Online Adaptation Models for Resource Usage Prediction in Cloud Network

Shaifu Gupta and Dileep Aroor Dinesh
School of Computing and Electrical Engineering
Indian Institute of Technology Mandi, Kamand - 175005, H.P., India
Email: shaifu_gupta@students.iitmandi.ac.in, addileep@iitmandi.ac.in

***Abstract*—Cloud computing provides rapid on-demand access to shared services over the Internet. Elasticity is a key feature of cloud that allows the system to dynamically adapt to workload changes such that the resource allocation sufficiently models the resource usage as close as possible. It is a highly challenging job as a number of users enter and leave the cloud environment dynamically. Predicting the usage of different resources in advance helps the service providers in better capacity planning and satisfy the needs of users. In this paper, we analyze the multi-step ahead CPU usage predictions of different existing time series prediction models. To model the highly varying cloud workloads and counter the effect of error propagation in iterative multi-step ahead predictions we proposed and analyzed online adaptation of time series CPU usage prediction models. We also use fractional differencing to capture long range dependence in the data. We evaluate the proposed adaptive models on Google cluster trace.**

I. Introduction

Cloud computing is growing rapidly as a widely accepted paradigm for accessing vast resources over the Internet. It offers a highly scalable and 'pay-as-you-use' environment. Elasticity of cloud allows it to automatically adapt itself to meet the changing workload requirements. Since a number of users dynamically enter and leave the cloud environment to access the services offered by the cloud network, there is a need of timely allocation of resources from cloud service provider's end [1]. It is not efficient to keep all the resources active to satisfy the incoming user needs. It leads to over-provisioning and wastage of energy. On the other hand, allocation of less resources leads to service degradation. Effective monitoring and prediction of resource usage helps the service administrators to determine the appropriate number of running machines and take actions to turn on/off machines [2]. Insight into future usage of different resources helps in better capacity planning and make desirable decisions for allocating resources. Resource prediction can be seen as a time series prediction problem. The future usage of resources is predicted by recognizing the trends in the past usage. Different statistical models have been explored for resource prediction in cloud for varied applications. PRedictive Elastic ReSource Scaling (PRESS) [3] uses a discrete Markov model for resource usage prediction. AGILE [4] extends PRESS by using wavelet analysis along with Markov model for predicting usage of resources in cloud. Auto-regressive integrated moving average (ARIMA) model [5] and feed-forward neural network [6] are also used for prediction of usage of different resources in cloud. Most of the existing approaches for resource usage prediction in cloud have assumed that the observations separated by a long time span are independent of each other. However, [7] analyzed the cloud workloads using Google cluster trace [8] and identified long range dependence in the cloud workloads. The Fractional differencing method is a technique that captures long range dependence in time series data [9]. In our work, we extend the identified prominent approaches of resource usage prediction in cloud to capture long range dependence in the time series resource usage values. We use Hurst exponent [10] computed using a widely adopted rescaled range analysis method to perform fractional differencing on the time series data.

One-step ahead resource usage predictions are not sufficient to support the objective of resource prediction in cloud. Far-ahead knowledge of forecast aids the service providers to make better decisions for capacity planning and provisioning of resources. Multi-step ahead predictions are more challenging than single step ahead predictions. It has been observed that iterative multi-step ahead prediction causes accumulation of errors and results in inaccurate predictions [11]. In [3] the prediction models are repeatedly updated using new measurement samples when the number of consecutive mis-predictions of the resource prediction models increases beyond a certain limit. The main motivation is to update the time series prediction model based on new observations to closely model the real context and counter the error propagation. In the present work, we propose to analyze multi-step ahead predictions of different existing time series prediction models and also propose two novel approaches to develop online adaptive models for time series resource prediction in cloud. To the best of our knowledge, the adaptation of time series prediction models have not been explored for resource usage prediction in cloud network. The effectiveness of all the techniques proposed in this paper is demonstrated for prediction of CPU usage in Google cluster trace [8].

The paper makes the following contributions towards resource prediction in cloud network. First contribution of our work is to analyze the multi-step ahead predictions of different time series prediction models. The second contribution is the approaches proposed to build online adaptive time series models for resource usage prediction in cloud and the third contribution of our work is to analyze the two different variants of proposed online adaptive time series prediction models.

978-1-5090-5356-8/17/$31.00 © 2017 IEEE

The rest of the paper is organized as follows. In Section II, we review the time series models for resource usage prediction in cloud network. In Section III we present the dataset used in our studies. In Section IV, we analyze the multi-step ahead predictions of different time series prediction models and present the online adaptive model for resource prediction in cloud. Section V presents the results of the experiments performed for validating the proposed methods. Conclusions and future work are presented in Section VI.

II. Overview of time-series models for resource usage prediction in cloud network

In this section, we present an overview of some of the dominant approaches that have explored the domain of resource usage prediction in cloud using time series prediction methods. Gong et al. [3] uses Markov model and signature-driven approaches for prediction of resources in Google cluster trace. Fourier transform and dynamic time warping are used to extract signature patterns in resource usage data. Markov model is used for resource prediction for workloads where a repeating pattern is not present. The usage of the desired resource metric is divided into N bins and a transition probability matrix $\mathbf{P}$ of size $N \times N$ is computed. The probability of the next bin is predicted using Chapman-Kolmogorov equations as

$$\pi_t = \pi_{t-1}\mathrm{P} \tag{1}$$

where π_t and π_{t-1} denote the probability of a state at time t and $t-1$ respectively. AGILE [4] uses wavelet transform along with Markov models for prediction of resources in cloud. Wavelet transform provides multi-resolution analysis in both time and frequency domains which captures all the frequencies in time series along with their location. Zhang et al. [5] uses a linear model called auto-regressive integrated moving average (ARIMA) method for prediction of resource consumption values. The value at time instant t, $x_t^{'}$ is predicted as a weighted sum of previous lags.

$$x_t^{'} = a_0x_{t-1}+a_1x_{t-2}+...+a_{p-1}x_{t-p}+b_0\epsilon_{t-1}+\ldots+b_{q-1}\epsilon_{t-q} \tag{2}$$

where x_i is the actual value at time instant i, a, b are the weights of the ARIMA model. p and q are the number of lags used and ϵ are the error terms associated with the model. iOverbook [6] uses feed-forward neural network for resource usage prediction. The analytic equation of a neural network can be expressed as:

$$x_t^{'} = \sum_{j=1}^{q} \alpha_j g\left(\sum_{i=1}^{p} \beta_{ij}x_i\right) \tag{3}$$

where x is the input, p is the number of inputs, q is the number of neurons in the hidden layer, $g(.)$ is the activation function and α, β represent the connection weights that will be learned during training. Feed-forward neural network provides a non-linear mapping between input and output values and allows to capture non-linear patterns in the time series.

The approaches presented above assume the time series to be stationary and memoryless [7]. However [12] have shown that, workloads in a cloud environment are highly dynamic and heterogeneous in nature. It is specified that deeper patterns may exist in resource usage data. To learn these patterns, a significant portion of the workload is required. Conventional time series models cannot effectively capture such real workload patterns. This places a requirement for sophisticated long memory based time series models. In [7] the presence of long range dependence in Google cluster trace is identified using the detrended fluctuation analysis method. A wavelet-based technique is used to compute fractional order derivate from wavelet coefficients. After estimating the fractional order derivate, linear regression method is used to generate final forecasts. The predicted value at time instant t, is given as:

$$x_t^{'} = a\sum_{j=1}^{p}\binom{\alpha}{j}x(t-j)+b \tag{4}$$

where x is the actual time series, a , b are the linear regression parameters, α is the fractional order exponent and p is the number of past lags considered for generating out-of-sample forecasts.

In the present work, we use rescaled range analysis [13] method to perform fractional differencing [9] and capture long range dependence in cloud workloads. Rescaled range analysis is a widely used method to compute the value of Hurst parameter (H) and identify long range dependence in a time series. Long range dependence characterizes strong temporal correlations in past lags of the time series. It is also called long memory or persistence. To capture long range dependence in the data, fractional differencing [9] method is applied. Fractional differencing is defined as an infinite binomial series expansion.

$$x_t^{'} = x_t - dx_{t-1} + \frac{d(d-1)}{2!}x_{t-2} - \frac{d(d-1)(d-2)}{3!}x_{t-3}\ldots \tag{5}$$

where x_i represents the value at time instant i and $x_t^{'}$ is the fractional differenced value of x_t. d is the difference parameter and it takes a fractional value of $H-0.5$ [9]. The forecasts are made using fractionally differenced data and the predicted results are integrated to convert them back to original domain. Multi-step ahead time series prediction helps to analyze better the future trends in the time series. In our work we use iterative method to analyze multi-step ahead predictions of aforementioned methods. We propose online adaptation of resource usage prediction models to handle varying cloud workloads and counter the effect of error propagation. The proposed methods will be validated on Google cluster trace presented in next section.

III. Dataset

In 2011, Google released a cluster usage dataset called Google cluster trace [8]. This dataset is based on a cluster of about 12500 machines and provides run time information of different jobs arriving to the cluster for a 29-day period. This workload is heterogeneous in nature. The cluster includes different machines which vary in number of cores, RAM per machine etc. The jobs arriving at the cluster are scheduled for execution on different machines. The workload provides information about scheduling events of different jobs. Data on arrival, submission, execution and termination of different jobs is provided along with their time-stamps. The whole dataset is divided into six tables namely: machine events, machine attributes, job events, task events, task resource usage, and task constraints. Task resource usage table provides usage of different resources like CPU, memory and disk at different instants. In the proposed work, CPU usage is analyzed at different time instants to generate multi-step ahead forecasts.

IV. Proposed online adaptive model for resource usage prediction in cloud network

In this section, we address the issues involved in the online adaptation of models for resource usage prediction in cloud network. Cloud workloads are highly dynamic and time varying in nature. Training the models once using past resource usage is not sufficient to model the changing workloads. It is necessary to update the time series prediction model regularly to capture the new patterns in the usage of different resources. Moreover, iterative multi-step ahead resource usage prediction results in accumulation of errors [11]. Therefore, there is a need to adapt the time series prediction model to counter the effect of error propagation. This notion of adaptation of resource usage prediction models is not reported in cloud network. In this work, we propose to analyze the multi-step ahead predictions of different existing resource usage prediction models and develop an online adaptive model for resource usage prediction in cloud network.

Multi-step ahead prediction includes forecasting two or more future values in a time series. They are generated using an iterative method where the predicted value is repeatedly passed back to the model to generate the next step forecast [11]. An M step ahead forecast requires M one-step ahead predictions. M is also called prediction horizon. For example, x_{t+1} can be predicted using p past lags and some function $f(.)$ as $x_{t+1} = f(x_t, x_{t-1}, x_{t-2}, \dots, x_{t-p+1})$. In the next step, x_{t+1} is passed as input to the model and x_{t+2} can be predicted as $x_{t+2} = f(x_{t+1}, x_t, x_{t-1}, \dots, x_{t+1-p+1})$. This iterative procedure is repeated till the prediction x_{t+M} is generated. In the present work, we analyze the multi-step ahead forecasts of different resource prediction models viz. PRESS, AGILE, ARIMA and non-linear autoregressive neural network (NARNN). Figure 1 shows the prediction horizon vs prediction error in NARNN. It can be observed that as the prediction horizon increases, the accuracy of time series prediction decreases. While generating the multiple steps ahead forecast, even the small errors observed in the present predictions are propagated to future predictions leading to more erroneous and uncertain predictions [11].

To mitigate the effect of error propagation in iterative CPU resource usage prediction, there is a need to periodically update the time series prediction model based on new changes observed in the real environment. The model needs to be periodically updated to reflect the workload changes and

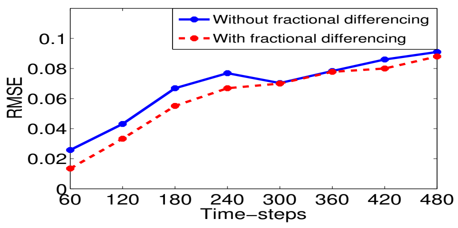


Fig. 1. Prediction horizon vs prediction error on CPU usage in NARNN

closely model the real resource utilization values. Updating the resource prediction models regularly improves the prediction ability of the time series model. It enables the model to utilize new values and improve the quality of predictions. Retraining the prediction model again and again after observing some set of real resource usage values results in a big run time overhead. The time series model can be updated by adapting the model based on the real values of CPU usage. The rest of the predictions can be adjusted based on the error observed in the predictions to get better forecasts [14]. An online ARIMA model is proposed in [15]. Here, the batch ARIMA model is converted into online model to deal with streaming characteristics of time series data. The parameters of the model are updated at each step using online gradient descent and online newton step algorithms. This method is validated on real data of monthly registration of private cars and daily index of Dow Jones Industrial Average (DJIA). In this method, the learning rate parameter remains fixed during the entire online prediction process. In our approach, we propose self adjustment of the learning rate parameter at each adaptation step to reduce the error between real CPU usage values and predicted values in the observation window. An online dictionary based kernel affine projection algorithm for time series prediction is proposed in [16]. In [15], [16] the parameters of the model are updated at every observation . In the present work, we propose two novel approaches for updating the parameters of time series model. In the first approach we update the parameters of the model, after a sufficient number of samples are observed and in the second approach we update the parameters of time series prediction model, for every incoming real value at every step.

In the present work, we explore both the variations of online adaptive models using gradient descent (GD) [17] and Levenberg Marquardt algorithm (LMA) [18] for updating the model parameters. We use the error computed between predicted values and newly observed real predictions to adjust the learned weights of the models. Let $\{\hat{x}_{t+1}, \hat{x}_{t+2}, \dots, \hat{x}_{t+M}\}$ be the M step ahead prediction of the model. Suppose m is the number of observations that we partially observed. Here, $m \geq 1$ and is called observation window. We need to adapt the model based on the error observed in the partially observed set of predictions and update the rest $M-m$ number of predictions to match closely with the real resource usage. The weights are adjusted such that the mean square error (MSE) between actual and predicted values is minimized.

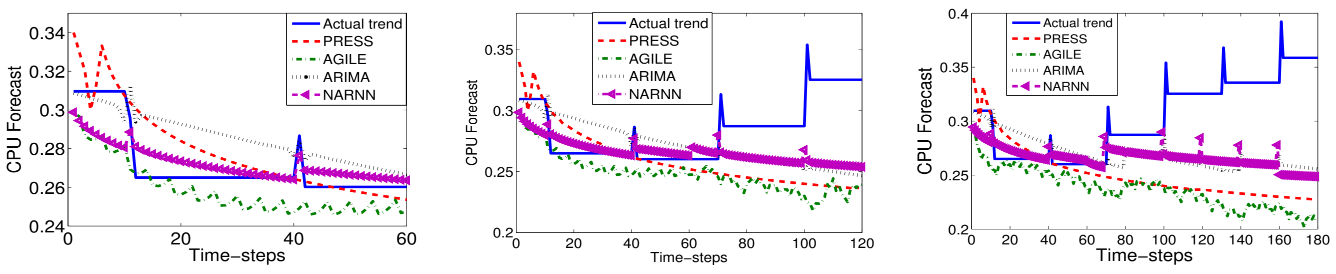


Fig. 2. Multi-step ahead out-of-sample forecasts on CPU usage at (a) 60 (b) 120 (c) 180 steps

The MSE is given as

$$Err = \frac{\sum_{t=1}^{m}(\hat{x}_t - x_t)^2}{m} \tag{6}$$

where x_t represents the actual observed resource usage values and $\hat{x}_t$ indicate the predicted values. The weights are adjusted using the gradient descent method as given below

$$w_{t+1} = w_t - \eta\frac{\partial Err}{\partial w} \tag{7}$$

where w_{t+1} and w_t represent the weights at time instants $t+1$ and t respectively and η is the learning rate. The amount of change in w (weight) is given by partial derivative of the error encountered in the prediction w.r.t. weights of the model.
In addition to the gradient descent method, we explore the LMA weight update rule. It is a combination of gradient descent and Gauss-Newton method [18]. For this method, the weight update equation is given as

$$w_{t+1} = w_t - \eta(J^TJ + \mu I)^{-1}J^TErr \tag{8}$$

where J is the Jacobian of the derivatives of residuals w.r.t. parameters of the model and μ is the combination parameter and I is the identity matrix. Small values of combination parameter result in Gauss-Newton update and high values result in gradient descent update to the weights of the model [18]. The results of multi-step ahead out-of-sample predictions of different models and online adapted predictions using gradient descent and LMA are presented in Section V.

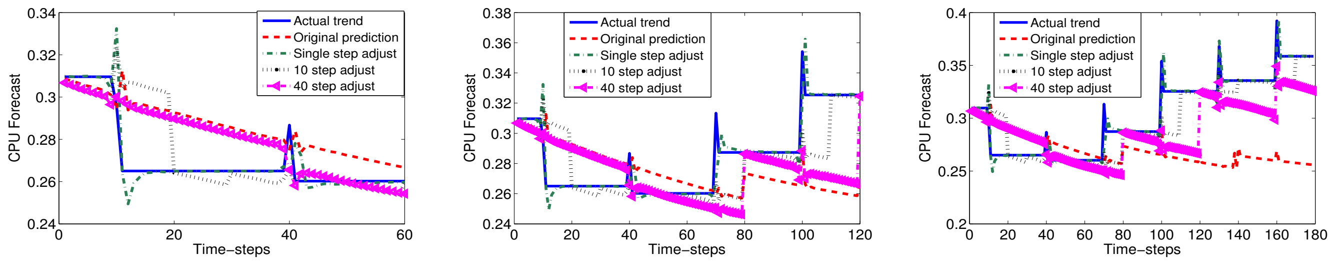
V. Studies on Resource Prediction in Cloud

In this section, we present the results of the experiments performed for time series prediction of cloud workloads. The experiments are conducted on Google cluster trace [8]. In this study we analyze time series CPU resource utilization data for generating multiple step ahead out-of-sample forecasts. In the present studies, we took 60480 samples (7 days) for training the time series prediction models, the next 20 (3 min) samples of the time series are used as validation data for parameter selection and we generate and analyze the out-of-sample predictions of the succeeding 60 (10 min), 120 (20 min) and 180 (30 min) steps ahead. Resource usage values are aggregated at 10 sec time interval. To compute the value of fractional difference parameter (d), we analyze the the wavelet coefficient method (used in [7]) and the rescaled range analysis [13] method. We observed that both the methods gave comparable RMSE on out-of-sample predictions. As rescaled range analysis is a widely adopted method, we use it in our study for computing the value of fractional differencing parameter and hence generate multi-step ahead out-of-sample forecasts. Table I presents the RMSE for the iterative multi-step ahead out-of-samples forecasts of (i) PRESS (ii) AGILE (iii) ARIMA and (iv) NARNN without and with fractional differencing. It can be observed that the prediction results of each model are improved after performing fractional differencing. Fractional differencing captures the long memory in the time series and leads to better predictions. In the rest of the studies, we consider predictions obtained only using fractional differencing. It is also observed that NARNN predicts CPU usage better than other models. We can observe that as the prediction horizon is increased from 60 to 180 steps, the error in the resource usage prediction increases. This is because of the accumulation of errors in recursive multi-step ahead prediction. Figure 2 show the results of multi-step ahead predictions of PRESS, AGILE, ARIMA and NARNN resource prediction models at 60, 120 and 180 steps ahead. We can see that the prediction at 120 and 180 steps ahead is more erroneous due to accumulation of errors than the prediction obtained at 60 steps.

TABLE I
RMSE of multi-step ahead prediction of CPU usage

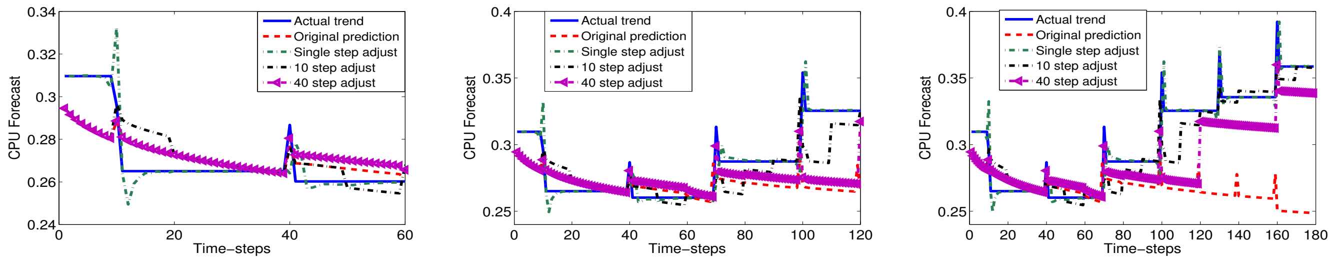
	Without fractional differencing			With fractional differencing		
	60	120	180	60	120	180
PRESS	0.0419	0.0685	0.0876	0.0260	0.0558	0.0753
AGILE	0.0386	0.0539	0.0761	0.0159	0.0535	0.0732
ARIMA	0.0340	0.0444	0.0570	0.0198	0.0308	0.0562
NARNN	0.0258	0.0431	0.0565	0.0135	0.0301	0.0551

To minimize the effect of accumulation of errors and handle changing workloads, we explore two variants of proposed approaches for adapting the time series prediction models (i) To adapt the time series prediction model after the actual value of every prediction is observed- at very step (ii) To adapt the time series prediction after observation of a fixed number of samples - after multiple steps. In the rest of the studies, we explore the gradient descent method and LMA to update the model parameters in ARIMA and NARNN (as both methods perform better than PRESS and AGILE). Table



(a) (b) (c)

Fig. 3. Adjusted online out-of-sample forecasts of CPU usage in ARIMA at (a) 60 (b) 120 (c) 180 steps



(a) (b) (c)

Fig. 4. Adjusted online out-of-sample forecasts of CPU usage in NARNN at (a) 60 (b) 120 (c) 180 steps

TABLE II
RMSE OF ADJUSTED ONLINE OUT-OF-SAMPLE PREDICTIONS IN ARIMA AND NARNN. HERE m IS OBSERVATION WINDOW

m	60 step ahead				120 step ahead				180 step ahead			
	ARIMA		NARNN		ARIMA		NARNN		ARIMA		NARNN	
	GD	LMA	GD	LMA	GD	LMA	GD	LMA	GD	LMA	GD	LMA
1	0.0070	0.0067	0.0068	0.0067	0.0072	0.0069	0.0071	0.0069	0.0078	0.0072	0.0076	0.0071
10	0.0144	0.0133	0.0128	0.0119	0.0170	0.0163	0.0140	0.0137	0.0173	0.0169	0.0146	0.0132
20	0.0151	0.0147	0.0130	0.0121	0.0181	0.0171	0.0162	0.0153	0.0192	0.0187	0.0183	0.0179
40	0.0162	0.0153	0.0133	0.0126	0.0237	0.0231	0.0229	0.0221	0.0249	0.0240	0.0243	0.0234

II presents the RMSE error of ARIMA and NARNN after updating the weights of the models after different number of real values of predictions (m) are observed (as explained in Section IV). We can see that the RMSE of the adjusted values is lower than actual predictions (Table I) leading to more accurate predictions. The results of LMA are better than gradient descent method for weight updation. Figures 3 and 4 show the out-of-sample predictions obtained after online adapation of prediction models using different number of observations using Levengberg-Marquardt algorithm. We can see that the adjusted out-of-sample forecasts are more closer to the real CPU usage values than the original multi-step ahead predictions. Figures 5 and 6 compare the RMSE and execution times of gradient descent and LMA in ARIMA. We observed similar trend in the NARNN. As LMA provides a greater flexibility in choosing the parameters, the RMSE in LMA is lower than gradient descent method. However, this flexibility comes at the cost of increased execution time. Updating the weights of time series prediction model at every step is a computational intensive task and involves high run time overhead. To update the weights of the prediction model, suitable values of η and μ need to be computed at every step so that the error in the current prediction is minimized and accurate next multi-step ahead predictions can be generated.

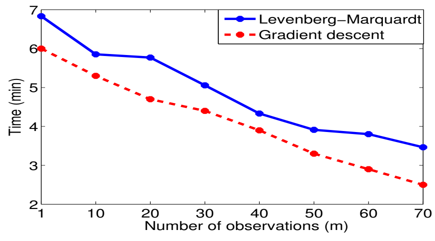


Fig. 5. Plot of number of observations vs execution time of CPU usage forecast in ARIMA in 180 step-ahead forecast

Figure 7 shows the plot of number of observations we observe before updating and the RMSE obtained. We see that as the number of steps we observe before updating increase, the error increases. Figure 8 presents the trend observed between number of predictions generated before updating the model and the execution time. We can see that as the number of observations, we pass before updating increase, the execution time decreases. Therefore, comparing the Figures 7

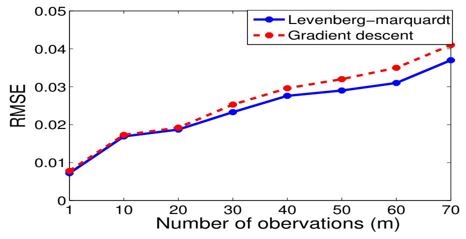


Fig. 6. Plot of number of observations vs RMSE of CPU usage forecast in ARIMA in 180 step-ahead forecast

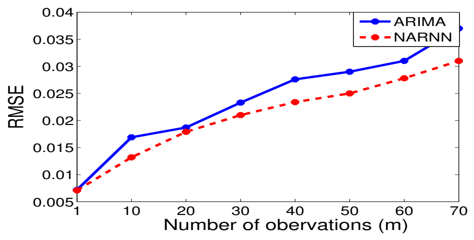


Fig. 7. Plot of number of observations vs prediction error of CPU usage forecast in ARIMA and NARNN in 180 step-ahead forecast

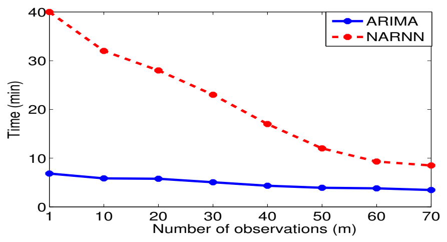


Fig. 8. Plot of number of observations vs execution time of CPU usage forecast in ARIMA and NARNN in 180 step-ahead forecast

and 8 we can say that there exists a trade-off between the run time overhead observed and accuracy of time series prediction. A suitable value of number of steps to be observed before updating can be selected such that the observed error and the run time overhead are within permissible limits.

VI. Conclusions

Forecasting resource usage is valuable for better scheduling and load balancing in Cloud. In our work, we explored the presence of long range dependence in the cloud workloads using Hurst exponent. We analyzed the multiple step ahead prediction resource prediction values of different prediction models and observed that as the prediction horizon increases, the errors in the predictions of models also increases. Therefore, we proposed updation of model parameters based on the error in the predicted and new observations. We analyzed gradient descent algorithm and LMA algorithm for updation of model parameters and observed that LMA performs better than gradient descent. We also analyzed the trade-off between the RMSE obtained and the run time overhead observed at different sizes of observation windows and conclude that a suitable size of observation window can be chosen such that the error and the run time overhead are within permissible limits. In the future, we would like to generalize our work to other time series prediction models such as Markov models.

Acknowledgements

We are deeply grateful to our Director Prof. Timothy A. Gonsalves for his invaluable guidance and support throughout the work.

References

[1] Q. Zhang, L. Cheng, and R. Boutaba, “Cloud computing: state-of-the-art and research challenges,” *Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, 2010.

[2] T. Kaur and I. Chana, “Energy efficiency techniques in cloud computing: A survey and taxonomy,” *ACM Computing Surveys*, vol. 48, no. 2, pp. 22:1–22:46, 2015.

[3] Z. Gong, X. Gu, and J. Wilkes, “PRESS: PRedictive Elastic ReSource Scaling for cloud systems,” in *International Conference on Network and Service Management (CNSM)*, 2010, pp. 9–16.

[4] H. Nguyen, Z. Shen, X. Gu, S. Subbiah, and J. Wilkes, “AGILE: elastic distributed resource scaling for infrastructure-as-a-service,” in *10th International Conference on Autonomic Computing (ICAC)*, 2013, pp. 69–82.

[5] Q. Zhang, M. F. Zhani, S. Zhang, Q. Zhu, R. Boutaba, and J. L. Hellerstein, “Dynamic energy-aware capacity provisioning for cloud computing environments,” in *International Conference on Autonomic Computing (ICAC)*. ACM, 2012, pp. 145–154.

[6] F. Caglar and A. Gokhale, “iOverbook: Intelligent resource-overbooking to support soft real-time applications in the cloud,” in *IEEE 7th International Conference on Cloud Computing (CLOUD)*, 2014, pp. 538–545.

[7] M. Ghorbani, Y. Wang, Y. Xue, M. Pedram, and P. Bogdan, “Prediction and control of bursty cloud workloads: A fractal framework,” in *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, 2014, pp. 1–9.

[8] C. Reiss, J. Wilkes, and J. L. Hellerstein, “Google cluster-usage traces: format + schema,” Nov. 2011.

[9] J. R. Hosking, “Fractional differencing,” *Biometrika*, vol. 68, no. 1, pp. 165–176, 1981.

[10] H. E. Hurst, “Long-term storage capacity of reservoirs,” *Transactions of the American Society of Civil Engineers*, vol. 116, pp. 770–808, 1951.

[11] L. Zhang, W.-D. Zhou, P.-C. Chang, J.-W. Yang, and F.-Z. Li, “Iterated time series prediction with multiple support vector regression models,” *Neurocomputing*, vol. 99, pp. 411–422, 2013.

[12] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, “Heterogeneity and dynamicity of clouds at scale: Google trace analysis,” in *Third ACM Symposium on Cloud Computing SoCC*. ACM, 2012, pp. 7:1–7:13.

[13] M. S. Granero, J. T. Segovia, and J. G. Prez, “Some comments on Hurst exponent and the long memory processes on capital markets,” *Physica A: Statistical Mechanics and its Applications*, vol. 387, no. 22, pp. 5543 – 5551, 2008.

[14] H. Shen, J. Z. Huang, and C. Lee, “Forecasting and dynamic updating of uncertain arrival rates to a call center,” in *International Conference on Service Operations and Logistics, and Informatics (SOLI)*. IEEE, 2007, pp. 1–6.

[15] C. Liu, S. C. Hoi, P. Zhao, and J. Sun, “Online ARIMA algorithms for time series prediction,” in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[16] C. Richard, J. C. M. Bermudez, and P. Honeine, “Online prediction of time series data with kernels,” *IEEE Transactions on Signal Processing*, vol. 57, no. 3, pp. 1058–1067, 2009.

[17] H. Guo and S. B. Gelfand, “Analysis of gradient descent learning algorithms for multilayer feedforward neural networks,” *IEEE transactions on circuits and systems*, vol. 38, no. 8, pp. 883–894, 1991.

[18] H. Yu and B. M. Wilamowski, “Levenberg–marquardt training,” *Industrial Electronics Handbook*, vol. 5, no. 12, p. 1, 2011.