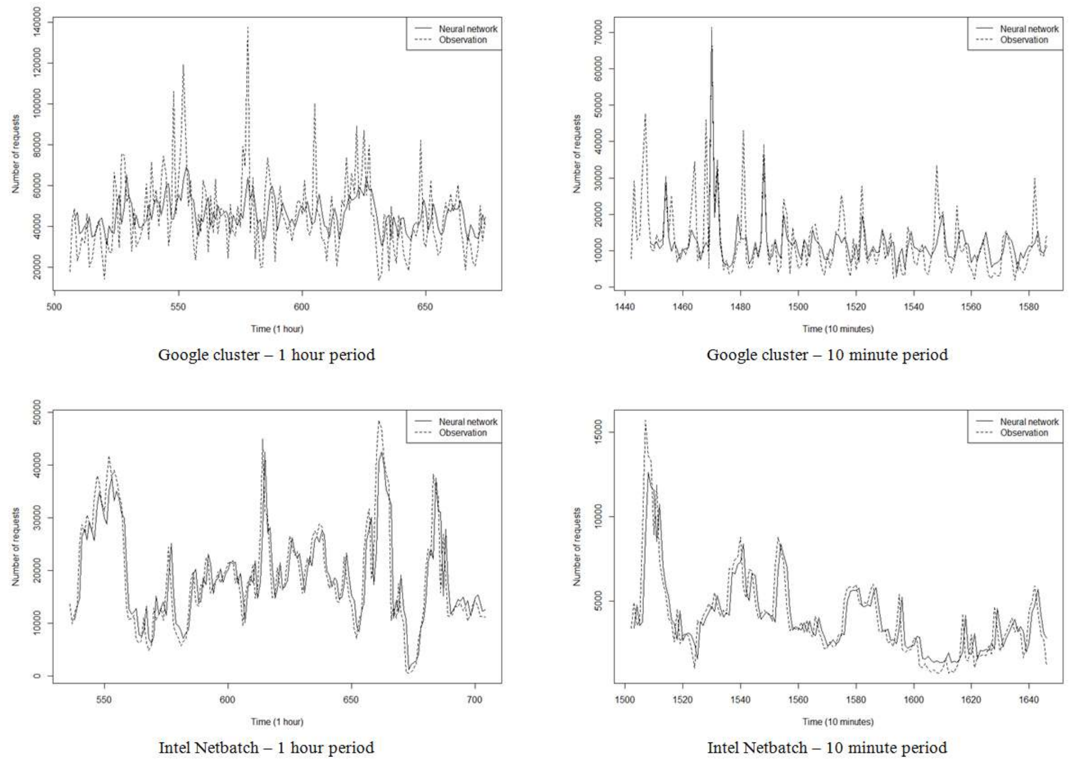
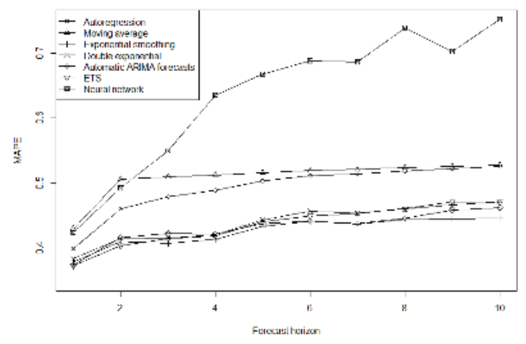
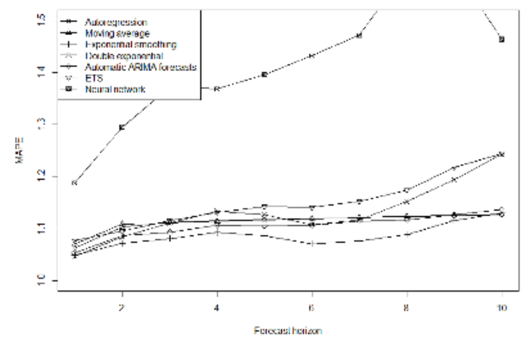


Figure 19: Out-of-sample forecast versus observed value for neural network autoregression

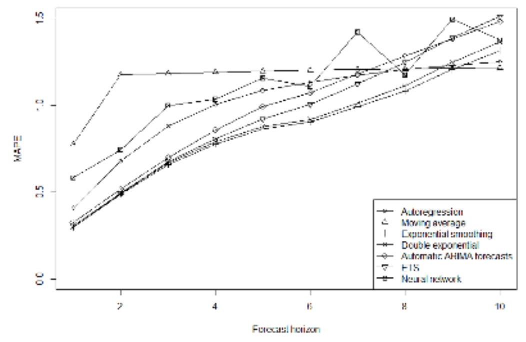
5.2 Rolling Forecast Origin Cross-Validation



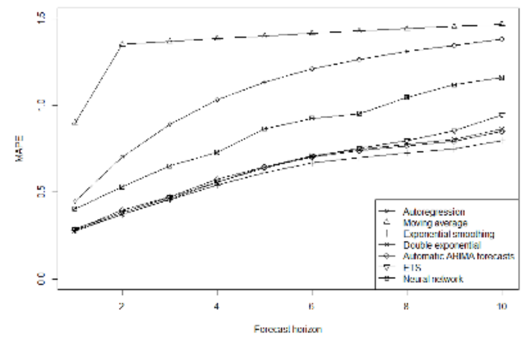
(a) Rolling forecast origin cross-validation for Google cluster traces-1 hour



(b) Rolling forecast origin cross-validation for Google cluster traces-10 minutes



(c) Rolling forecast origin cross-validation for Intel Netbatch traces-1 hour

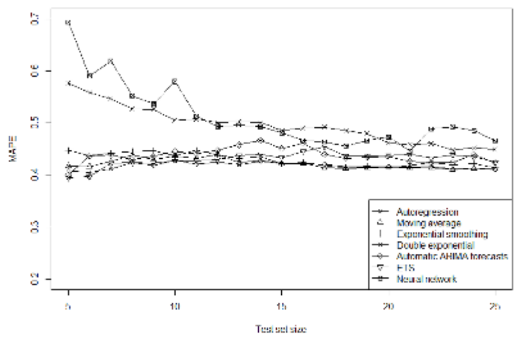


(d) Rolling forecast origin cross-validation for Intel Netbatch traces-10 minutes

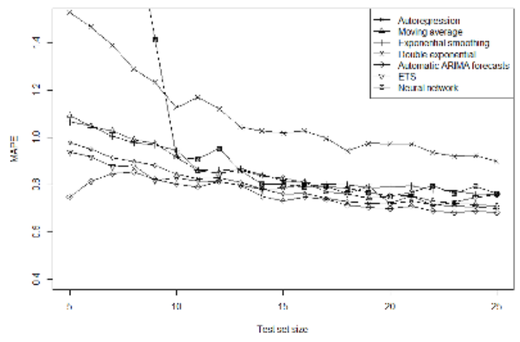
Figure 20: Rolling forecast origin cross-validation.

Rolling forecast origin cross-validation is included to evaluate a forecasting models ability to accurately produce a number of predictions, ten for this study. After training the model with a test set of length 100, ten future values are predicted and the MAPE for each prediction is recorded. This process is repeated with the test set starting one point later until no more values can be predicted.

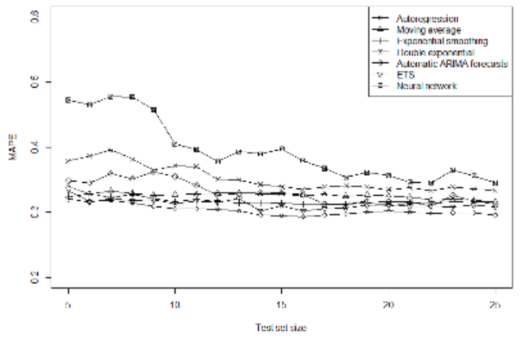
The average MAPE for each forecast horizon length are presented in figure 20. It can be seen in the figures that as the forecast horizon increases in length, the forecasts become less accurate, as one would expect. Some models were extremely inaccurate as forecast horizon increased, especially the neural network. The relative performance of a model with respect to the other models tends to stay consistent i.e. a model that outperforms another model at a given forecast horizon tends to outperform that model when a larger forecast horizon is considered.



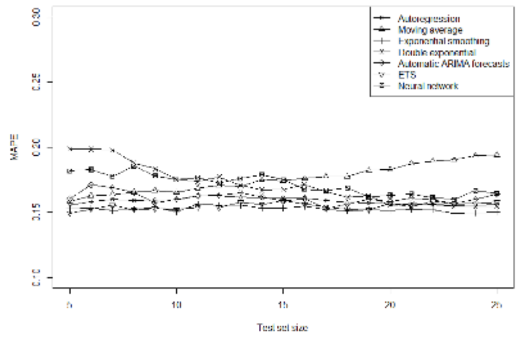
(a) Effect of training set length for Google cluster traces-1 hour



(b) Effect of training set length for Google cluster traces-10 minutes



(c) Effect of training set length for Intel Netbatch traces-1 hour



(d) Effect of training set length for Intel Netbatch traces-10 minutes

Figure 21: Effect of training set length.

5.3 Training Set Length Analysis

In order to observe the effect of increasing the training set length the following test sequence is performed. As one would expect, forecasting accuracy tends to improve as the more historical data is included in training a forecasting model. However, the benefits of increasing the training set length appears to saturate for all tested forecasting models at larger lengths. Most models converged to a stable lower forecasting error level with between 10 and 15 training set points. Although double exponential and neural network both produced relatively accurate results, these methods require more points to achieve stable accuracy.

6 Conclusion

In this work an overview of cloud computing concepts has been presented including an explanation of the dynamic configurability afforded by virtualization. By using an automated resource scaling algorithm, a cloud applications can best exploit the benefits of on-demand elasticity, reducing wasted computing resources and avoiding SLA violations. This work has also presented a review of time series forecasting and several time series forecasting techniques that could be utilized to implement a proactive dynamic resource scaling algorithm. Obtaining accurate prediction results is crucial to the efficient operation of an automated resource scaling algorithm.

These forecasting methods were compared for their ability to forecast cloud computing workloads including Google cluster data and Intel Netbatch logs. Out-of-sample forecasts of the computing logs were taken and compared in terms of their accuracy. Although no one model always produced the most accurate results, some models did tend to perform better than others including first order autoregressive and neural network autoregression. It was found that accuracy of the forecasts produced by a model depends on the workload being considered and the frequency at which the workload is being considered. A rolling forecast origin cross-validation was performed for each of the forecasting models to test the accuracy of multiple future predictions made with the same test set. Although the accuracy of a model varied with different trace and frequency, some models could produce useful predictions two or three time periods into the future. In analyzing the effects of the training set length on prediction accuracy, it was observed that most of the tested models only needed 10 to 15 points to produce their best accuracy results.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[2] M. J. Kavis, *Architecting the cloud: Design decisions for cloud computing service models (SaaS, PaaS, AND IaaS)*. John Wiley & Sons, 2014.

[3] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Proc. of the 2008 Grid Computing Environments Workshop (GCE'08), Austin, Texas, USA*. IEEE, November 2008, pp. 1–10.

[4] J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann Publishers Inc., 2003.

[5] R. Prodan and T. Fahringer, *Grid Computing: Experiment Management, Tool Integration, and Scientific Workflows*. Berlin, Heidelberg: Springer-Verlag, 2007.

[6] "AWS — Amazon Elastic Compute Cloud (EC2) - Scalable Cloud Hosting," https://aws.amazon.com/, accessed: 2015-08-27.

[7] P. Mell and T. Grance, "The NIST definition of cloud computing," NIST Special Publication 800-145, September 2011, http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.

[8] "News Center - Microsoft," https://news.microsoft.com, accessed: 2015-08-27.

[9] A. J. Younge, R. Henschel, J. T. Brown, G. Von Laszewski, J. Qiu, and G. C. Fox, "Analysis of virtualization technologies for high performance computing environments," in *Proc. of the 4th IEEE International Conference on Cloud Computing (CLOUD'11), Washington, D.C., USA*. IEEE, July 2011, pp. 9–16.

[10] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki, and B. Falsafi, "Clearing the clouds: A study of emerging scale-out workloads on modern hardware," in *Proc. of the 17th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'13), Houston, Texas, USA*. ACM, March 2012, pp. 37–48.

[11] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[12] "Service Level Agreement Standardisation Guidelines," http://ec.europa.eu/digital-agenda/en/news/cloud-service-level-agreement-standardisation-guidelines, accessed: 2015-08-27.

[13] S. Dutta, S. Gera, A. Verma, and B. Viswanathan, "Smartscale: Automatic application scaling in enterprise clouds," in *Proc. of the IEEE 5th International Conference on Cloud Computing (CLOUD'12), Honolulu, Hawaii, USA*. IEEE, June 2012, pp. 221–228.

[14] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," in *Proc. of the 5th International Joint Conference on INC, IMS and IDC, Seoul, Korea*. IEEE, August 2009, pp. 44–51.

[15] S. Khatua, A. Ghosh, and N. Mukherjee, "Optimizing the utilization of virtual resources in cloud environment," in *Proc. of the 2010 IEEE International Conference on Virtual Environments Human-Computer Interfaces and Measurement Systems (VECIMS'10), Taranto, Italy.* IEEE, September 2010, pp. 82–87.

[16] E. Caron, F. Desprez, and A. Muresan, "Pattern matching based forecast of non-periodic repetitive behavior for cloud clients," *Journal of Grid Computing*, vol. 9, no. 1, pp. 49–64, 2011.

[17] "Google Finance: Dow Jones Industrial Average," http://www.google.com/finance?q=INDEXDJX:.DJI, accessed: 2015-08-27.

[18] K. Nikolopoulos, "Time-Series Forecasting: Chris Chatfield," *International Journal of Forecasting*, vol. 19, no. 4, pp. 754–755, 2003.

[19] "Common Approaches to Univariate Time Series," http://www.itl.nist.gov/div898/handbook/pmc/section4/pmc444.htm, accessed: 2015-08-27.

[20] "Smoothing Time Series," https://onlinecourses.science.psu.edu/stat510/?q=node/70, accessed: 2015-08-27.

[21] "Double Exponential Smoothing," http://www.itl.nist.gov/div898/handbook/pmc/section4/pmc433.htm, accessed: 2015-08-27.

[22] R. Hyndman, A. B. Koehler, J. K. Ord, and R. D. Snyder, *Forecasting with exponential smoothing: the state space approach.* Springer Science & Business Media, 2008.

[23] W. Pan, "Akaike's information criterion in generalized estimating equations," *Biometrics*, vol. 57, no. 1, pp. 120–125, 2001.

[24] "ARIMA modelling in R," https://www.otexts.org/fpp/8/7, accessed: 2015-08-27.

[25] D. Kwiatkowski, P. C. Phillips, P. Schmidt, and Y. Shin, "Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root?" *Journal of Econometrics*, vol. 54, no. 1, pp. 159–178, 1992.

[26] R. J. Hyndman and Y. Khandakar, "Automatic time series for forecasting: the forecast package for R," *Journal of Statistical Software*, vol. 27, no. 3, 2008. [Online]. Available: http://dx.doi.org/10.18637/jss.v027.i03

[27] 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, no. 1, pp. 155–162, 2012.

[28] N. J. Salkind, *Encyclopedia of research design.* Sage Publications, June 2010, vol. 1.

[29] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International Journal of Forecasting*, vol. 22, no. 4, pp. 679–688, 2006.

[30] J. Kupferman, J. Silverman, P. Jara, and J. Browne, "Scaling into the cloud," CS270-advanced operating systems, 2009, http://www.cs.ucsb.edu/j̃browne/files/ScalingIntoTheClouds.pdf.

[31] Z. Gong, X. Gu, and J. Wilkes, "Press: Predictive elastic resource scaling for cloud systems," in *Proc. of the 2010 International Conference on Network and Service Management (CNSM'10), Niagara Falls, Canada.* IEEE, October 2010, pp. 9–16.

[32] J. Huang, C. Li, and J. Yu, "Resource prediction based on double exponential smoothing in cloud computing," in *Proc. of the 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet'12), Three Gorges, YiChang, Hubei, China.* IEEE, April 2012, pp. 2056–2060.

[33] A. Chandra, W. Gong, and P. Shenoy, "Dynamic resource allocation for shared data centers using online measurements," in *Proc. of the 2003 International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS'03), San Diego, California, USA.* ACM, 2003, pp. 300–301. [Online]. Available: http://doi.acm.org/10.1145/781027.781067

[34] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao, "Energy-aware server provisioning and load dispatching for connection-intensive internet services," in *Proc. of the 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI'08), San Francisco, California, USA.* USENIX Association, 2008, pp. 337–350. [Online]. Available: http://dl.acm.org/citation.cfm?id=1387589.1387613

[35] H. Mi, H. Wang, G. Yin, Y. Zhou, D. Shi, and L. Yuan, "Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers," in *Proc. of the 2010 IEEE International Conference on Services Computing (SCC'10), Miami, Florida, USA.* IEEE, July 2010, pp. 514–521.

[36] R. G. Brown and R. F. Meyer, "The fundamental theorem of exponential smoothing," *Operations Research*, vol. 9, no. 5, pp. 673–685, 1961.

[37] R. Prodan and V. Nae, "Prediction-based real-time resource provisioning for massively multiplayer online games," *Future Generation Computer Systems*, vol. 25, no. 7, pp. 785–793, 2009.

[38] "Evaluating forecast accuracy," https://www.otexts.org/fpp/2/5, accessed: 2015-08-27.

[39] O. Shai, E. Shmueli, and D. Feitelson, "Heuristics for resource matching in intel's compute farm," in *Proc. of the 17th International Workshop on Job Scheduling Strategies for Parallel Processing (JSSPP'13), Boston, Massachusetts, USA, LNCS*, N. Desai and W. Cirne, Eds., vol. 8429. Springer Berlin Heidelberg, May 2014, pp. 116–135.

[40] J. L. Hellerstein, W. Cirne, and J. Wilkes, "Google cluster data," Google research blog, January 2010, http://googleresearch.blogspot.kr/2010/01/google-cluster-data.html.

[41] R. Ihaka and R. Gentleman, "R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, 2012," *Journal of Computational and Graphical Statistics*, vol. 5, no. 3, pp. 299–314, 1996.

Author Biography

Carlos Vazquez is from Killeen, TX. He earned his B.S. and M.S. degrees in Computer Engineering from the University of Texas at San Antonio. He plans on continuing his studies in computer science and engineering in the computer hardware industry.

Dr.Ram Krishnan received a Ph.D from George Mason University. He is currently an Assistant Professor of Electrical and Computer Engineering at UTSA. His area of research interest is computer and network security–specifically, access control (models and analysis) and security issues in cloud computing.

Dr.Eugene B John received a Ph.D from Pennsylvania State University. Currently he is Professor of Electrical and Computer Engineering at UTSA. His areas of research interests are Low Power VLSI Design, Computer Performance Evaluation, Cloud Benchmarking, Computer Architectures, Power-Aware and Reliable Systems.