Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 340 (2018) 41–54

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Failure Analysis Modelling in an Infrastructure as a Service (Iaas) Environment

Bashir Mohammed, Babagana Modu, Kabiru M Maiyama
Hassan Ugail, Irfan Awan[1]

School of Electrical Engineering and Computer Science,
University of Bradford Bradford, UK

Mariam Kiran[2]

Energy Science Network (ESnet)
Lawrence Berkeley National Lab
Berkeley, California, USA

Abstract

Failure Prediction has long known to be a challenging problem. With the evolving trend of technology and growing complexity of high-performance cloud data centre infrastructure, focusing on failure becomes very vital particularly when designing systems for the next generation. The traditional runtime fault-tolerance (FT) techniques such as data replication and periodic check-pointing are not very effective to handle the current state of the art emerging computing systems. This has necessitated the urgent need for a robust system with an in-depth understanding of system and component failures as well as the ability to predict accurate potential future system failures. In this paper, we studied data in-production-faults recorded within a five years period from the National Energy Research Scientific computing centre (NERSC). Using the data collected from the Computer Failure Data Repository (CFDR), we developed an effective failure prediction model focusing on high-performance cloud data centre infrastructure. Using the Auto-Regressive Moving Average(ARMA), our model was able to predict potential future failures in the system. Our results also show a failure prediction accuracy of 95%, which is good. We thus believe that our strategy is practical and can be adapted to use in existing real-time systems.

Keywords: Keywords: Failure Prediction, Iaas, Replication, HPC, Checkpointing

1 Introduction

Failure is the inability of a component or system to execute a required task or function according to its specification. As our growing needs for computing re-

[1] Email: b.mohammed1@bradford.ac.uk
[2] Email: mkiran@es.net

https://doi.org/10.1016/j.entcs.2018.09.004
1571-0661/

sources and performance keeps on fueling the development of high-performance systems and cloud data centres, there continue to be a growing need for an in-depth understanding of cloud system failure and errors as well as their statistical and empirical properties[3]. The understanding of this key properties can help in predicting possible failures and evaluating the effectiveness of different techniques for improving the overall system availability. It could also empower researchers in designing and developing a new state of the art solutions for both cloud service providers and customers. Cloud computing is a term used in describing a broad range of online services. According to the National Institute of Standards and Technology (NIST)[32], cloud computing is defined as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. A cloud computing service provides convenient, scalable, on-demand access to a pool of online computing resources, such as virtual machines, compute server clusters or networked storage[6]. It is pertinent to note that the size and complexity of cloud computing systems have made failure to become inevitable. These failures could be caused by the system having insufficient resources or an unexpected failure in the system itself. Therefore it is of critical importance to ensure the reliability and availability of such systems[25].

There is also an urgent need for cloud service providers (CSP) to offer a scalable, efficient and reliable on-demand resource to their customers in the presence of faults thereby fulfilling their service level agreement (SLA). Component failures within the cloud infrastructure are common, but large cloud data centres should be designed to guarantee a certain level of availability to the business system[24]. Infrastructure-as-a-Service (IaaS) cloud presents computational resources (e.g., CPU and memory), storage resources and networking capacity that ensures high availability in the face of such failures [1]. Cloud systems can have tremendous failure rates as they feature many servers that are geographically dispersed with a high workload. The availability of such systems can be quickly endangered if the failure is not sufficiently handled [1].To guarantee the availability of services to cloud users, cloud infrastructures should be designed such that they should have minimal or insignificant system downtime. Replication of data and checkpointing technique are some of the common existing strategies used to ensure availability of cloud services [2].

Failure prediction is necessary for predictive maintenance due to its ability to prevent failure incidents and maintenance costs[38]. Predictive maintenance is about anticipating failures and taking proactive actions[40]. Recent advances in machine learning and cloud storage have created a great opportunity to utilize the huge amount of data generated from cloud infrastructures which provide room to predict when a component is likely to malfunction or fail[3]. Currently, mathematical and statistical modelling are the prominent approaches used for failure predictions, these are based on equipment degradation, physical models and machine learning techniques respectively [29]. According to [4], cloud computing is usually associated with failures. The risk of failure can be viewed as the possibility of suffering loss or exposure in the cloud-computing life cycle [33]-[36]. Generally,

cloud computing risk management consists of processes, approaches, and techniques that are employed to reduce cloud computing risks failure.

There are three levels of essential services offered by cloud computing: Infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS).

- Infrastructure as a service (IaaS) is the most basic and important cloud service model under which virtual machines, load balancers, fault tolerance, firewalls and networking services are provided. The client or cloud user is provided with the capability to provision processing, storage, network and other fundamental computing resources to deploy and run arbitrary software such as operating system and applications. Common examples of these services include Rackspace, GoGrid, EC2, Google Apps, Concur, Cisco Webex, Citrix GoTo Meetings, Adobe Marketing Cloud, Facebook, Flickr) and Amazon cloud [8].
- Under the PaaS model, a computing platform including APIs, operating system and development environment are provided as well as programming language execution environment and web servers. The client maintains the applications, while the cloud provider maintains the service run-time, databases, server software, integrated server oriented architectures and storage networks. Various types of PaaS vendors offerings can include complete application hosting, development, testing and extensive integrated services that include scalability and maintenance. Some key players include Microsoft Windows Azure and Google Apps engine GoDaddy, Windows Azure, Apprenda, Google App Engine, Amazon Web Services, WordPress. The main benefit of these services includes focus on high-value software rather than infrastructure, leverage economies of scale and provide scalable go-to-market capability [39].
- SaaS provides clients with the capability to use provider application executing on a cloud infrastructure. An entire application is available remotely and accessible from multiple client devices through thin client interfaces such as web browsers. Cloud user does not manage or control the underlying cloud infrastructure [2]but providers install and operate the application software. Example providers for this service include Salesforce, Facebook and Google Apps, Amazon EC2, Rackspace, Microsoft Azure, Google Compute Engine and Amazon Web Services [9]-[11].

The aim of this research is to develop an accurate model for predicting future failure trends of system components in a high-performance data centre cloud infrastructure.

The main contribution of this work is as follows:

- To investigate the significance of analysing and predicting publicly available failure dataset.
- To develop and test time series models failure prediction, sensitivity and accuracy for a distributed computing system.

- To estimate the availability characteristics of the selected failure dataset in precise quantitative terms.
- To test and validate the prediction accuracy of our developed model using data in-production-failure dataset.

The remaining sections of this paper are organized as follows: Section 2 presents a brief summary of work related to this study. Section 3 presents a vivid description of our methodology, our proposed system model formulation and a brief overview of our dataset. Section 4 discusses the results, the analysis of the system failure distribution across the time under study and the failure prediction. Section 5 finally concludes the paper.

2 Related Work

Failures impact on availability and dependability of cloud data centre infrastructure, high-performance computing and distributed server systems has gained enormous attention in recent times. However, very few work has attempted to fully analyse and predict high performance and cloud failure data characteristics empirically. The authors in [12] have made a good attempt to analyse the failure data of a large-scale production cloud environment consisting of over 12,500 servers, which includes a study of failure and repair times and characteristics for both cloud workloads and servers, but they never looked at the failure correlation between workload intensity and size of the system respectively.
Authors in [13] characterizes the hardware reliability of Cloud datacenters from a number of data sources, but failed to analyse the failure of workloads and did not utilised publicly available dataset in their experiment. Kavulya et al.[14] present workload failure characteristics from a production MapReduce supercomputing cluster, but this work is confined to MapReduce type jobs and does not consider workload repair or server failure characteristics. They also did not utilise publicly available dataset in their work. The authors in [3] used the Bayesian network to predict failure probabilities. While the research seemed interesting, they did not disclose the dataset they used, thus making it hard to replicate or compare other Machine Learning (ML) Algorithms with their proposed model.[15] used an ensemble classifier to achieve hard drive failure prediction on a cloud infrastructure. The data they conducted their work on was acquired through two sources; Windows performance counts and Self-Monitoring, and Analysis and Reporting Technology (S.M.A.R.T or SMART). This research closely resembles the intended work. However, they only considered hard disk failure in their cloud architecture while business systems rely on other components (such as CPU, Disk, DIMM, Cable.etc) and not only hard drive. Recently,[16] used data acquired from cycles to predict Integrated Circuit (IC) failures. As in the case of [15] they also considered only one hardware failure occurrence. They analyzed fourteen (14) hardware samples but the data they used has not been made publicly available.

Our approach is to use publicly available hardware dataset to gain a machine learning (ML) classifier to predict hardware failures, contrary to most of the state-

of- the artworks. We decided to use a public dataset to enable other researchers in the field to compare their outcome with our obtained results. Furthermore, in this work we are not limiting our experiments to a single hardware, rather we attempt to predict several component failures. For a more comprehensive review of other works of literature by other scholars, the reader is referred to[17]-[25].

3 Methodology

This section presents a vivid description of our methodology. As the study focus on data-driven modelling of system failure frequency in a large-scale cloud-based infrastructure. Fundamentally, a failure data stream is usually time-dependent and are recorded over a given regular span of time. This scenario makes the possibility of applying time series models such as auto-regressive (AR) and moving-average (MA) and see the ramification[30]. For the preliminary analysis and the time series modeling[31], R programming language version 3.4.1 would be deployed.

3.1 Time Series

We define time series in the context of cloud-based infrastructure as a number of failures occurred to a system over a given period of time. Let $X_1, X_2, X_3, \cdots, X_t$ be the number of failures of a system, and is mathematically defined as:

$$X_t \;=\; f(X_{t-1}, X_{t-2}, X_{t-3}, \cdots, X_{t-n}) + \varepsilon_t \tag{1}$$

where X_t is the value of X at time t, then $X_{t-1}, X_{t-2}, X_{t-3}, \cdots, X_{t-n}$ represents the past values of X_t, and ε_t denotes white noise which has the distribution $\varepsilon_t \;\sim\; WN(0, \sigma^2)$. The ε_t is a stochastic term which does not follow any pattern and cannot be predicted. Basically, system failures are random, but it is rarely deterministic in a narrow sense from some identifiable causes.

For several decades, time series models have been utilized in all fields of study for prediction [28]. The models like autoregressive (AR), moving average (MA) and exponential smoothing ranging from linear to non-linear regression and a host of many others. Box and Jenkins [31] developed a classical time series model called autoregressive integrated moving average (ARIMA). These techniques were successfully applied in various domians such as data centre, complex industrial system and transportation networks and healthcare to predicts the failure of their systems [28]-[31].

3.1.1 Autoregressive process (AR)

Suppose the time series $\{X_t\}$ at time t has p past values $X_{t-1}, X_{t-2}, X_{t-3}, \cdots, X_{t-p}$, then AR process of order p is denoted by $AR(p)$ defined as:

$$X_t \;=\; \vartheta_1 X_{t-1} + \vartheta_2 X_{t-2} + \vartheta_3 X_{t-3} + \cdots + \vartheta_p X_{t-p} + \varepsilon_t, \tag{2}$$

where $\varepsilon_t \;\sim\; WN(0, \sigma^2)$ and ε_t is uncorrelated with X_r for each $r \;<\; t$. Using backshift operator (2) can be written in short form as concise by $X_t \;=\; \vartheta(L)^{-1}\varepsilon_t$.

In this process, the failure of a system dependent on the past causes or failure.

3.1.2 Moving average process (MA)

This process is memoryless because the failure of a system does not depends on past causes or failure. In many situations, cloud-based infrastructures failed erratically and this type of process could be best to describe such scenario. Let the time series $\{X_t\}$ is a moving average of order q denoted by $MA(q)$ and mathematically defined as:

$$X_t = \varepsilon_t + \phi_1\varepsilon_{t-1} + \phi_2\varepsilon_{t-2} + \phi_3\varepsilon_{t-3} + \cdots + \phi_q\varepsilon_{t-q} \tag{3}$$

where $\varepsilon_t \sim WN(0, \sigma^2)$ and $\phi_1, \phi_2, \phi_3, \cdots, \phi_q$ are constants. The X_t is a linear combination of $q+1$ white noise variables and are uncorrelated for all lags $s > q$, e.g X_t and X_{t-s}. The $MA(q)$ process can written in short form as $X_t = \phi(L)\varepsilon_t$

3.1.3 Autoregressive moving average (ARMA)

This process is a hybrid of AR and MA where the pattern of failure of a system can be attributed to two causes. The $ARMA$ model is a combination AR and MA models of order p and q respectively. Then, $ARMA(p, q)$ model is given by:

$$\vartheta(L)X_t = \phi(L)\varepsilon_t \tag{4}$$

3.2 Overview of NERSC dataset

This NERSC data [26] was collected with the purpose of providing failure specifics for I/O related systems and components in as much detail as possible so that analysis might produce some useful findings. Data were collected for storage, networking, computational machines, and file systems for production use at NERSC from the 2001-2006 time frame. The data was extracted from a database used for tracking system troubles, called Remedy, and is currently stored in a MySQL database and available for export to Excel format. As part of the SciDAC Petascale Data Storage Institute (PDSI) project Collaboration this is the failure data for the High-Performance Computing System-2 (MPP2) operated by the Environmental and Molecular Science Laboratory EMSL), Molecular Science Computing Facility (MSCF)[26], [27].

The MPP2 computing system has the following equipment and capabilities:

- HP/Linux Itanium-2
- 980 node/1960 Itanium-2 processors (Madison, 1.5 GHz) configured as follows:
- 574 nodes are "fat" compute nodes with 10 Gbyte RAM and 430 Gbyte local disk
- 366 nodes are "thin" compute nodes with 10 Gbyte RAM and 10 Gbyte local disk
- 34 nodes are Lustre server nodes (32 OSS, 2 MDS)
- 2 nodes are administrative nodes
- 4 nodes are login nodes

- Quadrics QsNetII interconnect
- 11.8 TFlops peak theoretical performance
- 9.7 terabytes of RAM
- 450 terabytes of local scratch disk space
- 53 terabytes shared cluster file system, Lustre

3.3 Proposed System Model

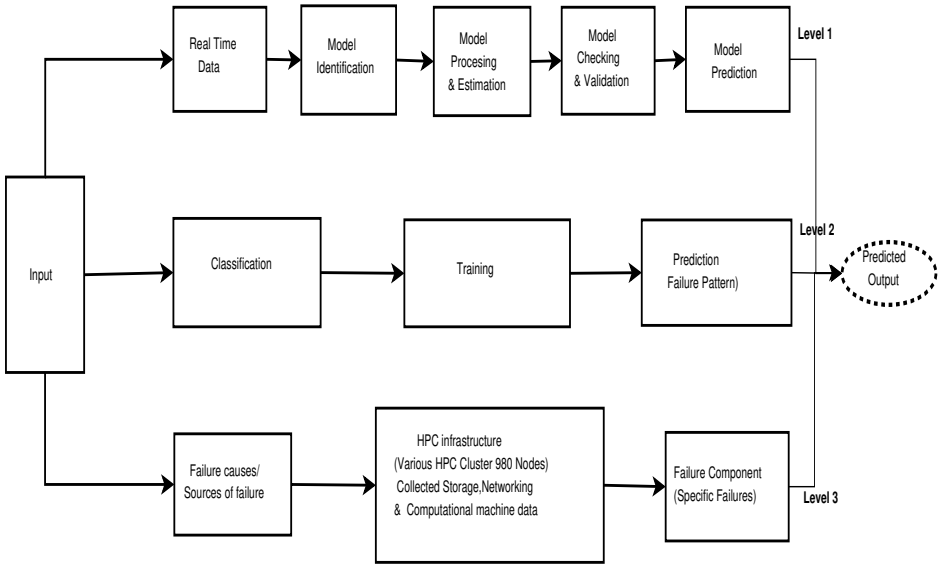

Fig. 1. Proposed System Model

Our proposed system model comprises three different levels as depicted in Fig.1. However, this research will only focus on Level-1 and Level-2. Level-3 is out of the scope of this work because the sources of failure are not considered.
Our failure prediction model is made up of the following main components:

- Identification
  The model identification is the first stage of building time series model after stationarity is achieved. With the aid of the autocorrelation function (ACF) and partial autocorrelation function (PACF), we can identify the appropriate model based on the pattern and order shown by the correlogram.
- Parameter Estimation
  After the appropriate model is identified, next is the estimation of parameters using some conventional techniques such as least squares method, maximum likelihood estimation and method of moment etc.
- Evaluation
  In this stage, the model is to be checked for accuracy and validation, even though postulation has been made that all models are wrong but some are better than others. For example, considering the properties of the residuals and check whether

the residuals from an ARMA is a normal regular distribution or random.

- Prediction
  At this stage, the identified model would be used to forecast future ahead. The estimated residuals of the model would be carefully examined to follow a white noise process.

4 Results and Discussion

In this section, we present the analysis of system failure distribution across the time under study. The model formulation and their properties, prediction and evaluation are also presented.

4.1 Preliminary analysis

In Fig.2, we present the frequency of pooled system failure using the histogram and also by superimposing a normal density function. We observed that the frequency of the system failure is fairly distributed normally. Furthermore, we performed normality test using Shapiro-Wilk test at $\alpha = 5\%$ level of significant and the $p - value = 0.02$. The null hypothesis has to be rejected and conclude that the frequency of the system failure is normally distributed.

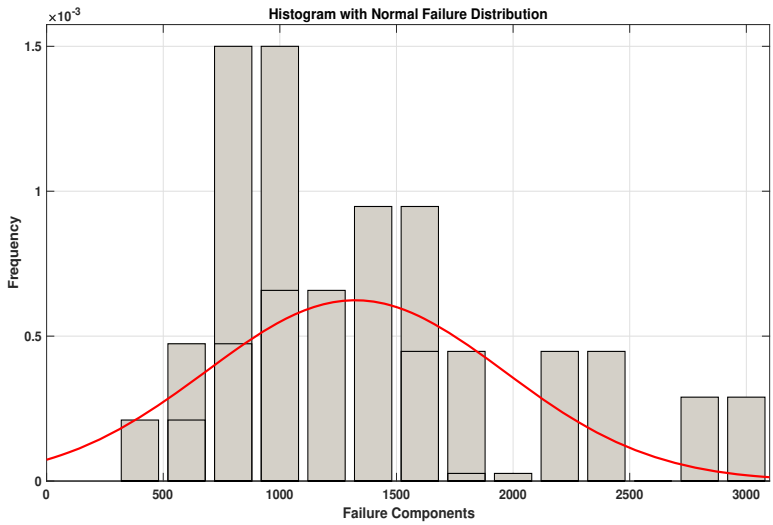

Fig. 2. Normality test of the failure data

4.2 System failure model

In Fig.3, we plot the frequency of time-dependent system failure in order to understand the pattern of its occurrence. The pattern of the system failure shows that it is not stationary as the mean and variance of the series keep changing over time.

To remedy this scenario, we need to deploy some technique of data transformations such as log-transformation, trig-transformation, differencing method and also discuss their properties.

We present the plots of system failure frequency transformation in Fig.4(a)–(d) using differencing, log, cosine and sine respectively. The log-transformation would not be appropriate in this case because there are some indefinite outcomes as a result of zero recorded system failure. However, the trig-transformation cosine and sine are also not appropriate. This is shown by the pattern of the system failure where the means and variances of the two series are considerably unstable. In this study, we choose differencing method over log, cosine and sine. This is because the pattern exhibited by the differencing shows that mean and variance of system failure series is fairly constant.

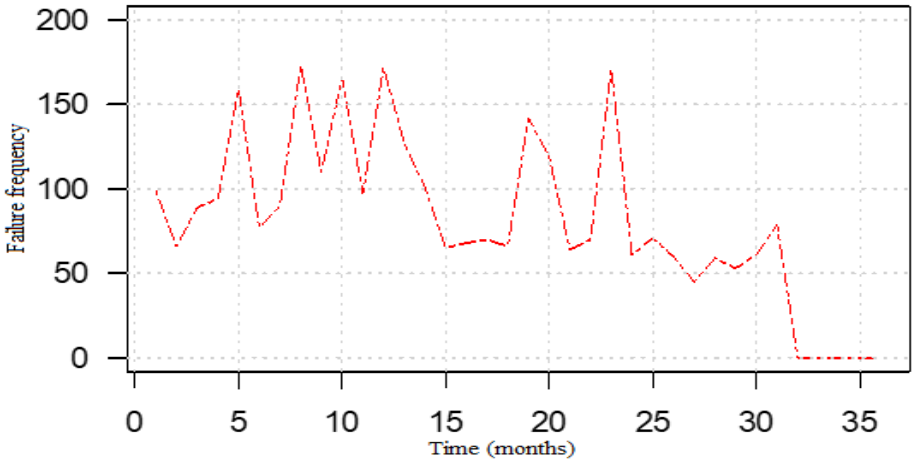

Fig. 3. Time Series Plot for Failed Components

In Table1, we present the summary of the frequency of the system failure series transformation using differencing, log-transformation and trig-transformation and also their limitations.

Table 1
Data Normalization Comparison Table

Transformation function	**Outcome**
Differencing	The mean and variance are stationary
Log	This transformation is indeterminant at some values
Cosine	The mean and variance are not stationary
Sine	The mean and variance are not stationary

4.3 Model Identification

We plots correlogram showing autocorrelation function (ACF) and partial autocorrelation function (PACF) of the system failure series (see Fig.5(a)–(b)). Using the

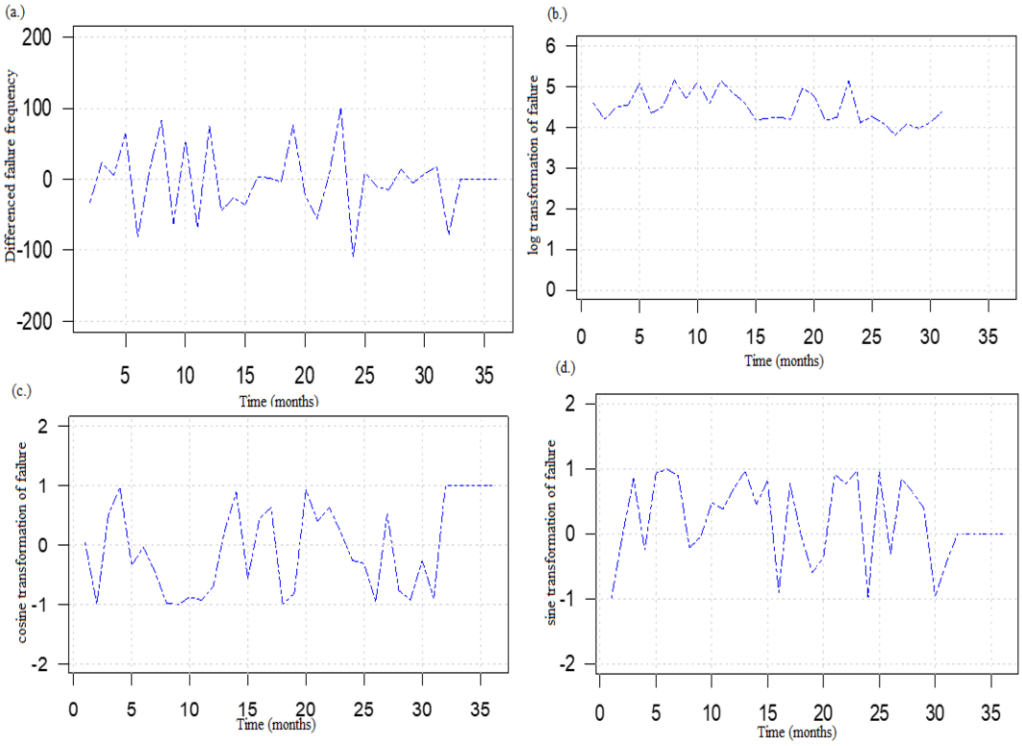

Fig. 4. Failure Component data transformation process

ACF correlogram, we were able to identify a moving average model of order 1, MA (1). While the autoregressive model of order 1 as well, AR (1). The combination of the two models gives ARIMA (1,1,1) model, where 1 at the centre is the number of times the model is differenced. The system failure frequency of the series was differenced ones before the stationarity was achieved. All the values of the auto-correlation that fall within the two blue dotted lines are indicating non-significant at 95%. While those values that fall outside the 95% confidence interval indicates that they are significant.

4.3.1 Estimation of Parameter

Having identified the ARIMA (1,1,1) model for the system failure frequency, we can then estimate the parameters of the model. The ARIMA (1,1,1) model is mathematically represented by

$$y_t = \phi y_{t-1} + \varepsilon_t + \vartheta_1 \varepsilon_{t-1} \tag{5}$$

where ε_t is the random shock occurring at time t, which has the distribution as $\varepsilon_t \sim WN(0, \sigma^2)$. We estimated the following parameters $\phi_1 = -0.1016$ and $\vartheta_1 = -0.5784$, log-likelihood = -178.3, AIC = 362.61, and the association standard errors of the model are 0.2531 and 0.2085 respectively. We therefore write the

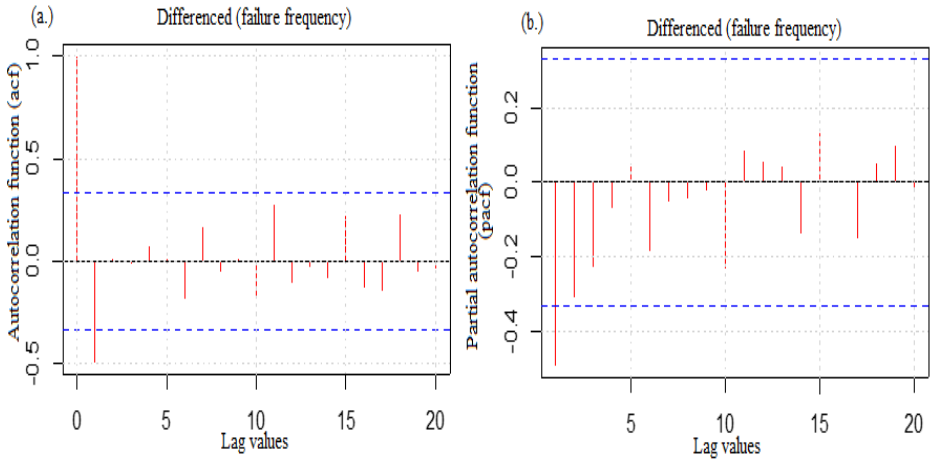

Fig. 5. ACF and PACF plots

ARIMA (1,1,1) model for system failure frequency as

$$y_t = -0.1016_{t-1} + \varepsilon_t - 0.5784\varepsilon_{t-1} \tag{6}$$

where $\varepsilon_t \sim WN(0, 1.534)$.

We present in Fig.6, the prediction capability of ARIMA (1,1,1) and we also evaluate the accuracy of the model. After evaluating the model, we were able to obtain the following indices RMSE = 38.6%, MAE = 31.6% and MASE = 23.5% respectively. This shows that the model is very robust as it gives error allowance of less than 40%.

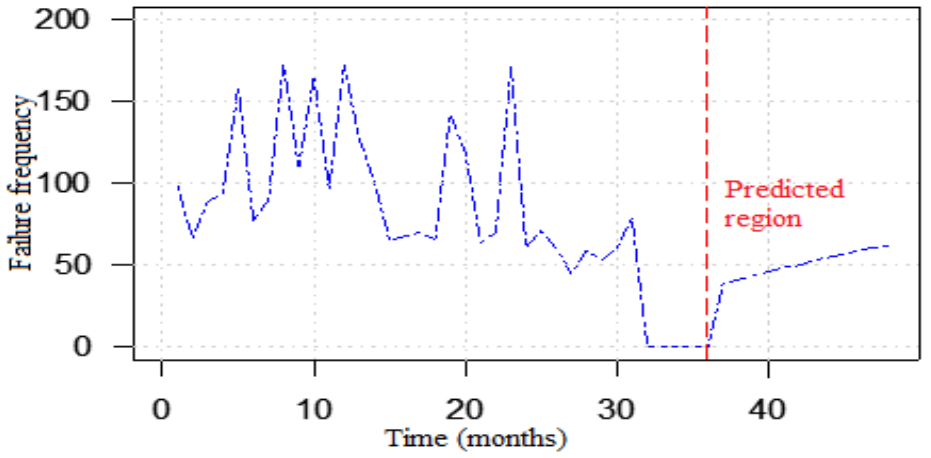

Fig. 6. This figure shows two regions, the first is the pattern of system failure frequency and while the second is the prediction region of the failure series..

5 Conclusion

In this research time series models for predicting failures have been examined and the results have been discussed. The significance of analysing and predicting component failures has been studied in this paper and verified through literature review. This research was undertaken with the prior motive to develop an adequate model for predicting future failure trends of components in a high-performance data centre cloud infrastructure. It evolved that ARIMA (1, 1, 1) fits the data well. The trend analysis indicated that the disk rate of failure is increasing at a fast rate. Therefore there should be more attention and focus on the disk component of the infrastructure. The model also proved that the model is adequate for predicting of monthly failure rates of components. Moreover, the model was found to have a good fit of 95% accuracy, hence appropriate for the study. In the future, we will further examine a huge real-time publicly available dataset like Google cluster by applying different machine learning approaches and comparing them to enable us to improve and get a more accurate prediction.

Acknowledgement

The authors would like to thank the anonymous reviewers. Their valuable comments and suggestions made the presentation of this paper much improved. One of the authors Bashir Mohammed is a Petroleum Technology Development Fund (PTDF) scholar. We would like to express our sincere gratitude to PTDF for their funding support.

References

[1] R. Ghosh, L. Francesco, F. Frattini, S. Russo, and S. T. Kishor, Scalable analytics for IaaS cloud availability,*IEEE Trans. Cloud Comput.*, vol. 2, no. 1, pp. 5770, 2014.

[2] T. Chalermarrewong, T. Achalakul, and S. C. W. See, The Design of a Fault Management Framework for Cloud,*2012 9th Int. Conf. Electr. Eng. Comput. Telecommun. Inf. Technol.*, pp. 14, 2012.

[3] A. Abu-Samah, M. K. Shahzad, E. Zamai, and A. Ben Said, Failure prediction methodology for improved proactive maintenance using Bayesian approach, IFAC Proc. Vol., vol. 48, no. 21, pp. 844851, 2015.

[4] A. Elzamly, B. Hussin, A. Samad, H. Basari, and C. Technology, Classification of Critical Cloud Computing Security Issues for Banking Organizations: A cloud Delphi Study, Int. J. Grid Distrib. Comput., vol. 9, no. 8, pp. 137158, 2016.

[5] C. Modi, D. Patel, B. Borisaniya, A. Patel, and M. Rajarajan, A survey on security issues and solutions at different layers of Cloud computing, J. Supercomput., vol. 63, no. 2, pp. 561592, 2013.

[6] D. Gnanavelu and D. G. Gunasekaran, Survey on Security Issues and Solutions in Cloud Computing, Int. J. Comput. Trends Technol., vol. 8, no. 8, pp. 126130, 2014.

[7] B. Wang, Y. Zheng, W. Lou, and Y. T. Hou, DDoS attack protection in the era of cloud computing and Software-Defined Networking, Comput. Networks, vol. 81, pp. 308319, 2015.

[8] Z. Pantic and M. Babar, Guidelines for Building a Private Cloud Infrastructure, ITU Tech. Rep. - TR-2012-153TR-2012-153, 2012.

[9] O. Sefraoui, M. Aissaoui, and M. Eleuldj, Cloud computing migration and IT resources rationalization, 2014 Int. Conf. Multimed. Comput. Syst., pp. 11641168, Apr. 2014.

[10] A. Sen and S. Madria, Off-Line Risk Assessment of Cloud Service Provider, 2014 IEEE World Congr. Serv., pp. 5865, Jun. 2014.

[11] S. Yadav, Comparative Study on Open Source Software for Cloud Computing Platform: Eucalyptus , Openstack and Opennebula, Res. Inven. Int. J. Eng. Sci. Vol.3, Issue 10, vol. 3, no. 10, pp. 5154, 2013.

[12] P. Garraghan, P. Townend, and J. Xu, An empirical failure-analysis of a large-scale cloud computing environment, Proc. - 2014 IEEE 15th Int. Symp. High-Assurance Syst. Eng. HASE 2014, pp. 113120, 2014.

[13] K. V. Vishwanath and N. Nagappan, Characterizing Cloud Computing Hardware Reliability, Proc. 1st ACM Symp. Cloud Comput. - SoCC 10, p. 193, 2010.

[14] S. Kavulya, J. Tany, R. Gandhi, and P. Narasimhan, An analysis of traces from a production MapReduce cluster, CCGrid 2010 - 10th IEEE/ACM Int. Conf. Clust. Cloud, Grid Comput., pp. 94103, 2010.

[15] A. Khan, B. Bussone, J. Richards, and A. Miguel, A practical Approach to Hard Disk Failure Prediction in Cloud Platforms, in 2016 IEEE Second International Conference on Big Data Computing Service and Applications, 2016, pp. 105116.

[16] G. H. Thomas Gentner, Klau p. Gungl, Patent US9319030 - Integrated circuit failure prediction using clock duty cycle recording and, 2016.

[17] S. Keke Gai, Meikang Qiu, Security-Aware Information Classifications Using Supervised Learning for Cloud-Based Cyber Risk Management in Financial Big Data, in 2016 IEEE International Conference on Intelligent Data and Security, 2016, pp. 197202.

[18] L. Zhang, K. Rao, R. Wang, and Y. Jia, Risk Prediction Model Based on Improved AdaBoost Method for Cloud Users, Open Cybern. Syst. Journal, 2015, vol. 9, pp. 4449, 2015.

[19] D. Pop, Machine Learning and Cloud Computing: Survey of Distributed and SaaS Solutions, Inst. e-Austria Timisoara, Tech. Rep 1, 2012.

[20] S. Bsch, V. Nissen, and A. Wnscher, Automatic classification of data-warehouse-data for information lifecycle management using machine learning techniques, Inf. Syst. Front., 2016. [

[21] D. Fall, T. Okuda, Y. Kadobayashi, and S. Yamaguchi, Risk Adaptive Authorization Mechanism (RAdAM) for Cloud Computing, J. Inf. Process., vol. 24, no. 2, pp. 371380, 2016.

[22] C. Guo, Y. Liu, and M. Huang, Obtaining Evidence Model of an Expert System Based on Machine Learning in Cloud Environment, J. Internet Technol., vol. 16, no. 7, pp. 13391349, 2015.

[23] Z. Amin, N. Sethi, and H. Singh, Review on fault tolerance techniques in cloud computing, Int. J. Comput. Appl., vol. 116, no. 18, pp. 1117, 2015.

[24] A. Pellegrini, P. Di Sanzo, and D. R. Avresky, Proactive Cloud Management for Highly Heterogeneous Multi-cloud Infrastructures, in 2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), 2016, pp. 13111318.

[25] B. Mohammed, M. Kiran, I.-U. Awan, and K. M. Maiyama, An Integrated Virtualized Strategy for Fault Tolerance in Cloud Computing Environment, 2016 Intl IEEE Conf. Ubiquitous Intell. Comput. Adv. Trust. Comput. Scalable Comput. Commun. Cloud Big Data Comput. Internet People, Smart World Congr., pp. 542549, 2016.

[26] B. Schroeder and G. Gibson, The computer failure data repository (CFDR), Reliab. Anal. Syst. Fail. Data , no. March, p. 6, 2007.

[27] B. Schroeder and G. Gibson, The Computer Failure Data Repository (CFDR): collecting, sharing and analyzing failure data, SC 06 Proc. 2006 ACM/IEEE Conf. Supercomput., no. March, p. 154, 2006

[28] T. Chalermarrewong, T. Achalakul, and S. C. W. See, Failure Prediction of Data Centers Using Time Series and Fault Tree Analysis, 2012 IEEE 18th Int. Conf. Parallel Distrib. Syst., pp. 794799, 2012.

[29] Q. Fan and H. Fan, Reliability Analysis and Failure Prediction of Construction Equipment with Time Series Models, J. Adv. Manag. Sci., vol. 3, no. 3, pp. 203210, 2015.

[30] Y. Zhou, Failure Trend Analysis Using Time Series Model, 2017 29th Chinese Control and Decision Conference., no. 1, pp. 859862, 2017.

[31] S. Ho, M. Xie, and T. Goh, A comparative study of neural network and Box-Jenkins ARIMA modeling in time series prediction, Comput. Ind. Eng., vol. 42, no. 24, pp. 371375, 2002.

[32] P. Mell, T. Grance, and T. Grance, The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology, Natl. Inst. Stand. Technol. Spec. Publ. 800-145 7 pages, 2011.

[33] Y. Jararweh, Z. Alshara, M. Jarrah, M. Kharbutli, and M. N. Alsaleh, TeachCloud: a cloud computing educational toolkit, no. 2012, pp. 116.

[34] R. Jhawar, V. Piuri, and I. Universit, Fault tolerance management in IaaS clouds, 2012 IEEE First AESS Eur. Conf. Satell. Telecommun., pp. 16, 2012.

[35] Bala A, Chana I. Fault tolerance-challenges, techniques and implementation in cloud computing. International Journal of Computer Science 2012; 9(1): 288293.

[36] S. Shen, A. Iosup, A. Israel, W. Cirne, D. Raz, and D. Epema, An availability-on-demand mechanism for datacenters, 2015 15th IEEE/ACM Int. Symp. Clust. Cloud Grid Comput., pp. 495504, 2015.

[37] B. Mohammed and M. Kiran, Analysis of cloud test beds using opensource solutions, 2015 3rd Int. Conf. Futur. Internet Things Cloud, pp. 195203, 2015.

[38] D. Sun, G. Chang, C. Miao, and X. Wang, Analyzing, modeling and evaluating dynamic adaptive fault tolerance strategies in cloud computing environments, J. Supercomput., vol. 66, no. 1, pp. 193228, 2013.

[39] B. Mohammed, M. Kiran, K. M. Maiyama, M. M. Kamala, and I.-U. Awan, Failover strategy for fault tolerance in cloud computing environment. Software: Practice and Experience., (2017). doi:10.1002/spe.2491

[40] V. S. Kushwah, S. K. Goyal, and P. Narwariya, A Survey on Various Fault Tolerant Approaches for Cloud Environment During Load Balancing, Int. J. Appl. Res. Inf. Technol. Comput., vol. 3, no. 3, pp. 385394, 2014.