Optimized Personalized Viable SLA Management Framework for Small and Medium Providers to Avoid SLA Violation in Cloud

A thesis submitted in fulfilment of the requirements for the award of the degree of Doctor of Philosophy by

Walayat Hussain

Faculty of Engineering and information technology

University of Technology Sydney

December 2016

Certificate of Authorship / Originality

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Candidate

Date

Abstract

In today's competitive world, service providers need to be customer-focused and proactive in their marketing strategies to create consumer awareness of their services. Cloud computing provides an open and ubiquitous feature of computing in which a large random number of consumers can interact with providers and request services. In such an environment, there is a need for intelligent and efficient methods that increase confidence in the successful achievement of business requirements. One such method is the Service Level Agreement (SLA), comprising service objectives, business terms, service relations, obligations and the possible action to be taken in the case of SLA violation. In the current literature, most of the emphasis until now has been on the formation of meaningful SLAs by service consumers, by which their requirements will be met. However, in an increasingly competitive market based on the cloud environment, service providers too need a framework that will form a viable SLA, predict possible SLA violations before they occur, and generate early warning alarms that flag a potential lack of resources. This is because when a provider and a consumer commit to an SLA, the service provider is bound to reserve the agreed amount of resources for the entire period of that agreement – whether the consumer uses them or not. It is therefore very important for cloud providers to accurately predict the likely resource usage for a particular consumer and to formulate an appropriate SLA before finalizing the agreement. This problem is more important for a small to medium cloud service provider which has limited resources at stake that must be utilized in the best possible way to generate maximum revenue. A viable SLA in cloud computing is one that intelligently helps the service provider to determine the amount of resources to offer to a requesting consumer, and there are number of studies on SLA management in the literature. The aim of this research is two-fold. First, it presents a comprehensive overview of existing state-of-the-art SLA management approaches in cloud computing, their features and shortcomings in creating viable SLAs from the service provider's viewpoint. From a thorough analysis, we observe that the lack of a viable SLA management framework renders a service provider unable to make wise decisions in forming an SLA, which could lead to service violations and violation penalties. To fill this gap, our second contribution is the proposal of the Optimized Personalized Viable SLA (OPV-SLA) framework which assists a service provider to form a viable

SLA and start managing SLA violation before an SLA is formed and executed. The framework also assists a service provider to take an optimal decision in service formation and allocate the appropriate amount of marginal resources. It also provides an informed decision system to service provider to manage risks of SLA violation when a system identifies and predict likely violation. We demonstrate the applicability of our framework in forming viable SLAs through experiments. From the evaluative results, we observe that our framework helps a service provider to form viable SLAs and later to manage them to effectively minimize possible service violation and penalties.

Acknowledgement

All praise to Allah, the most gracious, the most merciful, for giving me the strength, opportunity and determination to accomplish my dreams and able to complete the journey of PhD study. His blessings enlighten my ways, all the goals I have achieved so far are due to His mercy and all the mistakes are mine. I am thankful to Allah, for giving me this opportunity to complete my Doctoral Dissertation under the supervision of Dr. Farookh Hussain and Dr. Omar Hussain.

I wish to acknowledge the efforts of my family specially my late Father and my late Mother without whose support, efforts and sacrifices I would never have been the person that I am today. I am thankful to my wife that always supports me, to my brother and my sisters, their kind words and their unwavering belief in me. To all my poor Hazara community that have been a victim of terrorism from a long time.

I would like to thank Dr. Farookh Hussain, my principal supervisor, for all of his support, kindness, gratitude, motivation and wisdom; without his help I could not have completed this journey. I am indebted to my co-supervisor Dr. Omar Hussain for all of his assistance, attention, wise suggestions and prompt help whenever I was stuck with different problems. I would never have been able to complete the major part of my thesis without his help. His smiling face is always a source of solace for me.

I would like to thank the University of Technology Sydney (UTS) for awarding me scholarships. Many thanks to Quantum Computation and Intelligent System lab, Faculty of Engineering and Information Technology and Graduate Research School for awarding me travel grants to present my research work and especially to QCIS for giving me the facility of proofreading my research work.

I would like to thank my friends and colleagues; Osama, Supannada, Asma, Mohammad, Ali, Quynh with whom I had a memorable time in UTS. I would like to thank Ms. Belinda Glynn for her services to proofread this thesis.

Finally, I dedicate this thesis to my late parents.

Walayat Hussain

List of Publication

Journal publication

1. **Hussain**, **W**., Hussain, F.K., Hussain, O.K., & Chang, E. 2016, 'Provider-Based Optimized Personalized Viable SLA (OPVSLA) Framework to prevent SLA Violation', *The Computer Journal*. pp. 1-24. **(ERA, CORE – A* Journal).**

2. **Hussain**, **W**., Hussain, F.K., Damiani, E., Hussain, O.K. & Chang, E. 2016, 'Formulating and managing viable SLAs in Cloud Computing from a small to medium service provider's viewpoint: a state-of-the-art review', *ACM Computing Survey (CSUR)*. **(ERA, CORE – A* Journal). (Under review)**.

3. **Hussain**, **W**., Hussain, F.K., & Hussain, O.K. 2016, 'Risk Management Architecture to Avoid SLA Violation in Cloud', *Concurrency and Computation*. **(ERA, CORE – A Journal). (Under review).**

4. **Hussain**, **W**., Hussain, F.K., & Hussain, O.K. 2016, 'Comparative Analysis of the Accuracy of Cloud QoS Prediction Approaches', *Journal of Network and Computer Applications.* **(ERA, CORE – A Journal). (Under review).**

Conference publication

5. **Hussain**, **W**., Hussain, F.K., Hussain, O.K. 2014, 'Maintaining Trust in Cloud Computing through SLA Monitoring'. *21st International Conference on Neural Information Processing (ICONIP2014),* Springer, Kuching, Malaysia, pp. 690-697. (**ERA, CORE A-Rank)**.

6. **Hussain**, **W**., Hussain, F.K., Hussain. 2015, 'Profile-Based Viable Service Level Agreement (SLA) Violation Prediction Model in the Cloud'. *10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, IEEE, Krakow, Poland, pp. 268-272. **(Received the Best Paper Award).**

7. **Hussain**, **W**., Hussain, F.K., Hussain. 2015, 'Comparative Analysis of Consumer Profile-based Methods to Predict SLA violation'. *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, Istanbul, Turkey, pp 1-8. (**ERA, CORE A-Rank)**.

8. **Hussain**, **W**., Hussain, F.K., Hussain. 2015, 'Towards Soft Computing Approaches for Formulating Viable Service Level Agreements in Cloud'. *22nd International Conference on Neural Information Processing, (ICONIP 2015)*, Springer, Istanbul, Turkey, pp 639-646. (**ERA, CORE A-Rank)**.

9. **Hussain**, **W**., Hussain, F.K., Hussain. 2016, 'QoS Prediction Methods to Avoid SLA Violation in Post-Interaction Time Phase'. *11th IEEE Conference on Industrial Electronics and Application*, IEEE, Hefei, China. (**ERA A-Rank)**.

10. **Hussain**, **W**., Hussain, F.K., Hussain. 2016, 'Allocating Optimized Resources in the Cloud by a Viable SLA Model'. *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, Vancouver, Canada. (**ERA, CORE A-Rank)**.

11. **Hussain**, **W**., Hussain, F.K., Hussain. 2016, 'SLA Management Framework to Avoid Violation in Cloud'. *23rd International Conference on Neural Information Processing*. IEEE, Koyoto, Japan. (**ERA, CORE A-Rank)**.

12. **Hussain**, **W**., Hussain, F.K., Hussain. 2016, 'Risk Management Framework to Avoid SLA Violation in Cloud'. *11th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, IEEE, Asan, Korea. **(Received the Best Paper Award)**

13. **Hussain**, **W**., Hussain, F.K., Hussain. 2015, 'Transmitting Scalable Video Streaming over Wireless Ad Hoc Networks'. *29th International Conference on Advanced Information Networking and Applications (AINA)*. Gwangiu, South Korea. (**ERA, CORE B-Rank)**.

Table of Content

List of Figures

List of Tables

1 Introduction

1.1 Overview of Cloud Computing

Cloud computing is increasingly recognized and popular among business communities due to its elastic architecture and economical, easily accessible, scalable and flexible nature. Cloud computing provides the architecture through which a consumer can store, retrieve and process data and execute their application anytime and anywhere, regardless of the physical location of the servers (Hussain, Hussain & Hussain 2014). A cloud can be considered as a huge pool of virtualized resources, such as a platform, infrastructure and services that can be easily accessed and used by consumers (Chauhan et al. 2011). Cloud computing not only provides intelligent solutions to enterprise challenges but it also opens a door of opportunities for small businesses and end users. By using cloud computing, small businesses users can save significantly on the upfront cost of pre-planning the required resources, operation and maintenance costs, hardware and software costs, and additional costs for upgrading computing resources. It can also improve IT staff and employee productivity by enabling them to focus on core business activities (Etro 2009). Cloud computing provides consumers with on-demand self-service, whereby they can access virtually unlimited resources to satisfy their business needs (Mell & Grance 2011).

There are a number of definitions of cloud computing. The most widely accepted definition of cloud computing is that of the National Institute of Standards and Technology (Mell & Grance 2009). They define cloud computing as: "*A model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."* Cloud computing refers to online handling, organizing and accessing software applications and hardware application for storage, processing and infrastructure (Computing & Creeger 2009).

1.1.1 Emergence of Cloud Computing

Due to the elastic nature of a cloud and its wide range of services, the demand for and use of cloud computing is increasing every day. It is a rising and popular new technology in parallel computing because consumers can access resources anywhere and at any time irrespective of their location, timing or platform (Muchahari & Sinha 2012; Subashini & Kavitha 2011). While using cloud platforms, businesses and consumers benefit by increasing their capacity in a number of ways. For example, business cases collected by the Open Group (Group 2016) show that the cloud computing "pay-per-use" cost model decreases investment in resource planning and reduces upfront costs. Cloud users also report feeling free from the burden of managing IT resources and from the fear of running out of them. Soliant Consulting (Consulting 2015) published statistics that state that in 2015 about 63% of companies used a private cloud service, 88% of companies used public cloud services and 82% of companies used hybrid cloud services to perform their business activities. These figures are 74% higher than in 2014. In addition to this, it was estimated that by 2016 approximately 36% of all data would be stored in cloud and 70% of medium scale companies would enhance their business by switching to the cloud. In January 2016, a press release from Gartner (Gartner 2016) stated that the expected growth of the public cloud service market in 2016 is 16.5%, with a total amount of US $204 billion. This is an increase from the $175 billion spent in 2015. An amount of $677 billion will be spent on cloud computing between 2013 to 2016, including $310 billion on cloud advertising. In infrastructure as a service (IaaS) alone, the market is expected to grow from 31.9% to 38.4% in 2016 and the global cloud service market is predicted to grow by 13.6% in 2016, reaching $90.3 billion. The hype cycle for cloud computing in 2013, published by Gartner, is presented in Figure 1.1. This hype cycle outlines what type of cloud computing information technology (IT) leaders use for their business and which will become the foundation for the future benefit of cloud computing.

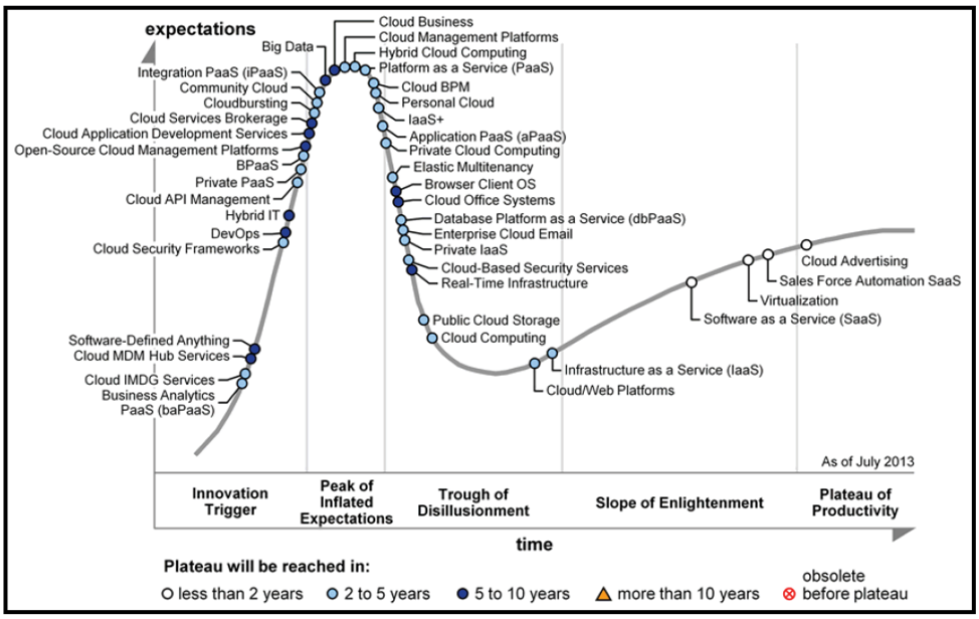


Figure 1.1: Hype cycle of cloud computing by Gartner (Macháček 2014)

1.1.2 Models of Cloud Computing

There are two models of cloud computing: cloud deployment models and cloud service models. These are explained below:

- *Cloud deployment models* describe the purpose and nature of access to the cloud. It presents the type of cloud environment based on location. There are three types of deployment models: public cloud, private cloud and hybrid cloud (Computing & Creeger 2009). These are presented in Figure 1.2.

  – *Public clouds* allow computing resources and services to be accessed over the internet by the general public. Due to its public nature, these clouds are the least secure of the three deployment models.

  – *Private clouds* allow computing resources and services to be accessed over a private network within an organization. These clouds are more secure than public clouds.

– *Hybrid clouds* combine various public clouds with different private clouds to achieve the functionality of both a public and a private cloud.

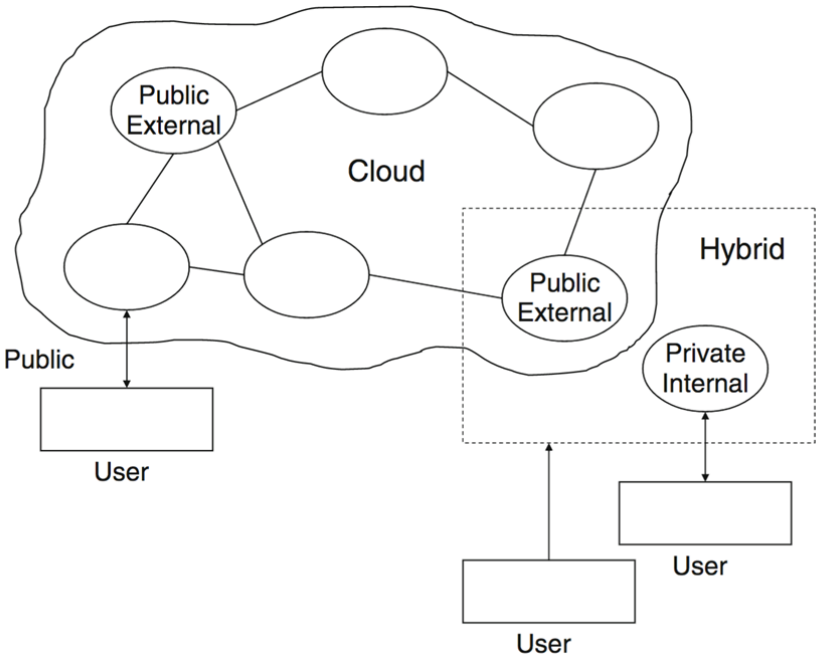


Figure 1.2: Three deployment models of cloud (Qian et al. 2009).

- *Cloud service models* describe the type of resources and services that are provided over the internet. There are three types of cloud service models: infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS). These are presented in Figure 1.3.
  - *Infrastructure as a service (IaaS):* IaaS provides access to hardware and all fundamental computing resources to users. The IaaS providers obtain and pool hardware resources to be provisioned as a service. Users can access these hardware resources through virtualization.
  - *Platform as a service (PaaS):* PaaS provides a platform for users to develop, deploy and execute their applications. It enables all PaaS users to build their software applications without worrying about a platform by sharing hardware and software resources between them.
  - *Software as a service (SaaS*): SaaS allows multiple users to access software applications using multitenant architecture. The SaaS

providers maintain, upgrade and host software applications on behalf of users. Different SaaS consumers share the same version of software and underlying hardware.

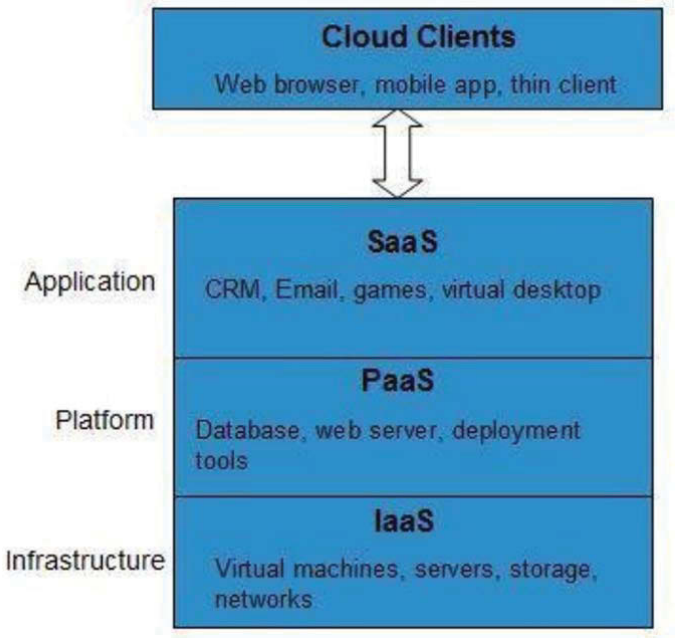


Figure 1.3: Cloud service models (Computing & Creeger 2009).

1.2 Service Level Agreement (SLA)

A service level agreement (SLA) is a contract between a service provider and a user that defines the level of service expected from the service provider. A typical SLA describes the relationship and roles of interacting parties, the agreed standards of service delivery (often called service level objectives or SLOs) and the obligations and penalties of violating parties (Liu, Squillante & Wolf 2001). For example, in their SLAs Amazon Elastic Compute Cloud (EC2) and Amazon Elastic Block Store (EBS) commit to provide an uptime of at least 99.95% for their services in a monthly billing cycle (*Amazon Web Service - Amazon EC2 Service Level Agreement*). EC2's SLA contains information about their service commitment and procedure for compensating the consumer if the commitment is not fulfilled. Microsoft Azure offers different SLAs for its offerings. It commits to a monthly uptime of at least 99.95% and 99.9% for its cloud services and remote applications respectively (*Microsoft Windows Azure - Service Level Agreement*). GoGrid promises an uptime of 100% for its network performance and defines a complete set of commitments, obligations, credit requirements and credit

limitations in their SLAs (*GoGrid – A datapipe Company, Serverpath - SLA*). In case of non-commitment to the formed expectations by the service providers, SLAs describe the applicable penalties.

1.3 Service Level Objective (SLO)

When a provider and a consumer form an SLA, it is comprised of many performance metrics, also called service level objectives (SLOs). An SLO describes the level of expectation and is used to measure the performance of a provider. Each SLO may be composed of one or many low-level metrics (Emeakaroha, Brandic, et al. 2010), as shown in Figure 1.4. The low-level metrics are different resource metrics like downtime and uptime that are combined together to form an SLO such as availability.

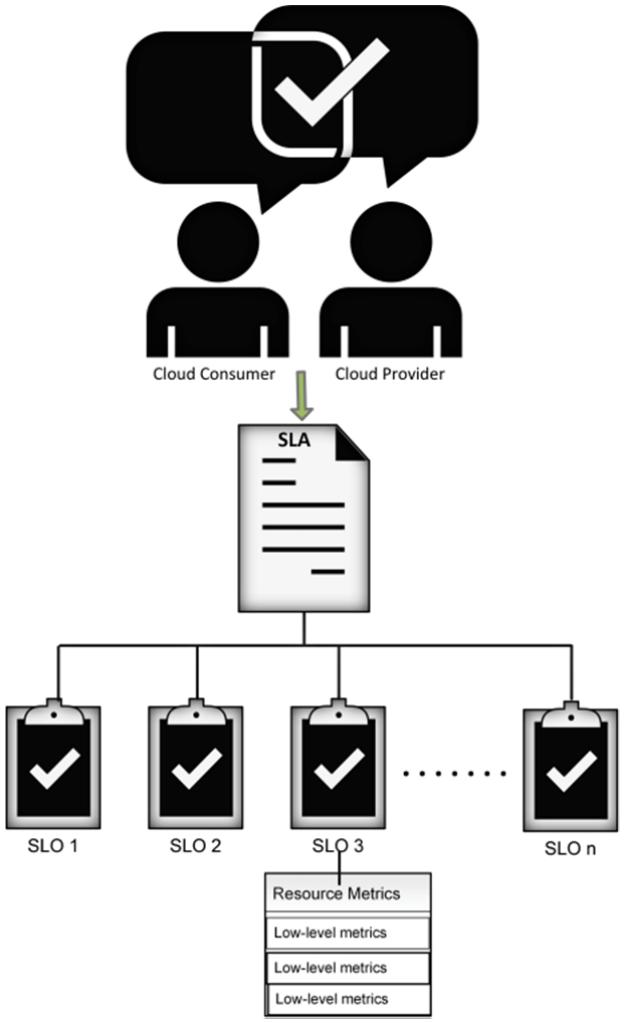


Figure 1.4: Relationship between a SLA, SLO and low-level metrics

For example, if availability is one of the SLOs, then it is composed of the low-level metrics of downtime and uptime in the following way: $1 - downtime/uptime$. For a business outcome to be positive, it is necessary that most if not all of the agreed SLOs be satisfied.

1.4 SLA Violation

An SLA violation is the un-fulfillment of the services contract (Wu & Buyya 2012) that was formed between the service provider and the consumer. The service provider is liable for penalty if the quality, availability and responsibilities are not met which are agreed on an agreement. There are three types of SLA violation (Rana et al. 2008): all or nothing provisioning, in which transactions are successful only when all SLOs are satisfied; partial provisioning, in which transactions are successful when some compulsory SLOs are satisfied; and weighted partial provisioning, in which transactions are successful by delivering those SLOs whose weights are greater than the threshold defined in the SLA.

1.5 Risk Management and SLA Violations

In any interaction the fear of loss or not achieving the desired outcome is known as risk (Hussain et al. 2008). Risk is a key determinant in the service continuation decision. In cloud computing, service providers may deal with a risk of service violation that would lead to violation penalties and a loss of reputation. To avoid violation penalties, a service provider assesses and analyses possible risks earlier, before an interaction, to determine whether it will achieve its desired outcome from the interaction or not. Due to the dynamic nature of cloud computing, managing its resources wisely and dealing with the risk of SLA violation intelligently is a challenging job for a service provider, particularly for small and medium cloud providers because unlike large enterprises they have limited resources. Risk and trust are interrelated, with each being an anterior or subsequent of each other (Mayer, Davis & Schoorman 1995). When there is an interaction between a consumer and a provider, the risk in that transaction is based on the degree of trust (Hussain et al. 2008). The risk action decision depends on the risk attitude of the service provider and the trustworthiness of the consumer (Hussain et al. 2016). The risk attitude of a service provider can be one of three types:

- *Risk averse:* A service provider is reluctant to take a risk
- *Risk neutral:* A service provider has a balanced attitude towards taking a risk.
- *Risk taking:* A service provider is likely to accept a risk.

Depending on the risk attitude of the service provider, the service provider deal with the risk in that way and take all necessary actions beforehand to avoid any violation.

1.6 Challenges of SLA Management from the Provider's Perspective

The increase in the cloud market produces new challenges and hurdles for cloud providers and cloud consumers. One of the main challenges is the SLA obligation, which is obligatory for both of the interacting parties. Due to the elastic features of cloud computing, consumers can form a number of SLAs with different providers depending upon their requirements. Consumers request two types of resources: required resources and marginal resources. Required resources are those resources that are compulsory for consumers to have in order to be able to run their business, whereas marginal resources are extra resources that are kept in reserve and are used by consumers in the case of an increase in business demands. Required resources are reserved and used, whereas marginal resources are reserved, not always used, and paid for only when used. Cloud consumers are disappointed when a cloud-based computation does not scale as they expect (Ardagna et al. 2014); on the other hand, cloud providers can bear significant losses when its marginal resources are reserved by a consumer for the duration of the agreement without being used.

There are two types of cloud provider: large enterprise providers that have virtually unlimited resources and offer a variety of services, and small or medium cloud providers that have limited resources and services. For large service providers that have unlimited resources, accepting consumers' requests for marginal and required resources is not as much of a problem as it is for the small cloud providers that have limited resources. Small service providers must make the best use of their limited resources and allocate them to consumers from whom they can maximize both utilization and revenue. This is not achieved when a consumer requests marginal resources that are not used in full. This leads to three

scenarios. First, the consumer is violating their commitment because of the agreements in an SLA for the reservation and usage of marginal resources; second, it is not allowing other perspective consumers to use those resources from the provider; and third, the provider does not obtain the financial returns from its resources that were kept in reserve and not used. This is similar to the cost of lost sales concept in logistics, where the provider would have earned revenue from its resources if the resources had been utilized wisely. However, a provider is required, according to the SLA, to keep the committed resources in reserve to avoid penalties, even though the consumer has not used them. Therefore, both parties need a mechanism that ensures that they will fulfil their commitment and alerts them to take appropriate action when it detects any violation. There is significant research (Brandic et al. 2010; Cicotti et al. 2011; Emeakaroha, Brandic, et al. 2010; Hussain et al. 2014) from service consumer's perspective on how to ensure the quality of service (QoS) of the SLOs; however, there are still many gaps about the effective formation and management of SLAs from the service provider's perspective.

Cloud SLA management is composed of two phases (Hussain et al. 2012): pre-interaction and post-interaction (as presented in Figure 1.5). Pre-interaction is the time phase from T-1 to T-m, which includes all steps taken prior to finalizing the SLA. The SLA is finalized at time T. Post-interaction is a time phase that starts from T+1 to T+n that include all steps taken when the SLA has been executed, such as service monitoring, violation prediction, risk management and penalty enforcement.

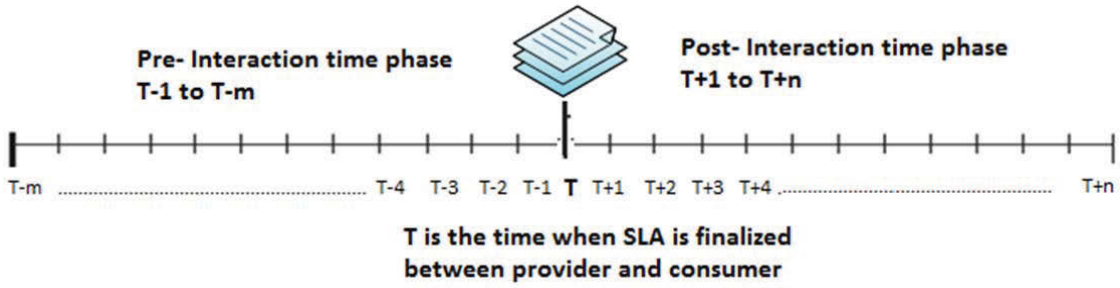


Figure 1.5: Pre-interaction and post-interaction time phases in an SLA

From a provider's perspective, SLA management is required in both time phases for better management of SLA and to avoid SLA violations. The challenges

related to SLA management in the pre-interaction and post-interaction time phases are presented in the following sections.

1.6.1 SLA Formation in the Pre-interaction Time Phase

In a cloud computing environment, consumers request services and resources from a service provider. The consumer puts forward their requirements for marginal and fixed resources, mandatory QoS parameters for different SLOs, the level of service and the type of deliverables from providers. A service provider needs a mechanism that can guide it to make wise decisions on whether to accept or reject consumers' requests. Additionally, once the request has been accepted, then the service provider needs an intelligent solution to help them in offering the amount of resources to make a viable SLA.

While many envision cloud computing as a paradigm of unlimited resources, in actuality in cloud computing the physical resources are shared by different users through virtualization. For small to medium service providers, making an appropriate decision concerning service formation and offering the right amount of resources is very important. The wrong decision could lead them to service violations and violation penalties. For optimal solutions, a provider needs to start the process of SLA monitoring in the pre-interaction phase when the consumer first requests the resources before finalizing and executing the SLA. Currently, there is no tool that a service provider can use to monitor SLA in the pre-interaction time phase by considering the reliability of consumer and the risk attitude of the provider, thus assisting the service provider to make a viable SLA. Existing SLA negotiation models, such as strategic SLA negotiation, automated SLA negotiation, multi-attribute negotiation, Markovian Arrival Process and Sandpiper, assist cloud stockholders to form SLAs in the pre-interaction time phase. However, none of these approaches consider the reliability of cloud users in committing to the SLAs formed during the negotiation phase; hence, a stakeholder's decision can only be based on the ability of the cloud service provider to commit to the requested resources, which does not guarantee the reciprocal adherence of service users to the terms of the SLAs.

1.6.2 QoS Monitoring and Prediction in the Post-interaction Time Phase

Once the SLA has been formed, the consumer's resource usage is monitored in order to better predict future requirements. The provider can either follow the more formal quantitative prediction methods (Waters 2011) or choose a prediction algorithm. This choice plays an important role in managing the provider's risk. Due to the elastic nature of the cloud, an optimal prediction method would help small and medium service providers avoid a shortfall in resources before it leads to a service violation. The previous QoS history plays a key role in predicting future QoS. There are many QoS prediction methods that have different prediction accuracy at different time intervals with different data patterns. For the most accurate predictions, a service provider needs to use an optimal prediction method that can give an optimal result with different data patterns. The service provider assesses the difference between the runtime QoS parameters and the agreed QoS parameters and, if it finds any discrepancies between them, then it takes the appropriate action to manage the risk of SLA violation. There are few published works that discuss QoS prediction using cloud datasets and there is no work that discusses how to manage a risk when a provider finds a variation in QoS parameters that considers the reputation of the consumer and the risk attitude of a provider. These existing approaches were unable to guide a service provider in their decisions about service formation with a consumer and thus service providers form SLAs with consumers who may have flawed intents that will lead to financial loss.

1.6.3 Managing the Risks of an SLA Violation

When a service provider identifies a difference in QoS parameters, then it is very important for it to identify, estimate and mitigate risks to avoid an SLA violation. If a service provider underestimates risk and it occurs, then it would lead to a service violation. Therefore, the service provider needs an optimal method that can accurately assess and estimate the likely risk of SLA violation. In this thesis, a novel framework is proposed to manage the risk of SLA violations by considering three inputs: the risk attitude of a provider, the reliability of a consumer and the predicted trajectory. From these three inputs, an output of risk mitigation action is generated.

1.7 Research Objectives

The previous sections have described the need for a viable SLA management framework that works from a service provider's perspective, assisting the provider in decision-making about consumers' requests, offering resources to consumers, monitoring its runtime behaviour, mitigating the risk of SLA violations, providing an alarm when service violations occur and recommending appropriate action when it detects the risk of an SLA violation. This thesis has five objectives:

- Objective 1: To propose a viable SLA lifecycle that starts the process of SLA monitoring from the pre-interaction time phase.
- Objective 2: To develop an intelligent approach to form a viable SLA in the pre-interaction time phase that determines the probability of a given consumer potentially violating SLA by taking into account their previous reputation of SLA violations.
- Objective 3: To evaluate the QoS history of requesting consumers and develop a methodology for predicting the future QoS based on different patterns of a dataset.
- Objective 4: To develop a framework that manages service providers' risk of possible SLA violations when it finds discrepancies between agreed and predicted QoS parameters.
- Objective 5: To evaluate and validate the proposed approach.

1.8 Research Significance

In the cloud computing environment, it is essential for a cloud provider to manage, monitor and maximize the utilization of cloud resources by ensuring security, efficiency, high availability, trustworthiness and reliability as well as fulfilling all service level objectives (SLOs) as defined in the service level agreement (SLA). For successful business relations, it is vital to predict any service violation before it affects the consumer or the provider. The service provider needs to know the consumer's likely usage of different cloud resources before finalizing an SLA. To the knowledge of the researcher and based on a through literature review, the existing literature does not present a comprehensive SLA management framework from a provider's perspective that is able to form a

personalized viable SLA based on the previous profile of the requesting consumer as well as predict and manage the risks of likely service violations.

The significance of the thesis lies in the main objective of this thesis, which is to form a personalized viable SLA management framework which intelligently forms SLA based on consumers' past history, selects an optimized prediction method for precisely detecting violations and, when it predicts a risk of SLA violation, then managing and mitigating it by using an optimized risk management framework. There are number of different aspects of this thesis that highlight its significance:

- The thesis develops an optimized framework for service providers that assists cloud providers in deciding whether or not to accept a consumer's request.
- The thesis proposes a framework that helps service providers in forming an optimized viable SLA by offering a maximum limit of marginal resources based on the available resources.
- This study presents a methodology of forecasting QoS data based on existing data and selects an optimized QoS method with optimal parameters at different datasets.
- The study helps service providers in the effective management of risk to avoid SLA violations.
- The study helps service providers to mitigate any possible SLA violations before the actual violation occurs and helps them to avoid violation penalties.

1.9 Thesis Structure

This thesis is comprised of nine chapters. This chapter defines cloud computing, SLA, SLO, SLA violations and risk management before discussing the challenges of SLA management from the provider's viewpoint and outlining the research objectives. The significance of the study is also briefly defined. The rest of the thesis is organized as follows.

Chapter 2 contains a thorough study of the related literature that discusses different aspects of SLA management, particularly SLA monitoring in cloud

computing. The literature is then categorized into four categories depending on their features and working attributes. The features and shortcomings of each method are identified and mentioned in each section and the chapter concludes with an intensive comparison of all of the related approaches.

Based on a thorough survey of related literature in Chapter 2, in Chapter 3 the shortcomings and the gaps are identified. The concept and terminologies that are used in this thesis are also described. The chapter also discusses the background of the research problem and outlines the associated research questions. At the end of the chapter the research approach that is used in this thesis to solve the problem is described.

Chapter 4 describes the provider-side viable SLA management framework. The chapter outlines the steps for a viable SLA management framework and provides an overview of the proposed solution. The chapter contains a brief description about all of the modules of the pre-interaction and post-interaction time phases that combine to form the framework.

Chapter 5 describes the components of the pre-interaction time phase: the identity manager module and the optimized viable SLA module. The chapter shows how by using the framework a provider is able to make a decision about consumers' requests and is then able to form a viable SLA. The chapter discusses fuzzy rules and the fuzzy inference system (FIS) that are used to form a framework and the study validates the approach by using existing datasets.

Chapter 6 describes the time series QoS forecasting in the post-interaction time phase that helps the service provider to select an optimal prediction algorithm. The neural network prediction method, stochastic prediction method and time series prediction methods are used and their prediction accuracy is analyzed at different time intervals with various patterns.

Chapter 7 describes the risk management module in the post-interaction time-phase. The chapter discusses managing the risks of likely SLA violations and recommends an appropriate action to mitigate them. The chapter also examines risk management decisions using an FIS that takes three inputs and then generates an output.

Chapter 8 describes the software implementation of the proposed framework. Chapter 9 summarizes the thesis and highlights future research directions.

1.10 Conclusion

Cloud computing is an emerging technology that offers diverse services to its consumers. Typical service paradigms include infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS). Due to the wide range of services available, the popularity and demand of the cloud is increasing day by day; however, the resulting increase in consumer population creates many challenges for providers. Among them is the formation of a viable SLA that offers resources wisely, assures the best prediction algorithm for managing risk and advises appropriate actions to prevent SLA violations. This thesis develops a personalized viable SLA management framework that can assist and guide service providers to form a viable SLA from the stage of service formation to the stage of service violation mitigation, and help providers avoid SLA violations and violation penalties.

2 Literature Review

2.1 Introduction

In the previous chapter the basic concepts of cloud computing, service level agreements, service level objectives and important research areas related to this research were highlighted. The purpose of this chapter is to present existing methods related to the topic and to provide a background on the SLA management in cloud computing. SLA management comprises of different activities to avoid SLA violation, which include SLA formation, SLA monitoring, SLA violation prediction and SLA-based decision-making. Various approaches in the literature focus on each of these activities in the SLA management process.

SLA management spans two different time phases: the pre-interaction start time phase and post-interaction start time phase (as described in Chapter 1). In the pre-interaction phase, the service provider and service consumer negotiate on the specifics of the SLOs which form the SLA and in the post-interaction phase the service provider (if managing the SLAs in a proactive manner) monitors the SLOs against the formed levels on a real-time basis. Most existing approaches for SLA management (Emeakaroha, Brandic, et al. 2010; Jaramillo, Ardagna & Anisetti 2015; Katsaros et al. 2012; Rana et al. 2008) start the process when both parties have formed their agreement; however, for effective SLA management it is vital for service providers in the pre-interaction phase to carefully negotiate the SLOs in order to maximize the chances of commitment, reduce the chances of SLA violation and gain maximum financial returns (Feng & Buyya 2016; Messina et al. 2016).

In this chapter, the related literature is divided into different sections. Depending on the working attributes of the related existing approaches for SLA management, they are classified into three classes: self-manageable cases based on the reasoning approach, the trust model approach, and the proactive monitoring approach. In the next section, the two SLA management time phases – pre-interaction and post-interaction – are described. The chapter is organized as follows. Section 2.1 introduces the chapter. Section 2.2 outlines the existing cloud SLA management lifecycles. Section 2.3 describes the proposed classification of

SLA monitoring approaches. Section 2.4 reviews the related literature on self-manageable case-based reasoning approaches. Section 2.5 examines the related literature on trust-based model approaches and Section 2.6 discusses the proactive SLA monitoring approaches. Section 2.7 discuss risk management approaches in cloud SLA management. Section 2.8 presents a critical evaluation of the existing literature, which is followed by the chapter conclusion (Section 2.9). The existing literature lacks a thorough review of the existing work on SLA management along different dimensions. This chapter addresses these gaps by presenting a thorough review of the existing work on provider-side cloud SLA management.

2.2 Cloud SLA Management Lifecycle

The cloud SLA management lifecycle is an essential element of the provisioning of cloud services. There is a strong association between SLA and cloud service lifecycle, i.e. the attainment, process and termination of services. The SLA management lifecycle is defined as the process or processes that allows an organization to manage, synchronize, control, provision and support single or multiple SLOs during their SLA (*The International Foundation for Information Technology*).

A basic SLA lifecycle, as described by authors (Ron & Aliko 2001), is comprised of three phases: finding the best-matched service provider, accessing the services and terminating the SLA when delivery is complete. A consumer finds a suitable service provider and enters into a standard, non-negotiable SLA, which terminates at the end of the agreement. A more thorough SLA lifecycle is outlined by authors (Wustenhoff & BluePrints 2002), who describe a SLA lifecycle that is comprised of three phases: creation, operation and removal. In the creation phase, consumers first search for a suitable service provider that offers all of the services they require. Selecting a service provider is a critical for service consumers, because there are many service providers and it is important for the consumer to adapt an intelligent method for appropriate service selection (Sun et al. 2014). In the service creation phase, both parties define an SLA, which contains service definitions, service objectives, SLA parameters and violation penalties. Once the SLA is agreed upon, the operation phase begins where real-time performance is monitored against agreed benchmarks. In the removal phase, the SLA is terminated either on completion of the service or in the case of violation; in the

latter case, penalties are enforced. The authors further divided the above-mentioned three phases of SLA into six steps: finding a suitable service provider; describing the SLA; formalizing the agreement; monitoring the SLA; terminating the SLA; and applying penalties to the violating party. Authors (Wu & Buyya 2012) propose a SLA lifecycle that comprises six steps: choosing a suitable service provider; reaching mutual agreement; formulating an SLA; monitoring the runtime of the SLA; determining the success or failure state and terminating the SLA; and enforcing penalties for any violating party. Authors (Rehman, Hussain & Hussain 2015) propose a user-side SLA management framework that collects QoS information, predicts likely violations and generates recommendations. In all of these approaches, an SLA is formed first before its SLOs are monitored for the possibility of a violation. An automated service level agreement management system was proposed by authors (Lu et al. 2015), in which the lifecycle of an SLA is automatically managed by agents. To achieve a fault-tolerant system, they developed a hierarchy of agents and established relationships between them. They propose an SLA lifecycle comprising of three phases: creation, operation and removal. These three phases were then further decomposed into six steps: three in the creation phase, one in the operation phase and two in the removal phase. The authors define an SLA as the process during which the consumers negotiate with service providers for a required service, then both parties consent and establish their agreement. During SLA monitoring, the service provider monitors the behaviour of the consumer and detects any violations.

SLA monitoring is an essential element of the SLA management lifecycle. The next section describes the classification of SLA monitoring approaches, identifying the features and shortcomings for each approach.

2.3 Classification of SLA Monitoring Approaches

SLA management involves many activities, of which monitoring is an essential element as it is a prerequisite for contract governance. Monitoring the difference between the agreed SLOs and the value delivered during runtime performance will lead to the detection of possible service violations. Exiting literature presents various approaches for detecting possible SLA violations (Emeakaroha, Calheiros, et al. 2010; Emeakaroha, Netto, et al. 2012; Sakr & Liu 2012; Wu, Garg & Buyya 2011). However, the objects of analyses in the existing research vary from each

other, thereby varying the classifications of their SLA management analysis. For example, some approaches focus either on the consumer (Hussain et al. 2014; Sakr & Liu 2012) or the provider (Emeakaroha, Brandic, et al. 2010; Emeakaroha, Calheiros, et al. 2010) for detecting possible SLA violations. Others, such as authors (Antonescu, Robinson & Braun 2013; Liu, Xu & Miao 2011; Yun 2013) consider the problem of SLA management in terms of an optimization problem to increase the consumer's satisfaction by ensuring the provisioning of the promised QoS and to increase the revenue of a provider. Authors (Sahal, Khafagy & Omara 2016) classify SLA violation management into two classes: SLA management for cloud and SLA management for cloud-hosted big data analytic applications. They mainly focused on monitoring for the single layer while optimizing services by considering QoS parameters.

However, the focus for SLA management in this research is interested in the perspective of forming viable SLAs first and then later managing them, the existing approaches are classified into the following three classes: *the self-manageable case-based reasoning (CBR) approach, the trust model-based approach* and *the proactive SLA management approach*, as shown in Figure 2.1. These classifications are made based on the functionality, working attributes and the methodology employed to manage the SLAs for whose assurance.

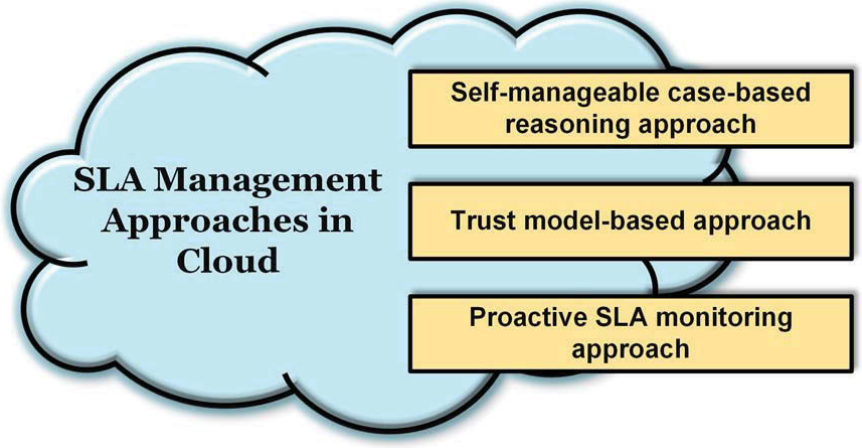


Figure 2.1: Classification of SLA management approaches in cloud computing

The three classes can be further described as follows.

1. *Self-manageable case-based reasoning (CBR) approaches*

CBR approaches refers to those techniques that use self-manageable case-based reasoning to manage SLAs when a variation between the agreed and runtime behavior is detected (Al Falasi, Serhani & Dssouli 2013; Brandic et al. 2010; Emeakaroha, Brandic, et al. 2010; Emeakaroha, Ferreto, et al. 2012; Emeakaroha, Netto, et al. 2012; Haq, Brandic & Schikuta 2010; Katsaros et al. 2012; Mosallanejad & Atan 2013). These approaches use previous knowledge to find solutions for managing SLA violations in the future. Some of these approaches use a hierarchical self-healing approach (Emeakaroha, Ferreto, et al. 2012) that detects any violation and then manages the SLA in a hierarchical way. The hierarchical system tries to prevent potential violations by reacting autonomously before notifying the end user. Approaches which come under this class are discussed in Section 2.4.

2. *Trust model-based approaches*

   This class incorporates those techniques that use trust or reputation for managing the SLAs. Reputation or trustworthiness is a key element in SLA management as it assists in the selection of a reliable service provider (Badidi 2013; Hammadi & Hussain 2012). Existing literature proposes techniques for a consumer to score the reliability of a provider (Alhamad, Dillon & Chang 2010; Badidi 2013; Chandrasekar et al. 2012; Fan & Perros 2013) using approaches like the IP-based method (Wang et al. 2011), the adaptive credibility model (Noor & Sheng 2011) and the trust management model (Fan & Perros 2013). Approaches which come under this class are discussed in Section 2.5.

3. *Proactive SLA management approaches*

   Proactive SLA management approaches for SLA management refer to those techniques in which the likelihood of SLA violation is predicted before the actual violation and the service provider is alerted to take all necessary actions to avoid the violation from occurring. In this category, authors use various SLA monitoring approaches (Ciciani et al. 2012; Egea et al. 2015; Hussain et al. 2014; Pacheco-Sanchez et al. 2011; Romano et al. 2011; Wood et al. 2009) to predict likely violations of SLOs and to issue an early warning to cloud providers for early remedial action. Approaches which come under this class are discussed in Section 2.6.

Using these classes, a comprehensive survey of the approaches for SLA management is presented from the viewpoint of making viable SLAs. The theoretical review method is used to analyse and present existing theories, examine the relationship between them and identify the gaps with reference to the novel framework developed in this research. Different scientific databases are used for this study such as Compendex, IEEE Xplore, Association for Computing Machinery Digital Library, CiteSeerX, Google Scholar and other related databases. This research discusses SLA management from different stakeholders' points of view and presents related concepts in an articulated manner that helps to identify the individuality of different approaches along with their features and shortcomings.

2.4 Self-Manageable Case-Based Reasoning Approach

The case-based reasoning (CBR) approach is a problem-solving method in which new problems are solved based on the solution of previous similar problems (Aamodt & Plaza 1994). This method has been widely used for decision-making in a variety of complex, dynamically changing complex and unstructured problems (Chua, Li & Chan 2001). However, there are certain drawbacks of the CBR approach, such as adaption, processing time and storage. The CBR approach gives reasonable solutions to manage SLA; however, it does not give an optimum solution (Cheetham, Varma & Goebel 2001). For SLA management, authors (Brandic et al. 2010; Emeakaroha, Brandic, et al. 2010; Haq et al. 2009) have applied the CBR approach to identify the likelihood of SLA violations occurring and use previous solutions for early remedial actions. Many techniques in the literature utilize such an approach to quantify the degree of fulfillment of SLOs for SLA management. Some of these are discussed in the following subsections and a summary is presented in Table 2.1.

2.4.1 Low-Level Metrics to High-Level SLAs (LoM2HiS)

Mapping low-level hardware resource metrics to high-level SLA parameters (LoM2HiS) is proposed in (Emeakaroha, Brandic, et al. 2010). Mapping is either simple or complex, based on a pre-defined rule stored in a mapped metrics repository. In simple one-to-one mapping, low-level resource metrics are mapped directly to the SLO's fulfillment without any further processing. In complex mapping, predefined rules are used to map resource metrics to SLOs. Such rules

define thresholds for runtime SLA management and determine SLA violations. A runtime monitor accesses the repository and uses mapped metrics values to check the service status. The values are compared against the corresponding thresholds and if a violation is detected, it alerts the enactor component to take preventive action. With this approach, although the system is capable of detecting SLA violations, there is no mechanism that shows how to rectify an error when a violation occurs. The authors define very few rules for converting low-level metrics to SLA parameters and, in the case of a violation, it is a challenge for the system to find which low-level metrics need to be checked in order to address the violation.

2.4.2 Detecting SLA Violation Infrastructure (DeSVi)

An automatic SLA violation detection infrastructure called Detecting SLA Violation infrastructure (DeSVi) is proposed in (Emeakaroha, Netto, et al. 2012), DeSVi manage and predicts SLA violations using resource management. The architecture is made up of three components: an automatic virtual machine deployer that is responsible for arranging all the required resources for a requested service and organizing its deployment on a virtual machine; an application deployer that is responsible for executing the requested resource; and a LoM2HiS framework to plot hardware-level metrics against SLA parameters. LoM2HiS performs SLA monitoring, which comprises the following modules: a runtime monitor, services, an agreed SLA repository, mapped metrics, a host monitor and infrastructure resources. The runtime monitoring module communicates with the consumer and service provider. Monitoring starts when both parties agree on the SLA and the service provider establishes a mapping rule for LoM2HiS. Consumers request services from the runtime monitoring module, which loads the corresponding SLAs from the SLA repository module. A monitoring agent is used to collect observables, compute the resource metrics and send them to the runtime monitoring module, which maps the low-level metrics and stores the results of the mapping in a repository module. The runtime monitoring module uses these mapped values to monitor the service status and to plot the degree of fulfillment of the SLOs. In the case of an SLA violation, the runtime monitoring module notifies a knowledge component for an early remedy. Although LoM2HiS helps to detect possible service violations, the system is unable to give a recommendation for its correction.

2.4.3 Layered Approach for the Prevention of SLA Violations in Self-Manageable Cloud Infrastructures

A hierarchical layered approach (LAYSI) to SLA management is proposed in (Brandic et al. 2010). The authors propose a bottom-up approach for propagation of violation. There are two main components that are responsible for SLA violation propagation: the knowledge base and the SLA manager. The knowledge base compares the violation threshold generated based on a utility function with the current system status and triggers a reactive action when it detects a violation threat. The reactive action is based on case-based reasoning and tries to solve the problem from past experiences. The system is multi-layered, so that when a particular layer is unable to suggest a reaction then the problem needs to be propagated to the upper layers. The SLA manager is responsible for propagating SLA violation threats to the upper layers. It receives violation notifications from the lower layer and accesses the current layer's knowledge base for reactive action. If no action is found, it notifies the upper layer. Sometimes the violation propagation continues until the top layer, which informs the user by triggering the need for renegotiation or ending the service. The layered architecture assists in the better correction of errors; however, further description is needed that tells how each layer rectifies an error once a violation occurs.

2.4.4 Holistic SLA Validation

An holistic SLA validation framework is proposed in (Haq, Brandic & Schikuta 2010). The framework combines three SLA management techniques: mapping low level resource metrics to high level SLOs (LoM2HiS) model (Emeakaroha, Brandic, et al. 2010), a hierarchical layer model (LAYSI) (Brandic et al. 2010) and the rule-based SLA aggregation and validation (Haq et al. 2009). Once both parties have agreed on an SLA, the framework combines LoM2HiS with all of the services that keep track of SLA violation threats. In the case of an SLA violation, it follows the LAYSI model by trying to fix the problem at the current layer or propagate it to upper layers for its corrective measures. When the framework detects any violation, it finds out the reason and imposes a penalty on the service provider. The method imposes penalties on violating parties; however, it is not described how the problem is rectified once it occurs.

2.4.5 Cloud Application SLA Violation Detection (CASViD)

Cloud Application SLA Violation Detection architecture (CASViD) (Emeakaroha, Ferreto, et al. 2012) manages SLAs at the application level. It comprises the provision of services, setting up services, monitoring services and detecting SLA violations. To detect violations, CASViD finds an effective measurement interval to identify the resource consumption of each application. Effective measurement is conducted by sampling time intervals and checking the applications at each interval. If the utility of the current time interval is greater than the previous interval, the current interval is set as an effective measurement interval. The process continues until the end of the interval. The monitoring agent in each node monitors the application and sends information to the SLA management module. The SLA management module accesses the database and retrieves the SLA with its violation threshold. The module then compares the current SLA with the predefined threat threshold to analyze future SLA violation threats. The threat threshold is defined manually by the provider and indicates future SLA violations, and the system reacts proactively to avoid violations. Although having a measurement interval helps for better management of an SLA, the system lacks a reaction based on previous records.

2.4.6 SLA Management Using the Sky Framework

Falasi et al. (Al Falasi, Serhani & Dssouli 2013) propose an architecture capable of managing the multilevel maintenance and monitoring of SLAs in a federated cloud environment. Their proposed architecture, based on the Sky framework, consists of a sky broker, the socialization module and the federation module that adaptively implements SLAs to manage changes in the federated cloud environment. A performance evaluation report for each dependent SLA in a federated cloud is used to ensure that the primary SLA is preserved and all relevant parties gets updated when changes occur. The authors describe an SLA life cycle which lacks the stages of pre-SLA negotiation, monitoring steps and renegotiation after an SLA violation occurs. Moreover, the system does not alarm a service provider in the case of a possible violation so it can arrange for necessary actions for early remedy.`

2.4.7 Hierarchical Self-healing SLA (HS-SLA)

Another hierarchical self-healing SLA management framework is proposed in (Mosallanejad & Atan 2013). Each SLA is connected to the related layer of the cloud. The service provider in each layer of the cloud has one or more copy of the SLA. Each upper layer is dependent on a lower layer. Two QoS parameters, response time and throughput, are considered to measure the efficiency of their system. The SLA is monitored by the monitoring function available in each SLA. When it detects a possible violation, the system tries to resolve the issue by switching to other resources in that layer but, if the layer is unable to resolve the problem, it informs the other SLAs in the upper layer. The system tries to prevent violations before they affect the end users. However, the approach is reactive in SLA management. Furthermore, due to vendor lock-in and a lack of standards in real-life scenarios, most cloud service providers do not allow the application to migrate from one service provider to another in the case of service violations.

2.4.8 Fault-tolerant Actor System SLA Management Framework

(Lu et al. 2015) propose an actor system to manage the SLA life cycle. The agent creates a parent and child relationship that helps to propagate an error message upward for its resolution. The proposed actor system is responsible to manage the complete SLA process. In the case of a violation, a concerned actor tries to rectify the error if it is unable to solve it then inform its immediate supervisor actor in the hierarchy. However, it is not mentioned how SLA management would be controlled if the single actor system that is responsible for the whole processes crashes. Moreover, there is no violation prediction mechanism and, in the case of a service violation, the concerned layer propagates a violation report to upper layers for remedial action.

2.4.9 Mulit-layer Monitoring

A self-adaptive SLA management mechanism is proposed in (Katsaros et al. 2012) that monitors SLAs on the basis of monitoring time intervals and parameters. The proposed mechanism manages both application and infrastructure layers and triggers on-the-fly reconfiguration. The management mechanism comprises six components that are arranged according to the three layers of a typical cloud stack. The PaaS layer includes a monitoring framework service (MFS) and the

monitoring central index (MCI). The MFS is responsible for monitoring applications and performs corrective actions in the case of violation. The MCI is a repository storing parameter values. The infrastructure monitoring service (IMS) resides in the IaaS layer and is responsible for collecting such values and sending them to the MCI. The SaaS layer comprises of three components: the monitoring service instance (MSI), the monitoring index service (MIS) and the data collector (DC) that completes them with additional data like name, value and unit of measure and publishes them in the local repository. Self-adaption allows both the data collector and the infrastructure monitoring service to readjust resources or monitor time intervals. The IaaS monitoring layer is based on the low-level information and related metrics; however, the authors did not describe those metrics. The self-adaption depends on the custom policies with the help of experts but these policy functionalities are integrated with the monitoring module which does not provide flexibility or a user-friendly policy enforcement mechanism. The approach does not provide scalability performance.

Table 2.1 presents a summary of the approaches that come under this class, the description of each approach, their features and their limitations in forming a viable SLA.

Table 2.1: Self-managed case-based reasoning approaches

SLA management approach	Description of the approach	Features of the approach	Issues/ limitations of the approach
LoM2HiS framework (Emeakaroha, Brandic, et al. 2010)	Converting low-level metrics to high-level SLA parameters and compare	Capable of detecting future SLA violations by comparing SLA objectives with the threat threshold values. When a system detects SLA violation it inform enactor components to take early remedial action An automatic SLA management and enforcement.	When a violation occurs, system does not describe error correction method. Only two rules are discussed for converting resource metrics to SLA parameters. In case of service violation, it is very difficult to state that which low-level metrics need to be checked to address the violation. The criteria for a threat threshold are not defined and system is unable to prevent violation once it occurs.
DesVi framework (Emeakaroha, Netto, et al. 2012)	Resource management by LoM2HiS, which is able to monitor and detect SLA	Flexible and reliable management of SLA. Early detection of SLA violation using a threat threshold value.	It is only capable of managing a single cloud data center. There is a lack of reactive action based on the best measurement interval. limitations of LoM2HiS discussed

SLA management approach	Description of the approach	Features of the approach	Issues/ limitations of the approach
	violation. Effective SLA management interval.	Reactive action by the case-based reasoning approach.	above. No mechanism is described to select optimal measurement intervals. System is unable to prevent violation once it occurs.
LAYSI Approach (Brandic et al. 2010)	SLA management by LoM2HiS. Bottom-up approach for the propagation of an SLA violation threat.	Proactive alarming. Violation threat of SLA. Self-manageable. Propagation of violation threat to layer of concern.	Does not describe how a system rectifies an error once a violation occurs. It uses CBR approach that has its own limitations. Lacks description, on which basis the threat threshold is defined for violation detection. limitations of LoM2HiS discussed above. The approach did not describe how a system reacts when actual violation occurs.
Holistic SLA validation Approach (Haq, Brandic & Schikuta 2010)	Combination of LoM2HiS, LAYSI and SLA aggregation and validation framework.	Features of LoM2HiS, LAYSI. Consistency check for SLA Penalty enforcement on violating party.	Limitations of LoM2HiS and LAYSI approaches discussed above. Layered bottom up approach for violation propagation, but no mechanism on how each layer manage it. limitations of CBR approach. Study focuses only on basic design and lacks details on a real implementation and system is unable to manage violation once it occurs.
CASViD (Emeakaroha, Ferreto, et al. 2012)	Detecting SLA violation based on threat threshold. It manage and monitors the SLA at the application layer.	Monitors and detects SLA violation at the application layer. Determines an effective measurement interval.	Manually defined threat threshold by a provider. There is a lack of reactive action based on previous knowledge. SLA management focuses on post-interaction phase only.
Sky framework model (Al Falasi, Serhani & Dssouli 2013)	Handles multilevel SLA management in a federated cloud environment.	Management of a multilevel SLA. Dynamic SLA validation and deployment.	SLA life cycle lacking the pre-SLA management. No procedure defined for SLA management once the violation occurs. Lacks negotiation and renegotiation after SLA violation.

SLA management approach	Description of the approach	Features of the approach	Issues/ limitations of the approach
HS-SLA (Mosallanejad & Atan 2013)	It manages SLA by monitoring violations and in a hierarchical way and tries to prevent violation by migration to another provider or propagating the violation to an upper layer.	Self-healing of SLA violation, which includes SLA monitoring, SLA violation detection and all necessary actions to rectify a violation. Follow a layered hierarchical pattern for violation propagation and prevention.	The system reacts once the violation occurs, there is no mechanism to predict violation before actual violation. Migrating to other service providers, itself have many issues, which is not wise suggestion for a consumer. It did not describe what necessary actions to be taken when a violation occurs.
Fault-tolerant actor system (Lu et al. 2015)	An actor system automatically manage complete SLA life cycle that follow a hierarchical structure for fault-tolerant effective and efficient SLA management.	Actor system accomplishes better SLA management. It follows a layered hierarchical structure for fault-tolerant SLA management structure and in case of service violation, the violation propagated to upper layer for its possible solution.	No prediction of SLA violation, the remedial action performed when there is actual violation occurs. Complete SLA system depends on single actor system; however, there is no explanation on how the SLA management works if there is any problem occurs with actor system. Authors did not describe what are the necessary action that the system takes for avoiding SLA violations.
Multilayer monitoring (Katsaros et al. 2012)	It aggregates the QoS from the application and infrastructure layer in the platform layer.	Measure QoS parameters at infrastructure and application levels. It self-configures the monitoring time interval and monitoring parameter. Runtime adaptability of resource provisioning, estimation and decision. SLA management is done at each layer	No information about low-level metrics. Policy functionalities are integrated within the monitoring module and lacking convenient policy imposition system. Need to enhance scalability performance of the system using load balancing method.

2.5 Trust Model-based Approach for SLA Management

Due to the large number of service providers and consumers in cloud computing, it is a great challenge to establish and maintain trust relationships. Trust in the provider, trust representation, and the criteria for trust calculations are only some

of the issues that concern consumers. Trust has a life cycle that includes establishment, maintenance and termination. Fachrunnisa and Hussain (Fachrunnisa & Hussain 2013) propose a proactive performance management mechanism for trust maintenance by introducing third party agents and trust-level metric recalibrations. The third party agent is responsible for SLA administration and performance management and compares the actual behavior with the agreed behaviour defined in the SLA. Both parties recalibrate their trust to calculate the final trust level. Trust can be calculated either in monetary form or in reputation form. Reputation is based on the reliability of a service provider. The consumers score providers on each successful or unsuccessful transaction. If the consumers give good feedback on every successful transaction, this will result in a high reputation value for the provider and conversely for unsuccessful ones, poor feedback will result in a low reputation value. This scoring method allows false, biased and unreliable feedback to skew the results, which can change the reputation of the provider (Fan & Perros 2013). Wang et al. (Wang et al. 2011) propose that the transaction validity can be verified by analyzing the IP address of consumers. They suggest an iterance and IP monitoring mechanism that collects and records the IP address of the service provider and consumer and makes its analysis based on the IP region, the IP record and the transaction validity. The system cancels multiple feedbacks from the same IP region. In this way a system differentiates between the biased and true consumer's feedback. Although the approach helps in differentiating consumers' feedback, the study did not describe how a system would function if there were multiple consumers who give genuine feedback from the same region or the same organization.(Quillinan et al. 2010) propose that trust between a provider and a consumer can be maintained by managing a trust from the consumer's and the provider's sides. A trusted third party monitors communication between consumers and providers; however, it cannot determine the internal state of either the consumer or the provider. The provider-side trust model has access to the internal state of the provider and can take measures to avoid any violation. Many techniques in the literature utilize trust as an approach for SLA management. Some of them are described in the following subsections. A summary is presented in Table 2.2.

2.5.1 Adaptive Credibility Model

(Noor & Sheng 2011) propose an adaptive credibility model by offering trust as a service that differentiates between consumers' credible and biased feedback by using consumers' capability and majority consensus. It considers unanimous feedback and measures whether a specific score is close to the majority of the feedback. The proposed framework has two components – *a credible module* and a *distributed trust feedback assessment and storage module*. The first module is responsible for distinguishing between true and biased feedback by considering the majority consensus feedback and the second module stores the feedback assessment in a distributive way. The proposed framework is comprised of three layers that use service oriented architecture to offer trust as a service. The authors tried to address the issue of differentiating between true and biased feedback; however, the model is only feasible for a service provider who has provided services for a long time and has enough existing feedback to satisfy the majority consensus. The authors also did not describe how much feedback is considered as a majority consensus, if there is a certain group of consumers who particularly aim to give false feedback and target to any particular service provider, then how the system manages the situation. The proposed model is suitable for consumers who have a previous history of service usage and feedback. However, for a consumer who is new or has just started using services, the feedback has a lower impact even if it is true feedback. There is no mechanism that bootstraps for new consumers.

2.5.2 Reliability-based Trust Management Model

(Fan & Perros 2013) propose a trust management model by filtering feedback based on two factors, familiarity and consistency. The factors are calculated from the trust feedback value of consumer and the duration of services that the consumer used. The two factors are multiplied with each other to calculate the trusted feedback of consumers. The proposed trust management system is divided into two sections: a provider and a consumer section. The provider section deals with connecting the consumer with a provider domain and, in the consumer section, the system collects cloud service information. The framework allows the consumer and the provider to establish a trusted relationship for service selection and classification. The authors proposed a trust value range from 1 to 5. Only

consumers whose reliability factor exceeds predefined thresholds were able to assess providers. The familiarity of the consumer depends on their service usage history in a certain period. A consumer who has used services a long time ago will not have a reliable assessment. The authors did not describe the timeframe for deciding the early past and distant past, and no bootstrapping mechanism is defined for consumers who have just subscribed for services or have no previous record. There is no mechanism defined for threshold formation and comparing consumers against that. The authors did not justify the selection of two parameters for decision-making and there are many other parameters which, if considered, can significantly improve results.

2.5.3 Trust Mining Model

(Marudhadevi, Dhatchayani & Sriram 2014) propose a trust-mining model that calculates the degree of trust based on the subjective and objective rating of consumers. The model calculates a trust value based on certain attributes like the number of successful and unsuccessful responses, the average response time and the number of complaints from consumers. The system considers the feedback of consumers about each services. The model helps service consumers to select trustworthy cloud services and acts as a decision-making system that helps consumers to decide whether to continue with the same provider or to switch to another provider. The model also helps the service provider to monitor the services that can help to sustain a trusted association with a consumer. Rough set and Bayesian inference are used to generate the prediction results. The proposed approach calculates trust at two levels. In the first level, the system uses existing data about a provider and calculates a trust value of it and once a transaction is completed then the consumer gives feedback based on which the second level of trust is calculated using the Bayesian inference theorem. Depending on the trust value at level 1 and level 2, a consumer decides whether to continue the service or not. The approach is reactive and trust is calculated once the provider violates its commitment; moreover, the authors did not describe how to alarm or invoke the service provider in the case of degradation to rectify the errors. Additionally, no procedure is defined for SLA violations and penalty enforcement.

2.5.4 Dynamic Trust Calculation Method using Markov Chains

(Chandrasekar et al. 2012) present an effective SLA management and QoS monitoring technique to monitor trust in a provider. For the provider to be trusted it should have a previous service profile for the services that it offers and QoS is the extracted from previous SLAs. The authors propose a dynamic trust calculation method based on Markov chains. They assigned different weights to different parameters of QoS based on their importance and calculated their cost by multiplying assigned weights to the difference of the actual and expected values. Cost was calculated regularly and, based on this cost, the Markov chain can find itself in three states: a steady state, an unsteady state or a failure state. The trust value is computed at regular intervals. When a provider reaches maximum trust, any extra trust is banked as a surplus to be used when there is a failure, before affecting the maximum trust value. The authors propose a 50% trust value for untrusted providers; however, the authors were unable to describe how a system handles instances where a provider with a poor trusted value starts a business with new details. Moreover, the authors focus only on the bandwidth required for transmitting QoS by analyzing through the Markov chain model without comparing other different models.

2.5.5 SLA-based Trust Model

(Alhamad, Dillon & Chang 2010) propose an SLA-based trust model to assist the cloud consumer to select the most suitable cloud provider based on its trustworthiness value. Their proposed model comprises of different components such as SLA agent, a cloud consumer module, a cloud service directory and a cloud provider module. Each of these components performs different functions that combine to form the trust model. The authors propose a common directory where all cloud providers register themselves that can help consumers to find a suitable provider. The authors defined a set of SLA metrics for each layer of the cloud and used them with the trust value to calculate the suitability of the provider; however, there is no method described for choosing such parameters. There are a number of other different parameters which, if considered, can better help a consumer to choose an appropriate service provider. The study lacks a description of the process of SLA agreement and the process of negotiation, which is very important for a cloud consumer. Moreover, the paper proposed a concept yet did

not describe the criteria, methodology or the evaluation and implementation for calculating a trust value.

2.5.6 Cloud Service Registry and Discovery Model

Trust, privacy and security are some of the factors that hinder adaptation in cloud computing. (Muchahari & Sinha 2012) propose a trust management architecture called the cloud service registry and discovery (CSRD) model that acts as a monitoring agent between the consumer and provider. The framework comprises of three modules: a registry module, a trust-calculating module and a dynamic trust-monitoring module. The registry module registers service providers and service consumers and lists them based on their trust values. The trust-calculating module calculates the trust value of a provider by considering the feedback of credible consumers and credible cloud service providers. The feedback depend on QoS parameters and SLA. Consumer credibility is the product of the total number of services consumed and their duration. Provider credibility is calculated based on service duration length and total number of services offered. The feedback of all consumers and providers that have a credibility value higher than the mean of the total calculated credibility value are considered as reliable feedback; however, the mean credibility value can be biased and a consumer or provider who is new their feedback are not considered at all. This approach calculates trust dynamically using a standard deviation of duration, which is considered as inversely proportional to trust. The proposed approach is suitable for a system that has existing records of consumers and providers; however, in real-life scenarios there are many factors that can influence the feedback of one provider for others. The authors do not define the criteria for credibility and the approach is based on a conceptual framework without any validation and implementation.

2.5.7 Trust and Risk Assessment Model

(Hammadi & Hussain 2012) propose a risk and reputation assessment framework for a third-party cloud service provider that use two inputs to help a cloud consumer in the decision-making process concerning whether to continue or recompose services: the trust of a provider and the risk of service level degradation. The proposed framework has three layers. A third party defines the time for QoS assessment and then divides the period into pre- and post-interaction

phases. The selection of a provider depends on user recommendations. Credible users receive a reputation request from the third party and reply with a trust value based on the provider's previous record which is stored in the information repository. However, the authors did not describe how to calculate the credibility of consumers or how to deal with biased and real consumers' feedback. The authors proposed a fuzzy logic approach by considering three inputs: a credibility value, a time-delay value and a recommendation opinion. The third-party SLA monitoring component aggregates all reputation values from all recommending users and calculates the final reputation value of the service provider. It also accesses the runtime SLA parameter and compares it with the threshold to identify the probability of failure. However, the literature did not describe the threshold parameters and the issues while migrating to other cloud providers.

Table 2.2 presents a summary of the approaches that come under this class. A description of each approach, their features and their limitations in forming a viable SLA is outlined.

Table 2.2: SLA-based trust model approach

SLA management approach	Description of the approach	Features of the approach	Issues/ limitations of the approach
Adaptive credibility model (Noor & Sheng 2011)	The adaptive credible model offers trust as a service by considering consumers' capability and majority consensus to distinguish between true and biased feedback.	Offers trust as a service that helps service provider to distinguish between true and biased feedback. Proposed model offers distributed feedback management to avoid the hurdles linked with the centralized system. The feedbacks of an experienced consumer have a higher value than other consumers.	The limit of majority is not described and if a certain group targets any provider then the system cannot tackle a problem. The authors considered only two factors for measuring the reliability of feedback. The proposed framework is suitable for existing consumers but there is no mechanism for how to bootstrap a new consumer.
Reliability based trust management model (Fan & Perros 2013)	Trust management framework that considers the familiarity and consistency of consumer and differentiates	Differentiates between true and biased consumer feedback. The system helps new consumers select an appropriate service provider.	The timeframe is not defined for deciding the early past and distant past. No mechanism defined for bootstrapping new consumers.

SLA management approach	Description of the approach	Features of the approach	Issues/ limitations of the approach
	between true and biased feedback.		The threshold for comparison is not defined.
Trust mining model (Marudhadevi, Dhatchayani & Sriram 2014)	The model helps cloud consumers to choose a reliable service provider during the negotiation phase. The approach uses rough set and Bayesian inference to calculate trust.	Calculates trust at two levels: before signing an agreement and during transaction. This approach ensures the consumer receives the services it expects and allows provider to monitor the performance. The framework suggests consumers keep using services or switch to another provider.	Reactive SLA management approach. No mechanism that shows how a provider can mitigate violation. The approach did not describe the procedure for violation penalties and its enforcement. Switching from one provider to another has many issues like data integrity and compatibility.
Markov chain model (Chandrasekar et al. 2012)	Trust in the service provider can be determined by employing a state-monitoring approach and establishing dynamic trust using the Markov chain model.	Effective monitoring technique using state monitoring and derived monitoring techniques. Dynamic trust calculations based on the deviation from actual value. Trust is represented by a numeric value, which is added or subtracted based on the performance behaviour.	Providers with a low reputation can start a business with different details and they would be able to get a 50% trust value. The approach was tested only using the Markov chain method without comparing with other methods. There is no method proposed if a system detects a deviation between agreed and monitored QoS.
SLA based trust model (Alhamad, Dillon & Chang 2010)	SLA-based trust model that help cloud consumers to select a suitable provider based on their trustworthiness values.	Trust model helps consumer to select a reliable cloud provider. The framework determines the responsible party in case of service violation and determines violation penalties. Credibility metrics define the trustworthiness of the provider.	The framework lacks the process of negotiation and SLA formation. The management of the service directory is not defined. How to differentiate between true and biased feedback for calculating provider's trust value is not described.
CSRD model (Muchahari & Sinha 2012)	The framework calculates the trust value of each provider based on the credible feedback from	CSRD and dynamic trust can overcome security, privacy and trust problems for the adaptation of the cloud. Trust of provider and	The model doesn't support a third party provider, which is generally needed in many real-time applications.

SLA management approach	Description of the approach	Features of the approach	Issues/ limitations of the approach
	consumers and providers. It keeps track of the dynamic trust value with respect to time and transaction.	consumer is calculated and updated dynamically.	Considering providers' feedback for other providers has many issues. The approach is applicable for existing providers and consumers. The criteria for credibility is not defined. The approach is abstract without any implementation and evaluation.
Trust and risk assessment (Hammadi & Hussain 2012)	Selects reliable cloud providers based on their reputation value and monitors runtime performance as defined in the SLA.	The framework helps the consumer to select an appropriate cloud service provider. A framework that provides real-time assessment for SLA monitoring. QoS assessment in both pre-interaction and post-interaction time phases.	No methodology defined for calculating credibility of consumer. No distinction between true and biased feedback. The parameters of the threshold are not defined. The issue of vendor lock-in while migrating to other cloud providers.

2.6 Proactive SLA Monitoring Approaches

A significant amount of research has been done on proactive and reactive SLA management in cloud computing (Al Falasi, Serhani & Dssouli 2013; Chandrasekar et al. 2012; Quillinan et al. 2010). In proactive SLA management, the system detects a possible SLA violation before it occurs and performs all necessary action to avoid actual violation. In reactive SLA management, the system monitors the SLA at runtime only. Predicting a possible SLA violation requires proactive SLA management to identify any discrepancies between parties and apply all necessary actions for possible remedy before the parties are affected. Proactive management approaches can be self-monitoring, self-healing, use a case-based reasoning approach, predict a violation based on QoS parameters, or use mathematical approaches. A summary of some of the proactive SLA management approaches is described in the following sub-section and presented in Table 2.3.

2.6.1 SLA Violation Prediction by QoS Prediction

As discussed earlier, an SLA is composed of one or more SLOs. Each SLO may comprised of one or many QoS measurements. For example, throughput is one of the SLOs defined in an SLA which depends on multiple components, each with a QoS throughput measurement. The prediction of QoS parameters plays a key role in avoiding SLA violations in an SLA management framework. When a provider predicts a difference between the agreed and actual QoS parameters, then it takes all necessary actions to manage it. The following subsections outline a few approaches that use QoS prediction to manage SLA in the cloud.

2.6.1.1 QoS Prediction by CloudPred

(Zhang, Zheng & Lyu 2011a) propose a neighbourhood-based collaborative approach. They presented the idea of sharing local cloud component usage with all users to calculate global usage. By using QoS data from the nearest neighbour and applying both user-based and item-based collaborative filtering approaches, they were able to predict the QoS for a particular user. First, they collected QoS data using the concept of user-collaboration in which all users send their previous web service QoS data to a central repository. Then, users with similar QoS data are grouped using Pearson's correlation coefficient. The significance weighting of the top N users reduces the influence of less similar users. Once similar users are identified, the QoS values are predicted based on the user-based and item-based collaborative filtering method. The approach is based on the assumption that consumers have used same QoS parameters for same services in past; however, in reality this will vary. Moreover, the study did not cover the criteria for monitoring and predicting intervals, which is very important for the decision-making process.

2.6.1.2 QoS Monitoring-as-a-Service (MoNaaS)

QoS monitoring as a service was proposed by (Cicotti et al. 2015; Romano et al. 2011), who offer the QoS monitoring ability to cloud consumers. The proposed model monitors the performance of cloud providers depending on a stream processing framework for quick and timely responses. The proposed framework operates on top of SRT-15 platform (Cicotti et al. 2011) that works in two-tier architecture: business logic tire and data tire. The business logic tire is composed of two modules: QoS monitoring and QoS checking modules. The QoS

monitoring module controls the monitoring process, managing the database scheme, parsing and adding a timestamp for digital signature. The QoS checking module is responsible for executing the monitoring algorithm, monitoring all QoS parameters, and, in the case of violation, notifying the QoS manager to manage the pool of QoS detectors that use the monitoring algorithm to parse the input with reference to defined ontologies. The authors did not mention how the prediction algorithm predicts QoS parameters. The prediction intervals are not defined and the authors did not describe what the actions are that are performed by a consumer or provider. The approach works only if a cloud consumer and cloud provider are using the SRT-15 platform. In another work, (Chaudhuri, Maity & Ghosh 2015) proposed a Hierarchical Modified Regularized Least Squares Rough Support Vector Regression (HMRLSRSVR) approach and considered previous service history to predict QoS parameters. A soft computing approach is used to validate their approach on a public dataset.

2.6.1.3 Local Neighbourhood Matrix Factorization (LoNMF) Method

(Lo et al. 2015) propose a local neighbourhood matrix factorization (LoNMF) method to predict QoS parameters. The authors used two level selection processes to select the nearest neighbours. The first level identifies and selects all those neighbours that have a close physical distance from the requesting user. In the second level, the system uses PCC and the top-K nearest neighbour selection algorithm to find similar neighbours. After the selection of the nearest neighbours, the system combines domain knowledge and an extended matrix factorization method for personalized QoS prediction. (Qi et al. 2015) integrate network and service neighbourhood information with the matrix factorization method to predict personalized QoS parameters.

2.6.2 Workflow SLA Violation Detective Control Model (WSVDC)

(Sun et al. 2013) propose Workflow SLA Violation Detective Control Model (WSVDC). The authors consider a utility function that measures the level of satisfaction and give control charts for each SLA. Performance is measured against four QoS parameters: response time, cost, reputation and reliability. They used the Western Electric rule to manage service behaviour and detect service violations. In statistical process control, the Western Electric Rules are decision

rules for detecting "out-of-control" or non-random conditions on control charts. Observables that lie outside control limits (typically at ±3 standard deviations) draw the monitor's attention on the service as they may predict future violations. The approach did not describe how the control charts and control rules are formed and it guarantees an optimal monitoring mechanism. The study only considers four SLA variables; however, in real life there are many other attributes which cannot be ignored. Moreover, the study did not describe reputation and reliability calculation mechanisms.

2.6.3 RaaS-based Early Warning Framework

(Hussain et al. 2014) propose Risk Assessment, as a service-based early warning framework. The framework detects future violations of SLAs based on SLO parameters or performance metrics. Their approach helps consumers to govern deviations in performance and assists them in avoiding violations before they occur. The framework comprises a number of modules. The early warning system monitors the difference between actual performance and predicts performance over a period of time. Depending on the difference between performance and the potential risk to the user, the system may suggest migrating the service. Autoregressive integrated moving average (ARIMA) and exponential smoothing methods are used to forecast the quality of cloud services. The QoS SLA violation detector determines the deviation between the QoS expected curve (QEC) and the QoS observed curve (QoC). To predict future violations, the previous QEC value is sampled into a different time interval based on the assumption that observing a previous pattern of behaviour can predict behaviour in the future. The risk propensity of all users is determined by three attitudes i.e. risk averse, risk neutral and risk taking. Recommendations to discontinue a service are dependent on the output of the fuzzy inference system. The decision-making module checks the direction of a deviation between the observed QoS value and the expected QoS. An output value of 1 shows the service is acceptable to the user, while a value of 0 shows the service is not. Although the approach covers both pre- and post-interaction phases, the approach lacks SLA negotiations and the formation of viable SLA that assures QoS parameters and appropriate actions to mitigate and avoid violations. Migration to other service providers raises issues such as vendor lock-in and data compatibility issues.

2.6.4 SLA Violation Prevention by Cross-layer Adaptation

(Schmieders et al. 2011) propose a cross-layer adaptation to manage SLA and prevent SLA violations of the service-based application. Service management is performed by a service level agreement monitor (SALMon) that compares the retrieved QoS with the expected QoS value in service-based applications. If a violation is detected, the SALMon sends a notification to the specification and assumption based detection (SPADE) module. This notification contains the assumption for the violation and the violating value. SPADE checks the requirements. If the requirements are not satisfied, the service-based application adapts to avoid any delays. If the requirements are fulfilled, the adaptation strategy engine (ASE) module is activated. Within the ASE module, each agent gathers information and negotiates the decision to adapt with the others. When the system detects a violation, a related process agent is activated to choose an adaptation strategy. These adaptations include service replacement, SLA re-negotiation or service infrastructure adaptation. The adaption strategies are very limited. For service replacement adaption, it is not guaranteed that every time a suitable replacement will be found.

2.6.5 Machine Learning-based Regression Technique

(Leitner, Wetzstein, et al. 2010) propose runtime SLA violation prediction, based on a regression model using existing data. This approach predicts a likely violation before an actual violation based on previous QoS data for each SLO. Prediction can be done at multiple checkpoints. At each checkpoint, there may be one of three types of information:

- *Fact data*: the data which are known at the checkpoint. This data are used as an input for determining unknown data.
- *Unknown data:* the data which are not known at the time of prediction. For accurate prediction results, it is necessary to know all related data.
- *Estimate data*: estimate data are all those data which are not available at the time of prediction but can be estimated. The checkpoint predictor module uses the fact data to approximate a numerical value for each SLO using a machine learning technique, such as regression. Approximation can be done on existing, known QoS and instance data. The prediction is then represented as a graphical user interface and the prediction manager

manages the entire life cycle of prediction i.e. its initialization, maintenance and termination. All predictions are stored in the database for future analysis.

The authors defined checkpoints for describing where the prediction should carry out that are based on assumption that is triggered by their proposed component 'hook' and 'checkpoint predictor'. However, these checkpoints do not have a factual basis, as for the prediction of SLA violation the approach should predict early enough that a provider or consumer performs early actions to avoid actual violations. Moreover, the predictions only work when existing data is available. The authors only used a machine learning method for prediction and did not describe the dataset. There are other different other prediction methods which give optimal result. The approach did not describe any method once the system predicts a violation or if an actual violation occurs. The authors propose a conceptual framework without any evaluation and implementation of their approach.

2.6.6 Prediction of Violation by Workload Analyser

(Ciciani et al. 2012) propose a workload analyzer for the cloud TM project that is able to anticipate future workload fluctuations and hence predict SLA violations by monitoring resource data. The workload analyzer manages and classifies consumption data at both the infrastructure and platform layer. It combines the data from all nodes of the cloud TM platform, then filters and correlates them. Once the data is gathered, it makes a complete workload outline of all applications, describing the current and future need for hardware and software resources. It uses various statistical functionalities to predict the future tendency of workload variations and generates a user alert to potential violations of their SLA. However, when the violation is predicted there is no mechanism that outlines what remedial actions need to be taken to avoid an actual violation. The proposed approach works only when a provider and a consumer are using a cloud TM project.

2.6.7 Resource Management by Heuristic Policies

(Cardellini et al. 2011) propose heuristic policies for application service management to produce an optimal solution. The approach automatically manages

resources at the application level while considering both QoS objectives and resource utilization. They proposed proactive and reactive heuristic policies that use a prediction algorithm based on the recursive least square algorithm to predict the workload for future time slots. The authors evaluated their approach using only a stochastic workload model. The policy is capable of detecting SLA violations but is unable to prevent them.

Table 2.3 presents a summary of approaches that come under this class, a description of each approach and their features and outlines their limitations in forming a viable SLA.

Table 2.3: Proactive SLA monitoring approaches

SLA management approach	Description of the approach	Features of the approach	Issues/limitations of the approach
CloudPred (Zhang, Zheng & Lyu 2011a)	User-based and item-based collaborative filtering methods are applied to predict future QoS values and avoid service violations.	Time-aware personalized QoS predictions for different consumers. Predicts QoS value based on previous experience.	The basis for selecting monitoring and prediction intervals is not defined. The approach works for evaluating consumers using the same QoS parameters for the same services; however, in real time this may vary.
QoS-MONaaS [52]	The framework offers monitoring as a service to all cloud consumers to monitor QoS parameters and detect service violations.	The QoS monitoring service allows consumers to monitor runtime services and predict violations in advance. The framework has a feature of complex event processing and content-based routing.	The approach only works when both the provider and the consumer are using the SRT-15 platform. Prediction intervals are not defined. No process is defined once the system detects a QoS violation.
WSVDC (Sun et al. 2013)	The approach uses the SLA utility function and control charts to identify the difference in workflow composition and the risks of cloud services to improve the quality of cloud services and performance.	Proactively detects SLA violations based on runtime monitoring parameters. It helps an enterprise to detect faults in its system and adjusts workflow in the case of changing providers. It improves workflow reliability.	The formation of control charts and control rules are not defined. Does not assure for an optimal monitoring mechanism. No procedure defined for reliability and reputation calculation. The study only considers four SLA variables but there may be other important variables that need to be examined. There is a need for a detective model that considers multiple criteria.

RaaS (Hussain et al. 2014)	ARIMA and exponential smoothing are used to predict QoS, the result of which helps the consumer to decide whether to continue with the same provider or migrate to some other service provider.	Generation of an early warning to alarm consumers about likely service violations. The approach uses FIS by considering the risk attitude of consumers and suggest service continuation or migration.	Migration from one provider to other provider raises the issues like vendor lock-in and data compatibility issues. Approach lacking a methodology for suggesting appropriate actions once consumer detects a violation. The pre-interaction phase lacks a negotiation process for QoS parameters and the formation of a viable SLA.
Cross-layer adaptation (Schmieders et al. 2011)	Discusses the assumption from the service-based application context to check whether real-time data correlates with it or not. In the case of a difference between agreed and predicted SLA, the approach chooses the appropriate adaption strategy to avoid it.	Cross layer adoption and prevention of SLA violation. Both the consumer and a provider benefit from service-based application. The adaption strategies minimizes the impact of service violation.	For service replacement, it is not always guaranteed a suitable replacement will be found. The adaption strategies are very limited. The approach does not cover the process once the violation occurs. Migrating to other service providers has many issues.
Machine learning regression (Leitner, Wetzstein, et al. 2010)	Runtime prediction of SLA violation for composite services using existing QoS data. Predicts SLA violations based on the prediction checkpoints.	Predict SLA violations of composite services. Checkpoints describe the execution of composite services and define the input of prediction. In case of violation it alerts the consumer about the likely violation.	The selection of checkpoints for prediction are not justified. Prediction only works if existing data is available. The dataset is not defined. No procedure on how to avoid actual violations when a system detects a likely violation.
Workload analyzer (Ciciani et al. 2012)	Workload analyzer predicts future workload and demand for resources. Statistical data are gathered from different nodes to develop workload profile.	Anticipates future workload fluctuations for SLA violation prediction. Generate an alarm when a violation is detected.	The approach only works when the consumer and provider are using the cloud TM platform. When violation detected no suggestion is made for appropriate remedial actions.
Resource management by heuristic policies (Cardellini et al. 2011)	The approach manages resources on runtime using proactive and reactive heuristic policies to help the cloud provider to manage	Helps application service provider to manage resources. Improved workload prediction model.	No procedure defined for SLA management once the violation occurs. There are no criteria for monitoring intervals.

	its resources to avoid violations.		

2.7 Managing Risk of SLA Violation

Risk management is an essential process that recognizes, evaluates and mitigates possible risk for long-term business success and the better management of business operations, and enables an optimal decision-making process which fulfills a desired objective. A risk is defined as the possibility or likelihood of the occurrence of failure or any negative impact on a business due to the vulnerabilities of a business (Hussain 2010). From a provider's perspective, this could be the loss of assets or reputation or any financial loss that a provider might experience by forming an SLA with a consumer. It is very important for the service provider to identify risks and estimate the amount of loss due to occurrence of those risks in an optimal way and to take all of the necessary actions to eliminate or minimize its effect and to avoid SLA violations. Compensation for SLA violation varies from case to case. In the case of a large module where a service provider cannot fulfill the required resources or when a provider wants to share the risk, then the main module is divided into sub-modules which are assigned to different resource providers. Apart from the SLA with the consumer, a service provider makes an SLA with each other provider covering violation compensation and the payment percentage for each successful transition. (Gu, Zhang & Tao 2012) propose an SLA violation compensation mechanism that defines the compensation agreement during the negotiation phase, indicating whether a resource provider is liable to pay compensation when a violation occurs. The amount of compensation to be paid by each service provider should be less than the total amount of compensation.

Trust and risk are directly related to one another. Online traders combine the trust and risk attitudes of online consumers to build promising consumer approaches and eventual good business behaviour (Pavlou 2003). Trust is one of the major factors that restrains consumers from choosing cloud services. Consumers feel hesitant to adopt cloud services because they find them unsecure and untrustworthy. The confidence of a consumer can be restored by a trust management framework that has consistent secure mechanisms (Sun, Chang & Li

2011). Trust can be either direct or indirect. A trustworthiness value which is determined by a consumer is direct trust. A trustworthiness value which is obtained from a third party is indirect trust. (Zhu et al. 2015) propose an authenticated trust and reputation calculation and management model that would help trusting parties to protect trust evaluation by considering historical trust, calculating the reputation of a provider and thus assist cloud service consumers in choosing an appropriate service provider.

(Kiran et al. 2011) propose a risk assessment methodology to help service providers and infrastructure providers with risk identification, risk minimization and risk monitoring. In the proposed approach, the risk assessment for the interaction between a service provider and an infrastructure provider comprises six steps. In the first step, a service provider determines the risk of all the infrastructure providers before sending an SLA request. In the second step, the infrastructure provider calculates the risk linked to the interaction when an SLA request is received from the service provider. In the third step, an infrastructure provider determines the risk associated with the transaction and calculates the probability of failure when the services are deployed. In the fourth step, the service provider calculates the risk of the offer received from the infrastructure provider. In the fifth and sixth steps, both the service provider and infrastructure provider continuously assess the risk and monitor the QoS parameters and low-level resources respectively.

The International Standard ISO/IEC 27005:2008 (27005 2008) provides guidelines for information security risk management. Guidelines included in the standard are establishing the context of risk management, assessing a risk, mitigating a risk and accepting a risk. The guidelines are for general risk management; however, organizations can use these guidelines to form their own risk management framework according to their own business requirements and commitments. (Zhang et al. 2010) propose a risk management model based on ISO/IEC 27001 and National Institute of Standard and Technology (NIST) management guidelines to identify the threats and vulnerabilities related to the cloud computing environment. The risk management framework they developed analyses, assesses and mitigates risk to help the service provider achieve better management of the SLA. The risk assessment is comprised of four steps: defining the likelihood of vulnerabilities and associated threats, determining the magnitude

of a risk, finding the level of a risk and taking all necessary actions to mitigate risk. The proposed approach was unable to elaborate on the control matrix for risk management, proactive risk detection or its early remedy or the implementation of their framework.

(Albakri et al. 2014) propose a security risk management framework that helps consumers to evaluate a risk and contribute to the risk assessment process. The framework allows consumers to define the legal requirements, identifies the risk factors and obtain feedback from a service provider. A service provider does not rely solely on risks identified by the consumer but determines all possible risks linked with the system. The framework is comprised of two parts: cloud consumers, which make up one part; and a cloud service provider, which is comprised of four modules. There is a common interface through which a cloud provider can communicate with cloud consumers. The authors claim that the framework is more realistic than other risk management models because of the participation of all stakeholders in a process; however, the study lacks validation and practical implementation of a framework.

(Morin, Aubert & Gateau 2012) identify the risks and challenges linked with cloud computing. They propose an SLA risk management framework that helps manage risk in runtime cloud computing environments to enhance governance, risk and compliance. Although their contribution highlighted important security issues, it lacked practical implementation and an evaluation of the proposed approach to validate its risk management framework.

(Wang, Liu & Liu 2012) propose a quantitative risk assessment model that defines risks, assets and vulnerabilities and makes a quantitative calculation to estimate the possibility of a risk occurrence. The model quantifies a risk based on the existing best practice approach. Although the study describes the vulnerabilities and risks of the cloud environment, the model is unable to assess assets. In addition to this, the authors do not describe how cloud providers impact how the system obtains its data.

The attack-defense tree (ADT) method was used to handle the threat and risk analysis in cloud computing environments (Wang et al. 2012). The ADT method considers the attack cost of vulnerabilities and the defense cost of defensive

actions. The model calculates the defensive cost based on the attacking cost with a ratio of success. The model uses Bayesian network theory to determine the amount of risk and help defenders take appropriate action; however, the suitability of the framework depends on the availability of authentic statistical data and the selection of a proper defense method.

2.8 Critical Evaluation of Existing Approaches on SLA Management and Gaps in the Literature

This section presents a comparative analysis of SLA management approaches from the viewpoint of an SME service provider to form viable SLAs in order to proactively manage possible SLA violations. The approaches are compared according to the basic parameters that are required for SLA management. They are the focus of SLA management process (whether pre- or post-interaction phase), their ability to predict future QoS to detect possible SLA violations, their approach defined as a process when a possible SLA violation is detected and recommendations for possible actions. The comparisions are presented in Table 2.4.

Table 2.4: Critical evaluation of existing SLA monitoring approaches

Source	SLA management Process		Predict SLA / SLO /QoS	Procedure defined when it detect violation threat	SLA violation recommendation
	Pre-interaction	Post-interaction			
Emeakaroha et al. (Emeakaroha, Brandic, et al. 2010)	✖	✔	✔	✖	✖
Emeakaroha et al.(Emeakaroha, Netto, et al. 2012)	✖	✔	✔	✖	✖
Brandic et al. (Brandic et al. 2010)	✖	✔	✔	✔	✖
Haq et al. (Haq, Brandic & Schikuta 2010)	✖	✔	✔	✔	✖
Emeakaroha et al. (Emeakaroha, Ferreto, et al. 2012)	✖	✔	✔	✔	✖
Mosallanejad et al. (Mosallanejad & Atan 2013)	✖	✔	✖	✔	✖
Katsaros et al. (Katsaros et al. 2012)	✖	✔	✖	✖	✖
Al Falasi et al. (Al Falasi, Serhani & Dssouli 2013)	✖	✔	✖	✖	✖
Chandrasekar et al. (Chandrasekar et al. 2012)	✖	✔	✔	✖	✖
Alhamad et al. (Alhamad, Dillon & Chang 2010)	✖	✔	✖	✖	✖
Wang et al. (Wang et al. 2011)	✖	✔	✖	✖	✖
Hammadi and Hussain (Hammadi & Hussain 2012)	✖	✔	✖	✖	✖
Muchahari and Sinha (Muchahari & Sinha 2012)	✖	✔	✖	✖	✖

Source	SLA management Process		Predict SLA / SLO /QoS	Procedure defined when it detect violation threat	SLA violation recommendation
	Pre-interaction	Post-interaction			
Cicotti et al. [52]	✖	✔	✔	✖	✖
Romano et al. (Romano et al. 2011)	✖	✔	✔	✖	✖
Sun et al. (Sun et al. 2013)	✖	✔	✔	✖	✖
Hussain et al. (Hussain et al. 2014)	✔	✔	✔	✔	✔
Leitner et al. (Leitner, Wetzstein, et al. 2010)	✖	✔	✔	✖	✖
Ciciani et al. (Ciciani et al. 2012)	✖	✔	✔	✖	✖
Cardellini et al. (Cardellini et al. 2011)	✖	✔	✔	✖	✖
Son et al. [10]	✔	✖	✖	✖	✖
Silaghi et al. [11]	✔	✖	✖	✖	✖
Badidi [14]	✔	✖	✖	✖	✖
Pacheco-Sanchez et al. [15]	✔	✖	✔	✖	✖
Wood et al. [16]	✖	✔	✖	✔	✖
Schmieders et al. (Schmieders et al. 2011)	✖	✔	✔	✔	✖
Noor and Sheng (Noor & Sheng 2011)	✖	✔	✖	✖	✖
Fan and Perros(Fan & Perros 2013)	✔	✖	✖	✖	✖

The comparative study of the existing literature demonstrates that a variety of approaches can be used for SLA monitoring, including a mutually agreed third party, monitoring at the provider or consumer side, and the hierarchical self-monitoring of SLAs. In almost every approach, SLA monitoring is performed periodically. In the case of discrepancies, violations are recorded and relevant parties are informed. The existing literature considers different methods for violation prediction, such as formula-based mapping between SLOs and resource metrics, defining threat thresholds or applying different mathematical approaches, such as prediction, exponential smoothing, ARIMA, the Markov chain theory and the recursive least square (RLS) approach to avoid a possible violation of an SLA. The majority of approaches perform SLA monitoring in the post-interaction time phase when both parties have finalized the SLA. Although existing approaches try to avoid SLA violations and maintain a trusted relationship between both parties, these approaches could be further improved by forming a viable SLA based on an intelligent determination of the likely violations by the consumer before entering into an agreement.

In addition to this, this comparison demonstrates that while the literature describes the SLA lifecycle, taking into account most of the phases of interaction between the consumer and provider, there are still many gaps in existing literature that

need to be filled. One of these gaps is the formation of a viable SLA between the consumer and the provider that enables the service provider to predict likely violations of the SLA prior to its formation and offer the amount of resources wisely to consumers, based on their reliability value and contract duration. The proposed viable SLA lifecycle is outlined in Chapter 4 of this thesis.

2.9 Conclusion

In this chapter, an extensive survey of the literature on SLA management in cloud computing was conducted. The chapter starts by discussing existing approaches to the SLA lifecycle. The different approaches to SLA management were divided into four classes based on their working attributes. Each of the classes were described and all of the methods that reside in each classes were discussed. The features and shortcomings of each approach were also identified. Finally, the existing literature was evaluated based on different attributes and it was found that there are still many gaps in SLA management from the service provider's viewpoint.

In Chapter 3, the gaps in the literature are identified and the problem associated with the existing literature is formally defined. In Chapter 4, a solution overview is proposed by describing the need for a viable SLA and then outlining how service providers can manage SLA in an optimal way by following a viable SLA lifecycle both in the pre-interaction and post-interaction time phases.

3 Problem Definition

3.1 Introduction

As discussed in earlier chapters, in the cloud computing environment, it is very important for a service provider to form a viable SLA with a consumer that guarantees a particular level of the provision of services. This allows the service provider to obtain the maximum financial returns from its resources without creating any SLA violations. A provider can get these financial benefits by offering its marginal resources wisely and by starting the process of SLA monitoring from pre-interaction time phase. In the pre-interaction phase, the monitoring process should consider the consumer's aptitude to execute SLA without committing any violations. The consumer's profile is one of the key elements in determining a likely service violation or non-commitment to the promised marginal resource. Therefore, the consumer's profile is used as an input for a decision-making recommender system that assists a service provider in the formation of a service agreement with the consumer. Once the provider is able to execute a viable SLA with the consumer, it then needs to monitor the runtime QoS parameters of each agreed SLO. To do this, the service provider needs an optimal prediction method that determines the likely resource usage behaviour of the consumer for future intervals. When a provider finds discrepancies between the agreed and the predicted QoS parameters, it then needs to make a crucial and timely decision based on certain parameters of the provider and the consumer to manage the risk of SLA violation.

To achieve the above-mentioned tasks, a service provider needs to have the consumer's profile history, a comprehensive viable SLA framework, an optimal prediction method and an intelligent risk management mechanism in order to develop a decision-making mechanism to manage and mitigate a risk of SLA violation. Chapter 2 extensively discussed the existing SLA monitoring approaches. This demonstrated that there are still many gaps within the existing literature. The shortcomings in the existing literature are described in the following section.

3.2 Gaps in the Literature

As mentioned in the preceding chapter, there is a significant amount of research in literature that focuses on SLA management and monitoring frameworks from the perspectives of both the service provider and the service consumer; however, there are still significant gaps that need to be filled. In this chapter, the gaps and shortcomings of existing SLA monitoring approaches from a service provider's perspective are identified. These are listed below:

- The existing literature focuses on SLA monitoring from a consumer's perspective, particularly when they finalize and execute SLA. Existing literatures were unable to describe a viable SLA which assures a provider the maximum utilization of marginal resources in order to obtain the maximum profit.
- Most of the literature presents post-SLA violation recommendations; however, none of the methods describe how to generate pre-SLA violation recommendations.
- Existing literature were unable to determine the maximum amount of marginal resources to be offered to a consumer, depending on its past performances which is required to form a viable SLA.
- Most of the existing literature either predicts or generates the post-interaction SLA violation recommendation; however, none of them are able to predict SLA violations before finalizing SLA.
- Most of the existing SLA monitoring approaches use a case-based reasoning (CBR) approach for recommendations; however, there are different issues linked to the CBR approach such as adaptation, processing time and storage that the existing literature is unable to answer.
- The existing SLA management lifecycles lack the inclusion of the viable SLA formation phase, which is an essential phase for the cloud service provider to avoid SLA violations.
- There is no work that describes how to identify, estimate and mitigate possible risks of SLA violations by considering the reliability of a consumer, the risk attitude of the provider and the the predicted trajectory of resource usage.
- Most of the existing literature is unable to validate their approaches on a real cloud dataset.

3.3 Key Terms and Concepts

This section defines the key terms and concepts which are used to formally define a problem in this thesis.

3.3.1 Cloud Computing

Cloud computing is the delivery of on-demand shared computing resources over the internet. It has a pool of shared computing resources and provides pervasive and on-demand access to them on pay-for-use basis.

3.3.2 Service Level Agreement (SLA)

A service level agreement (SLA) is the key agreement between the service provider and the service consumer that defines overall services, objectives, commitment, deliverability, obligations and violation penalties. By pursuing the SLA, both parties can track the discrepancies between the agreed and committed services.

3.3.3 Service Level Objective (SLO)

The individual performance metrics of the SLA are called service level objectives (SLOs). SLOs are the detailed measureable characteristics of the SLA and are comprised of one or many quality of service (QoS) measurements like throughput, response time, availability.

3.3.4 Cloud Service Provider

A cloud service provider is an entity or organization that offers cloud computing resources/services to other individuals or businesses.

3.3.5 Cloud Services

Cloud services are virtualized computing resources that are made available to cloud service consumers on demand through the internet by a cloud service provider.

3.3.6 Cloud Services Consumer

A cloud services consumer is an individual or organization that has formally contracted with a cloud service provider to use their cloud services.

3.3.7 Elastic Computing

Elastic computing is the ability of the cloud service provider to scale up or scale down cloud services by provisioning or de-provisioning resources with reference to changes in business requirements.

3.3.8 Virtualization

Virtualization refers to the creation of virtual resources such as infrastructure, computer hardware, operating systems, servers and computer networks while sharing underlying physical hardware resources.

3.3.9 Cloud SLA Management from the Provider's Side

Cloud SLA management from the provider's side is the process of negotiating, executing, monitoring and mitigating SLA violations by a cloud service provider.

3.3.10 SLA Violation Penalties

SLA violation penalties are the financial or reputation losses of a cloud service provider due to noncompliance of its commitments.

3.3.11 QoS Parameters

QoS parameters are the measurable characteristics on which the overall performance of the cloud service depends.

3.3.12 Time Interval

A time interval is the division of the execution time over equal consecutive slots.

3.3.13 Risk Propensity of the Service Provider

The risk propensity of the service provider is the tendency of the cloud service provider to deal with the risk of SLA violations.

3.3.14 Risk Averse

Risk averse describes the attitude of a service provider who wishes to avoid any risk. Such providers are reluctant to take risks.

3.3.15 Risk Neutral

Risk neutral describes the balanced behaviour of a service provider to deal with a risk which is neither risk averse nor risk taking.

3.3.16 Risk Taking

Risk taking describes the tendency of a service provider to take risks despite their consequences and dangers.

3.3.17 Risk Management

Risk management is the process of identifying, estimating and mitigating risks of SLA violation.

3.3.18 Reliability of Service Consumer

The reliability of service consumers is the trustworthiness value of the cloud service consumers in previous transactions with the cloud provider.

3.3.19 Time Period

The time period is the duration of time in which the service provider is bound to provide cloud services to the service consumer.

3.3.20 Time Phases

Time phases are the interval of time which is used as a unit of time over which the SLA management period is divided.

3.3.21 Pre-interaction Time Phase

The pre-interaction time phase is the cloud SLA management before the execution of the SLA.

3.3.22 Post-interaction Time Phase

The post-interaction time phase is the cloud SLA management after the execution of the SLA.

3.3.23 Predicted Trajectory

The predicted trajectory is the curved path of the predicted QoS for a particular time period.

3.3.24 Transaction Trend

The transaction trend of the consumer is the number of successful transactions divided by the number of the total transactions.

3.3.25 Consumer's Profile

The consumer's profile is the consumer's previous transaction history with the service provider.

3.3.26 Agreed Threshold

The agreed threshold is described in the SLA and is mutually agreed on by the consumer and the provider.

3.3.27 Safe Threshold

The safe threshold is a customized threshold defined by the provider for its safe side to alarm it and invoke the risk management module when a runtime QoS reaches or exceeds it.

3.4 Problem Definition

In earlier chapters, it was described in detail that to avoid violation penalties and to maintain its reputation value a service provider needs a viable SLA management framework that assesses and monitors SLA in the pre-interaction and the post-interaction phases. Due to elastic nature of cloud computing, it is crucial that small and medium providers organize and manage their resources in an optimal way. A service provider needs to form an intelligent viable SLA that

ensures the service provider has the maximum utilization of its reserved resources in order to maximize their profit. When a provider is able to form an SLA, then it need an optimum QoS prediction and runtime monitoring mechanism that assesses the QoS parameters and, in case of any difference between the agreed and predicted QoS parameters, notifies the provider and generates recommendations for an early remedial action.

Thus, a viable SLA management framework is comprised of two time phases: the pre-interaction time phase and the post-interaction time phase. In the pre-interaction phase, a provider chooses an appropriate consumer by considering the consumer's previous profile and offering resources accordingly. Consumers' request for required and marginal resources from a small or medium service provider are then granted with a condition that marginal resources should be kept in reserve for the requested consumer and will be used when there is a need for them. If the consumer violates its commitment by not using its marginal resources, a provider can lose a significant amount of revenue by not using its resources wisely. The mismanagement of resources could lead to service violations and violation penalties. The consumer's profile is one of the key factors to determine SLA violation (Hussain, Hussain & Hussain 2015b). A consumer in multiple transactions with no violation history is far more reliable for service execution than a consumer with a history of multiple violations. By considering a consumer's profile, a provider can determine whether or not to accept a consumer's request. When a provider decides to accept a consumer's request, then the next question is the quota for the marginal resources. All these questions occur within the pre-interaction time phase when a provider has received consumer's request for SLA formation but before the services are being provided.

When both parties have agreed on all SLOs and signed an SLA for execution, the next issue is the prediction of QoS parameters for future intervals and monitoring the runtime behaviour of the consumer for ongoing QoS parameters. A provider need an appropriate prediction algorithm that intelligently predicts the QoS parameters so it can make decisions about the provision of services.. The runtime QoS monitoring is the key to monitoring the likely behaviour of consumers for future intervals and is used as an input for recalibrated prediction results. The predicted QoS parameters are compared with the agreed ones. In case of a

difference, it informs the service provider so it can take appropriate action to mitigate a risk of an SLA violation.

The next challenge for a service provider is managing the risk of SLA violation when it finds a difference between actual and committed QoS parameters. Risk is the probability of losing something, such as reputation or financial returns, due to certain actions (Williams & Heins 1985). Risk is linked with uncertain, unpredictable or uncontrollable outcomes. For the appropriate management of a risk, the service provider needs to identify, predict and assess all risks in the best possible way in order to minimize their impact. Inadequate risk management can result in a service violation and violation penalties for the service provider. Therefore, a service provider needs an optimal risk management framework that intelligently identifies, estimates and mitigates all possible risks. The risk mitigation includes all necessary actions to eliminate or lessen the impact of a risk.

Based on the above discussion, the research problem that is addressed in this thesis is described as follows:

How to develop a provider-based optimal viable SLA management framework that assists the service provider in forming a viable SLA with the consumer that monitors the runtime behaviour of the consumer over the interaction time period, chooses an optimal prediction method to predict QoS parameters and uses an intelligent risk management framework to eliminate or minimize the probability of SLA violations and violation penalties.

The above broad research question can be decomposed into four specific research questions. These are presented in next section.

3.5 Research Questions

This sections outlines the research questions which are addressed in this thesis in order to achieve the objectives as mentioned in Chapter 1. The research questions are as follows.

Research Question 1: How can a provider form a viable SLA in the pre-interaction time phase?

Providers need a model that allows them to form a viable SLA formation in the pre-interaction time phase that intelligently creates an optimized personalized SLA by taking into account the consumer's reputation and profile history. One part of the research deals with this question.

Research Question 2: How can a provider select an optimized prediction method to detect SLA violations?

There are a number of different prediction methods that can be used to predict SLA violations. However, the results of the prediction methods vary from one another depending on different inputs. There is need for effective methods by which the service provider can predict instances of deviations in the QoS to be delivered. To answer this question, the research compares and analyses the accuracy of different prediction methods at various time intervals.

Research Question 3: How can providers manage the risk of possible SLA violations?

Once a service provider is able to predict a likely service violation, then it is very important for the service provider to manage the risk associated with it. Depending on the provider's attitude towards risk, the reliability of the consumer and the amount of loss, a provider will decide to either avoid or accept a risk.

Research Question 4: How can the proposed approach be evaluated and validated?

To evaluate the effectiveness and applicability of the approach proposed in this research, it is necessary to implement and validate all components of the framework and to combine them into a prototyping system based on the proposed methodology. This will verify its applicability and soundness. To do this, a prototype software implementation of the approach is developed that uses a dataset from existing cloud and related datasets, depending on which prototype is tested.

3.6 The Research's Approach to Problem-Solving

This thesis develops and subsequently tests and validates a methodology for viable SLA management from the service provider's perspective, bearing in mind

all research questions mentioned in Section 3.5. When developing a methodology, it is necessary to follow a systematic scientific method in order to ensure that it has a robust scientific basis. Therefore, this section describes the existing scientifically-based research methods and outlines the reason for selecting a particular research method.

3.6.1 Categories of Research Methods

The research in information system is categorized into two categories, namely the science and engineering approach and the social science approach.

3.6.1.1 Science and Engineering Approach

The science and engineering approach combines the scientific and the engineering methods to solve problems. The scientific method uses various techniques to investigate phenomena, get new information and combining them with the previous knowledge (Goldhaber & Nieto 2010). Galliers and Lands (Galliers 1990) state that in the engineering method, the goal is to make something that works. Generally, the engineering-based approach is divided into three levels: conceptual, perceptual and practical. These are described below.

- The conceptual level is the first level of the engineering approach. It deals with creating new concepts and perceptions through evaluation.
- The perceptual level is the second level of the engineering approach. It formulates a new method by designing and making a system to solve the problem.
- The practical level is the third level of the engineering approach. It deals with experimenting, testing and validating by using real world examples.

When solving a problem, the science and engineering method deals with new techniques, methodologies and concepts to form a novel framework that not only tells us about the problem but also how to solve it.

3.6.1.2 Social Science Approach

The social science research approach follows systematic plan to find evidence to prove or disprove the articulated assumptions based on gathered data (Burstein & Gregor 1999; McTavish & Loether 2002). This approach is usually conducted

through surveys or interviews and is divided into two parts: a quantitative methodology and a qualitative methodology.

- The quantitative method deals with quantifying evidence and depends on the statistical analysis of obtained data to validate claims.
- The qualitative method deals with direct observations and communication through interviews and surveys. The data obtained is used for analysis.

The social science research approach helps the researcher to understand social and cultural issues within the research domain. This research approach only implies the extent of acceptability of the methodology or may be some time the reason of it as well; however, unlike engineering-based approach, this research approach does not propose the methodology for solving a problem (Kaplan & Maxwell 2005).

This thesis deals with the development of a novel methodology for SLA management that clearly falls into the domain of science and engineering research.

3.6.2 The Choice of the Science and Engineering-Based Research Method

This thesis follow a science and engineering-based research method as the research methodology for the proposed solution development. This is presented in Figure 3.1.

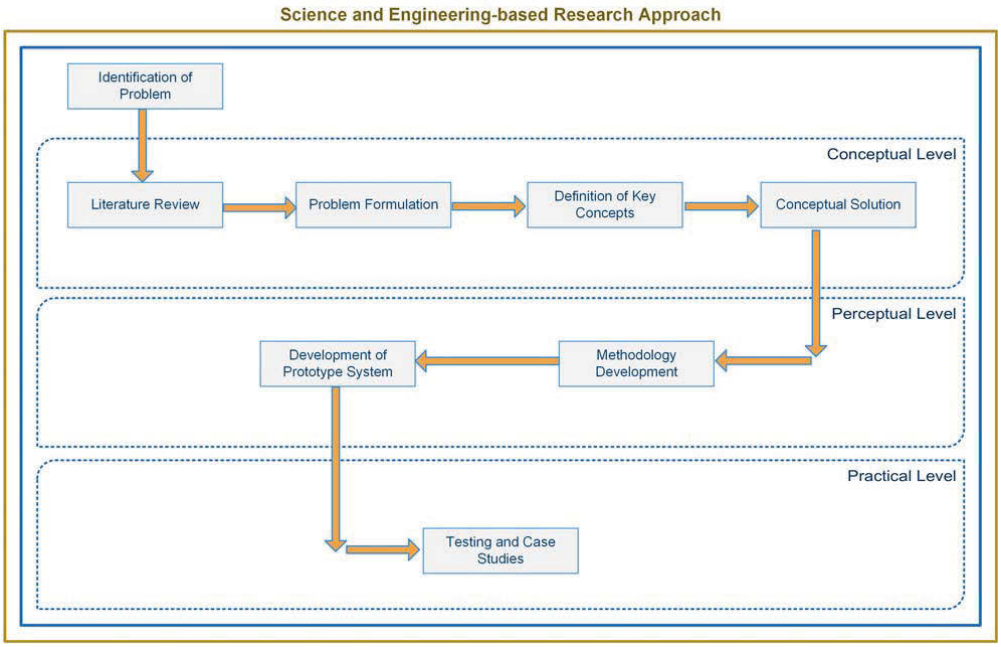


Figure 3.1: A science and engineering-based research method

The research starts by identifying research problems in the area of cloud SLA management from the provider's perspective. Existing literature is analysed with reference to this study. Based on a thorough review of existing literature, the research gaps are identified and the problem that needs to be addressed was formulated. The related key concepts that are used in developing the conceptual solution are defined in Chapter 4. These steps – the literature review, problem formulation, defining key concepts and a conceptual solution – are part of the conceptual level. The next level is the perceptual level, which is comprised of methodology development and the development of a prototype system. The methodology addresses various parts of cloud SLA management including viable SLA formation, threshold formation, QoS monitoring, QoS prediction and risk management.. The last level is the practical level, in which the prototype system is used together with the developed case studies to evaluate and validate the proposed approach.

The proposed research contains the development of a novel framework that requires the identification of problems and analysis of the developed concepts through validation. Therefore, a research method proposed by Nunamaker and Paradan (Nunamaker Jr, Chen & Purdin 1990) was followed. This method is comprised of three components: problem identification, conceptual solution and prototype development. The output of the validation of the developed prototypes forms the foundation for the assessment of research outcomes.

3.7 Conclusion

The chapter starts with the identification of the gaps related to the provider's side SLA management. The key terms and concepts that are used in designing and evaluating the framework were defined. A formal definition of the problem that is addressed in this research was presented. The identified problem was then divided into four research questions that need to be addressed in order to solve the defined problem of provider's side cloud SLA management. Furthermore, various research approaches are described and discussed. The science and engineering methodology is the most suitable approach related to this research's requirements for solving problems, therefore that is the methodology chosen for this research.

In the next chapter, an overview of the solution proposed to solve the problems address in this chapter is presented.

4 Solution Overview

4.1 Introduction

Cloud computing not only provides intelligent solutions to enterprise challenges, it also opens a door of opportunities for small businesses and end users. In small businesses, users can save significantly on the upfront cost of pre-planning the required resources, operation and maintenance costs as well as reducing their hardware and software costs, saving additional costs on upgrading computing resources and improving IT staff and employee productivity by enabling them to focus on core business activities (Etro 2009). Cloud computing provides consumers with on-demand self-service whereby they can access virtually unlimited resources to satisfy increasing business needs (Mell & Grance 2011). However, while such features free consumers from the worry of managing resources, they present a resource management predicament for cloud service providers, who need to manage their available resources among requesting consumers in the best possible way to maximize efficiency.

As mentioned in Chapter 2, there is a significant amount of research in the literature that has made attempts and advances to solve the problems of SLA management from a service consumer and service provider perspective. However, it is apparent from the discussion in Chapters 2 and 3 that there are still significant gaps concerning SLA management from the provider's perspective that need to be filled. Chapter 3 provides the four research questions that are aimed at solving this crucial problem.

This chapter presents an overview of proposed solutions. It is organized as follows. Section 4.2 proposes the definition of a viable SLA management from the provider's perspective. Section 4.3 describes the steps for forming a viable SLA with the consumer. Section 4.4 outlines the proposed solution. Section 4.5 presents a description of the viable SLA formation. Section 4.6 discusses the threshold formation, runtime QoS monitoring and QoS prediction module as well as the risk management module to avoid SLA violation. Section 4.8 concludes the chapter.

4.2 Definition of Provider's Side Viable SLA Management

As discussed previously, cloud SLA management is different from the perspective of the service provider than it is from the consumer's perspective. However, not much attention given in the literature. Therefore, before describing the proposed solution for viable SLA management for service providers, a specific definition of provider's side viable SLA management is provided. The definition is as follows:

> Provider side viable SLA management is the process that assists the service provider in the decision-making process at two time phases. The first phase is the pre-interaction time phase in which the service provider decides whether or not to accept the consumer's request, and, if affirmative, then an optimal viable SLA is formed by offering a maximum limit of marginal resources based on the available resources. In the post-interaction time phase, the runtime performance of offered services is monitored and optimal prediction methods are applied to predict and manage the risk of likely SLA violations.

The process of a viable SLA management is composed of two phases. In the first phase, the provider is more concerned with the formation of a feasible workable SLA that ensures the provider receives maximum financial returns from its resources that were kept in reserve and not used. In the second phase, the provider has already executed SLA with the consumer and wants to monitor and predict the runtime behaviour of the consumer for the period of a contract. When the provider finds any differences between the agreed and predicted behaviour, then it estimates and mitigates the risk of SLA violation. Therefore, the provider's side of viable SLA management is comprised of three main tasks:

1. Formation of a viable SLA with the consumer.
2. Monitor the runtime QoS parameters and predict QoS parameters for future intervals.
3. Manage the risk of SLA violation when there are any discrepancies between the agreed and runtime behaviour of the consumer.

The next section describes the steps linked with the formation of a viable SLA.

4.3 Steps in Viable SLA Management Framework

Forming a viable SLA is very important for the service provider, particularly for small or medium providers, so that the provider can manage their resources wisely. In this section, it is argued that in order to minimize SLA violations and monitor SLA in an efficient way, processes need to start in the SLA negotiation phase. As outlined in Chapter 2, there are a number of proposed different approaches and recommendations for proactive and reactive SLA violation detection; however, almost all of these approaches are designed to detect SLA violation once the SLA is formed between the provider and the consumer. For better management of SLA, the process of monitoring should start when the provider receives a request for services from the consumer. By establishing an extended timeframe for SLA monitoring, the service provider has the opportunity to observe the past commitment and/or behaviour of a cloud consumer and, based on that, can design informed and viable SLAs that have a high chance of success.

This is common practice in many business domains, such as finance, where service contracts (the financial counterparts of SLAs) are formed only with consumers that have a high probability of leading to a positive outcome even at pre-negotiation stage, and preventive actions are taken for their successful outcome (Saunders 1985). This concept is much rarer in cloud computing, where SLAs tend to involve limited negotiation and are formed on the fly. However, SLAs for multi-tenant cloud services need careful planning and monitoring to protect the cloud provider from breaching its obligations when many users place requests at the same time. This is particularly beneficial for small cloud service providers who do not have the vast infrastructure at their disposal that big cloud service providers do, hence they need to form viable SLAs with service consumers to better manage their resources and to avoid the chance of SLA violation.

The process of a viable SLA formation is comprised of eight steps, which are described here and presented in Figure 4.1.

> Step 1 - Resource / service request received from consumer: Consumer requests service and/or resource requirements in a formal manner.

Parameters accompanying the request include type, quantity, duration and importance.

Step 2 - Resource allocation criteria determination: When the provider receives the request from the consumer, then unbeknownst to the consumer, the provider uses intelligent algorithms to determine the trust value of the requesting consumer along with the time the resources are requested for. These criteria play a crucial role in determining whether to allocate part of or full resources to the consumer during negotiation.

Step 3 - Analysis of request based on the resource allocation criteria determination: In this step, the provider compares the criteria determined for the consumer against its defined threshold values. Step 4 - Provider decision to accept, reject or negotiate: Depending on the criteria determination result, in this step the provider may decide to:

a. accept the request as is;
b. provisionally accept the request but negotiate to formalize amount of resources to be offered; or
c. reject the service request – especially if the resource allocation criteria indicates the consumer is likely to violate the service agreement.

Step 5 - Formulate the SLA: After the negotiation and re-negotiation steps, both parties come to a mutual agreement and a SLA is formed.

Step 6 - Threshold formation: Once the interaction with the provider and user starts, based on the agreed thresholds in SLA, the service provider forms a customized threshold to alarm for early possible service violations.

Step 7 - Runtime QoS monitoring and QoS prediction: In this step, the QoS parameters for future intervals are predicted and monitored against the runtime QoS parameters. In case of variation between predicted and observed QoS parameters invoke risk management module for immediate necessary actions for SLA management.

Step 8 - Risk of SLA violation: identifying possible risks of SLA violation, estimating the severity of a risk, and calculating ways to mitigate it.

Section 4.4 provides an overview of a proposed viable SLA management framework that would assist cloud providers in management decisions regarding SLA formation, execution, monitoring, violation detection and mitigation.

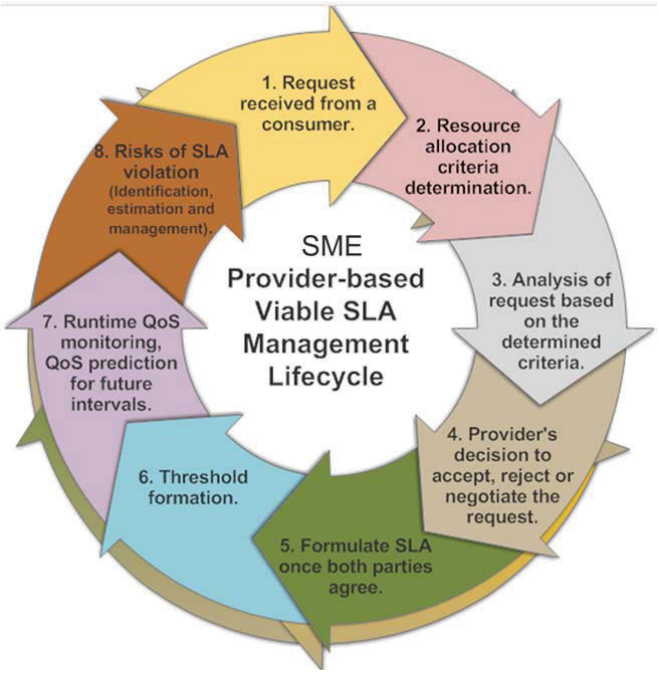


Figure 4.1: Viable SLA life cycle from a SME provider's perspective

4.4 Overview of the Proposed Solution

As discussed in previous chapters, the problem of provider side viable SLA management has different folds and many challenges that have not been identified and explained in the existing literature such as decision about service formation with a consumer, the amount of marginal resource offer to a consumer, necessary action when a system identify risk of SLA violation and a complete SLA management framework that start the SLA monitoring in pre-interaction time phase. This section presents an extensive overview of the framework that is proposed to meet these challenges in the cloud computing environment. The solution is named the Provider Side Viable SLA Management Framework.

The proposed OPVSLA management framework is comprised of two time phases: the pre-interaction time phase and the post-interaction time phase. The pre-interaction phase consists of all of the steps and processes that occur before finalizing and executing SLA, which include the authentication and validation of the consumer and forming an intelligent viable SLA with them. The post-interaction phase is comprised of steps and processes where both parties have

agreed on SLA and executed it. The processes are threshold formation, runtime QoS monitoring, QoS prediction and a risk management module. An overview of the proposed solution with all its components is presented in Figure 4.2.

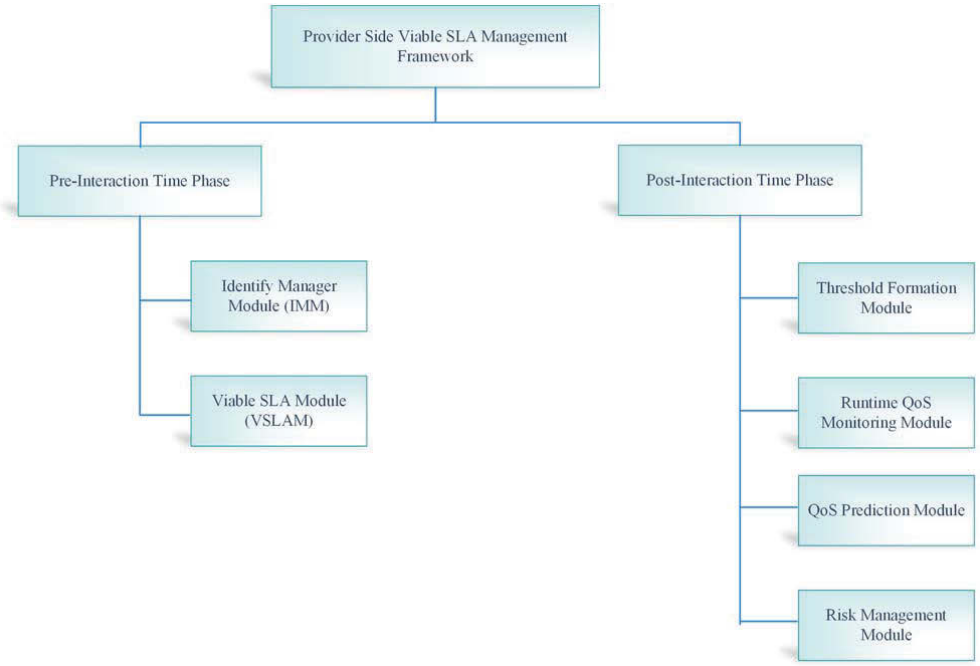


Figure 4.2: Overview of the whole solution of the Provider Side Viable SLA Management framework

As presented in Figure 4.2, the proposed framework is comprised of different modules that work together in both time phases to achieve the desired objective of viable SLA formation and management. Different modules have multiple components that combine together to form the viable SLA management framework, as presented in Figure 4.3. Depending on process of the workflow, the above-mentioned modules are structured into three sections as follows:

- Section 1: Viable SLA formation.
- Section 2: QoS monitoring and predicting.
- Section 3: Risk management of SLA violations.

Section 1 is situated in the pre-interaction time phase and it deals with the authentication of the consumer, the decision on whether to accept the consumer's request and the formation of the viable SLA to ensure maximum utilization of resources and minimum chances of SLA violation. Sections 2 and 3 resides in the post-interaction time phase and deal with the activities that occur after an SLA is formed between the consumer and the provider. The activities are runtime

threshold formation, QoS monitoring, QoS prediction, comparing prediction results with the threshold value and, in case of QoS degradation, generating an early warning to the provider along with type of action to eliminate or minimize the risk of SLA violation and violation penalties.

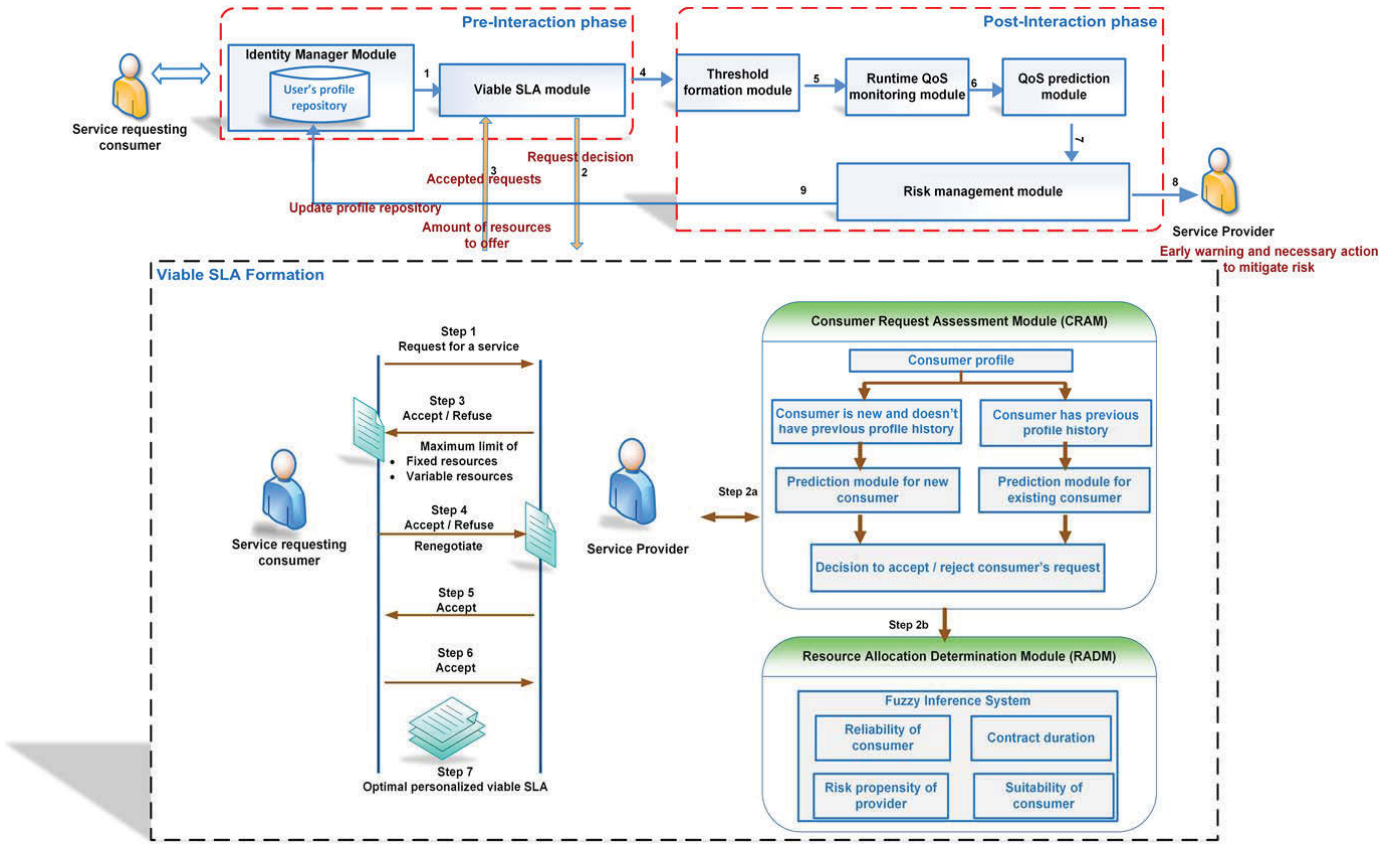


Figure 4.3: The proposed OPVSLA management framework and the flow of information between various modules

The steps involved in two phases of the SLA management are described in Table 4.1.

Table 4.1: Activities in both phases of OPVSLA management framework from SME provider's perspective

Pre-interaction phase	Post-interaction phase
1. Expression and classification of consumer's requirements in a formal manner: • Service/resource requirement • Duration of service/resource • Prioritization of different components • Amount of resource/service required 2. Resource allocation criteria determination by provider. 3. Provider determines to accept / reject / negotiate consumer's request. 4. Provider determines the capacity of resource / service offered to consumer. 5. Form the SLAs.	6. Provider determines the threshold value for resource usage. 7. Real-time monitoring of consumer's behavior. 8. Comparison between real-time performance and expected performance. 9. Generation of an early warning to alert cloud provider to avoid possible service violation. 10. Identify, estimate and manage the risk of possible SLA violation by generating recommendation. 11. Update trust value of consumer after the SLA agreement is finished for future formation of SLAs.

Each section of the framework is explained in detail in the following sections of this chapter.

4.5 Overview of the Section 1: Viable SLA Formation

There are two modules – the identify manager module (IMM) and the viable SLA module (VSLAM) – that work together to form a viable SLA with the consumer, as presented in Figure 4.3. Unlike existing approaches, the proposed approach starts the process of SLA monitoring in the pre-interaction time phase, when the consumer requests services from the provider.

When the provider receives a request from the consumer, the request is moved to the first module of the framework's IMM, where the consumer is authenticated and validated. There is one of the possible conditions here: either the consumer is new to the provider or it has existing previous records. In either condition, the consumer is validated and the request is forwarded to the VSLAM.

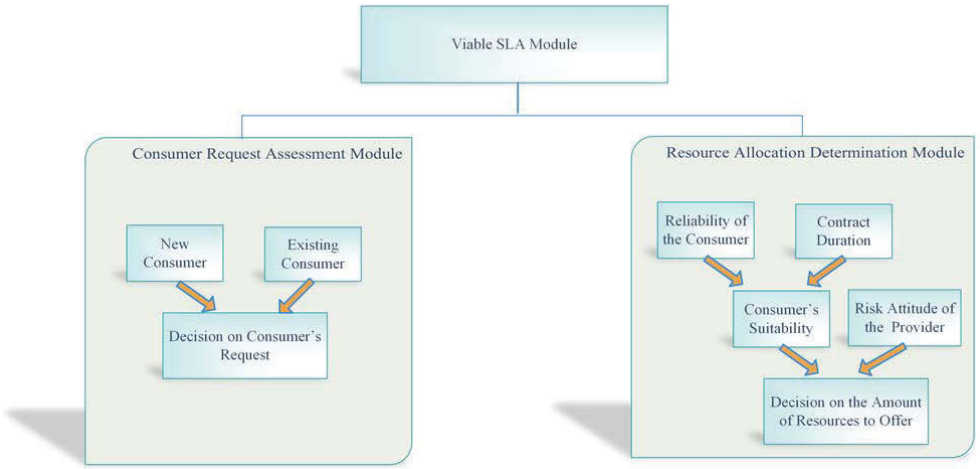


Figure 4.4: The viable SLA module

The VSLAM is the key module which helps the service provider in the decision of whether or not to accept or the consumer's request and, if affirmative, then the amount of resources to offer to them. The detail of VSLAM is presented in Chapter 5 of this thesis. The VSLAM comprises of two sub-modules, the consumer's request assessment module (CRAM) and the resource allocation determination module (RADM), as presented in Figure 4.4.

4.5.1 Consumer's Request Assessment Module (CRAM)

CRAM helps the service provider to make a decision about the consumer's request. CRAM considers the transaction trend (T_{trend}) of the requesting consumers to determine whether to allocate them resources or not. The value of T_{trend} is calculated from the number of successful transactions divided by the total number of transactions. For existing consumers, CRAM considers the consumer's previous profile (Walayat Hussain 2015) and calculates their T_{trend} value. For a new consumer who does not have a previous profile, CRAM determines the top-K nearest neighbours that are similar to the requesting consumer's and, from their T_{trend} value, calculates the likely T_{trend} value of the requesting consumer (Hussain et al. 2016; Hussain, Hussain & Hussain 2016a). The T_{trend} value represents the trustworthiness value of the consumers. The T_{trend} value of the requesting consumer is compared with the defined threshold value. CRAM either accepts a request or not. If the T_{trend} value is more than the defined threshold value, then the consumer's request is accepted; otherwise, it is not. In the case of acceptance, the service provider then decides the amount of resources to offer to the accepted

consumer. RADM helps the service provider to determine the amount of resources to offer to consumers. RADM is explained in Section 4.5.2.

4.5.2 Resource Allocation Determination Module (RADM)

RADM helps the service provider to decide the amount of resources to offer to the consumer. The cloud provider offers two type of resources: static and marginal. Due to the elastic nature of the cloud, it is very important for the service provider to offer the amount of its marginal resources wisely (Hussain et al. 2016). The decision on the amount of resources to offer depends on the reliability value of the consumer and the duration of the contract for which the consumer is requesting for services. RADM uses a fuzzy inference system (FIS) that takes three inputs – the reliability of the consumer, the duration of the contract and the risk propensity of the service provider – and determines the amount of marginal resources offer to the consumer. The provider informs the consumer about the status of its request and the amount of resources that will be offered to it. The consumer accepts, rejects or renegotiates the offer depending upon its circumstances. When both parties agree, a formal SLA is signed and executed. Once the SLA is formed, then the monitoring process shifts to the post-interaction time phase.

4.6 Overview of Section 2: QoS Monitoring and Predicting

The need for precise QoS predicting and QoS monitoring is very important for the service provider so that it can manage the risk of SLA violation and avoid violation penalties. As discussed earlier, Sections 2 and 3 consist of all those components and activities which are concerned at the post-interaction time phase when both parties have executed SLA. Therefore, this section is designed to accomplish these tasks and it has three components for this purpose:

- Threshold formation module;
- QoS monitoring module; and
- QoS predicting module.

The modules associated with this section and the exchange of the information among them are presented in Figure 4.5.

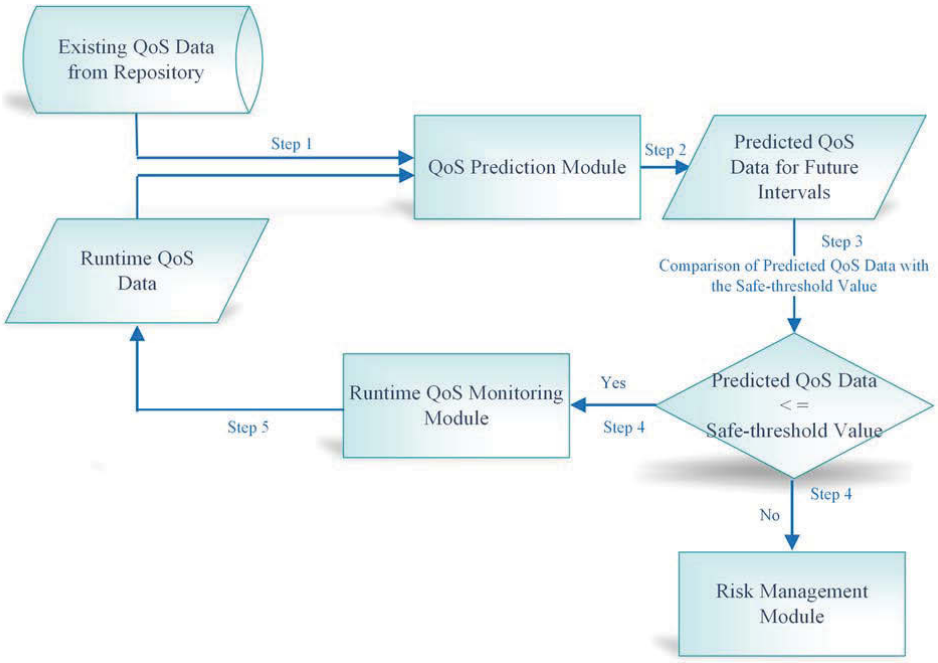


Figure 4.5: Working of Modules in Section 2 of the Framework

The first component in Section 2 of the framework deals with the formation of the threshold. Two thresholds are proposed to assist the service provider in avoiding service violation, namely:

- Agreed threshold (T_a)
- Safe threshold (T_s)

T_a is the agreed threshold value defined in SLA and agreed by the consumer and the provider for all SLOs. T_s is the threat threshold or the provider's customized threshold that is used as an early warning to alarm the provider to take any action if required when the predicted QoS reach or exceed it.

The second and third component of this section deals with the prediction and monitoring of QoS parameters. The runtime QoS monitoring module is responsible for monitoring the runtime QoS parameters for each agreed SLO. The obtained runtime QoS parameters are sent to the QoS predicting module, where they are used to recalibrate the QoS prediction for future intervals. The QoS prediction module uses the previous behaviour of the consumer obtained from the repository along with the values obtained from the QoS monitoring module to predict future intervals. The QoS prediction is performed using various prediction algorithms such as ARIMA, simple exponential smoothing, simple moving

average, weighted moving average, the Holt-Winters double exponential smoothing method, the extrapolation method and the neural network approaches. The predicted QoS are sent to the decision-making module, which compares the difference between the predicted and agreed QoS parameters. If the difference is less than the T_s value, then the process of monitoring continues but if the difference is greater than the T_s value, then the risk management module is activated to generate an alarm to the provider thus providing an alert for a possible risk of service violation and suggesting a possible remedial action, if required.

4.7 Overview of Section 3: Risk Management of SLA Violation

Section 3 of the proposed framework deals with the risk management of SLA violations. Risk is an important factor that cannot be ignored by the service provider. The risk management module uses an FIS that considers three input values – the reliability of the consumer, the risk attitude of the provider and the predicted trajectory – and, based on these three inputs, estimates the amount of risk and necessary action to mitigate the risk of an SLA violation. An overview of the risk management module is presented in Figure 4.6.

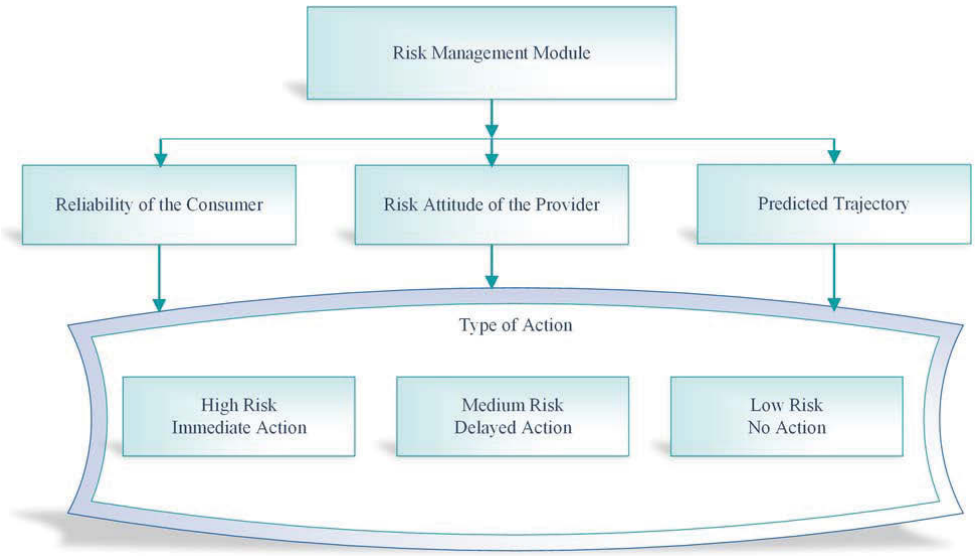


Figure 4.6: Risk management module

When the risk is assessed as high then the provider takes immediate action to mitigate the chance of actual violation. When the risk is assessed as medium, then depending on the nature of inputs the module decides either to take an action or ignore it by accepting the risk. When the risk is assessed as a low risk, then the

provider does not take any action and accepts the risk. In this way, the risk management module helps the service provider to manage the risk of SLA violation by arranging the deficient resources within a certain time frame before affecting either of the parties. At the end of the transaction the reliability value of the consumer is updated in the repository.

4.8 Conclusion

In this chapter, a general overview of the proposed viable SLA management framework was presented. All of the modules and components that work together to solve the problems identified in Chapter 3 were identified. The chapter first presents a generic definition of the provider side viable SLA management, which is followed by the steps in forming a viable SLA. The proposed framework is then described before being divided into three sections based on the process of their workflow. The sections are viable SLA formation, QoS monitoring and prediction and risk management to avoid SLA violation.

In the next chapter, the viable SLA module is presented in detail and its application in the context of the cloud service provider is thoroughly discussed.

5 Viable SLA Module in the Pre-interaction Time Phase

5.1 Introduction

The VSLAM reside in the pre-interaction phase of OPVSLA management framework. As discussed in previous chapters, the existing literature on SLA management mostly focuses on post-interaction SLA monitoring approaches in which the process of SLA monitoring starts when both parties have executed the SLA (Ferretti et al. 2010; Hussain et al. 2014; Leitner, Michlmayr, et al. 2010; Romano et al. 2011). Although these approaches are significant for predicting SLA violations, none of them define an optimal viable SLA management framework that can help a service provider form a viable SLA and start the process of SLA monitoring prior to interaction, i.e. when the consumer first requests resources.

This module helps the service provider to make optimally viable SLA by

a) deciding whether to offer the requested marginal resources to the consumer, and
b) if affirmative, deciding the level of resources to offer.

To make such a decision, the consumer's previous profile history is a key element in determining likely service violation or non-commitment to the promised marginal resources and it is used as an input for the decision-making recommender system, which assists the service provider in the formation of a service agreement with the consumer. This can be understood better by considering a scenario in a financial institution. Let consider that consumer A needs a sum of money and requests a loan from a bank. In making its decision, the bank has to consider two criteria: first, whether to approve or reject the request; and second, if the request is approved, the limit it would place on the offer to the consumer. The first priority for every financial institution in making such a decision is to know the previous profile or credit history of the consumer. If the consumer has a bad credit history, the bank may not approve the loan or may approve a lesser amount than requested and/or may impose extra measures and conditions. A similar case applies for a cloud service provider when it has to form a viable SLA with a consumer. In such a scenario, it needs to evaluate the consumer's profile for usage history and commitment to previous SLAs before

finalizing an agreement, because once the SLA is drawn up, the provider is obligated to reserve the agreed resources for the consumer, whether or not the consumer uses them. Failure to do this would result in a service violation by the provider that would have an impact on their reputation or significantly affect their profit. To avoid these negative scenarios, a viable SLA module assists providers to make viable and economic SLAs with consumers.

The chapter is organized as follows. This section introduces the chapter. Section 5.2 discusses the identity manager module, while Section 5.3 provides an overview of the viable SLA module. Section 5.4 outlines the calculation of the reliability and the transaction trend of the consumer. Section 5.5 describes the decision-making system concerning whether to accept or reject the consumer's request. Section 5.6 describes the amount of resources to offer to accepted consumers. Section 5.7 discusses the FIS and all inputs and outputs for the decision system, while Section 5.8 validates the module, Sections 5.9 and 5.10 discuss the scenario of requesting consumers and Section 5.11 concludes the chapter.

5.2 Identity Manager Module (IMM)

When the cloud provider receives a request from a consumer, the request and consumer's details are sent to the IMM, as shown in Figure 5.1. The IMM holds the consumer profile repository, which contains a record of all of the users that have previously formed an SLA with the provider. The IMM is responsible for checking whether a consumer is new or whether it already has a profile history with the provider. After authentication, the IMM forwards the request to the viable SLA module (VSLAM) along with the consumer's profile and reliability value, if they exist.

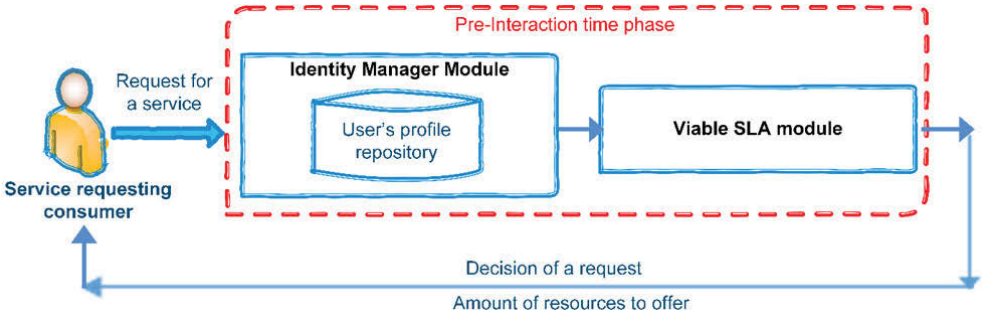


Figure 5.1: Pre-interaction time phase of viable SLA management framework

5.3 Viable SLA Module

The viable SLA module is the key module in the SLA management framework and it assists cloud providers in two ways. First, it assists in making a decision on whether to accept or reject a consumer's request based on the information received from IMM. Second, if the decision is to accept the request, it assists the cloud provider by recommending how many resources to offer. To make a decision on whether to accept or reject a consumer's request, the VSLAM determines the consumer's transaction trend. The transaction trend (T_{trend}) is discussed in Section 5.4. For existing consumers that have a profile of previous resource usage, the transaction trend is determined according to this profile. For a new consumer that has no resource usage profile or reliability value with the service provider, the module uses intelligent methods such as the user-based collaborative filtering method and top-K nearest neighbour selection to identify similar users that match the consumer's profile pattern. It then uses the resource profiles of these users to determine the consumer's transaction trend.

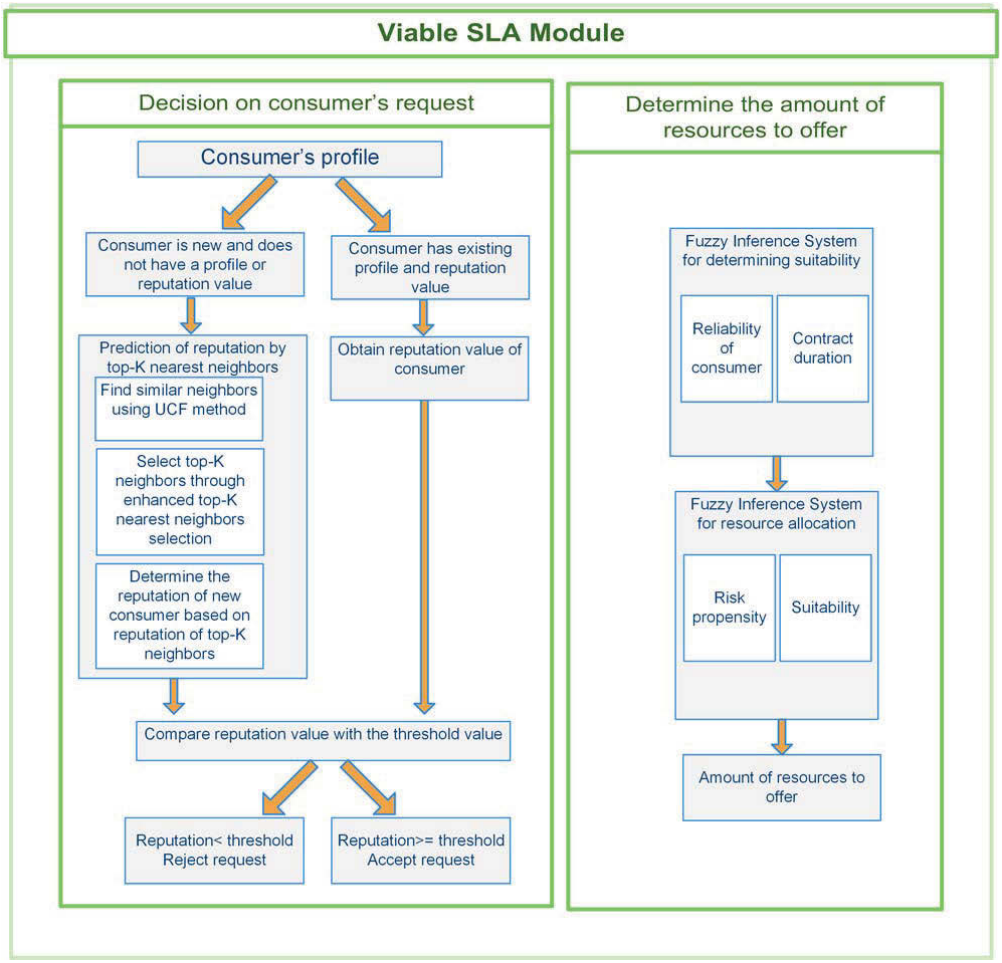


Figure 5.2: Components of a viable SLA module

To be able to accept a consumer's request, the transaction trend must be greater than or equal to the threshold value. The threshold is the amount of success ratio defined by a provider, and a provider decides whether to accept a request based on that value. To determine the extent of resources to be offered, VSLAM uses a fuzzy inference system (FIS) that takes two variables, the reliability of the consumer and the duration of the contract as inputs, to determine the suitability of the consumer. It then combines this information with the risk propensity of the provider to ascertain and recommend the amount of resources to be offered to the consumer. The framework and the working of the OPV-SLA framework are represented in Figure 5.2.

The complete process of the OPV-SLA framework comprises six steps, as shown in Figure 5.3. These steps are explained as follows:

Step 1. Consumer requests a service: In this step, the consumer requests resources from the service provider. The service request contains three components:

- The purpose of service requirement;
- A proclamation of service requirements; and
- The grading of each service requirement.

The first component, the purpose of service requirement, is the process of identifying the needs of the cloud provider and cloud consumer within their relationship.

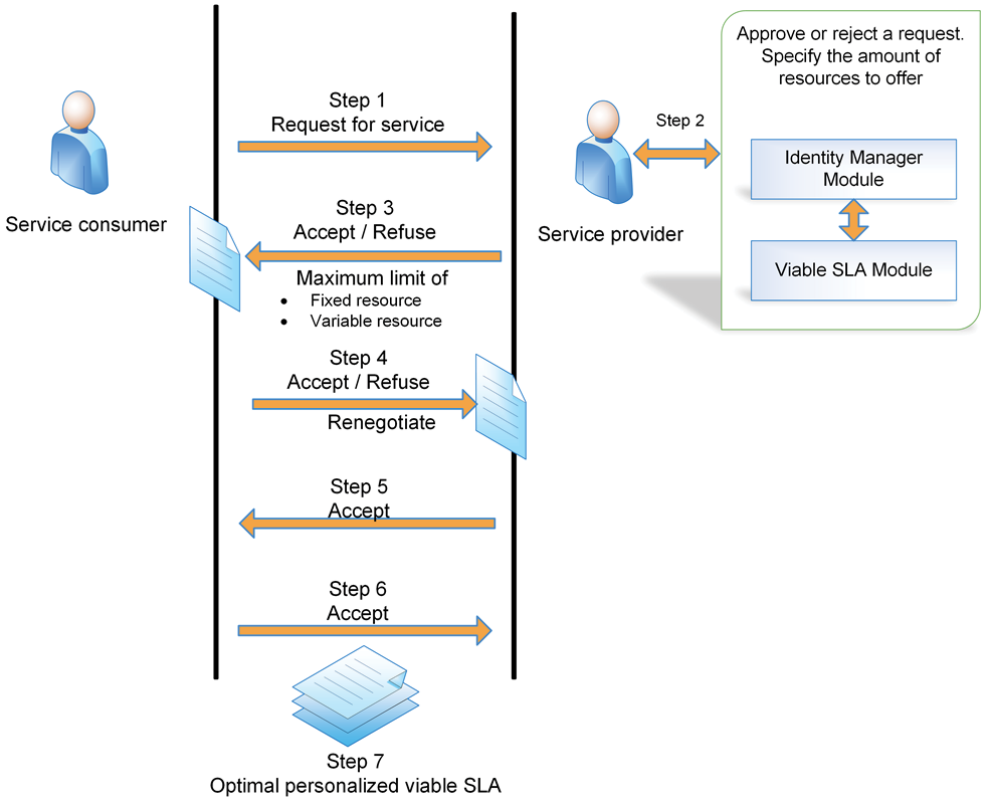


Figure 5.3: Steps of forming optimized personalized viable SLA

The second component, a proclamation of service requirements, is the process of defining all service requirements and identifying all service criteria and the quality of service. In the third component, grading the service requirement prioritizes each service requirement, i.e. so that some requirements and objectives have more importance than others (Fachrunnisa & Hussain 2013). All three components are combined to form a service requirements proposal

that contains all the service objectives, their respective weights, the quality of service, the amount of all required and marginal resources and the time frame. The requirement proposal is submitted by the consumer for approval by the provider.

Step 2. Service provider receives service request: The cloud provider receives a request from a consumer and forwards the request to the identity manager module (IMM). The IMM is responsible for the authentication and validation of requesting consumers with two possible conditions:

- The consumer is new and does not have a previous resource usage profile with the service provider.
- The consumer has a previous resource usage profile and has a reliability value.

In both cases, the request is forwarded to the viable SLA module (VSLAM), which decides whether to accept or reject the request and, if accepted, to then determine the amount of resources to be offered to accepted consumers using a fuzzy inference system (FIS).

Step 3. Decision on consumer's request and the amount of resources to offer: In this step, the decision from Step 2 is communicated to the consumer.

Step 4. Consumer's decision to accept, reject or renegotiate the offer: The consumer receives the decision of the provider. The consumer either accepts or rejects the decision and the amount of resources offered by the provider or renegotiates with the provider.

Steps 5 and 6. Decision and agreement from both sides: The parties negotiate and renegotiate with each other until they agree upon all the terms of the agreement.

Step 7. Viable SLA formation: When both parties agree upon the SLA requirements, they form an optimal viable SLA that offers a suitable amount of resources to the consumer based on the latter's reputation value.

5.4 Ascertaining the Reliability and the Transaction Trend of the Consumer

Cloud computing is flexible in nature, allowing consumers to request virtualized resources managed by a cloud provider over the internet at any time. The SLAs formed between consumers and providers operate under two payment terms:

- Consumers pay for the requested resource for the capacity they use; or
- Consumers pay a fixed amount for defined resources.

Small or medium service providers, which have limited resources, need to consider the reliability of the cloud consumer in relation to the amount of resources offered, even though the fixed payment method allows consumers to pay for a fixed level of resources.

The reliability of a consumer depends on their commitment to the promised resource usage in any transaction. If the consumer has a successful past record of commitment to the previously defined objectives, it has a high reliability value; likewise, the reverse is true – a poor commitment record reflects poor reliability. When an SLA is formed between a provider and consumer for required and marginal resources, the provider is bound to reserve that resource for the consumer. If the consumer agrees to the required or marginal resource usage and fails to honor this, the service provider should decrease the consumer's reliability value accordingly at the end of the transaction. The reliability of the consumer is represented using a reliability value, which ranges on a scale from 0 to 100. The reliability value 0 is the basic value and the reliability value 100 is the highest value. We assumed that the reliability of a consumer is not less than 0 and is not greater than 100.

With the successful completion of each transaction, the provider increases the reliability of the consumer by one point, and decreases reliability by one point with every unsuccessful transaction. It is considered that the reliability value of the consumer is time dependent. In order to maintain its reliability value, a consumer needs to conduct a transaction with its provider in a certain time frame, otherwise it will lose reliability points. The time frame in the proposed model is a one-year period. When a consumer does not perform any transaction with a provider in a period of one year then its reliability value decrease by some factor.

We set an arbitrary value of two points such that, a consumer loses two points from its reliability value for each year that it does not interact with the provider. Based on the reliability values of the consumer, existing consumers are classified into three classes: bronze, silver and gold. The ranges of reliability scores for each class of consumer are shown in Table 5.1.

Table 5.1: Categories of consumer and their reliability value

	Category	Reliability value
1	Bronze	0–40
2	Silver	41–70
3	Gold	71–100

Table 5.1 shows that bronze is a basic category with a reliability value that ranges from 1 to 40. Silver is the second category, with a reliability value ranging from 41 to 70, and gold represents the most trustworthy consumers, who have a reliability value of 71 to 100. When a consumer requests resources from a service provider, the IMM checks whether the consumer is a new consumer or has a previous resource usage profile. Using the usage profile, IMM determines the *transaction trend* value, which calculates the number of successful transactions divided by the total number of transactions. Mathematically, Equation 5.1 determines the transaction trend value:

$$\text{Transaction trend} = (T_{succ}/T_n) * 100$$

Equation 5.1

This value is then used to make a decision as to whether to accept or reject the service request.

5.5 Decision to Accept or Reject the Consumer's Request

In this section, a scenario is considered in which a cloud provider has limited resources to offer; therefore, the provider needs to consider both its required and marginal resource allocation. The provider takes a consumer's profile, which is a key factor for determination of service violation, into account. It is assumed that a consumer who has previously violated its commitment is more likely to violate its commitments again than consumers who have not previously violated

commitments. The provider therefore considers the previous profile of the consumer before determining whether to accept or reject the request.

5.5.1 Decision on the Existing Consumer's Request

The existing consumer who requests a service already has a previous resource usage profile. Information about existing consumers is stored in the profile repository of existing consumers in the IMM, as shown in Figure 5.1. VSLAM retrieves the consumer's history from the repository and calculates its transaction trend. The value of the transaction trend is compared with a threshold value. The threshold is the percentage of a success ratio determined by the service provider. If the transaction trend is equal to or greater than the threshold value, VSLAM accepts the request; otherwise, the request is rejected. For example, a requesting consumer has completed 10 transactions with a provider. In five transactions, the consumer has performed well with no service violation, while in the other five transactions it has performed poorly. The transaction trend in this case is 50%. The threshold value defined by the provider is 50%. VSLAM compares the value of the transaction trend with the threshold value, which in this case is equal. Therefore, VSLAM accepts the request. Similarly, VSLAM accepts all those requests whose transaction trend value is equal to or greater than 50%. This can be represented by the equation below.

$$(T_{succ}/T_n)* 100 \geq \text{threshold}$$

Equation 5.2

where T_n is the total number of transactions

and T_{succ} is the number of successful transactions.

For acceptance of a request, T_{succ} / T_n must be greater than or equal to the threshold value.

5.5.2 Decision on the New Consumer's Request

If the new consumer does not have a previous resource usage profile and or a transaction trend value, the provider considers the consumer's nearest neighbours to find the top-K nearest neighbours and examines their resource usage profiles (Hussain, Hussain & Hussain 2015a). The transaction trend for the new consumer is predicted from the top-K nearest neighbours' profile patterns. The prediction

result is highly dependent on the selection of nearest neighbours, because dissimilar users can decrease prediction accuracy. The top-K nearest neighbours are selected from a set of consumers with the maximum number of similarities to a new consumer. The process of selecting the top-K nearest neighbours and determining the reputation of a new consumer is comprised of the following steps.

5.5.2.1 Selection of Similar Neighbours

To find the nearest neighbours, the user-based collaborative filtering method (UCF) is used. UCF is widely used in user-based recommender systems. In this study we assume that a new consumer have nearest neighbours that have similar profile as a new consumer. The UCF is used to determine the likely transaction trend of a new consumer based on its nearest top-K resource usage profiles (Breese, Heckerman & Kadie 1998; Herlocker et al. 1999; Wang, De Vries & Reinders 2006). The criteria for the nearest neighbours' selection is to select all those neighbours that have similar profile patterns and who have previously requested similar services. The selection of nearest neighbours is represented by Equation 5.3.

$$NN = \frac{\sum_i^n [sim(r, n_i) * \text{rs}_i\{ n_i \in \text{N} \mid \text{rs} \in \text{R}\}]}{\boldsymbol{n}}$$

Equation 5.3

where *NN* are nearest neighbours,

r is the requesting consumer,

n_i is a i^{th} nearest neighbour from a set of all neighbours N,

rs_i is the i^{th} resources used by that neighbour, and

R is a set of all resources.

To measure the strength of the similarity between a new consumer and an existing consumer, the Pearson correlation coefficient (PCC) (Breese, Heckerman & Kadie 1998) is used, as represented in Equation 5.4.

$$Sim(r, r_a) = \frac{n(\sum r r_a) - (\sum r)(\sum r_a)}{\sqrt{[n\sum r^2 - (\sum r)^2][n\sum {r_a}^2 - (\sum r_a)^2]}}$$

Equation 5.4 (Breese, Heckerman & Kadie 1998)

The PCC compare a new consumer with existing consumers based on predefined criteria such as region, age, services etc. In the above equation, *r* is a requesting consumer and r_a is a set of nearest neighbours. $Sim(r,r_a)$ shows the strength of the similarity between requesting consumer *r* and all its nearest neighbours r_a. PCC is commonly used in various linear regression and recommender systems because of its prediction accuracy. PCC ranges from -1 to +1 where -1 represents a negative linear relationship, 0 represent no linear relationship and +1 represents a positive relationship. In the classical top-K algorithm, the nearest neighbours are selected from a set of neighbours that have PCC values from -1 to 1. However, prediction accuracy is significantly affected by the selection of negative PCC values because some neighbours have a limited number of similarities or even no similarities. To improve prediction accuracy, only those neighbours whose PCC value is positive are selected. The enhanced top-K nearest neighbours can be determined using Equation 5.5:

$$TKN_{enh}(r) = \{r_a | r_a \in T_k(r), siml\,(r_a, r) > 0, r_a \neq r\}$$

Equation 5.5

where TKN_{enh} is the enhanced top-K nearest neighbours,

r is a requesting new consumer, and

$T_k(r)siml\ (r_a,r)$ is the similarity between a consumer *r* and a set of traditional top-K nearest neighbours.

5.5.2.2 Transaction Trend of the New Consumer

To measure the transaction trend of new consumers, all top-K nearest neighbours, their transaction trends and the degree of similarity are considered that they have to the new consumer. The transaction trend is calculated by Equation 5.6:

$$T_{trend}(rn) = \frac{1}{n}\left[\sum_{i=1}^{n} TKN_{enh}\,(i)\{PCC(i) * T_{trend}(i)\}\right]$$

Equation 5.6

where $T_{trend}(rn)$ is a transaction trend for the requesting new consumer,

$TKN_{enh}(i)$ is the i^{th} enhanced top-K nearest neighbour,

$PCC(i)$ is the PCC value for i^{th} nearest neighbours, and

T_{trend} (i) is a transaction trend for i^{th} nearest neighbours that starts from 1 and moves to n where n is the total number of top-K nearest neighbours.

5.5.2.3 Decision to Approve or Reject the New Consumer's Request

VSLAM obtains the likely transaction trend for a new consumer from the above steps based on the transaction trends of the consumer's nearest neighbours. The transaction trend for a new consumer is compared with a threshold value. A request is accepted if the value of the transaction trend is equal to or greater than the threshold value, otherwise it is rejected. The provider sets the value of the threshold. Equation 5.7 represents the approval of a request:

$$\text{accept request} = T_{trend}(rn) \geq \text{threshold}$$

Equation 5.7

where $T_{trend}(rn)$ is the determined transaction trend value of a new consumer.

To accept a request, $T_{trend}(rn)$ must be greater than or equal to the threshold value. By accepting a request, it means that a provider accept consumer's request for further processing to offer the amount of resources otherwise if the value of T_{trend} is less than the threshold value the request is rejected without further processing. It should be noted that even if a request to assign resources is accepted, it does not guarantee that the user will be given or allocated the amount of resources that is requested. This depends on the resources the provider has at that given period of time, how many users are requesting those resources, and the time period for which they are requesting them. This is explained in the next section.

5.6 Determining the Amount of Resources to Offer

To determine the amount of resources to offer, VSLAM considers the reliability value of the consumer and the duration of the contract. As discussed earlier, consumer reliability is the successful commitment of a consumer in previous transactions that is updated by the provider at the end of a transaction. The second

factor for resource allocation is contract duration, which is the time period for which resources are reserved for a consumer. When only one consumer requests resources, the provider offers the requested or available resources to the consumer. However, if there is more than one consumer, the provider categorizes consumers into three levels of suitability – low, medium and high – based on reliability value and the duration of the consumers' contracts. To maximize its profit, the provider gives preference to requests from highly reliable consumers who reserve marginal resources for a minimum period. FIS is used to determine the suitability of consumers and the amount of resources offered to each suitability level. FIS uses two inputs, the suitability of consumers based on reliability and the contract duration, and generates the amount of resources to offer. The FIS for determining consumer suitability and resource allocation is described in the following subsections.

5.6.1 Fuzzy Inference System for Determining the Value of the Consumer Suitability and the Amount of Resources to Offer

The notion of fuzzy set was proposed by Zadeh (Zadeh 1965). In contrast to a crisp value for classifying an object in two ways – as a member or non-member of a set – a fuzzy set allows for the differentiation of objects according to the degree of membership value, which ranges from zero to one. Zadeh proposes a framework based on his fuzzy set theory (Zadeh 1973) by defining linguistic variables and characterizing simple and complex relations using a fuzzy algorithm. Mamdani and Assilian (Mamdani & Assilian 1975) outline an inference system using fuzzy set theory to control a steam engine and boiler. The Mamdani approach is widely used in decision support systems.

To determine the consumer's suitability and the amount of resources to be offered, a FIS using the Mamdani approach is utilized. Fuzzy rules are applied at two levels. The suitability value of resource-requesting consumers is determined at the first level, and the amount of resources to be offered is determined at the second level. The structure of the fuzzy rules is shown in Figure 5.4. The inputs at each level are fuzzified through fuzzy set and by using fuzzy rules. The output value, which is the suitability value of the consumer and the amount of resources to offer, is then obtained. The output is defuzzified using the centroid method to

obtain a crisp value. The working of the FIS for the decision-making module is represented by a flow chart in Figure 5.5.

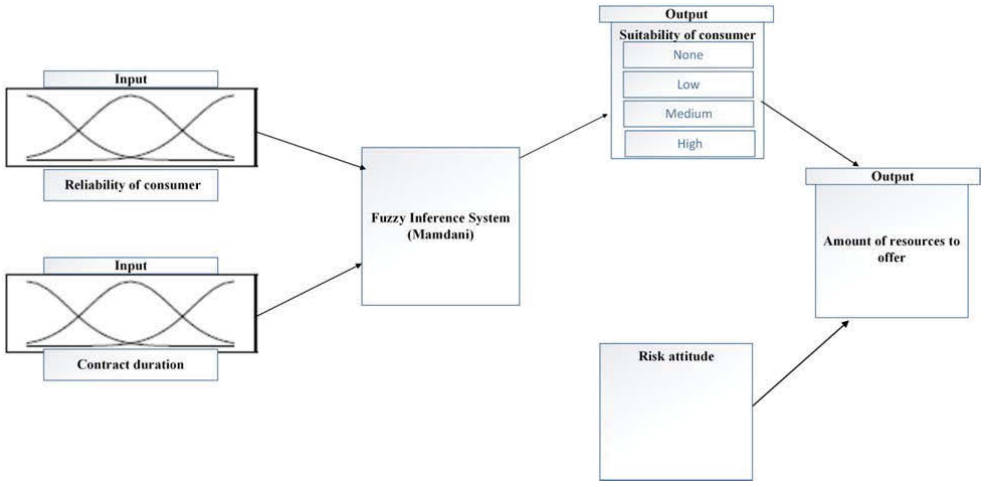


Figure 5.4: FIS for resource allocation decision

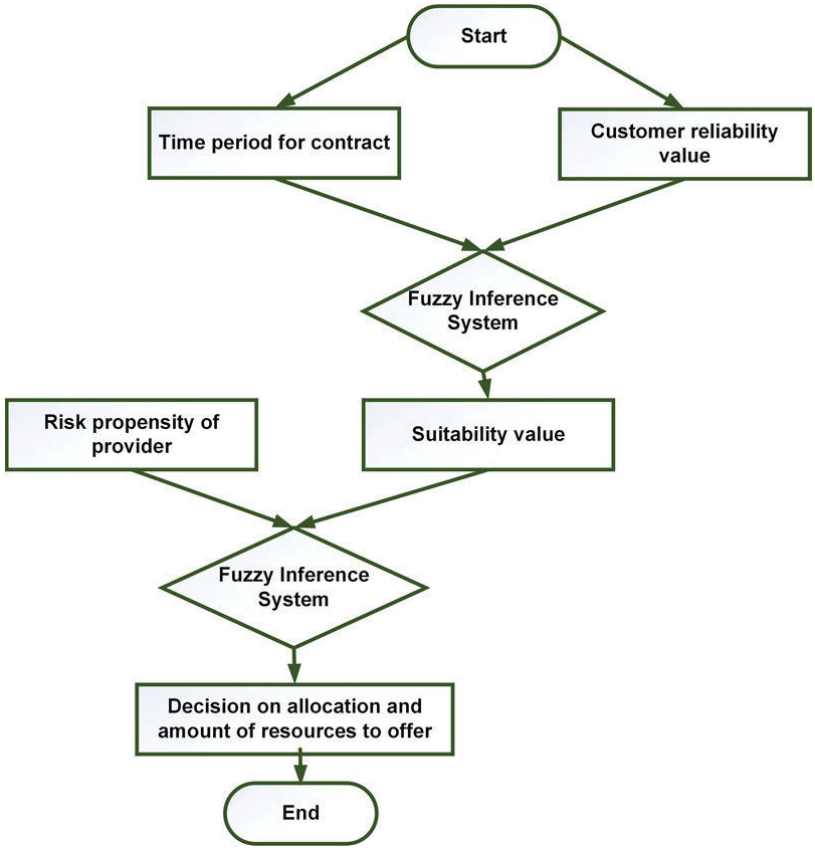


Figure 5.5: Flowchart for the FIS decision-making module

5.6.2 Fuzzy Inference System for Ascertaining the Suitability of Consumers

5.6.2.1 Fuzzy Set for the Input "Reliability of Consumers"

It is very important for a service provider to ascertain the reliability value of a consumer when making a decision about the amount of resources to be offered. Consumers with a high reliability value have more opportunity to reserve request resources compare to consumers with a low reliability value if the time period they request is small. The reliability of a consumer is defined over a universal set (*US*) as presented in Equation 5.8:

$$\boldsymbol{US = \{rl \,|0 \leq rl \leq 100; rl \in R\}}$$

Equation 5.8

where *rl* is the reliability of a consumer in a range of 0 to 100.

The universal set of users is divided into three predicates: gold, silver and bronze. The numerical range for a reliability value is as shown in Table 5.1 and the membership function is represented in Figure 5.6.

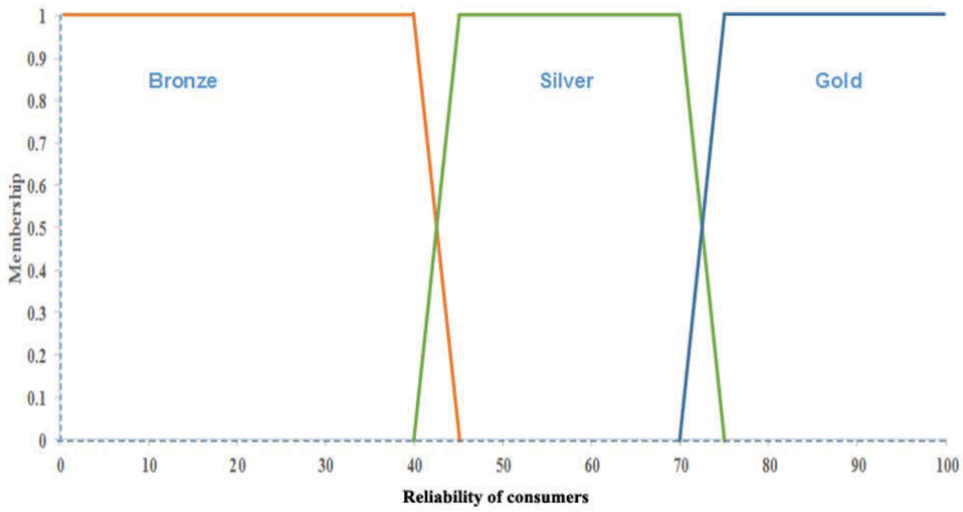


Figure 5.6: Membership function for the input value "Reliability of consumers

5.6.2.2 Fuzzy Set for the Input "Contract Duration"

The second input for FIS is the contract duration. Contract duration is categorized into three types: short, medium, and long. The numerical range for contract duration is as follows: a short contract is between 0 to 4.5 months; a medium

contract is between 4 to 8.5 months; and a long contract is between 8 to 12 months or more. The membership function for the input "contract duration" is represented in Figure 5.7.

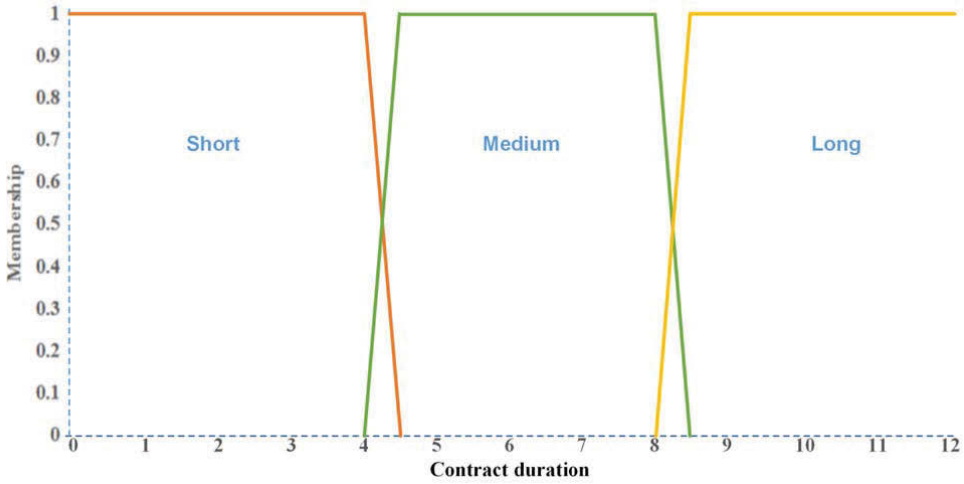


Figure 5.7: Membership function for the input value "contract duration"

5.6.2.3 Fuzzy Set for the Output "Suitability"

The first output of the FIS is the suitability value. The suitability value represents to the provider the suitability of the consumer for the assigning of resources by considering the user's reliability value and the duration for which it is requesting resources. The suitability value is classified into four classes in ascending order: *None* < *Low* < *Medium* < *High*. The numerical value for these classes ranges from 0 to 100. A consumer with a numeric value of 0 has no suitability and a consumer with a numeric value of 1 has high suitability. The membership function for the output "suitability" and the numerical range for each class of suitability are represented in Figure 5.8.

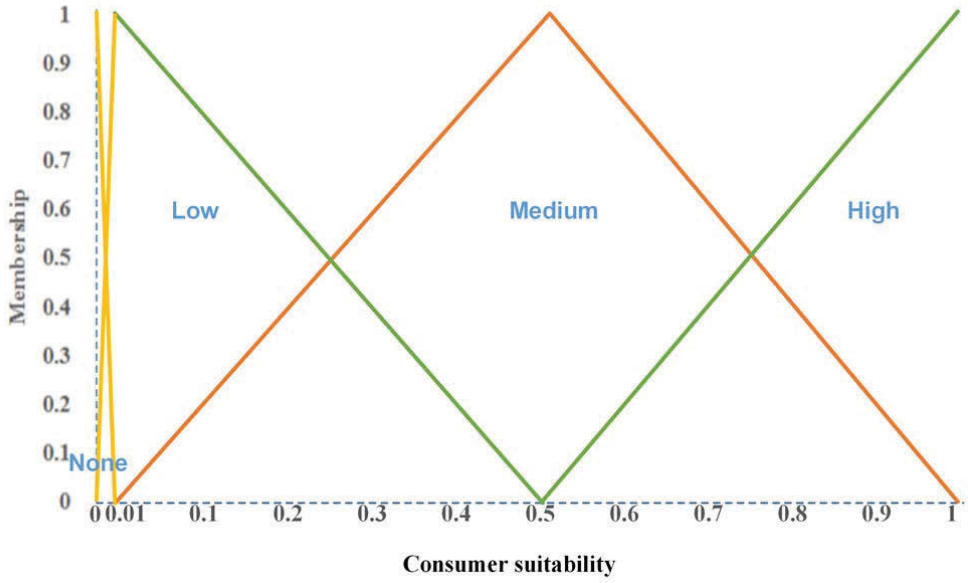


Figure 5.8: Membership function for the output "suitability of consumer"

5.6.3 Fuzzy Inference Rules for Determination of Suitability

A Mamdani-type fuzzy inference system is used for the detection of suitability in consumers. There are nine FIS rules for determining the suitability value of a consumer, using an IF-THEN structure and AND operator. These rules are represented in Table 5.2.

Table 5.2: Fuzzy rules for determining suitability value and resource allocation

	Reliability		Contract duration		Suitability
If	bronze	and	short	then	low
If	silver	and	short	then	medium
If	gold	and	short	then	high
If	bronze	and	medium	then	none
If	silver	and	medium	then	medium
If	gold	and	medium	then	medium
If	bronze	and	long	then	none
If	silver	and	long	then	low
If	gold	and	long	then	medium

Once the suitability of a consumer has been determined, requesting consumers are ranked in ascending order according to their suitability for being assigned resources. The next step is to determine how much to allocate according to the suitability value of consumer. This process is described in the following section.

5.7 Using Fuzzy Inference System to Ascertain the Amount of Resources to Offer the Requesting Consumers

The FIS is used for assisting providers to make a decision concerning the amount of resources to offer the requesting consumers. This takes two inputs, as shown in Figure 5.4: the suitability of requesting consumers and the risk propensity of the provider. The input risk propensity of the provider represents its attitude towards taking risks, which plays an important role in deciding the amount of resources to offer in relation to the amount requested. The risk propensity of the service provider is represented across three fuzzy sets, which in an ascending order are *Risk Averse* < *Risk Neutral* < *Risk Taking*. This shows that a provider with a risk propensity value of *Risk Averse* needs consumers who have high suitability for resource allocation compared to providers who have a risk propensity value of *Risk Taking*. The range over which the risk propensity of the consumer is drawn is 1 to 5 and the membership function is as shown in Figure 5.9.

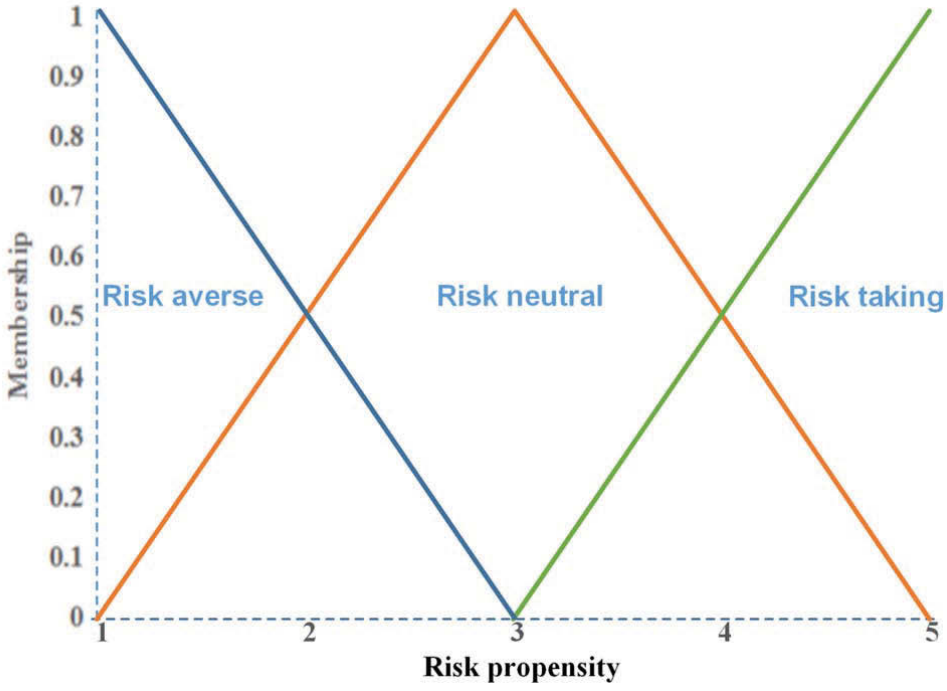


Figure 5.9: Risk propensity of a provider

It can be seen from Figures 5.8 and 5.9 that the suitability and risk attitude of the requesting consumers and providers respectively can have an overlap of more than one membership function; for example, in the case of risk propensity, risk

averse = 0.4 and risk neutral = 0.6. In such a case, the maximum risk propensity level of a consumer is selected according to their increasing level (i.e. RA < RN < RT), on which the suitability value of a consumer requiring resources is ascertained. This is achieved by fuzzy rules, which represent the required suitability value (RSV) of consumers in fuzzy membership functions so that the provider can allocate resources according to its maximum risk attitude (MRA), as shown in Table 5.3. In Table 5.3 and 5.4 the notation H represents high, M represents medium s, L represents low and N represents none.

Table 5.3: Fuzzy rules that represent the required suitability value of consumers depending upon the maximum risk attitude of providers

	MRA		RSV
If	RA=1	then	H = 1
If	RN = 0.1	then	H = 0.9
If	RN = 0.2	then	H = 0.7
If	RN = 0.3	then	H = 0.6
If	RN = 0.4	then	H = 0.5
If	RN = 0.5	then	H = 0.4
If	RN = 0.6	then	H = 0.3
If	RN = 0.7	then	H = 0.2
If	RN = 0.8	then	H = 0.3
If	RN = 0.9	then	H = 0.1
If	RN = 1	then	M = 1
If	RT = 0.1	then	M = 0.5
If	RT = 0.2	then	M = 0.7
If	RT = 0.3	then	M = 0.6
If	RT = 0.4	then	M = 0.4
If	RT = 0.5	then	M = 0.3
If	RT = 0.6	then	L = 1
If	RT = 0.7	then	L = 0.8
If	RT = 0.8	then	L = 0.5
If	RT = 0.9	then	L = 0.3
If	RT = 1	then	L = 0

Once the required suitability of a requesting consumer (RSV) has been determined according to the maximum risk attitude of the provider (MRA), the amount of resources to be offered or allocated (ALLOC) can be determined by the fuzzy rules defined in Table 5.4. The fuzzy predicates over which ALLOC is determined are NONE (N), MARGINAL (M), PARTIAL (P) and FULL (F), as shown in Figure 5.10.

Table 5.4: Fuzzy rules for determining the resources to offer consumers

	CS		ALLOC		N_ALLOC		ALLOC
If	N	then	N				
If	L	then	M	if	0	else	N
If	M	then	P	if	0	else	N
If	H	then	F	if	0	else	N

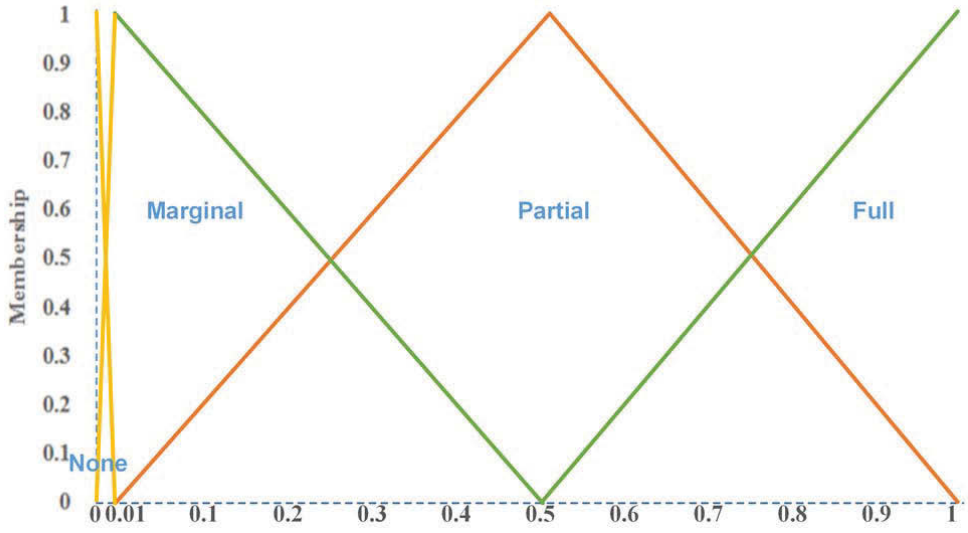


Figure 5.10: Membership function for determining the output "allocation to the consumer"

The abbreviation CS in Table 5.4 represents the current suitability value of the requesting consumer, ALLOC represents the degree of membership from the resources to be allocated, and N_ALLOC is a variable which has a value of 0 or 1. It should be noted that the suitability of the requesting consumer can span two membership functions; for example, Low = 0.6 and Medium = 0.4. It is possible that based on the maximum risk attitude of the provider, one level of consumer suitability may be acceptable while the other is not. For example, if a requesting consumer has a suitability value represented in fuzzy membership function as L = 0.4 and M = 0.6 and the maximum risk attitude of the consumer as RT = 0.4, then it can be seen from Table 5.10 that the required RSV for the acceptance of the resource request is M = 0.4. However, the current suitability (CS) values of the consumer indicate that only one membership function (M = 0.6) adheres to the provider's requirements, hence only that one needs to be considered when a decision is made on how much to allocate to the user. In other words, the membership value of L = 0.4 is omitted when the amount of resources to offer to the consumer is determined. This is represented by the variable N_ALLOC, whose value is determined by the rules shown in Table 5.5. This variable determines whether the requesting consumer can be allocated resources based on the *current* suitability value of the consumer and the maximum risk attitude of the provider and, if resources can be allocated, then how much.

Table 5.5: Fuzzy rules for determining the value of the variable N_ALLOC according to the MRA and CS of the provider and consumer respectively

	MRA		CS <		N_ALLOC		N_ALLOC
If	RA = 1	and	H = 1	then	1	else	0
If	RN = 0.1	and	H = 0.9	then	1	else	0
If	RN = 0.2	and	H = 0.7	then	1	else	0
If	RN = 0.3	and	H = 0.6	then	1	else	0
If	RN = 0.4	and	H = 0.5	then	1	else	0
If	RN = 0.5	and	H = 0.4	then	1	else	0
If	RN = 0.6	and	H = 0.3	then	1	else	0
If	RN = 0.7	and	H = 0.2	then	1	else	0
If	RN = 0.8	and	H = 0.3	then	1	else	0
If	RN = 0.9	and	H = 0.1	then	1	else	0
If	RN = 1	and	M = 1	then	1	else	0
If	RT = 0.1	and	M = 0.5	then	1	else	0
If	RT = 0.2	and	M = 0.7	then	1	else	0
If	RT = 0.3	and	M = 0.6	then	1	else	0
If	RT = 0.4	and	M = 0.4	then	1	else	0
If	RT = 0.5	and	M = 0.3	then	1	else	0
If	RT = 0.6	and	L = 1	then	1	else	0
If	RT = 0.7	and	L = 0.8	then	1	else	0
If	RT = 0.8	and	L = 0.5	then	1	else	0
If	RT = 0.9	and	L = 0.3	then	1	else	0
If	RT = 1	and	L = 0	then	1	else	0

5.8 Validation of OPV-SLA Framework to Determine the Decision on the Consumer's Request

To validate the proposed VSLAM, a use case of three consumers that request resources from a provider is considered. The intention of this use case is to show how the VSLAM framework helps the provider to first determine which users to offer resources to according to their transaction trend value and then to assist with the decision on the resource amount to be offered to those consumers. A time series dataset is considered from (Zhang, Zheng & Lyu 2011b) comprising of 142 users using 4532 web services for 64 time intervals. For this study, a throughput is considered as one of SLO. Let us consider that the users requesting resources (throughput) are those with IDs 106, 34 and 5. Two of the requesting users (106 and 34) have a past interaction history with the service provider while the other

(5) does not. The first step in the VSLAM is to ascertain the transaction trend of the consumers. A mean absolute deviation (MAD) and root mean square error (RMSE) are used to measure prediction accuracy. In the next subsection, the process of determining the transaction trend for each user is discussed.

5.8.1 Measuring Prediction Accuracy

MAD and RMSE are used as a benchmark for measuring prediction accuracy. MAD measures and captures how much the predicted result varies from the mean value. The absolute value is used to obtain a positive value. MAD is defined by Equation 5.9.

$$MAD = \frac{1}{n}\sum_{i=1}^{n}|x_i - \bar{x}|$$

Equation 5.9

where n is the total number of observed values,

$\bar{x}$ is a mean of observed value, and

x_i is the i^{th} individual value.

RMSE is a quadratic scoring rule that calculates the average magnitude of the error. In RMSE, the difference between the actual and predicted values is squared, averaged and then a square root is applied to the average. This gives higher weights for large errors as a result of the squaring. RMSE is defined by Equation 5.10.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \hat{x}_i)^2}{n}}$$

Equation 5.10

where n is the total number of observed values,

x_i is the observed throughput value for time interval from 1 to n, and

$\hat{x}_i$ is the predicted throughput.

5.8.2 Determining the Transaction Trends of Consumers who have an Interaction History

The transaction trends for consumer numbers 106 and 34 are determined by Equation 5.1, in which the number of successful transactions is divided by the total number of transactions. Let assume for the SLO throughput that the number of past transactions of each user with the provider is as shown in Table 5.6; thus, the transaction trend values for consumers 106 and 34 are 55% and 10% respectively.

Table 5.6: Transaction trends of requesting consumers

#	Consumer Number	No. of successful transactions	No. of violated transactions	Transaction trend ($(T_{succ}/T_n)*100$)
1	106	5	4	55%
2	34	2	18	10%

5.8.3 Determining Transaction Trends for New Consumers

To determine the transaction trend for a new consumer (consumer number 5), the system finds the top-K nearest neighbours and then considers the weighted average of transaction trends from all top-K nearest neighbours.The process comprises the following steps:

Step 1: Selection of the top-K nearest neighbours

User-based collaborative filtering method is used and selects all those neighbours that have similar properties and who have previously used the same services. The K nearest neighbour is a widely used algorithm for classification or regression. It is a non-parametric algorithm that make decisions based on a complete training dataset. Top-K is the number of consumers that have maximum similarity to a new consumer. As mentioned earlier, prediction accuracy depends on the selection of nearest neighbours. The traditional top-K algorithm selects neighbours with both positive and negative PCC values. An improved top-K nearest neighbour algorithm is used that selects only those nearest neighbours whose PCC value is positive. In the current scenerio, consumer 5 is requesting a service that has the service ID 1942 and the top-K neighbours are those consumers that have maximum similarity to consumer 5 and have all previously used service ID 1942.

Step 2: Limit of the top-K value

Once the set of nearest neighbours has been obtained, the next issue is the limit of top-K neighbours. To find out the impact of top-K in prediction accuracy, the study analysed neighbours with different top-K values. It starts with two neighbours and increases to 20 neighbours. The prediction accuracy for each nearest neighbour is compared using MAD and RMSE, as represented in Table 5.7.

Table 5.7: RMSE and MAD for top-K nearest neighbours

TOP-K	RMSE	MAD
2	0.820856	0.607037
3	0.802337	0.635128
4	0.793873	0.641566
5	0.860021	0.690758
6	0.694338	0.579813
7	0.724104	0.607998
8	0.686049	0.583164
9	0.668933	0.550948
10	0.707908	0.628897
11	0.735434	0.657866
12	0.764868	0.681089
13	0.759547	0.678688
14	0.739035	0.655641
15	0.773842	0.688224
16	0.777221	0.690981
17	0.765566	0.678838
18	0.763594	0.676608
19	0.826001	0.735019
20	0.815077	0.722747

Figure 5.11 shows the RMSE of the top-K nearest neighbours and Fig. 5.12 represents the MAD for top-K nearest neighbours.

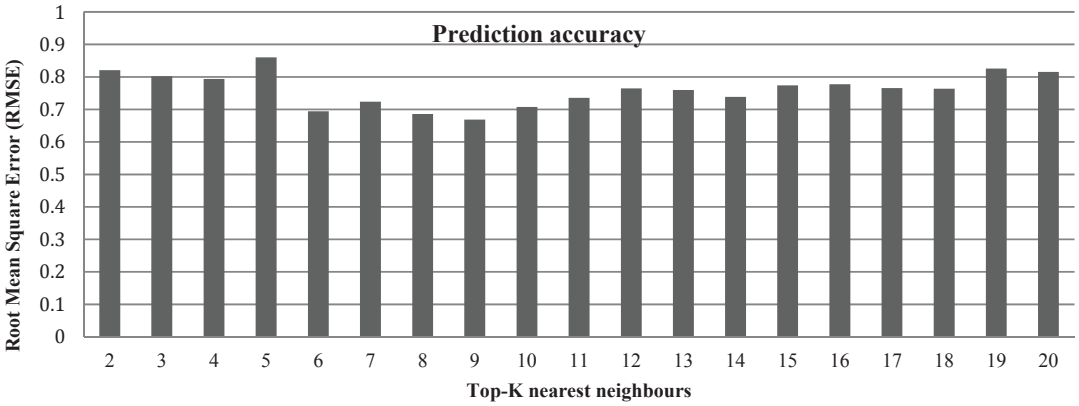


Figure 5.11: RMSE of top-K nearest neighbours

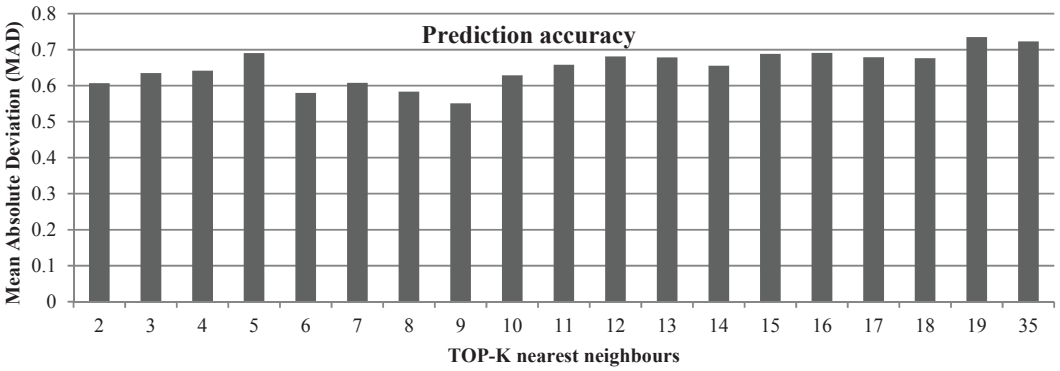


Figure 5.12: MAD of top-K nearest neighbours

It can be seen from Table 5.7 and Figures 5.11 and 5.12 that the top-K nearest neighbour with a value of 9 gives the optimal result with RMSE 0.668933 and MAD 0.550948. When the number of neighbours increases or decreases from 9, fewer similar users are included, which decreases prediction accuracy. For the rest of the experiment, the value of top-K as 9 is considered.

Step 3: Transaction trend of a new consumer

The throughput (SLO) values of top-K nearest neighbours who have used service ID 1942 for 64 time intervals is represented in Figure 5.13. The x axis in Figure 5.13 represent time interval which spans over 64 intervals and the y axis represent the throughput value. As discussed earlier for an optimal result we need to select nine neighbours.

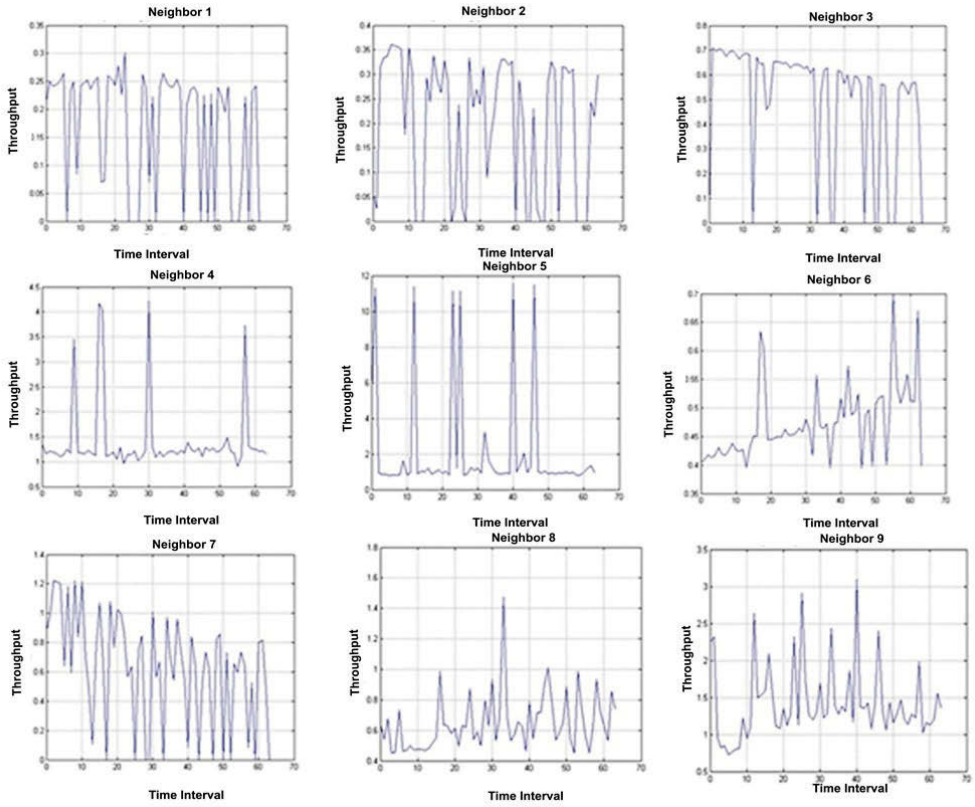


Figure 5.13: Throughput values of top-K nearest neighbours for 64 time intervals

Due to the lack of data, random data is generated for the transaction trend of existing consumers and merged with existing data to calculate a transaction trend value for a new consumer. The exisiting dataset does not have the data for the number of successful transactions and number of violated transactions, therefore a random data is generated for the status of the transaction and linked it with the profile of consumer. The top-K nearest neighbours of consumer 5 are represented in Table 5.8, which shows their PCC values, the number of successful and violated transactions and the transaction trend.

Table 5.8: K nearest neighbours with PCC values, number of successful and violated transactions and the transaction trend

#	**Nearest neighbours** Consumer #	**PCC**	**No. of successful transactions**	**No. of violated transactions**	**Transaction trend**
1	106	0.9999779	5	4	55.00%
2	94	0.9999685	5	1	83.33%
3	34	0.9999406	2	18	10.00%
4	82	0.9999311	4	2	66.67%
5	63	0.9997260	1	27	3.57%
6	23	0.9992655	1	8	11.00%
7	92	0.9992068	10	4	71.42%
8	81	0.9992068	7	4	63.64%
9	15	0.9992068	0	1	0.00%

Using Equation 5.6 and taking a weighted average, the transaction trend value for a new consumer is:

$$T_{trend} = 40.50\%$$

From Tables 5.6 and 5.8, the obtained value of a transaction trend for existing consumers 106 and 34 and for a new consumer 5 are 55%, 10% and 41% respectively. The threshold value set by the provider is 40%. The value of the transaction trend is compared with the threshold value. In this scenario, the transaction trend of consumer 106 and consumer 5 is greater than the threshold value, while consumer 34 has a lower transaction trend value than the threshold value. The system recommends the service provider should consider the requests from consumers 106 and 5 further and reject the request from consumer 34 due to the lesser value of its transaction trend. The next step is for the provider to determine the amount of resources to allocate to consumers 105 and 5. This is ascertained by using the FIS, as shown in the next section.

5.9 Ascertaining the Suitability of Requesting Consumers Using FIS

To ascertain the suitability of requesting consumers, the first step in the FIS is to transform the transaction trend and the contract duration values into linguistic terms. Using the fuzzy membership functions, the reliability value for user 106 is quantified as a membership value of S = 1, while for user 5 it is quantified as a membership value of B = 0.8 and S = 0.2. This level of quantification is shown in Figures 5.14 and 5.15.

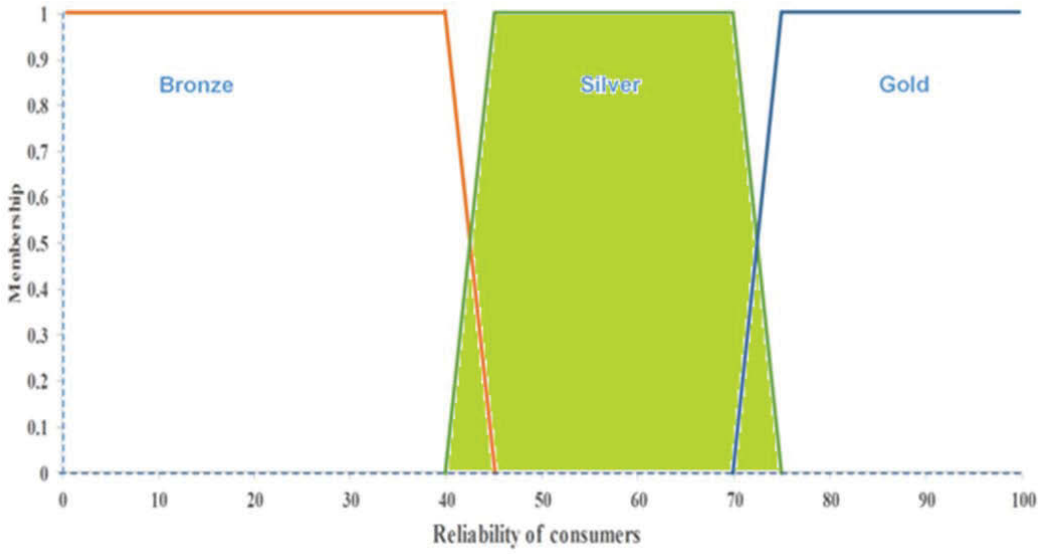


Figure 5.14: Reliability in terms of fuzzy predicates of consumer 106

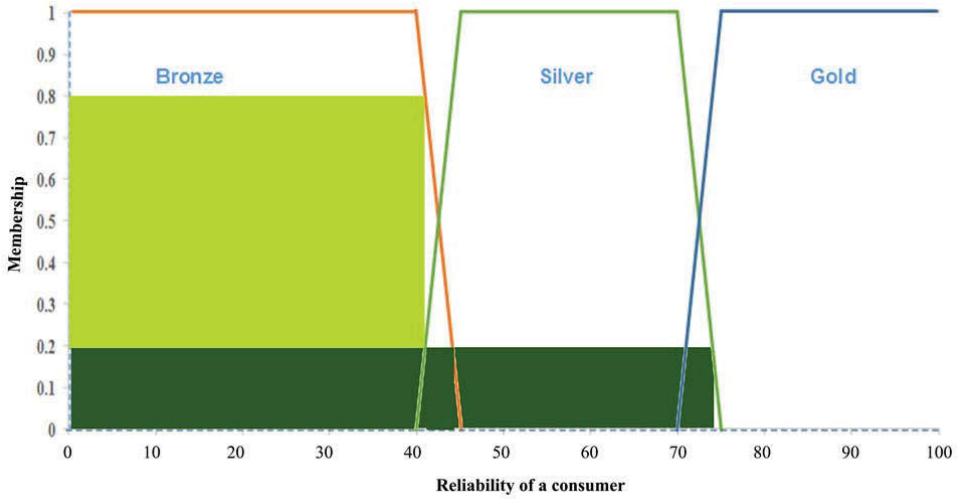


Figure 5.15: Reliability in terms of fuzzy predicates of consumer 5

Let us assume that the time period for which each user is requesting resources is 4.2 months. When translated into fuzzy linguistic terms using the membership function defined in Section 5.6.2.2, this equates to the predicates of S = 0.6 and M = 0.4, as shown in Figure 5.16.

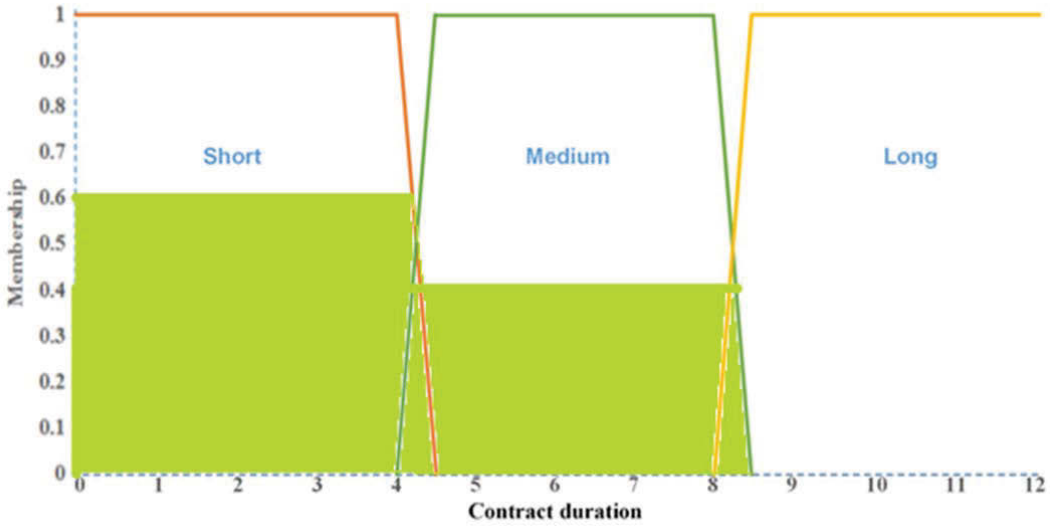


Figure 5.16: Contract duration in terms of fuzzy predicates

The suitability of each requesting consumer is ascertained using the fuzzy rules defined in Table 5.2. As mentioned in Section 5.6.2.3, the suitability value of the consumer represents the likelihood that the provider will allocate resources. The various rules mentioned in Table 5.2 are fired for each user and the suitability values for user 106 and user 5 in terms of fuzzy predicates are M = 1 and N = 0.4, M = 0.4 and L = 0.6 respectively, as shown in Figures 5.17 and 5.18.

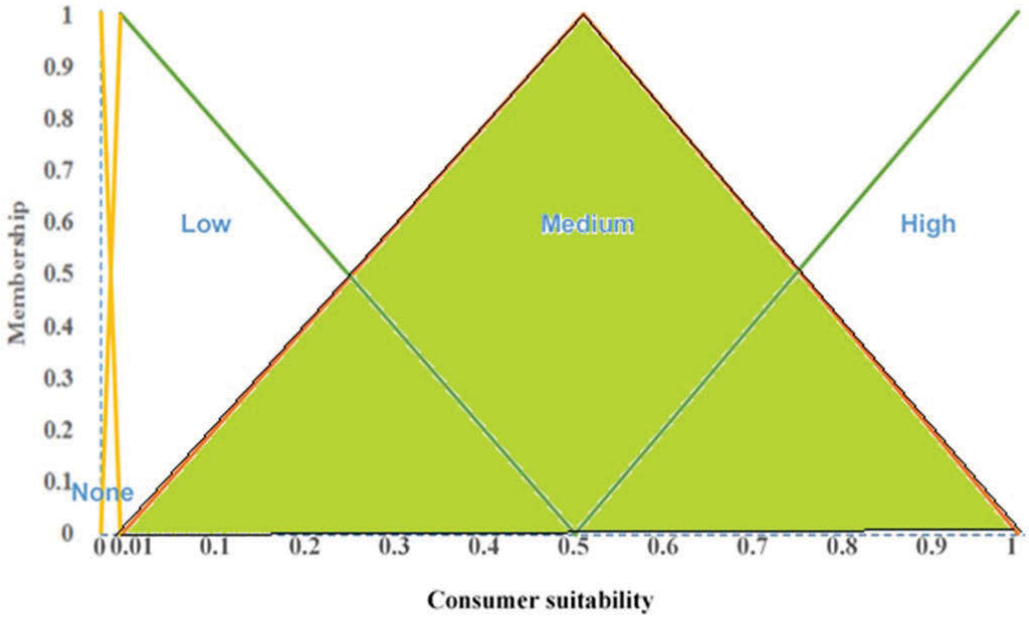


Figure 5.17: Suitability in terms of fuzzy predicates of consumer 106

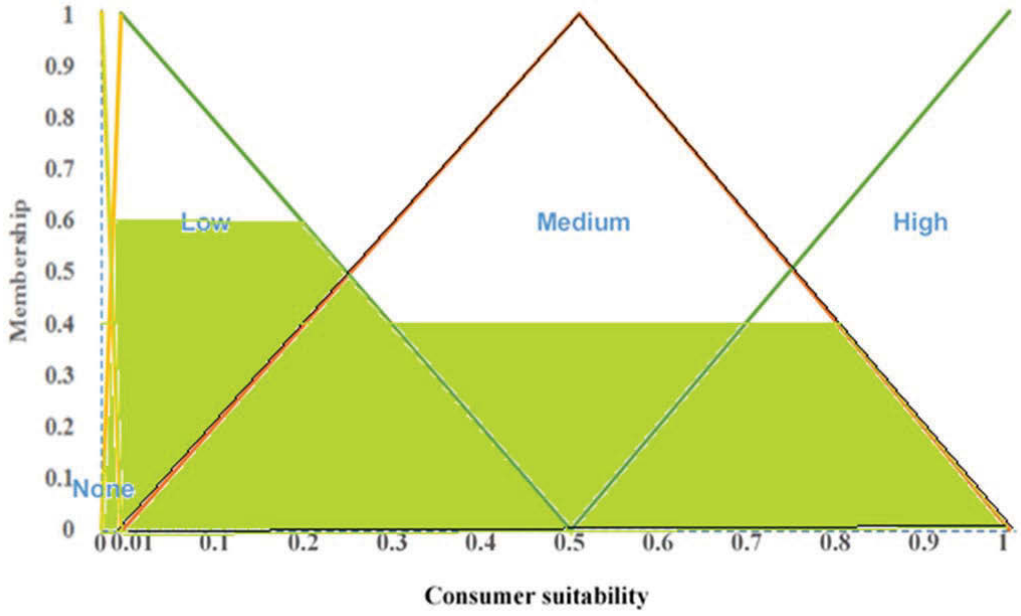


Figure 5.18: Suitability in terms of fuzzy predicates of consumer 5

Once the level of suitability of each requesting consumer has been ascertained, the next step is to determine the amount of resources to grant to the consumer. This is done by using the risk propensity of the service provider, as shown in the next section.

5.10 Ascertaining the Amount of Resources to Grant Each Requesting Consumer

Let us consider that the risk propensity of the service provider on a scale of 1 to 5 is 4. As seen in Figure 5.19, this translates to fuzzy predicates of RN = 0.5 and RT = 0.5.

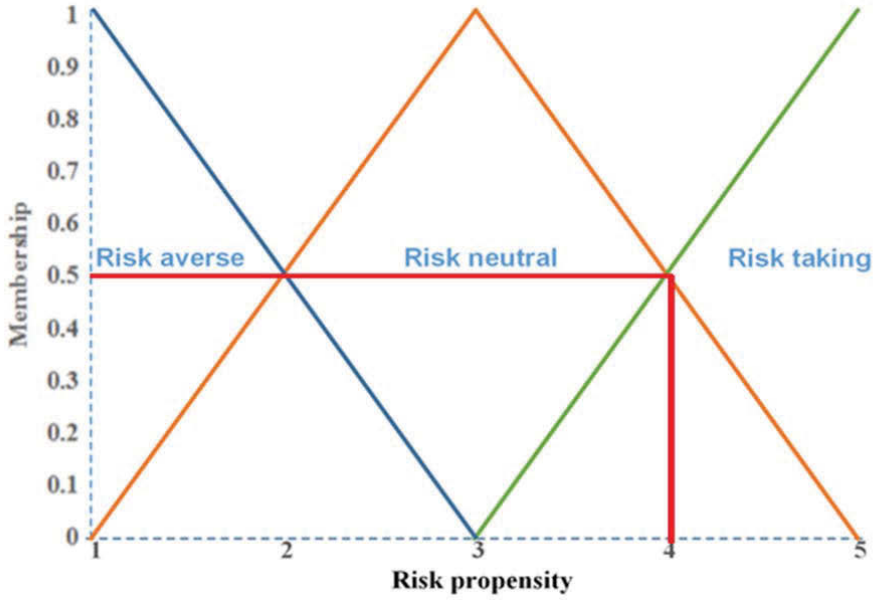


Figure 5.19: Fuzzy predicates for the risk propensity of the service provider

The MRA of the service provider is RT = 0.5 and the RSV from Table 5.3 is M = 0.3. This value shows that according to the MRA of the service provider, service users are required to have a suitability value of M = 0.3 or higher for the request to be approved. It can be seen from Figures 5.17 and 5.18 that the total suitability of consumer 106 is higher than the required RSV, but in the case of consumer 5 it is only partially higher than the required limit. To ascertain the resources to offer to the requesting consumers, we therefore fire the fuzzy rules defined in Tables 5.4 and 5.5 to obtain the representation shown in Figures 5.20 and 5.21 for users 106 and 5 respectively. The fuzzy predicate *Partial* for user 106 fires with strength of 1, while the fuzzy predicates of *None* and *Partial* for user 5 fire with strengths of 1 and 0.4 respectively.

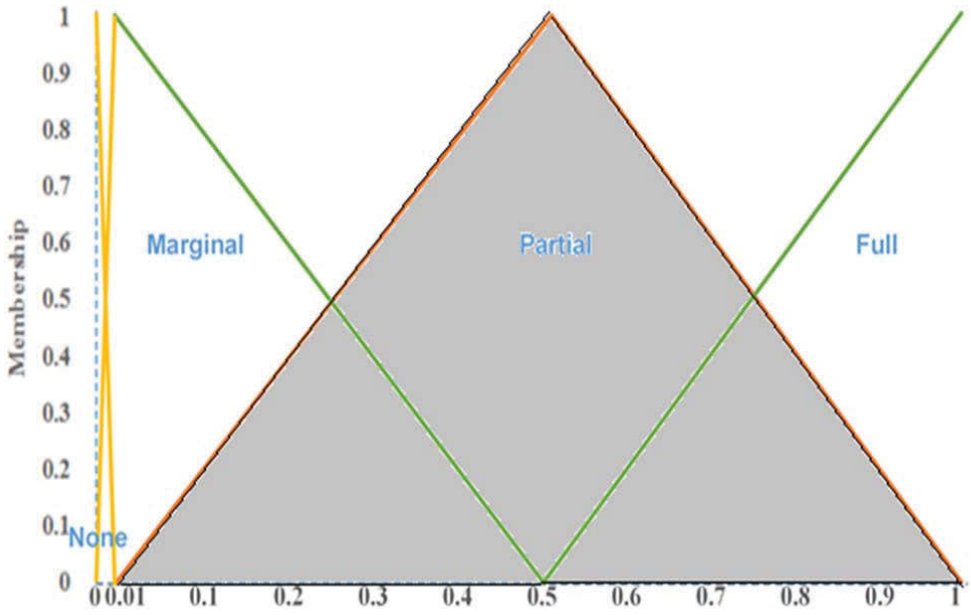


Figure 5.20: Allocation in terms of fuzzy predicates of consumer 106

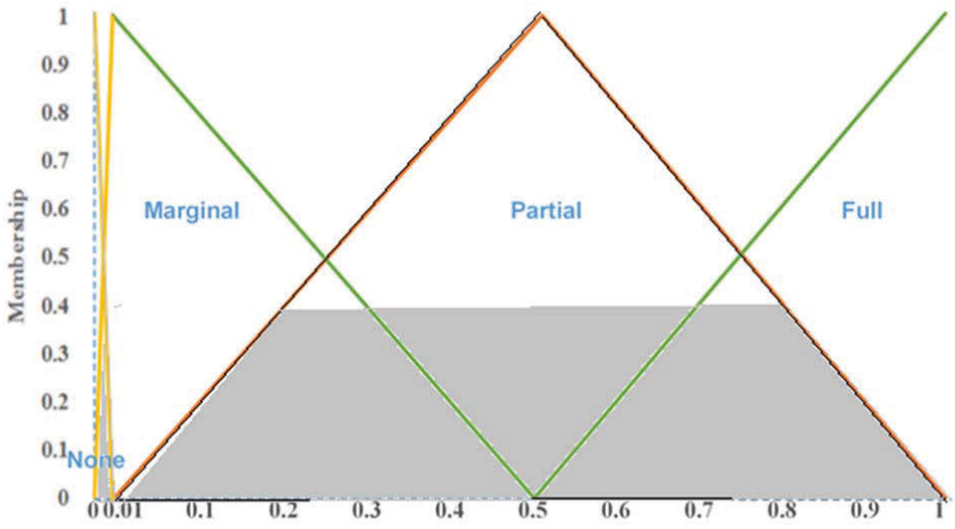


Figure 5.21: Allocation in terms of fuzzy predicates of consumer 106

Defuzzifying the shaded regions of the figures, the crisp values of 51% and 50% are obtained for users 106 and 5 respectively. These values show that by considering the suitability value of user 106 and the provider's risk propensity, it can approve 51% of the resources that it is asking for. Similarly, by considering

user 5's suitability value and the provider's risk propensity, the provider can approve 50% of the resources requested.

5.11 Conclusion

SLA monitoring is a key challenge in business today, particularly in the cloud business environment. Due to the elastic nature of the cloud, a consumer can request service provision anytime and anywhere. It is therefore very important for cloud providers – especially small and medium cloud providers with limited resources – to manage their resources efficiently and form viable SLAs. The VSLAM helps service providers make decisions about whether to accept requests from consumers and whether to offer SLAs. If an SLA is to be offered, the module assists service providers in determining the optimal parameters for making their offer. A consumer's profile is used to select top-K nearest neighbours for making decisions about service formation, and it further used FIS to determine the amount of resources to offer. From the evaluation results it is observed that by forming viable SLAs, a service provider is able to manage its reserved resources in the most productive way.

In the next chapter, different prediction algorithms are evaluated at different time intervals by using various patterns of real cloud dataset, and an optimal prediction method that would help the service provider with accurate decision-making to avoid SLA violations is chosen.

6 Analysis of QoS Prediction Approaches in the Post-interaction Time-phase

6.1 Introduction

In cloud computing, service level agreements (SLAs) are legal agreements between a service provider and consumer that contains a list of obligations and commitments that need to be satisfied by either party during the transaction. From a service provider's perspective, a violation of defined commitment would lead to monetary and reputation penalties. Therefore, effective prediction methods are needed by which the service provider can predict instances of deviations in the QoS that will be delivered and appropriately take steps so that quality promised in the SLAs is maintained. Quality of Service (QoS) is a key factor to measure service level objectives (SLO) in an SLA. It is a critical parameter that combines with other QoS parameters to form a performance metric.

The focus of this chapter is on future QoS determination that is done using QoS prediction techniques. There are many prediction techniques that can be used for future QoS determination; however, each technique will have different results. The choice of a prediction algorithm plays an important role in service providers managing SLA and avoiding SLA violations. A provider can reduce their risk by following more formal quantitative prediction methods (Waters 2011). Therefore, it is necessary to determine the appropriate QoS prediction method to be used based on its prediction accuracy at different time intervals.

The purpose of this chapter is to investigate the accuracy of the prediction approaches. For this study, six of the most commonly used prediction methods are considered: simple exponential smoothing, simple moving average, weighted moving average, Holt-Winter double exponential smoothing, extrapolation and the autoregressive integrated moving average (ARIMA). Their prediction accuracy is determined on a real cloud dataset from Amazon EC2 IaaS cloud services. Three QoS parameters are considered, Central ProcessingUnit (CPU), memory and Input Output (I/O) to predict their QoS values and compare them with the actual observed ones. To consider various possible patterns in the input QoS data and determine their effect on the output values, a dataset is divided into 10 time intervals, starting from five minutes to four weeks, which gives a dataset

with different patterns in it. To further analyse the prediction accuracy, the prediction accuracy of the above-mentioned methods is compared with neural network methods and stochastic methods such as cascade forward backpropagation, Elman backpropagation, generalized regression and Nonlinear Autoregressive Network with eXogenous inputs (NARX) to predict future -1 hour, 2 hours, 3 hours, 4 hours and 5 hours.

The chapter is divided into different sections. This section introduces the chapter. Section 6.2 discusses the time series data. Section 6.3 describes the six QoS prediction methods which are used in this study, while Section 6.4 presents the accuracy benchmark for forecasting methods. Sections 6.5 to 6.11 provides the accuracy validation of the predicted QoS data using the six prediction methods. Section 6.12 contains a comparison of the neural network, stochastic and time series prediction methods. Section 6.13 presents a comparative analysis of the prediction methods. Section 6.14 details a comparison of prediction accuracy of simple exponential smoothing method based on their freshness criterion and Section 6.15 concludes the chapter.

6.2 Different possible patterns in Time Series Data

Time series analysis is the process of measuring variables at a set period of time, which could be hourly, daily, weekly, monthly or some other regular interval of time. Time series data provides information on the previous behaviour of a system or the presence of data patterns in a time series and suggests an appropriate method for future data prediction (Bisgaard & Kulahci 2009; Jeffrey D. Camm 2015). Patterns in time series data help to select an optimal prediction method because each pattern has definite characteristics that can be predicted using a certain prediction method. Some of the common types of data patterns in time series data are described below (Miller & Hickman 1973). These data patterns are represented in Figure 6.1.

- *Horizontal pattern:* This occurs when the data changes arbitrarily over a certain period of time with a constant mean. These data follow a stationary time series in which the statistical properties of data are independent of the series (Miller & Hickman 1973).

- *Trend pattern:* When data show a shift towards higher or lower values over longer periods of time, it is called a trend pattern. Trends in data are due to an increase or decrease in values due to certain factors such as population increase or decrease or a variation in consumer preferences(Bisgaard & Kulahci 2009).
- *Seasonal pattern:* This occurs when data contain certain patterns that repeat themselves after a consecutive period of time due to seasonal variations (Bisgaard & Kulahci 2009).
- *Trend and seasonal pattern:* This occurs when data have characteristics of both trend and seasonality, for example, some data have an increasing or decreasing trend in certain seasons (Jeffrey D. Camm 2015).
- *Cyclic pattern:* This occurs when data have a repeated pattern above and below the trend-line but not for a fixed period with no seasonality(Jeffrey D. Camm 2015).

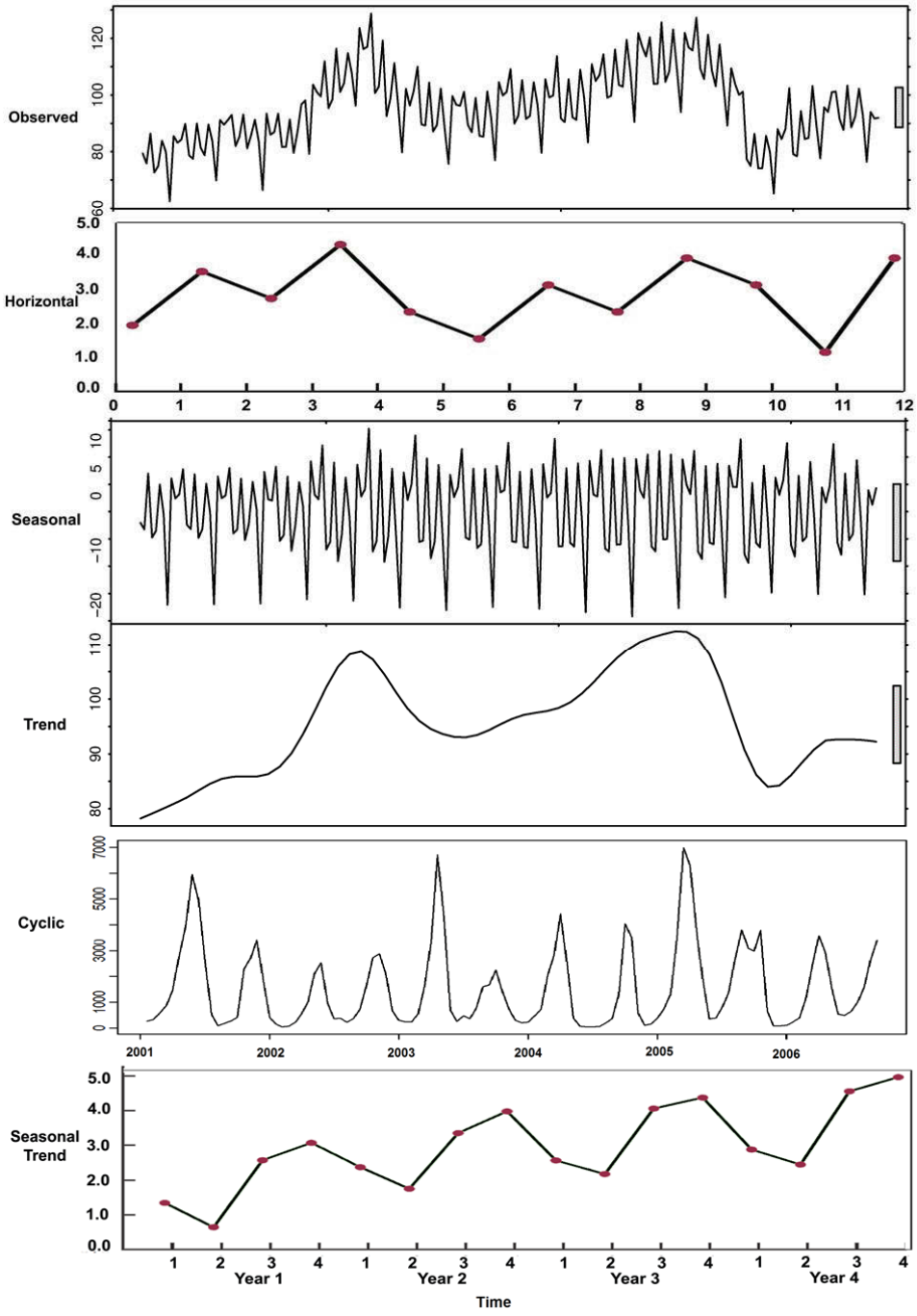


Figure 6.1: Different types of patterns in a time series

As discussed in Chapter 2, it is very important for service providers to predict QoS parameters and ensure that they comply with the QoS parameters defined in the SLA. However, if a provider knows about likely resource usage and identifies a difference between actual and predicted QoS parameters, then it can take appropriate measures to rectify the problem at a very early stage. In the next

section, six prediction methods are discussed and their prediction accuracy is analysed with different parameters on a real cloud dataset.

6.3 QoS Prediction Methods

In this study, neural network methods, stochastic method such as cascade forward backpropagation, Elman backpropagation, generalized regression, NARX and time series prediction methods such as simple exponential smoothing, simple moving average, weighted moving average, Holt-Winter double exponential smoothing, extrapolation and ARIMA are considered. The time series prediction methods are presented in Figure 6.2.

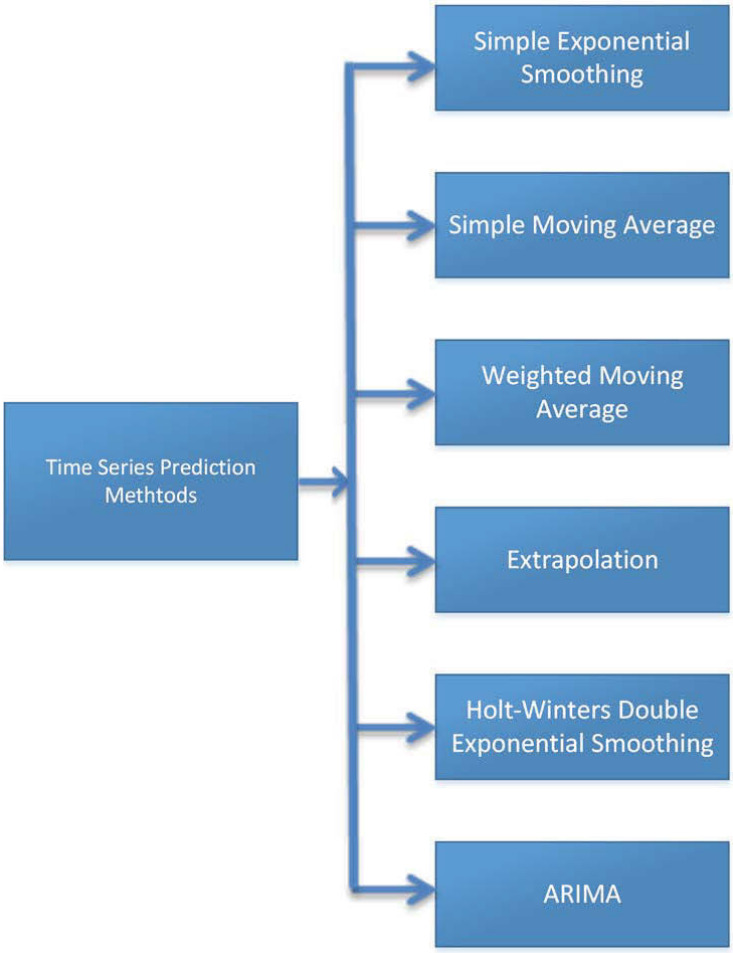


Figure 6.2: Time series QoS prediction methods

These 10 prediction methods are used to determine the prediction accuracy using times series QoS data from a real cloud dataset. These prediction methods are

widely used in time series datasets. A brief explanation of the prediction approaches and their related methods is given in the following subsections.

6.3.1 Simple Exponential Smoothing Method (SES)

Exponential smoothing is the most optimal forecasting method for state-space models (Gardner 1985; Hyndman & Khandakar 2007; Hyndman et al. 2002; Hyndman & Kostenko 2007; Ord, Koehler & Snyder 1997). Exponential smoothing was proposed by Brown (Brown 1959) for smoothing and predicting time series data. The basic purpose of SES is to smooth arbitrary fluctuations in time series data and give optimal results for short-range forecasting (Jeffrey D. Camm 2015). The future values are predicted by the weighted average of all previous observations where weights decrease on an exponential basis over time. In simple terms, higher weights are associated with the nearest past and smaller weights are related to the oldest observations. The smoothing function starts from the second observation and needs an initial value that most of the time is chosen as the first value of the series $F_{t-1}=y_t$. These weights are determined by a smoothing constant, as presented in Equation 6.1.

$$\hat{F}_{t+1} = \propto F_t + (1-\propto)\hat{F}_t$$

Equation 6.1

where $\hat{F}_{t+1}$ is the predicted value at time interval t+1,$\hat{F}_t$is the predicted value at time interval t, F_t is the actual value at time interval t and $\propto$is a smoothing constant range from 0<$\propto$<1.

Equation 6.1 shows that the predicted value at time interval t+1 is equal to the weighted average of the actual value and the predicted value at time interval t. The weight is assigned by the smoothing constant $\propto$. To better understand the process, we consider a time series data with three periods F_1, F_{12} and F_{13}. Let $\hat{F}_1$ be equal to the actual value at time interval 1 - $\hat{F}_1 = F_1$. The forecasted value for $\hat{F}_2$ is presented in Equation 6.2.

$$\hat{\boldsymbol{F}}_{\mathbf{2}} = \propto \boldsymbol{F}_{\mathbf{1}} + (\mathbf{1}-\propto)\hat{\boldsymbol{F}}_{\mathbf{1}}$$

Equation 6.2

$$= \propto F_1 + (1-\propto)F_1$$

$$= F_1$$

The predicted value at time interval 2 is the actual value of data in time period 1.

The predicted value for the third time period is presented in Equation 6.3.

$$\hat{F}_3 = \propto F_2 + (1-\propto)\hat{F}_2$$

Equation 6.3

$$= \propto F_2 + (1-\propto)F_1$$

The predicted value for fourth time period is presented in Equation 6.4.

$$\hat{F}_4 = \propto F_3 + (1-\propto)\hat{F}_3$$

Equation 6.4

$$= \propto F_3 + (1-\propto)(\propto F_2 + (1-\propto)F_1)$$

$$= \propto F_3 + \propto (1-\propto)F_2 + (1-\propto)^2 F_1$$

From Equation 6.4, we see that $\hat{F}_4$ is the weighted average of the first three values in a time series dataset. The sum of all of the weights is equal to 1. Each new smooth or forecasted value is the weighted average of the current observation and its previous observation. Similarly, the previous observation is calculated from the weighted average of the previous observation and the observations before the previous one. The process continues till the end of the time series data. Therefore, $\hat{F}_{t+1}$ is the weighted average of all the previous values in the time series dataset.

The weights assigned by various smoothing constant values are shown in Table 6.1.

Table 6.1: Prediction value with different values for the smoothing constant

Weights to	α = 0.2	α = 0.4	α = 0.6	α = 0.8
F_t	0.2	0.4	0.6	0.8
F_{t-1}	0.16	0.24	0.24	0.16
F_{t-2}	0.128	0.144	0.096	0.032
F_{t-3}	0.1024	0.0864	0.0384	0.0064

6.3.2 Simple Moving Average (SMA)

This is the most basic prediction method to smooth random fluctuations in time series data, similar to the exponential smoothing method. It considers the data from previous N time intervals, averages them and then uses the result to predict the future time interval. Equation 6.5 represents the working of a simple moving average prediction method.

$$\hat{F}_{t+1} = \frac{\sum_{i=t-k+1}^{t} F_t}{k}$$

Equation 6.5

$$= \frac{F_{t-k+1} + \cdots + F_{t-1} + F_t}{k}$$

where $\widehat{F}_{t+1}$ is the forecast value for future time interval $t+1$, F_t is the time series data at time interval t and k is the total time interval.

Each of the k previous values has a weight of $1/k$. When the size of the previous record k becomes bigger, each individual value of the recent past get less weight in order to have a smooth series forecast graph. The first period in F_{t-k+1} is one stage old. The second period is two stages old and so on till k term. The phrase term moving is used because each time a new value replaces the previous value in the equation, a new average is calculated. The average for each period changes or is moved based on the new data. The problem with this method is that it always lags behind the actual data. To use moving average, we need to select the number of time series k. The value of the time series k depends on the relevancy of the number of previous values. For a small number of previous values, a small value of k is considered and for a large number of previous values, a larger value of k is considered. Therefore, for a smaller number of datasets, k will track shift more quickly in a dataset, and a larger value of k gives the optimal result for smoothing random fluctuations.

6.3.3 Weighted Moving Average (WMA)

Unlike the SMA, which assigns an equal weight to all previous data, the WMA gives a higher weight to the nearest past data rather than the older data to calculate the average. We need to select a set of weighting factors such as $w_1, w_2, w_3 \ldots, w_k$, with the sum of all these weights being equal to 1.

$$\sum_{i=1}^{k} w_i = 1$$

The weights are used to determine the smoothed statistics value s_t as presented in Equation 6.6.

$$s_t = \sum_{i=1}^{N} w_i \, a_{t+1-i} \; = w_1 \, a_t + w_2 a_{t-1} + \cdots + w_N a_{t-N+1}$$

Equation 6.6

where *a* is a raw time series and *w* is a weighting factor.

Many technical analysts believe that assigning a greater weight to the recent past data rather than older data produces good prediction results. When using this method, the system reacts quickly when it detects any change. The following equation represents the weighted moving average:

$$F_t = \frac{\sum_{i=1}^{N} w_i * \; A_t}{\sum_{i=1}^{N} wi}$$

Equation 6.7

where *w* is the weighting factor,

A is the actual value,

F is the average value

and *N* is the total time period.

In the above Equation 6.7, we see that each measurement value is multiplied by the weighting factor, which varies from value to value. The total sum is divided by the sum of all the weight factors. Therefore, by assigning higher weights to recent past data and lower weights to older past data, we get values that are more responsive to recent changes. This method faces several issues, for example, the simple moving method cannot be used until the least *N* observation is made. Moreover, if data are missing from any stage of averaging, it is very difficult to precisely restructure a changing signal. To avoid this issue, usually an equal weight is assigned to all missed stages. In this study, we handle the issue in two steps.

Step 1: The first existing observations are used to calculate the predicted data. For example, when the user selects $k = 4$,

$s_1 \; only \; uses \; a_1$

$s_2 \; only \; uses \; (a_1, \; a_2 \,)$

$s_3\ only\ uses\ (a_1,\ a_2\ ,\ a_3\ )$

$s_4\ only\ uses\ (a_1,\ a_{2,}\ a_3\ ,\ a_4\ )$

$s_5\ only\ uses\ (a_2,\ a_{3,}\ a_4\ ,\ a_5\ )$

$s_6\ only\ uses\ (a_3,\ a_{4,}\ a_5\ ,\ a_6\ )$

Step 2: To assign weights to missing factors, the weights are calculated by multiplying the increasing factor and a number of weight factors. In this study, we take the value of the increasing factor *i* and the number of weight *w* factors as input from the user.

$$w_2\ = i\ w_1$$

$$w_3\ = i\ w_2$$

$$w_4\ = i\ w_3$$

$$\ldots\ldots\ldots..$$

$$w_N = i\ w_{N-1}$$

where *N* is a total value in a time series.

The sum of all the weighting factors is equal to 1

$$sum\ \{w_1, w_2, \ldots, w_N\} = 1$$

$$= w_1(1 + i + i^2 + i^3 + \ldots + i^{N-1})$$

$$= w_1 \frac{1 - i^N}{1 - i}$$

However, when a user gives a higher value of *k* or an increasing factor *a,* we face the problem of the smallest non-zero floating point number in Matrix Laboratory (MATLAB) which is 4.9407 x 10^{-324}. Therefore, the system generates a warning message to enter the smaller value for *k* or *a*.

6.3.4 Holt-Winter Double Exponential Smoothing Method (HWDES)

The HWDES method of prediction, proposed by Holt and Winter, deals with data that have a trend and seasonality. Seasonal data are time-series data that repeat after ever *N* time intervals. The Holt-Winter method comprises prediction equation and three smoothing equations, for level, seasonality and trend. There are

two methods in the Holt-Winter model that vary from each other based on seasonal components. These methods are the multiplicative seasonal component and the additive seasonal component (Kalekar 2004).

6.3.4.1 Multiplicative Seasonal Component and its Forecasting Equations

The multiplicative seasonal component is used when seasonal data changes proportionally with the time-series data or when there is a multiplicative change in seasonality. It is represented by Equation 6.8.

$$y_t = (p_1 + p_2 t) * SF_t + e_t$$

Equation 6.8

where p_1 is the permanent component,

p_2 is the linear trend component,

SF_t is a seasonal factor

and e_t is the error component.

Let D_t be the current decentralized level of the process at the end of time period t. The overall smoothing, the smoothing of the trend factor, the seasonal index and the forecast are represented below. The following terms are used in the equations:

$\widehat{D}_t$ is the estimate for the decentralized level of the process,

y_t is the time series data,

L is the length of the season,

$\bar{S}_{t-L}$ is the seasonal factor for the previous season,

S_t is the estimate of the seasonal component,

$\bar{T}_t$ is the estimate of the trend, and

and smoothing constant α, β, γ is greater than zero and less than 1.

i. *Overall smoothing*

The overall smoothing function is represented by the following Equation 6.9.

$$\widehat{D}_t = \propto \frac{y_t}{\bar{S}_{t-L}} + (1-\propto) * (\widehat{D}_{t-1} + \bar{T}_{t-1})$$

Equation 6.9

The time-series data is divided by the seasonal factor of previous season L. The prior value of the estimated trend is added to the previous value of the trend, which is further multiplied by the smoothing constant to estimate the decentralized level.

ii. *Trend factor*

The smoothing of the trend factor is calculated by taking the difference between the two successful estimates of the decentralized level, which is represented by the following Equation 6.10.

$$\bar{T}_t = \beta * (\bar{S}_t - \bar{S}_{t-L}) + (1 - \beta) * \bar{T}_{t-1}$$

Equation 6.10

iii. *Seasonal index*

The smoothing of the seasonal index is calculated by dividing the current seasonal data by the estimate for the decentralized level of the process and multiplying it by the smoothing constant and adding the previous seasonal data for that time period, which is presented in Equation 6.11:

$$\bar{S}_t = \gamma * (y_t/\bar{S}_t) + (1 - \gamma) * \bar{S}_{t-L}$$

Equation 6.11

iv. *Forecast of future value*

The forecast for the immediate next time period is represented by Equation 6.12.

$$y_t = (\widehat{D}_{t-1} + \bar{T}_{t-1})\bar{S}_{t-L}$$

Equation 6.12

and the forecast of the multiple time interval is represented by Equation 6.13.

$$y_{t+T} = (\widehat{D}_{t-1} + \mathrm{T} * \bar{T}_{t-1})\bar{S}_{t-L}$$

Equation 6.13

where T is the time period for which the forecast is performed.

6.3.4.2 Additive Seasonal Component and its Forecasting Equations

The additive seasonal component is used when there is a constant seasonal change in the data, irrespective of the overall level of time-series data. It is represented by Equation 6.14:

$$y_t = p_1 + p_2 t + SF_t + e_t$$

Equation 6.14

where p_1 is the permanent component,

p_2 is the linear trend component,

SF_t is a seasonal factor,

and e_t is the error component.

i. Overall smoothing

The overall smoothing function is represented by the following Equation 6.15.

$$\widehat{D}_t = \propto (y_t - \bar{S}_{t-L}) + (1 - \propto) * (\widehat{D}_{t-1} + \bar{T}_{t-1})$$

Equation 6.15

The time-series data is subtracted from the seasonal factor of previous season L and multiplied by the smoothing constant. The prior value of the estimated trend is added to the previous value of the trend, which is further multiplied by the smoothing constant to estimate the decentralized level.

ii. Trend factor

The smoothing of the trend factor for the additive seasonal component is calculated in a similar way to the multiplicative seasonal component. The following Equation 6.16 presents the trend factor for the additive seasonal component.

$$\bar{T}_t = \beta * (\bar{S}_t - \bar{S}_{t-L}) + (1 - \beta) * \bar{T}_{t-1}$$

Equation 6.16

iii. Seasonal index

The smoothing of the seasonal index is calculated by subtracting the current seasonal data from the estimate for the decentralized level of the process and multiplying it with smoothing constant and adding the previous seasonal data for that time period, which is presented in Equation 6.17.

$$\bar{S}_t = \gamma * (y_t - \bar{S}_t) + (1 - \gamma) * \bar{S}_{t-L}$$

Equation 6.17

iv. Forecast of future value

The forecast for the immediate next time period is presented by Equation 6.18:

$$y_t = \widehat{D}_{t-1} + \overline{T}_{t-1} + \bar{S}_{t-L}$$

Equation 6.18

where T is the time period for which the forecast is performed.

6.3.5 Autoregressive Integrated Moving Average (ARIMA)

ARIMA is the systematic method formulated by mathematical statisticians George Box and Gwilym Jenkins in the 1970s (Box, Jenkins & Reinsel 2011) to use with business and economic data. ARIMA is one of the most efficient methods of the autoregressive moving average (ARMA) method. ARMA is divided into two parts: autoregressive (AR) and moving average (MA). AR relates to the values of the dependent variables in a time series that have previously been defined or lagged. MA relates to the lags of the errors which are forecasted (Andrews et al. 2013). Both the AR and MA are used for stationary time series data. ARIMA is a model of the ARMA method which includes the seasonality component (Ziegel 1994). A non-seasonal ARIMA model is denoted by ARIMA (p,d,q) where p, q and d are positive integers and p is the order of AR, d is the degree of differencing and q is the order of MA. The seasonal ARIMA model is denoted by ARIMA (p,d,q) (P,D,Q) m where m is the total number of periods in each season and P,D and Q are the AR, differencing and MA of the seasonal component of ARIMA.

Several basic concepts need to be clarified before the ARIMA model is explained. These concepts are stationarity, differencing, autocorrelation, the AR model, the MA model and the mixed model.

6.3.5.1 Stationarity

Stationary time series data have uniform statistical properties over time. This data does not have any trends or seasonality over the time series. Stationary data have irregular cyclic behaviour throughout the time series and there is a constant variation in fluctuation over the time period. A unit test, for example, augmented Dickey-Fuller (Dickey & Fuller 1979), is used to measure the stationarity in time series data. In many cases, the process is unable to make accurate predictions when the data does not satisfy stationarity conditions.

6.3.5.2 Differencing

Differencing is an effective method for transforming non-stationary data into stationary data. The process of differencing is performed by subtracting a current value from its previous value. Differencing is calculated by the following Equation 6.19.

$$\bar{y} = y_t - y_{t-1}$$
Equation 6.19

where $\hat{y}$ is the differencing value,

y_t is the current value and

$y_{t-1}$is the previous value.

When the process of transforming is performed once, it is called *first differenced*. The process of first difference eliminates any trend from a series that grows at a constant rate, however, when a trend grows at an increasing rate, the data need to be differenced again to make it stationary. This process is called *second differenced*, as presented in Equation 6.20.

$$\bar{\bar{y}} = \bar{y}_t - \bar{y}_{t-1}$$
Equation 6.20

6.3.5.3 Autocorrelation

Autocorrelation measures the correlation between data over a time series. The numerical value ranges from -1 to +1, where +1 indicates more positive correlation and -1 indicates high negative correlation. To construct the ARIMA

model, it is essential to determine the pattern of autocorrelations and partial autocorrelations to decide whether to include the lags of a stationarized series or forecast errors in forming an equation (Andrews et al. 2013).

6.3.5.4 Autoregressive (AR) model

The autoregressive (AR) model predicts a future value by adding the previous value to the random error or white noise, as shown in Equation 6.21:

$$y_t = a + \psi_1 y_{t-1} + \psi_2 y_{t-2} + \psi_3 y_{t-3} + \cdots + \psi_n y_{t-n} + e_t$$

Equation 6.21

where y_t is the predicted value,

a is a constant value,

ψ is an autoregressive parameter,

y_{t-1} is a time series value lagged by one period of time,

n is the total number of observations, and

e is an error term or white noise.

6.3.5.5 Moving Average (MA) model

In the moving average (MA) model, the future value is predicted by taking the average of the previous n number of forecast errors, as shown in Equation 6.22.

$$y_t = b + \text{⫛}_1 e_{t-1} + \text{⫛}_2 e_{t-2} + \text{⫛}_3 e_{t-3} + \cdots + \text{⫛}_n e_{t-n} + e_t$$

Equation 6.22

where y_t is the predicted value, b is a constant value, ⫛ is a moving average parameter, e_{t-1} is an error lagged by one period of time, n is the total number of observations.

6.3.5.6 Mixed Model

The mixed model allows both the AR and MA models to work together to predict a future time series. ARIMA uses a mixed model that combines the AR model and

integration, which is the reverse method of differencing, and MA, to predict a future value and is represented as ARIMA (p,d,q).

6.3.5.7 Working of ARIMA

As previously mentioned, ARIMA integrates both the AR and MA models, as shown in Equation 6.23.

$$\check{y}_t = a + \psi_1 y_{t-1} + \psi_2 y_{t-2} + \psi_3 y_{t-3} + \cdots + \psi_n y_{t-n} + b + \Phi_1 e_{t-1} + \Phi_2 e_{t-2} + \Phi_3 e_{t-3} + \cdots + \Phi_n e_{t-n} + e_t$$

Equation 6.23

where $\check{y}_t$ is the predicted value.

The order of the autoregressive model, the number of differencing and the order of the moving average model is written as ARIMA(p,d,q). Therefore, ARIMA (1,1,2) is the first order of the AR model with the second order of MA and it has been single differenced.

6.3.6 Extrapolation Method (Exp)

The Extrapolation (Exp) method is the process of predicting future data based on the previous available data and it is used for both short and long-range forecasting. Unlike interpolation, which considers previous data between two known data points, extrapolation considers data beyond the range of known data points. For this reason, extrapolation produces noisy and insignificant output for long range forecasting (Armstrong 2001; Armstrong & Forecasting 1985). It is reliable, inexpensive, quick and effortlessly automated.

Selecting the data is an important part of the extrapolation process because this will decide how the data will be analysed. It is necessary to obtain data that characterize the forecast conditions. Choosing data is obvious; however, in a situation when there are frequent changes, it becomes more difficult to select. Data can be either historical data or simulated data. The process of extrapolation can only be applied to historical data. The precision of the historical data and the degree to which the underlying conditions that transform later on affect the accuracy of extrapolation. Simulated data is used when actual data is not available and are generated in realistic conditions or in a laboratory. Based on the results of

simulated data, a consumer assumes that future results will be similar to the current result. Short-range data, that is, data collected over a time period of less than a year, are adjusted by a seasonal adjustment to reduce error in prediction (Armstrong 2001). Some of the common methods of extrapolation are linear extrapolation, polynomial extrapolation and conic extrapolation.

6.3.6.1 Linear Extrapolation

In linear extrapolation, a tangent line is drawn at the end of the available data, which extends beyond the limit of a series. For optimal results, the predicted data need to be very far from the limit of the available data. Linear extrapolation is presented in Equation 6.24:

$$b(\tilde{a}) = b_1 + \frac{\tilde{a}-a_1}{a_2-a_1} * (b_2 - b_1)$$

Equation 6.24

where (a_1, b_1) and (a_2, b_2) are the end point of a series, $b(\tilde{a})$is the predicted value at point $\tilde{a}$.

6.3.6.2 Polynomial Extrapolation

Polynomial extrapolation determines the function value at some point $\tilde{a}$ on the x-axis, which is in the range of dataset *n* value. A polynomial curve is drawn at the end of the available data. The probability of error is very high when extrapolating a higher degree of polynomial data. Once error is included, it increases exponentially (Cabay & Jackson 1976).

6.3.6.3 Conic Extrapolation

A conic section can be created using five points near the end of the known data. This type of extrapolation is performed with a conic sections template (Myers 1994). Depending on the type of conic section, the loop may return back toward itself; however, a parabolic curve will not rejoin itself but may curve back toward the x-axis.

6.3.7 Cascade Forward Backpropagation

A cascade forward backpropagation network is a type of artificial neural network that is used for prediction of data. These networks are similar to feedforward

backpropagation networks; however, in this method it includes weight factors from the input and every previous layers to each successive layers (AL-Allaf 2012).

6.3.8 Elman Backpropagation

Elman networks are two-layer feedforward networks with an added input from the output of the context layer. Therefore, the backpropagation along with the hidden layer in the context layer are used for training the network (Elman 1990). The additional input allows an Elman network to recognize and generate temporal and spatial patterns (Goyal & Goyal 2012).

6.3.9 Generalized Regression

A generalized regression method is a part of the radial basis neural network. Unlike backpropagation it does not require any training procedure; instead, it approximates the arbitrary function between the input and output vectors and estimates directly from the set of training data. It is based on non-linear regression theory for function approximation (KIŞI 2006).

6.3.10 Non-linear Autoregressive Exogenous (NARX)

A non-linear autoregressive exogenous (NARX) neural network is a non-linear autoregressive model that takes the previous data of the same series and the current and previous data from the exogenous series as an input data to predict the future data (Gao & Er 2005).

6.4 Accuracy Benchmark for Forecasting Methods

The benchmark to measures the forecast accuracy: mean square error, root mean square error and mean absolute deviation. Prediction accuracy depends on forecast error, which is the difference between the observed and predicted values at time interval t. If Z_t and $\check{Z}_t$ represent the observed and predicted value at time interval t, then forecast error e_t is calculated using Equation 6.25.

$$e_t = Z_t - \check{Z}_t$$

Equation 6.25

A positive error indicates that the forecast method has underestimated the actual observation for time period *t*, and a negative error indicates that the forecast method has overestimated the actual observation. The three prediction accuracy methods that are used in this study are explained in the following subsections.

6.4.1.1 Mean Absolute Deviation (MAD)

Mean absolute deviation (MAD), also referred to as mean absolute error (MAE), is the mean absolute value of the forecast error. MAD does not consider positive or negative forecast errors and is calculated using Equation 6.26.

$$\text{MAD} = \frac{\sum_{t=b+1}^{a} |e_t|}{a-b}$$

Equation 6.26

where *a* is a number of observation in a time series and b is the number of past periods of time series.

6.4.1.2 Mean Square Error (MSE)

Mean square error (MSE) is the average of the squared forecast errors and is calculated by Equation 6.27.

$$\text{MSE} = \frac{\sum_{t=b+1}^{a} e_t}{a-b}$$

Equation 6.27

MSE gives a quadratic loss function as it squares and averages the different errors. MSE is therefore advantageous at the point when we would be concerned about huge errors whose negative results are proportionately greater than its equivalent smaller ones (Makridakis 1993).

6.4.1.3 Root Mean Square Error (RMSE)

Root mean square error (RMSE) is the square root of the MSE and is used to measure prediction accuracy, which is presented in Equation 6.28.

$$RSE = \sqrt{MSE} = \sqrt{\frac{\sum_{t=b+1}^{a} e_t}{a-b}}$$

Equation 6.28

6.5 Accuracy Validation of the Predicted QoS Data

In this section, the prediction accuracy of mentioned approaches is evaluated. The cloud dataset is used to evaluate the mentioned approaches, which are presented in below sections.

6.5.1 Cloud QoS data

The prediction accuracy of the six prediction approaches is tested using data from Amazon EC2 IaaS cloud services – EC2 US West, which covers a period of three years starting from 28 March 2012 to 28 March 2015. The data was collected from CloudClimate (CloudClimate) using the PRTG monitoring service (Monitor). The QoS parameters considered for this study are CPU, memory and I/O performance. By memory it means the RAM utilization for different servers. Figure 6.3 shows the QoS parameters for the time period in a graphical format as represented by the PRTG network monitor.

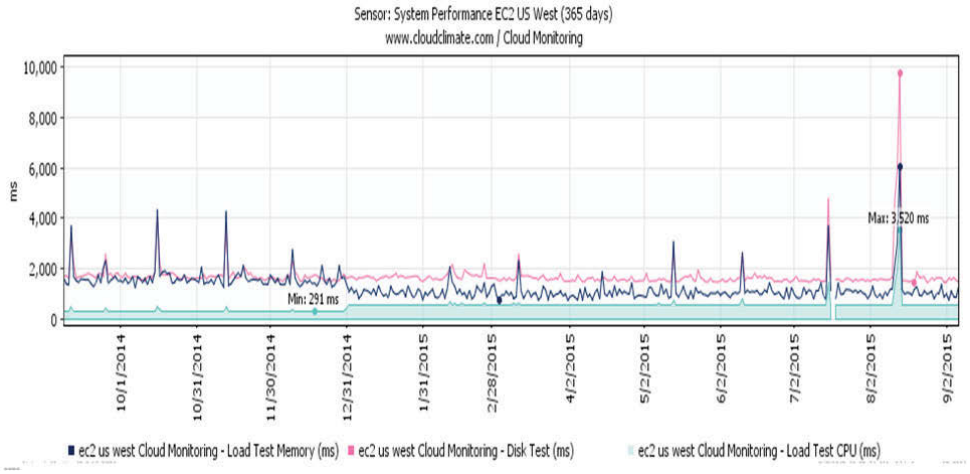


Figure 6.3: QoS parameters of EC2 US West (Monitor)

6.5.2 Measurement Interval of QoS Parameters and Determining the Patterns in them

It is very important for a service provider to be alerted to a likely violation of an SLA before it occurs so that appropriate action can be taken to avoid it. Each of the selected six prediction methods generates different prediction accuracy at various time intervals. As a small or medium-sized provider generally has limited available resources, this study will help a service provider to choose an appropriate prediction method that gives optimal results at various time intervals. To analyse the deviation between actual and predicted observations and to

measure prediction accuracy at different time intervals, the measurement intervals of dataset are divided into 10 subsets ranging from 5 minutes, 10 minutes, 20 minutes, 1 hour, 4 hours, 12 hours, 1 day (24 hours), 1 week, 2 weeks and 4 weeks. 5 minutes is selected as a minimum time interval because a provider takes around 5 to 10 minutes to set up a virtual machine in a cloud environment and a system can request a new virtual machine instance about 12 to 15 minutes prior to arranging the required resources (Islam et al. 2012). Therefore, a time interval of 5 minutes is the minimum possible time for the provider to take appropriate mitigating action when it detects that a violation is likely to occur. The prediction accuracy of all six-prediction methods is analysed at these 10 time intervals and determines the parameters at which they give optimal results.

The data patterns for CPU, memory and I/O at 5-minute time intervals are presented in Figures 6.4 to 6.6 respectively. For the 5-minute time intervals, there is a dataset of 1918 data points for CPU, memory and I/O.

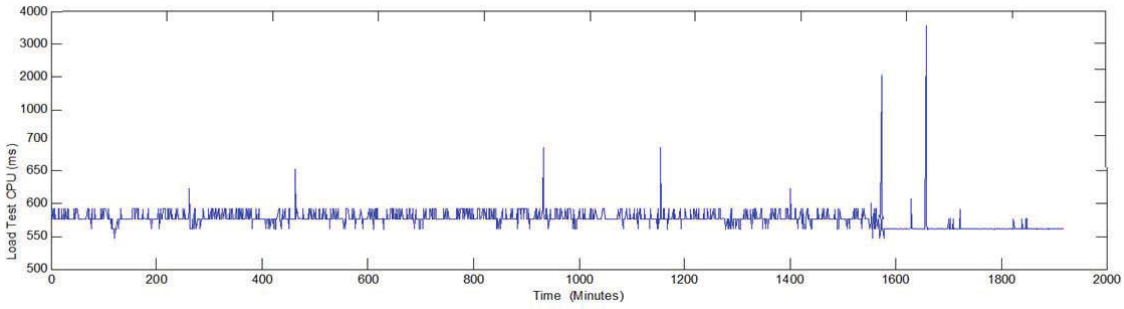


Figure 6.4 : CPU data pattern at 5-minute time intervals

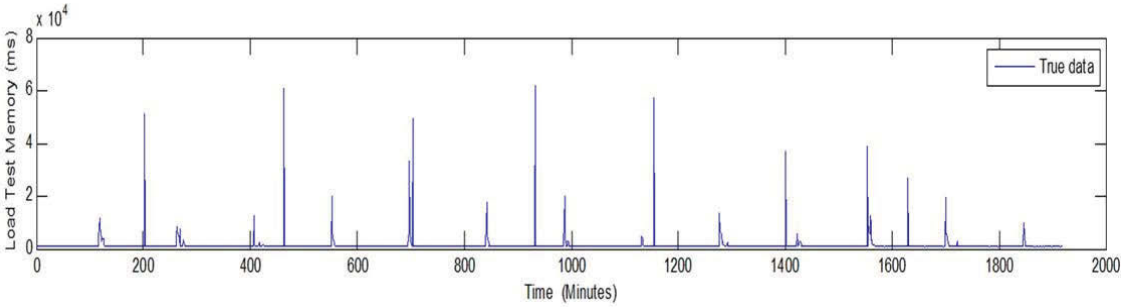


Figure 6.5: Memory data pattern at 5-minute time intervals

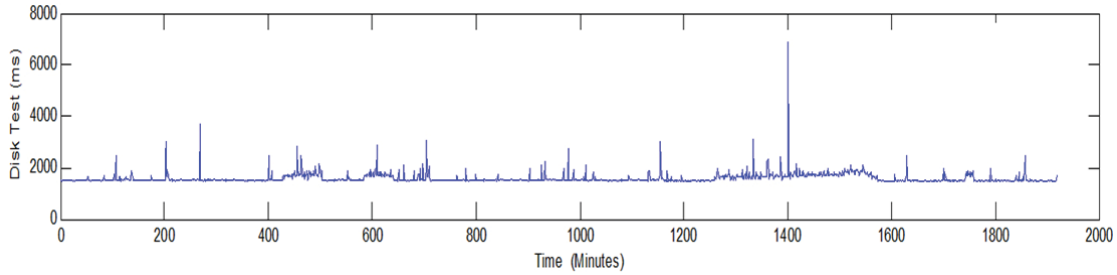


Figure 6.6: I/O data pattern at 5-minute time intervals

Figure 6.4 shows that the CPU data follows a horizontal pattern; however, we see that at the time intervals of 1552 and 1628, there is a sudden increase in the values of CPU. Due to this increase, RMSE and MAD give a higher value when we compared to the original and predicted values. The mean of CPU values to time interval point 1552 is 576 follows a horizontal pattern. Figure 6.5 shows that the memory data follow a cyclic pattern. We see that the pattern almost repeats itself after some time intervals with an increase in memory, however these are not at fixed periods. Figure 6.6 shows the I/O data for a period of 1918 data points and it can be seen that the data does not follow any particular pattern.

The data patterns for CPU, memory and I/O for the 10-minute time intervals are presented in Figures 6.7 to 6.9. For the 10-minute time intervals, we have a dataset of 960 data points. Figure 6.7 shows the CPU data follows a horizontal pattern with a mean value of 576. Figure 6.8 shows the memory data for 10-minute time intervals follows a cyclic pattern which almost repeats itself with an increase in memory, however these are not for fixed time periods. Figure 6.9 shows that the I/O data that does not have any pattern and they are distributed randomly throughout the time series.

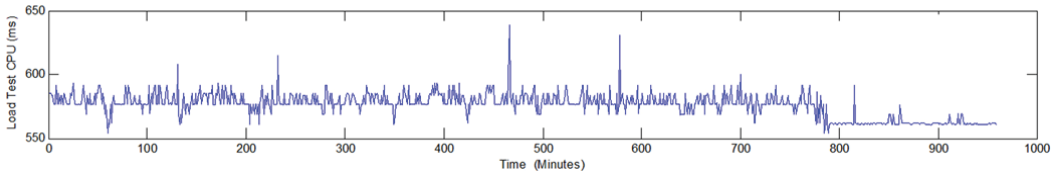


Figure 6.7 : CPU data pattern for 10-minute time intervals

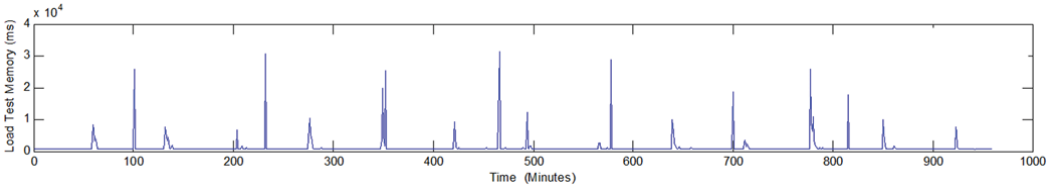


Figure 6.8: Memory data pattern for 10-minute time intervals

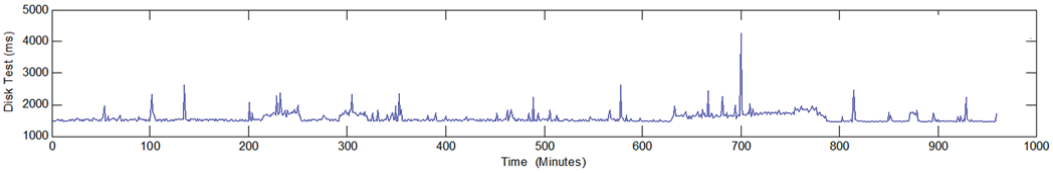


Figure 6.9: I/O data pattern for 10-minute time intervals

The data patterns for CPU, memory and I/O at 20-minute time intervals are presented in Figures 6.10 to 6.12. For 20-minute time intervals, we have a dataset with 482 data points for CPU, memory and I/O. Figure 6.10 shows that CPU has a horizontal data pattern with a mean value of 576. Figure 6.11 shows that memory has a cyclic data pattern with a mean value of 1078 and that the data pattern almost repeats itself after a few time intervals. Figure 6.12 shows that I/O data does not follow any pattern.

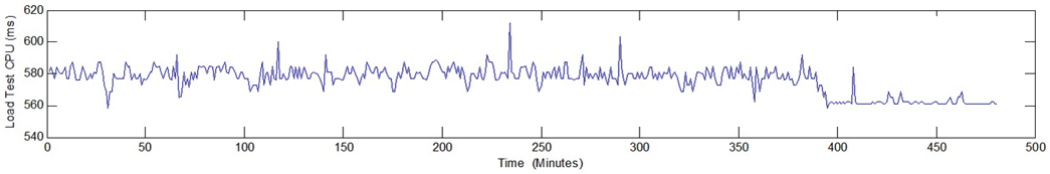


Figure 6.10 : CPU data pattern for 20-minute time intervals

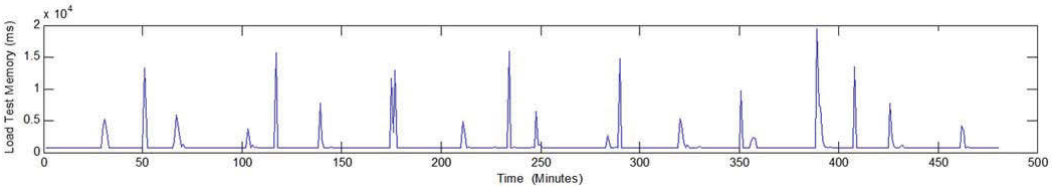


Figure 6.11: Memory data pattern for 20-minute time intervals

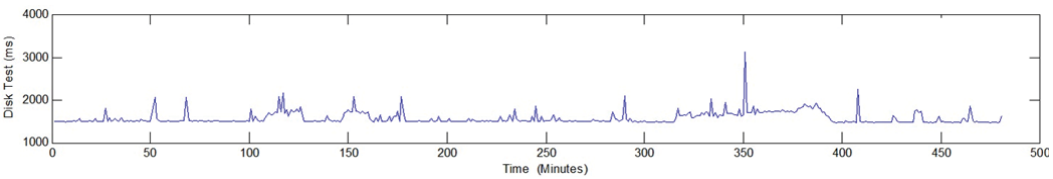


Figure 6.12: I/O data pattern for 20-minute time intervals

The data patterns for CPU, memory and I/O for 1-hour time intervals are presented in Figures 6.13- to 6.15. For 1-hour time intervals, we have a dataset with 161 data points for CPU, memory and I/O. Figure 6.13 shows that CPU has a horizontal data pattern with a mean value of 576. Figure 6.14 shows that memory has a cyclic data pattern that repeats itself after a few time intervals. Figure 6.15 shows that I/O does not follow any pattern.

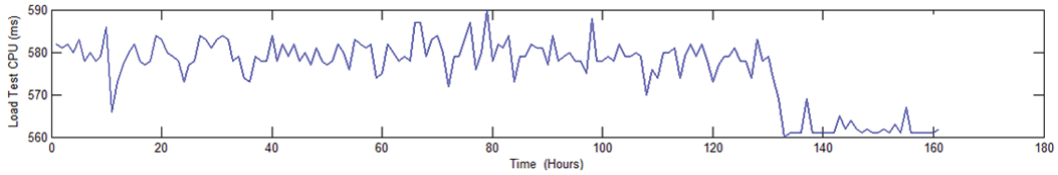


Figure 6.13 : CPU data pattern for 1-hour time intervals

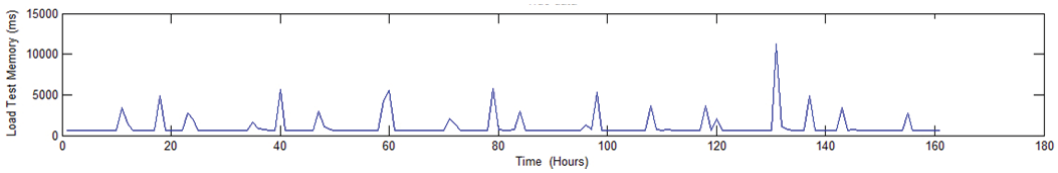


Figure 6.14: Memory data pattern for 1-hour time intervals

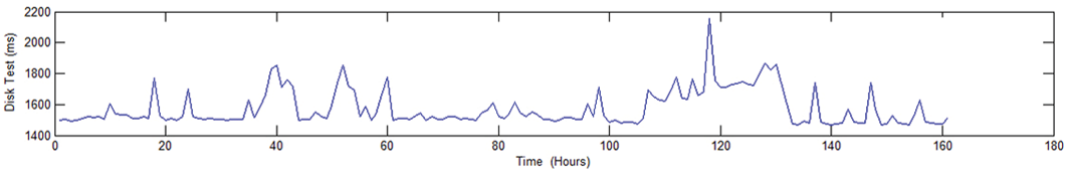


Figure 6.15: I/O data pattern for 1-hour time intervals

The data patterns for CPU, memory and I/O for 4-hour time intervals are presented in Figures 6.16 to 6.18. For 4-hour time intervals, we have a dataset with 186 data points for CPU, memory and I/O. Figure 6.16 shows that CPU has a horizontal data pattern with a mean value of 574. Figure 6.17 shows that memory has a cyclic data pattern that repeats itself after a few time intervals. Figure 6.18 shows that I/O has a horizontal pattern with a mean value of 1656, and a maximum and minimum value of 4833 and 1486, respectively.

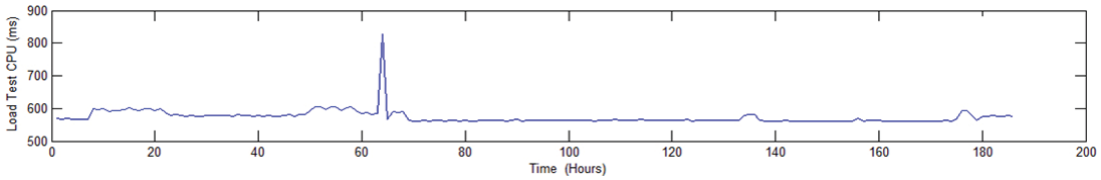


Figure 6.16 : CPU data pattern for 4-hour time intervals

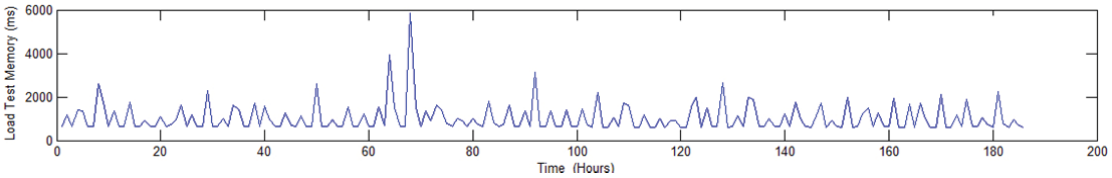


Figure 6.17: Memory data pattern for 4-hour time intervals

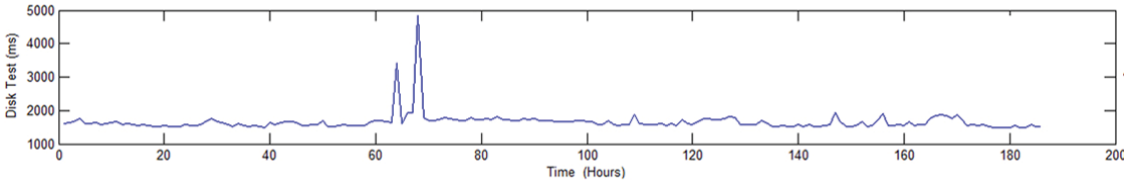


Figure 6.18: I/O data pattern for 4-hour time intervals

The data patterns for CPU, memory and I/O for 12-hour time intervals are presented in Figures 6.19 to 6.21. For 12-hour- time intervals, we have a dataset with 62 time intervals for CPU, memory and I/O. Figure 6.19 shows that after 25 time intervals the values of CPU are almost constant with very slight variation. Figure 6.20 presents a random data set for memory. Figure 6.21 shows that I/O has a horizontal pattern with a mean value of 1657.

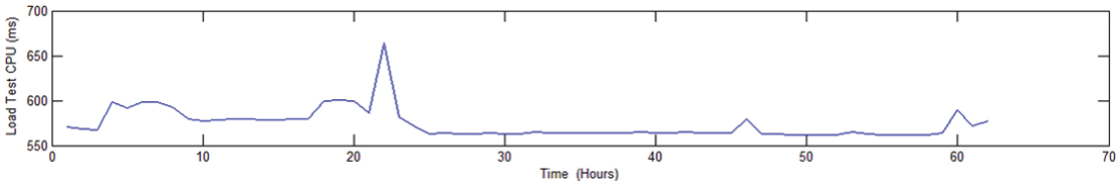


Figure 6.19 : CPU data pattern for 12-hour time intervals

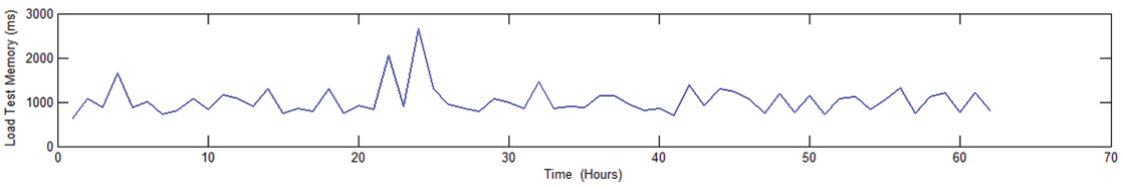


Figure 6.20: Memory data pattern for 12-hour time intervals

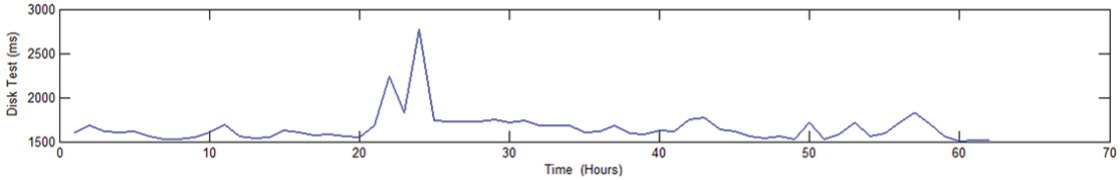


Figure 6.21: I/O data pattern for 12-hour time intervals

The data patterns for CPU, memory and I/O for 1-day time intervals are presented in Figures 6.22 to 6.24. For 1-day time intervals, we have a dataset with 30 time intervals for CPU, memory and I/O. Figure 6.22 shows that the data for CPU does not follow any pattern. Figure 6.23 shows a random data set for memory with a mean value of 1073. Figure 6.24 shows that I/O has a horizontal pattern with a mean value of 1664. The maximum value in this time series is 2593 and the minimum value is 1539 with a range of 1054.

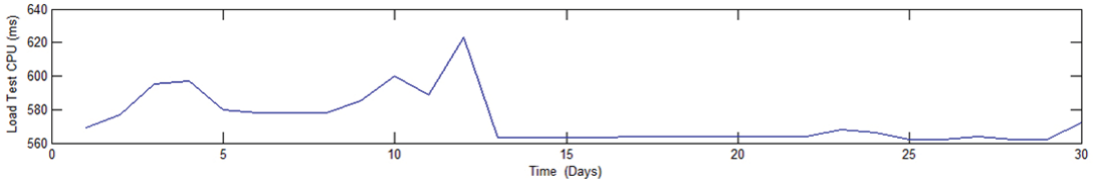


Figure 6.22 : CPU data pattern for 1-day time intervals

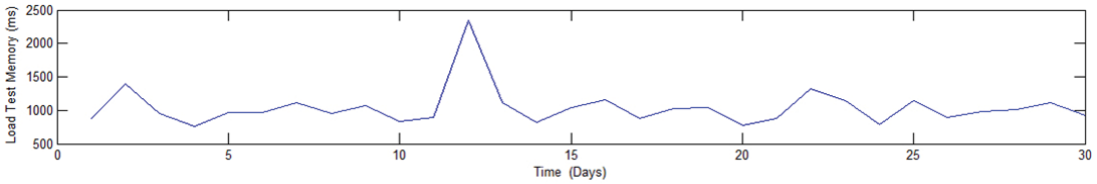


Figure 6.23: Memory data pattern for 1-day time intervals

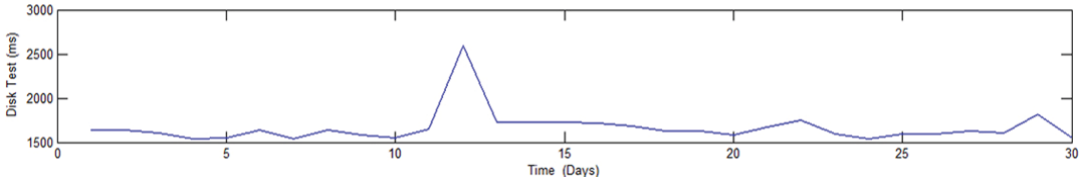


Figure 6.24: I/O data pattern for 1-day time intervals

The data patterns for CPU, memory and I/O for 1-week time intervals are presented in Figures 6.25 to 6.27. For 1-day time intervals, we have a dataset with

26 time intervals for CPU, memory and I/O. Figure 6.25 shows that the data pattern for CPU is half cyclic; that is, half of the dataset has a lower value and the other half has a higher value. Figure 6.26 shows a decreasing trend for memory data and Figure 6.27 shows that I/O follows no trend.

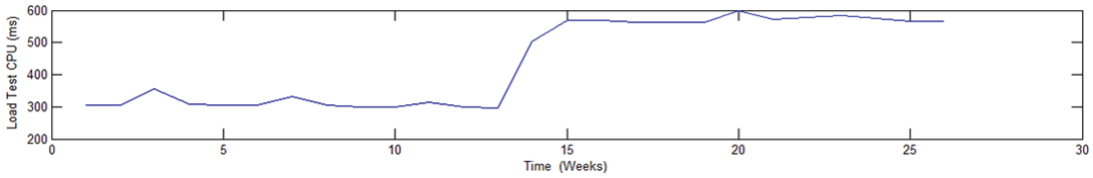


Figure 6.25 : CPU data pattern for 1-week time intervals

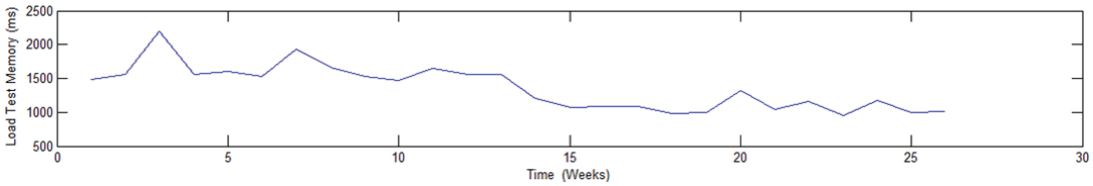


Figure 6.26: Memory data pattern for 1-week time intervals

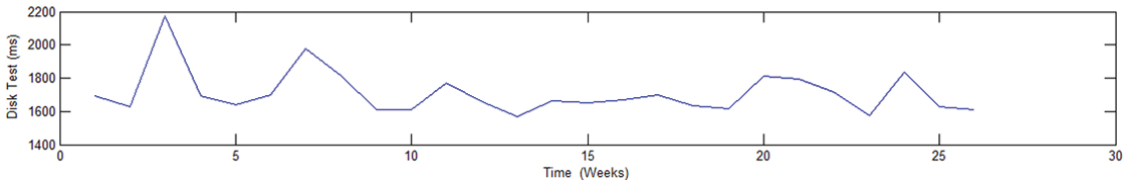


Figure 6.27: I/O data pattern for 1-week time intervals

The data pattern for CPU, memory and I/O for 2-week time intervals are presented in Figures 6.29 to 6.31. For 2-week time intervals, we have a dataset with 105 data points for CPU, memory and I/O. Figures 6.29, 6.30 and 6.31 show the data follows a seasonal pattern for CPU, memory and I/O which repeats itself after a consecutive period of time.

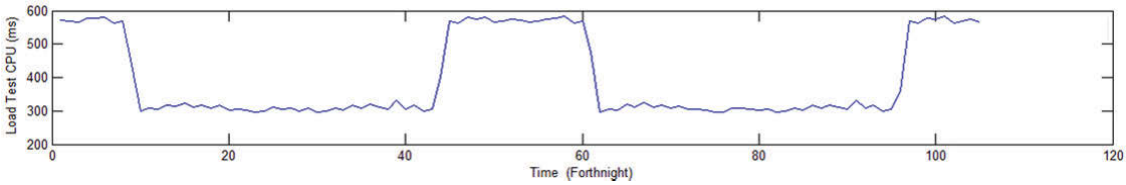


Figure 6.28 : CPU data pattern for 2-week time intervals

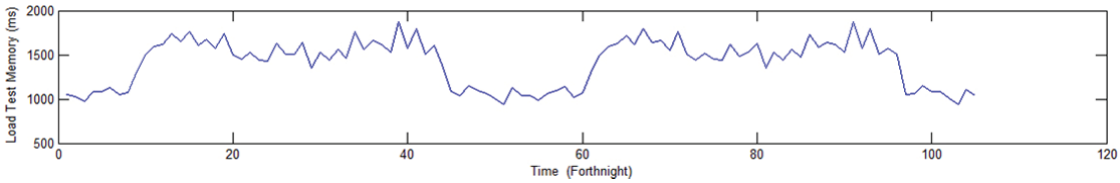


Figure 6.29: Memory data pattern for 2-week time intervals

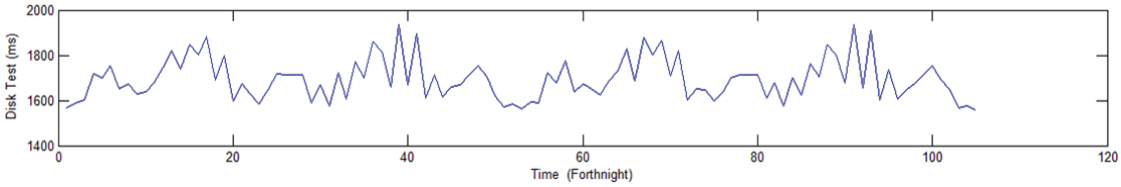


Figure 6.30: I/O data pattern for 2-week time intervals

The data pattern for CPU, memory and I/O for 4-week time intervals are presented in Figures 6.31 to 6.33. For 4-week time intervals, we have a dataset with 53 time intervals for CPU, memory and I/O. Figures 6.31, 6.32 and 6.33 show that the data for CPU, memory and I/O have a seasonal pattern that repeats itself after every consecutive period of time.

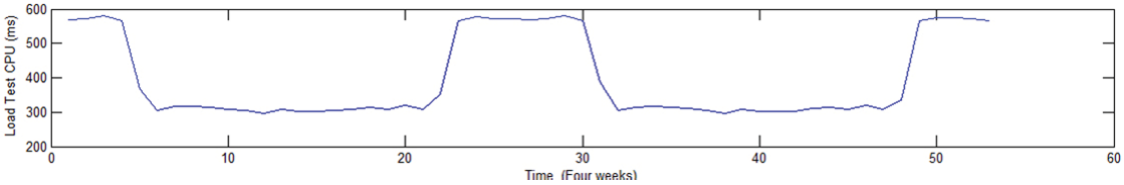


Figure 6.31 : CPU data pattern for 4-week time intervals

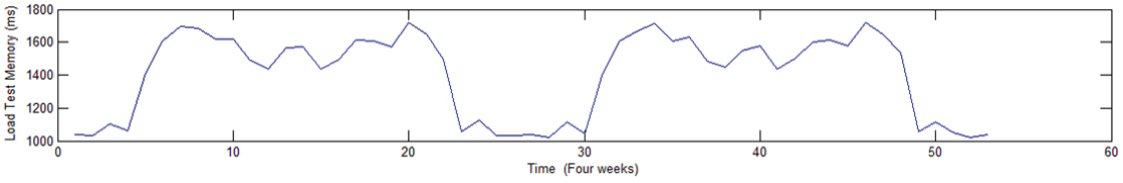


Figure 6.32: Memory data pattern for 4-week time intervals

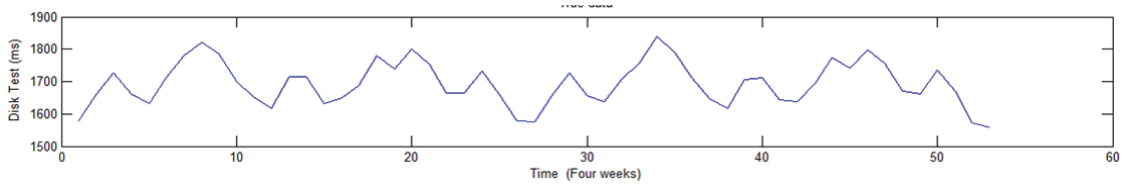


Figure 6.33: I/O data pattern for 4-week time intervals

Table 6.2 presents the data patterns observed for CPU, memory and I/O for all 10 datasets.

Table 6.2: Data patterns for CPU, memory and I/O for all datasets

Data pattern for	CPU	Memory	I/O
5 min	Horizontal	Cyclic	Random
10 min	Horizontal	Cyclic	Random
20 min	Horizontal	Cyclic	Random
1 hr	Horizontal	Cyclic	Random
4 hr	Horizontal	Cyclic	Horizontal
12 hr	Horizontal	Random	Horizontal
1 day	Random	Random	Horizontal
1 week	Cyclic	Trend	Random
2 weeks	Seasonal	Seasonal	Seasonal
4 weeks	Seasonal	Seasonal	Seasonal

6.6 QoS Prediction Accuracy using the Simple Exponential Smoothing (SES) Method

In this section, the prediction accuracy of the SES method is determined in order to forecast the QoS parameters. The existing literature advocates that different values of the smoothing constant α be used for prediction to represent the sensitivity of a forecast. Chopra and Meindle (Chopra & Meindl 2007) suggest that a value of α = 0.2 is the one that generates an optimal result. Schroeder et al. (Schroeder, Goldstein & Rungtusanatham 2013) recommend the value of α be between α = 0.1 and α = 0.3 to generate an optimal result. Heizer and Render (Heizer, Render & Weiss 2004) propose a range of α = 0.05 to 0.5 in order to produce an optimal prediction result. To observe the impact of smoothing constant α on prediction accuracy, the prediction result with all nine possible values are analysed for the variable. It starts with the value of 0.1 and increases to 0.9, and observes the prediction accuracy for each case by ascertaining MAD, MSE and RMSE. The prediction results for CPU, memory and I/O by using SES is presented in Appendix – A in Tables A1-A3.

6.7 QoS Prediction Accuracy using the Simple Moving Average Method

In this section, the prediction accuracy of the simple moving average method is determined to forecast the QoS parameters. Depending on the size of the dataset, the prediction accuracy is tested with different values of *k*, where *k* is the number

of observations. The values for MAD, MSE and RMSE for each time interval with a variable value of k is discussed in the following sections. The prediction results for CPU, memory and I/O by using SES are presented in Appendix – B in Tables B1- B3.

6.8 QoS Prediction Accuracy using the Weighted Moving Average Method

In this section, the prediction accuracy of the weighted moving average method is determined in order to forecast the QoS parameters. To analyse the impact of the number of observations and an increasing factor, the system takes two inputs from a user: the number of observations *k* that is used to consider average and weight factor α. A weight factor is used to assign the highest weight to recent past data and a lower weight to distant past data and the sum of all the weight factors is equal to zero. To analyse the impact of different values of *k* with respect to the weight factor α, we take 15 entries for each time interval. Depending on the size of the dataset, the value of *k* (the number of observations) and a weighted factor are selected. Three values of *k* are selected for each time interval in such a way that a first value of *k* takes the first two observations, the second value of *k* takes the mid-value of the dataset and the third value of *k* takes the last value of the dataset. For each value of *k*, it is evaluated with four values of weighted factors: 0.5, 1.5, 2, 5 and 10. The value for MAD, MSE and RMSE for CPU, memory and I/O is presented in Table C1 in Appendix –C.

6.9 QoS Prediction Accuracy using Extrapolation Method

In this section, the prediction accuracy of the extrapolation method is determined to forecast the QoS parameters. The prediction results for CPU, I/O and memory by using Extrpolation method are presented in Appendix – D.

6.10 QoS Prediction Accuracy using the Holt-Winter Double Exponential Smoothing Method

In this section, the prediction accuracy of the Holt-Winter double exponential smoothing method is determined in order to forecast the QoS parameters. To analyse the impact of smoothing factor α and trend smoothing factor β on prediction accuracy, all possible values of α ($0< \alpha<1$) and β ($0< \beta <1$) are considered for each time interval and take the value of MAD, RMSE and MSE.

Each time interval has 81 entries with all possible values of α and β. The values for MAD, MSE and RMSE for CU, memory and I/O are shown in Table E1 in Appendix -E. From Table E1 in Appendix - E, we observe that each time interval where the value of α is 0.9 and the value of β is 0.1 gives an optimal prediction result. Therefore, for an optimal value in each time interval, each value of α and β should be 0.9 and 0.1 respectively.

6.11 QoS Prediction Accuracy using the ARIMA Method

In this section, the prediction accuracy of the ARIMA method is determined to forecast the QoS parameters. The system takes as inputs the order of ARIMA (p), the degree of differencing (d) and the order of MA (q). Due to space limitations, the eight combinations of *p, d* and *q* are considered for each time interval, these being (0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1.0.1), (1,1,0) and (1,1,1). The values for MAD, MSE and RMSE for each time interval with different values for different combinations of *p,d* and *q* are presented in Table F1 in Appendix - F.

6.12 Comparative Analysis of Neural Network Methods and Stochastic Methods with the Time Series Prediction Methods

In this section, all of the 10 prediction methods that we discussed in Section 6.3 are analysed and compared. For this analysis, five datasets of a CPU are considered. The datasets from 1-100, 1-200, 1-300, 1-400 and 1-500 are selected and their prediction accuracy compared for next 10 time intervals. The dataset which is considered is from Amazon EC2 IaaS cloud services – EC2 US West, which was collected from CloudClimate (CloudClimate) by using PRTG monitoring service (Monitor). We consider a QoS parameter – CPU for every five-minute interval for the period of four days starting from 21/04/2015 4:00:00 to 25/04/2015 4:00:00 having 1007 observations.

6.12.1 Analysis of 10 Methods

Neural network methods are trained for 1007 observations and then they are allowed to predict for the future five hours respectively. The observed and predicted CPU response times from one hour to five hours are presented in Tables 6.3 to 6.7 and Figures 6.34 to 6.38 respectively.

Table 6.3: Observed and predicted values for the first hour, starts from 4:10:00 to 5:05:00

Time	4:10:00	4:15:00	4:20:00	4:25:00	4:30:00	4:35:00	4:40:00	4:45:00	4:50:00	4:55:00	5:00:00	5:05:00
Observed	576	577	576	577	577	577	576	577	593	577	592	576
CFBP	582.217	579.943	578.915	579.030	578.915	579.157	579.157	579.030	578.915	582.445	580.066	582.160
EBP	572.504	580.621	578.766	579.193	578.766	579.022	579.022	579.193	578.766	583.598	580.379	584.834
GR	580.965	580.661	579.228	578.991	579.228	579.270	579.270	578.991	579.228	582.362	580.612	582.889
NARX	581.984	580.127	579.375	579.200	579.375	579.437	579.437	579.200	579.375	582.009	580.317	581.900
SES	582.656	580.659	579.562	578.493	578.045	577.732	577.512	577.058	577.041	581.829	580.380	583.866
SMA	580.500	580.500	580.500	576.500	576.750	576.750	576.750	576.750	580.750	580.750	584.750	584.500
WMA	579.214	577.553	576.388	576.847	576.962	576.990	576.248	576.812	588.953	579.988	588.997	579.249
EXP	610.000	559.000	578.000	575.000	578.000	577.000	577.000	575.000	578.000	609.000	561.000	607.000
HWDES	577.662	577.050	576.084	576.880	576.970	576.982	576.085	576.887	591.378	578.573	590.651	577.580
ARIMA	575.983	576.984	575.983	576.984	576.984	576.984	575.983	576.984	593.000	576.984	591.999	575.983

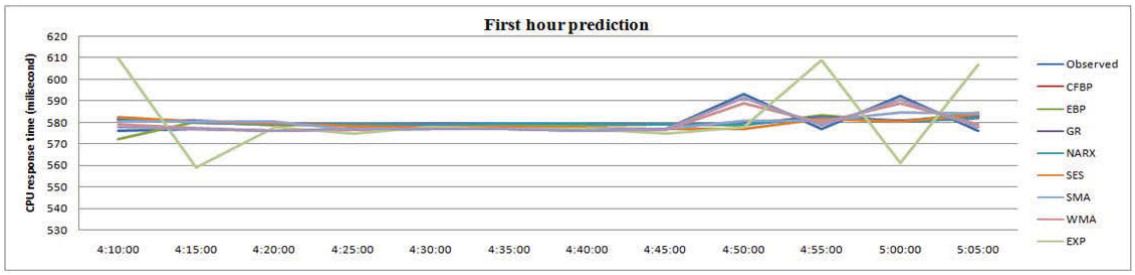


Figure 6.34: Observed and predicted values for the first hour

Table 6.4: Observed and predicted values for the second hour, starts from 5:10:00 to 6:05:00

Time	5:10:00	5:15:00	5:20:00	5:25:00	5:30:00	5:35:00	5:40:00	5:45:00	5:50:00	5:55:00	6:00:00	6:05:00
Observed	577	591	576	577	577	576	592	592	578	576	576	576
CFBP	579.980	578.915	581.885	580.012	578.915	579.157	579.030	581.928	582.541	580.236	579.245	578.786
EBP	582.256	578.766	582.752	581.448	578.766	579.022	579.193	580.040	583.162	580.357	579.771	578.256
GR	580.744	579.228	582.875	580.653	579.228	579.270	578.991	581.493	583.183	580.456	579.213	578.795
NARX	580.518	578.734	582.013	580.402	578.734	578.871	578.607	582.104	582.829	580.956	578.745	578.468
SES	581.506	580.154	583.408	581.186	579.930	579.051	578.136	582.295	585.206	583.045	580.931	579.452
SMA	580.500	584.000	580.000	580.250	580.250	576.500	580.500	584.250	584.500	584.500	580.500	576.500
WMA	577.562	587.641	578.910	577.478	577.119	576.280	588.070	591.017	581.254	577.314	576.328	576.082
EXP	560.000	578.000	605.000	561.000	578.000	577.000	575.000	608.000	592.000	564.000	574.000	576.000
HWDES	577.031	589.573	577.456	577.013	576.967	576.066	590.370	591.947	579.509	576.330	575.982	575.949
ARIMA	576.984	590.998	575.984	576.984	576.984	575.984	591.999	591.999	577.986	575.984	575.984	575.984

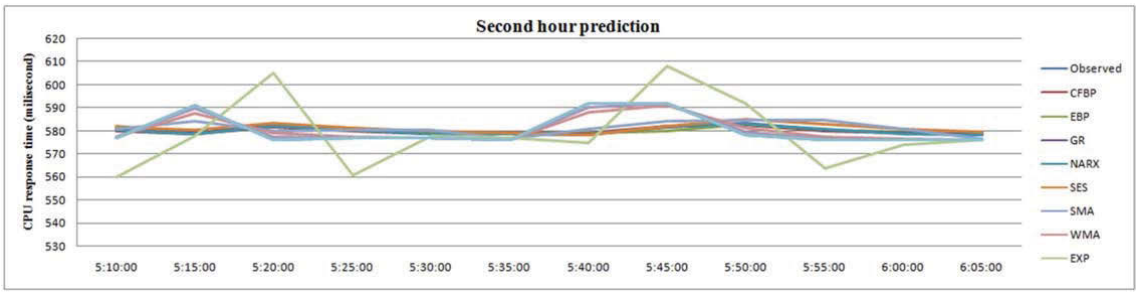


Figure 6.35: Observed and predicted value for the second hour

Table 6.5: Observed and predicted values for the third hour, starts from 6:10:00 to 7:05:00

Time	6:10:00	6:15:00	6:20:00	6:25:00	6:30:00	6:35:00	6:40:00	6:45:00	6:50:00	6:55:00	7:00:00	7:05:00
Observed	577	577	577	577	576	578	576	592	591	593	592	592
CFBP	579.739	579.888	580.089	580.089	580.089	579.942	580.043	580.120	583.117	583.217	583.641	583.162
EBP	578.256	578.766	579.022	579.022	579.022	579.193	579.048	579.771	580.040	581.710	585.342	584.299
GR	578.795	579.228	579.270	579.270	579.270	578.991	579.257	579.213	581.493	583.073	584.185	582.963
NARX	578.468	578.734	578.871	578.871	578.871	578.607	578.995	578.745	582.104	582.889	583.294	581.446
SES	578.416	577.991	577.694	577.486	577.340	576.938	577.257	576.880	581.416	584.291	586.904	588.433
SMA	576.250	576.500	576.750	577.000	576.750	577.000	576.750	580.500	584.250	588.000	592.000	592.000
WMA	576.771	576.943	576.986	576.996	576.249	577.562	576.391	588.098	590.274	592.319	592.080	592.020
EXP	576.000	578.000	577.000	577.000	577.000	575.000	580.000	574.000	608.000	590.000	595.000	591.000
HWDES	576.850	576.954	576.968	576.973	576.076	577.779	576.169	590.393	591.060	592.921	592.215	592.125
ARIMA	576.985	576.985	576.985	576.985	575.984	577.986	575.984	591.999	590.998	593.000	591.999	591.999

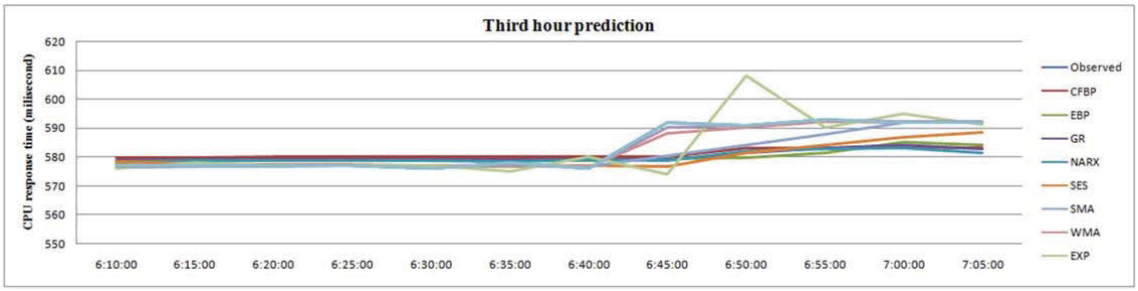


Figure 6.36: Observed and predicted value for the third hour

Table 6.6: Observed and predicted values for the fourth hour, starts from 7:10:00 to 8:05:00

Time	7:10:00	7:15:00	7:20:00	7:25:00	7:30:00	7:35:00	7:40:00	7:45:00	7:50:00	7:55:00	8:00:00	8:05:00
Observed	592	577	576	577	577	576	577	577	577	577	577	576
CFBP	584.077	584.077	581.771	580.503	580.468	580.718	580.503	580.468	580.718	580.718	580.718	580.718
EBP	583.162	583.162	581.232	579.193	578.766	579.022	579.193	578.766	579.022	579.022	579.022	579.022
GR	583.183	583.183	580.614	578.991	579.228	579.270	578.991	579.228	579.270	579.270	579.270	579.270
NARX	583.187	583.187	581.117	578.953	579.032	579.180	578.953	579.032	579.180	579.180	579.180	579.180
SES	589.503	590.252	586.276	583.193	581.335	580.035	578.824	578.277	577.894	577.626	577.438	577.307
SMA	592.250	588.250	584.250	580.500	576.750	576.500	576.750	576.750	576.750	577.000	577.000	576.750
WMA	592.005	580.751	577.188	577.047	577.012	576.253	576.813	576.953	576.988	576.997	576.999	576.250
EXP	592.000	592.000	562.000	575.000	578.000	577.000	575.000	578.000	577.000	577.000	577.000	577.000
HWDES	592.104	578.593	576.198	576.841	576.920	576.035	576.843	576.938	576.953	576.959	576.963	576.067
ARIMA	591.999	576.985	575.984	576.985	576.985	575.984	576.985	576.985	576.985	576.985	576.985	575.984

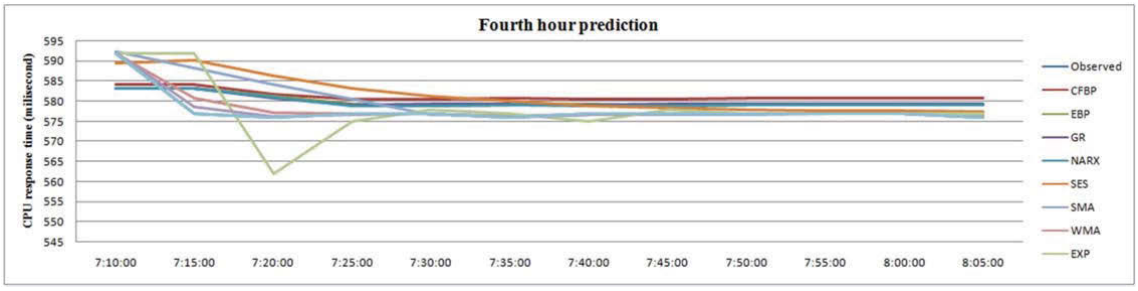


Figure 6.37: Observed and predicted value for the fourth hour

Table 6.7: Observed and predicted values for the fifth hour, starts from 8:10:00 to 9:05:00

Time	8:10:00	8:15:00	8:20:00	8:25:00	8:30:00	8:35:00	8:40:00	8:45:00	8:50:00	8:55:00	9:00:00	9:05:00
Observed	577	576	577	576	576	576	577	577	576	577	576	578

CFBP	578.337	578.310	578.337	578.310	578.337	578.101	578.101	578.310	578.540	578.337	578.310	578.337
EBP	579.193	578.766	579.193	578.766	579.193	578.256	578.256	578.766	579.022	579.193	578.766	579.193
GR	578.991	579.228	578.991	579.228	578.991	578.795	578.795	579.228	579.270	578.991	579.228	578.991
NARX	578.607	578.734	578.607	578.734	578.607	578.468	578.468	578.734	578.871	578.607	578.734	578.607
SES	576.915	576.940	576.658	576.761	576.533	576.373	576.261	576.483	576.638	576.446	576.613	576.429
SMA	576.750	576.500	576.500	576.500	576.250	576.250	576.250	576.500	576.500	576.750	576.500	576.750
WMA	576.812	576.203	576.801	576.200	576.050	576.013	576.753	576.938	576.235	576.809	576.202	577.551
EXP	575.000	578.000	575.000	578.000	575.000	576.000	576.000	578.000	577.000	575.000	578.000	575.000
HWDES	576.871	576.063	576.877	576.069	575.982	575.975	576.877	576.978	576.090	576.893	576.083	577.795
ARIMA	576.985	575.984	576.985	575.984	575.984	575.984	576.985	576.985	575.984	576.985	575.984	577.986

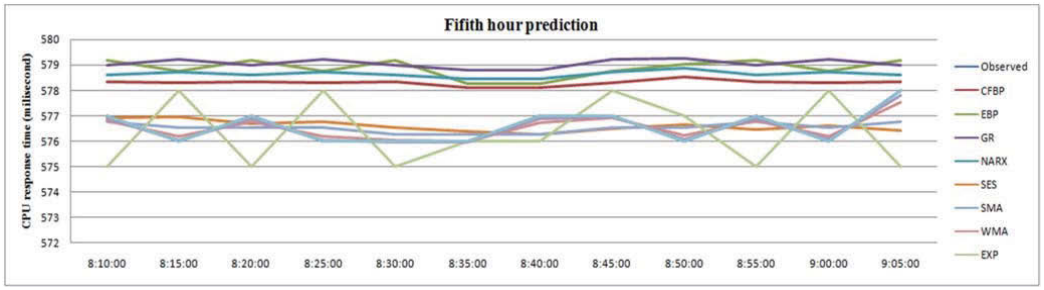


Figure 6.38: Observed and predicted value for the fifth hour

All the 10 prediction methods are compared using RMSE and MAD. This is presented in below Tables 6. 8 and 6. 9 and Figures 6.39 and 6.40.

Table 6.8: RMSE of each prediction method for all five intervals

Root mean square error (RMSE)					
Time	1st hour	2nd hour	3rd hour	4th hour	5th hour
CFBP	6.39483	6.549472	6.629322	6.268062	5.576587
EBP	6.57118	5.673175	6.828433	6.355071	5.86753
GR	6.276842	6.602214	6.629303	6.139847	5.54491
NARX	6.272158	6.570401	6.687622	6.086032	5.544366
SES	6.77886	7.082427	6.764071	6.47442	5.805765
SMA	5.343902	5.659156	5.202664	5.343902	4.446558
WMA	2.156221	2.091894	1.838725	2.156221	1.516133
EXP	19.71674	17.21312	14.69221	19.71674	11.71964
HWDES	1.009489	0.945184	0.819959	1.009489	0.670679
ARIMA	0.014699	0.014291	0.013539	0.014707	0.01418

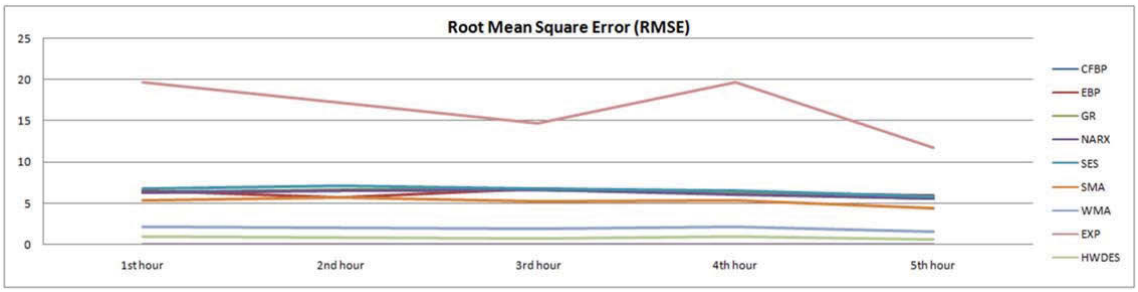


Figure 6.39: RMSE of all prediction methods

Table 6.9: MAD of each prediction method for all five intervals

Mean absolute deviation (MAD)					
Time	1st hour	2nd hour	3rd hour	4th hour	5th hour
CFBP	5.0823	5.327925	5.806919	5.733194	3.980838
EBP	5.197033	6.90745	5.536711	5.107492	4.592362
GR	5.084575	5.488321	5.806919	5.022108	4.449737
NARX	5.112658	5.309883	5.536711	4.954785	4.24715
SES	4.915972	5.755063	5.206801	5.23629	4.034413
SMA	3.854167	4.458333	3.729167	3.328125	2.7625
WMA	1.506637	1.486638	1.179733	1.00468	0.841057
EXP	14.08333	12.79167	9.972222	8.25	6.916667
HWDES	0.690422	0.620878	0.491847	0.422637	0.355741
ARIMA	0.013494	0.012933	0.011781	0.012396	0.01299

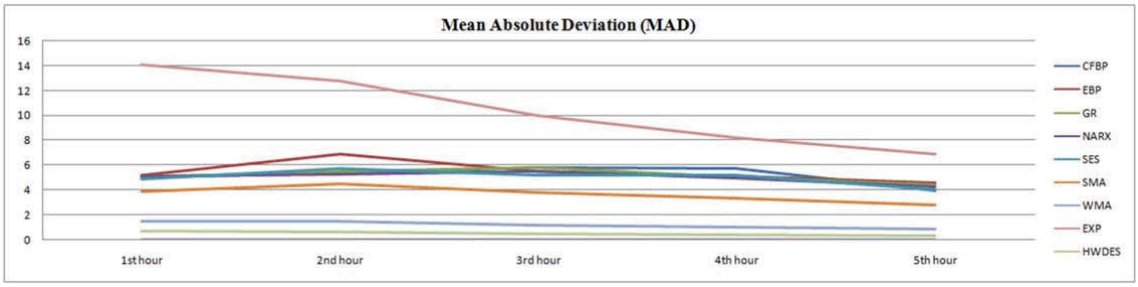


Figure 6.40: MAD of all prediction methods

6.13 Discussion and Comparative Analysis of the Prediction Methods

In this section, a comparative analysis of the all prediction methods is presented, followed by a discussion. In the following subsection, the time series prediction methods are compared with the neural network methods and the stochastic method. In the second following subsection, a time series of six approaches, which are used for Cloud QoS prediction, are compared against two criteria. The first criterion is the overall accuracy of the prediction approaches over the prediction period and the second criterion is freshness. Freshness represents the accuracy of results in the initial time periods of prediction as compared to later ones. As discussed earlier, the accuracy of the prediction approaches decreases with an increase in time so the freshness criterion aims to determine which parameter and variable in the prediction approach gives the most accurate results for the initial time slots. It should be noted that the study only tested the freshness accuracy for the results from the simple exponential smoothing method.

6.13.1 Neural Network, Stochastic and Time Series Prediction Methods

From the comparative analysis of 10 prediction methods used in the previous section, it is observed that among all the prediction methods ARIMA gives the most optimal results for all five time intervals. The Holt-Winter double exponential smoothing method generates the second optimal result. From Figures 6.39 and 6.40, we see that the extrapolation method gives the worst result out of all of the prediction methods because it considers data beyound the range of known data points, therefore it produce noisy results. The prediction value of all the neural network methods are similar to each other with slight variations, however we see that the prediction accuracy of these methods increases at the fifth time interval.

6.13.2 Comparative Analysis of a Time Series of Six Prediction Methods

Method 1, the SES method, is suitable for a dataset that does not follow any pattern. When there is a trend and seasonality in a dataset, its prediction accuracy decreases as observed for the CPU dataset in time periods of 1 week, 2 weeks and 4 weeks, all of which follow a seasonal pattern and therefore their RMSE and MAD are relatively larger than the other datasets. The value of smoothing factor α impacts significantly on prediction accuracy. When the value of α is 0, it is insensitive and when the value of α increases, it become more sensitive. When $\alpha = 1$, it is just one behind the naive forecast, which abruptly changes with a sudden change in the dataset. Depending on the nature of the dataset, different values of α give the optimal result. To analyse the impact of α, we evaluated the nine possible combinations from 0.1 to 0.9 for all 10 types of datasets. From the prediction results, it is observed that there is no specific value of α which produces an optimal result and, for each dataset with its own pattern, different values of α generate an optimal result.

Method 2, the SMA method, is suitable for data that have random variations. The prediction accuracy depends on the size of k (number of entries for mean), therefore the prediction accuracy of each method is analysed with a different number of k. 10 to 11 different values of k are considered to analyse the prediction accuracy, depending on the size of the dataset. The datasets are divided into 10 subsections that start from two entries, i.e. $k = 2$, and then after some time intervals

till the last value in a dataset. For each dataset (time intervals) when the size of k increases, the prediction accuracy decreases because when a shorter time span is used to find the average, it is more sensitive, and when there is any change in a dataset, it changes immediately. The longer the time span, the less sensitive it is, therefore a larger number of k produces more smooth data. Therefore, for each dataset, the smallest number of k, which we set as 2, gives the most optimal result.

Method 3, the WMA method, was used to analyse the prediction accuracy for 10 time intervals with two variable parameters: the number of observation k and increasing factor α. Depending on the size of the dataset, three values of k were selected: the start point with a $k = 2$ or $k = 3$, the mid-value of the dataset $k = m/2$ (m is the total number of data in the dataset) and the end point $k = m-1$ or $k = m-2$. To analyse the impact of the weight factor, the value of α is randomly set as 0.5, 1.2, 1.5, 2, 5 and 10. The value of α in this experiment is the difference in the weight between recent past data and distant past data. This means that when the value of α is 0.5, the weight of every recent past data is 0.5 times greater than the distant past. When α is 10, this means that the weight of every recent past data is 10 times greater than the distant past and the sum of all the weights is equal to 1. Therefore, the value of $\alpha = 10$ gives a higher weight to the most recent data than the value of $\alpha = 0.5$. Therefore, from the prediction results, it is observed that when the value of k and α increases, a better result is obtained because with larger data and assigning a higher weight to recent past data generates an optimal prediction result. It is observed that in some entries, the weight factor is smaller than the smallest non-zero floating-point value in MATLAB, therefore it does not generate any prediction results. This is indicated by N/A in the table.

Method 4, the Exp method, is used to compare the results of the 10 datasets. The results show that a time period of 1 hour for CPU, a time period of 4 weeks for memory and a time period of 1 day for I/O give most optimal prediction results.

Method 5, the HWDES method, was used to analyse prediction accuracy in 10 time intervals with two variable parameters α and β. In each dataset, 9 x 9 = 81 cases were analysed for a set of α and β with values of α as = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 and values of β as = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9. In each dataset, it is observed that with values of $\alpha = 0.9$ and $\beta = 0.1$, the system generates the most optimal prediction result.

Method 6, the ARIMA method, has a wide range of parameters which affects prediction. Eight sets of three parameters (p,d,q) are considered which are equal to (0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0) and (1,1,1). In each dataset, it is observed that the set of p,d,q = (0,1,0) gives the most optimal prediction result.

Based on the optimal prediction result for each prediction method with the optimal parameters, all the six prediction methods are selected and compared with each other to analyse their prediction accuracy. Table 6.10 presents a comparative analysis of the six prediction methods for the 10 time intervals. In the table, MT1 refers to method 1, the simple exponential smoothing method, MT2 refers to the simple moving average method, MT3 refers to the weighted moving average method, MT4 refers to the extrapolation method, MT5 refers to the Holt-Winters double exponential smoothing method and MT6 refers to ARIMA.

Table 6.10: Comparative analysis of six prediction methods and their optimal parameters with the prediction accuracy benchmark

		CPU test				Memory test				Disk test			
Time interval	Method	MSE	RMSE	MAD	at	MSE	RMSE	MAD	at	MSE	RMSE	MAD	at
5 minutes	MT1	5814.78	76.25	9.92	$\alpha = 0.1$	12237648.4	3498.24	694.3	$\alpha = 0.1$	35020	187.14	65.78	$\alpha = 0.1$
	MT2	3669.63	60.58	6.89	k = 2/1919	7432322.47	2726.23	414.9	k = 2/1919	20609.1	143.56	47.05	k = 2/1919
	MT3	0.0014	0.04	0.0033	k = 910/1919, $\alpha = 1.2$	0.36	0.6	0.04	k = 910/1919, $\alpha = 1.2$	30	5.48	0.18	k = 910/1919, $\alpha = 1.2$
	MT4	32468.23	180.19	20.21		65964876.61	8121.88	1186		160781	400.98	132.71	
	MT5	108.53	10.42	1.1	$\alpha = 0.9$, $\beta = 0.1$	220176.23	469.23	67.65	$\alpha = 0.9, \beta = 0.1$	569.37	23.86	7.58	$\alpha = 0.9, \beta = 0.1$
	MT6	0.01	0.09	0.04	p = 0, d = 1, q = 0	62.21	7.89	0.81	p = 0, d = 1, q = 0	0.23	0.48	0.16	p = 0,d = 1, q = 0
10 minutes	MT1	43.57	6.6	4.57	$\alpha = 0.2$	7008225.09	2647.31	652.5	$\alpha = 0.6$	21917.4	148.05	62	$\alpha = 0.3$
	MT2	24.57	4.96	3.22	k = 2/960	4156421.14	2038.73	447.6	k = 2/960	12400.3	111.36	44.03	k = 2/960
	MT3	0.0009	0.0303	0.0026	k = 950/960, α=2	0.19	0.44	0.04	k = 950/960, α= 2	14.5	3.81	0.16	k = 950/960, $\alpha = 2$
	MT4	201.13	14.18	9.18		35255761.54	5937.66	1242		96362.2	310.42	122.31	
	MT5	0.7	0.83	0.51	$\alpha = 0.9, \beta = 0.1$	119974.63	346.37	71.79	$\alpha = 0.9, \beta = 0.1$	341.76	18.49	6.97	$\alpha = 0.9, \beta = 0.1$
	MT6	0.0042	0.07	0.03	p = 0, d = 1, q = 0	142.41	11.93	1.59	p = 0, d = 1, q = 0	0.46	0.67	0.26	p = 0, d = 1, q = 0
20 minutes	MT1	27.15	5.21	3.69	$\alpha = 0.4$	4395116.12	2096.45	707	$\alpha = 0.1$	15148.1	123.08	59.5	$\alpha = 0.5$
	MT2	14.41	3.8	2.59	k = 2/481	2478822.13	1574.43	474.4	k = 2/481	7956.57	89.2	40.4	k = 2/481
	MT3	0.45	0.67	0.34	k = 240/481, $\alpha = 10$	0.09	0.29	0.02	k = 240/481, α = 10	12.73	3.57	0.18	k = 240/481, α = 10
	MT4	111.68	10.57	7.1		21669336.8	4655.03	1366		70749.7	265.99	119.27	

		CPU test				Memory test				Disk test			
Time interval	Method	MSE	RMSE	MAD	at	MSE	RMSE	MAD	at	MSE	RMSE	MAD	at
	MT5	0.4	0.63	0.42	α = 0.9, β = 0.1	72892.21	269.99	77.45	α = 0.9, β = 0.1	235.86	15.36	6.66	α = 0.9, β = 0.1
	MT6	0.01	0.07	0.04	p = 0, d = 1, q = 0	294.36	17.16	3.12	p = 0, d = 1, q = 0	1.23	1.11	0.47	p = 0, d = 1, q = 0
1 hour	MT1	15.39	3.92	2.81	α=0.4	1952440.66	1397.3	763.7	α=0.1	8274.6	90.96	57.91	α=0.7
	MT2	7.65	2.77	2.01	k = 2/161	1268127.97	1126.11	533.7	k = 2/161	4087.48	63.93	40.53	k = 2/161
	MT3	0.004	0.0632	0.0081	k = 155/161, α = 10	497.57	22.31	2.64	k = 155/161, α = 10	2.05	1.43	0.23	k = 155/161, α = 10
	MT4	58.92	7.68	5.74		10551904.78	3248.37	1533		24605.7	156.86	95.45	
	MT5	0.21	0.46	0.33	α = 0.9, β = 0.1	36283.73	190.48	85.07	α = 0.9, β = 0.1	99.15	9.96	6.08	α = 0.9, β = 0.1
	MT6	0.01	0.1	0.07	p = 0, d = 1, q = 0	588.84	24.27	6.99	p = 0, d = 1, q = 0	4.62	2.15	1.13	p = 0, d = 1, q = 0
4 hours	MT1	413.31	20.33	5.5	α = 0.2	466636.37	683.11	481.5	α = 0.1	78195.8	279.64	99.57	α = 0.5
	MT2	231.35	15.21	3.84	k = 2/186	333686.64	577.66	418.2	k = 2/186	45471.9	213.24	70.33	k = 2/186
	MT3	0.0087	0.0934	0.0114	k=180/181,α=10	252.55	15.89	2.21	k=180/181,α=10	0.81	0.9	0.15	k=180/181,α=10
	MT4	2174.31	46.63	10.55		2894397.14	1701.29	1263		408213	638.92	191.24	
	MT5	7.07	2.66	0.58	α=0.9,β=0.1	9743.62	98.71	68.08	α=0.9,β=0.1	1355.68	36.82	10.86	α=0.9,β=0.1
	MT6	0.2	0.45	0.2	p=0,d=1,q=0	138.59	11.77	5.14	p=0,d=1,q=0	16.99	4.12	1.4	p=0,d=1,q=0
12 hours	MT1	201.68	14.2	6.21	α=0.4	126211.77	355.26	239.25	α=0.1	30755.31	175.37	94.4	α=0.3
	MT2	93.42	9.67	4.33	k=2/62	60694.92	246.36	184.68	k=2/62	13498.48	116.18	63.74	k=2/62
	MT3	0.14	0.37	0.07	k=60/62, α=10	79.68	8.93	1.99	k=60/62, α=10	0.73	0.85	0.14	k=60/62, α=10
	MT4	767.32	27.7	11.26		794249.5	891.21	648.85		139100.21	372.96	168.63	

		CPU test				Memory test				Disk test			
Time interval	Method	MSE	RMSE	MAD	at	MSE	RMSE	MAD	at	MSE	RMSE	MAD	at
	MT5	2.65	1.63	0.68	α=0.9, β=0.1	2302.08	47.98	34.98	α=0.9, β=0.1	436.95	20.9	10.13	α=0.9, β=0.1
	MT6	0.36	0.6	0.29	p=0, d=1, q=0	273.75	16.55	10.22	p=0, d=1, q=0	46.28	6.8	2.75	p=0, d=1, q=0
1 day	MT1	90260.24	300.43	159.65	α=0.1	36542.23	191.16	89.91	α=0.1	20.36	4.51	0.9	α=0.9
	MT2	63670.95	252.33	145.37	k=2/31	21788.12	147.61	76.81		7.28	2.7	0.67	
	MT3	12.36	3.52	0.85	k=30/31, α=10	21.62	4.65	1.09	k=30/31, α=10	1.69E-28	1.30E-14	1.19E-14	k=30/31, α=10
	MT4	456537.19	675.68	448.61		170563.84	412.99	203.19		20.16	4.49	0.81	
	MT5	1718.08	41.45	27.15	α=0.9, β=0.1	602.19	24.54	12.04	α=0.9, β=0.1	0.21	0.46	0.16	α=0.9, β=0.1
	MT6	348.94	18.68	7.09	p=0, d=1, q=0	135.58	11.64	3.76	p=0, d=1, q=0	0.81	0.9	0.7	p=0, d=1, q=0
1 week	MT1	2258.07	47.52	22.42	α=0.9	46576.34	215.82	162.57	α=0.5	18485.61	135.96	97.02	α=0.1
	MT2	1359.15	36.87	17.71	k=2/26	21834.27	147.76	105.73	k=2/26	10647.83	103.19	76.22	k=4/26
	MT3	0	0.02	0.01	k=26/26, α=10	0.1	0.32	0.07	k=26/26, α=10	0.56	0.75	0.21	k=26/26, α=10
	MT4	18698.78	136.74	76.65		184055.43	429.02	334.44		229356.71	478.91	277.84	
	MT5	92.95	9.64	4.71	α=0.9, β=0.1	687.54	26.22	18.71	α=0.9, β=0.1	1052.61	32.44	17.37	α=0.9, β=0.1
	MT6	69.39	8.33	5.87	p=0, d=1, q=0	170.71	13.07	9.26	p=0, d=1, q=0	7.93	2.82	1.84	p=0, d=1, q=0
	MT1	1747.38	41.8	17.86	α=0.9	20191.53	142.1	110.22	α=0.6	7890.94	88.83	69.76	α=0.4
	MT2	1120.64	33.48	13.81	k=2/105	8325.92	91.25	68.08	k=2/105	2867.03	53.54	42.06	k=2/105

		CPU test				Memory test				Disk test			
Time interval	Method	MSE	RMSE	MAD	at	MSE	RMSE	MAD	at	MSE	RMSE	MAD	at
2 weeks	MT3	0.01	0.11	0.02	k=104/105, α=10	2.71	1.65	0.21	k=104/105, α=10	0.04	0.2	0.03	k=104/105, α=10
	MT4	2298.49	47.94	27.45		68895.15	262.48	213.17		43696.96	209.04	155.42	
	MT5	17.88	4.23	2.12	α=0.9, β=0.1	232.72	15.26	11.83	α=0.9, β=0.1	119.22	10.92	8.37	α=0.9, β=0.1
	MT6	34.89	5.91	4.07	p=0, d=1, q=0	161.28	12.7	8.49	p=0, d=1, q=0	19.03	4.36	2.75	p=0, d=1, q=0
4 weeks	MT1	3736.51	61.13	25.84	α=0.9	20753.18	144.06	97.09	α=0.9	3900.34	62.45	55.88	α=0.9
	MT2	2367.97	48.66	22.35	k=2/53	12359.18	111.17	76.19	k=2/53	2312.63	48.09	42.5	k=2/53
	MT3	0.01	0.09	0.02	k=52/53, α=10	0.27	0.52	0.1	k=52/53, α=10	2.11	1.45	0.25	k=52/53, α=10
	MT4	3945.95	62.82	33.12		51464.27	226.86	177.88		26003.22	161.26	114.22	
	MT5	29.53	5.43	2.68	α=0.9, β=0.1	229.07	15.13	11.18	α=0.9, β=0.1	83.1	9.12	7.2	α=0.9, β=0.1
	MT6	51.16	7.15	4.86	p=0, d=1, q=0	205.64	14.34	9.98	p=0, d=1, q=0	15.78	3.97	2.93	p=0, d=1, q=0

By examining Table 6.10, it can be seen that the weighted moving average method and the ARIMA method give the most optimal prediction results at different time intervals. The Holt-Winters double exponential smoothing method generates the second best optimal prediction results, followed by the simple moving average method. The simple exponential smoothing method and the extrapolation method give the least optimal prediction results. The prediction accuracy of each method for each time interval is arranged in ascending order in Table 6.11.

Table 6.11: Optimal prediction algorithm at different time intervals

Time interval	CPU		Memory		I/O	
	Prediction accuracy order	Prediction method	Prediction accuracy order	Prediction method	Prediction accuracy order	Prediction method
5 minutes	1	WMA	1	WMA	1	ARIMA
	2	ARIMA	2	ARIMA	2	WMA
	3	HWDES	3	HWDES	3	HWDES
	4	SMA	4	SMA	4	SMA
	5	SES	5	SES	5	SES
	6	EXP	6	EXP	6	EXP
10 minutes	1	WMA	1	WMA	1	ARIMA
	2	ARIMA	2	ARIMA	2	WMA
	3	HWDES	3	HWDES	3	HWDES
	4	SMA	4	SMA	4	SMA
	5	SES	5	SES	5	SES
	6	EXP	6	EXP	6	EXP
20 minutes	1	ARIMA	1	WMA	1	ARIMA
	2	HWDES	2	ARIMA	2	WMA
	3	WMA	3	HWDES	3	HWDES
	4	SMA	4	SMA	4	SMA
	5	SES	5	SES	5	SES
	6	EXP	6	EXP	6	EXP
1 hour	1	WMA	1	WMA	1	WMA
	2	ARIMA	2	ARIMA	2	ARIMA
	3	HWDES	3	HWDES	3	HWDES
	4	SMA	4	SMA	4	SMA
	5	SES	5	SES	5	SES
	6	EXP	6	EXP	6	EXP

Time interval	CPU		Memory		I/O	
	Prediction accuracy order	Prediction method	Prediction accuracy order	Prediction method	Prediction accuracy order	Prediction method
4 hours	1	WMA	1	ARIMA	1	WMA
	2	ARIMA	2	WMA	2	ARIMA
	3	HWDES	3	HWDES	3	HWDES
	4	SMA	4	SMA	4	SMA
	5	SES	5	SES	5	SES
	6	EXP	6	EXP	6	EXP
12 hours	1	WMA	1	WMA	1	WMA
	2	ARIMA	2	ARIMA	2	ARIMA
	3	HWDES	3	HWDES	3	HWDES
	4	SMA	4	SMA	4	SMA
	5	SES	5	SES	5	SES
	6	EXP	6	EXP	6	EXP
1 day	1	WMA	1	WMA	1	WMA
	2	ARIMA	2	ARIMA	2	HWDES
	3	HWDES	3	HWDES	3	ARIMA
	4	SMA	4	SMA	4	SMA
	5	SES	5	SES	5	SES
	6	EXP	6	EXP	6	EXP
1 week	1	WMA	1	WMA	1	WMA
	2	ARIMA	2	ARIMA	2	ARIMA
	3	HWDES	3	HWDES	3	HWDES
	4	SMA	4	SMA	4	SMA
	5	SES	5	SES	5	SES
	6	EXP	6	EXP	6	EXP
2 weeks	1	WMA	1	WMA	1	WMA
	2	HWDES	2	ARIMA	2	ARIMA
	3	ARIMA	3	HWDES	3	HWDES
	4	SMA	4	SMA	4	SMA
	5	SES	5	SES	5	SES
	6	EXP	6	EXP	6	EXP
4 weeks	1	WMA	1	WMA	1	WMA
	2	HWDES	2	ARIMA	2	ARIMA
	3	ARIMA	3	HWDES	3	HWDES
	4	SMA	4	SMA	4	SMA
	5	SES	5	SES	5	SES

Time interval	CPU		Memory		I/O	
	Prediction accuracy order	Prediction method	Prediction accuracy order	Prediction method	Prediction accuracy order	Prediction method
	6	EXP	6	EXP	6	EXP

6.14 Comparison of the Prediction Accuracy of the Simple Exponential Smoothing Method According to the Freshness Criterion

This section presents the observations on the best-input parameters to use for QoS prediction using the SES method when the goal is to include freshness in the prediction. To achieve this, the dataset is divided into sets of input values that range from 100 to 500 in sets of 1-100, 1-200, 1-300, 1-400 and 1-500. For each input value, first the error value at time slot t1 to t10 is determined and then the change in the error value over the time slots is plotted as a percentage of deviation with respect to the error value observed at time slot t1.

The plot of the error value over the nine time slots (future intervals) taking the inputs of 1-100 for different values of alpha is shown in Figure 6.41. From the figure, it is observed that when the deviation in error in time slot t2 exceeds 0.5%, then the value of alpha which gives the highest positive deviation (that is, shows an improvement in the prediction results) from the error observed in the first time slot results in having the best sustained prediction in the future time slots from time slot t1 (100 in the figure). This observation holds true when datasets 1-200 are considered and predict the future QoS values by using the SES method, as shown in Figure 6.42. From this figure, it is noted that even though the alpha value of 0.9 leads to the highest possible positive deviation, exceeding 0.5%, it goes into the negative region between time slots t2 and t3 but it is the first value to come back in the positive region. Thus, when the freshness of the prediction results is more important, then using the value of alpha which gives the largest deviation, exceeding more than 0.5% in time slot t2, ensures the most optimal results.

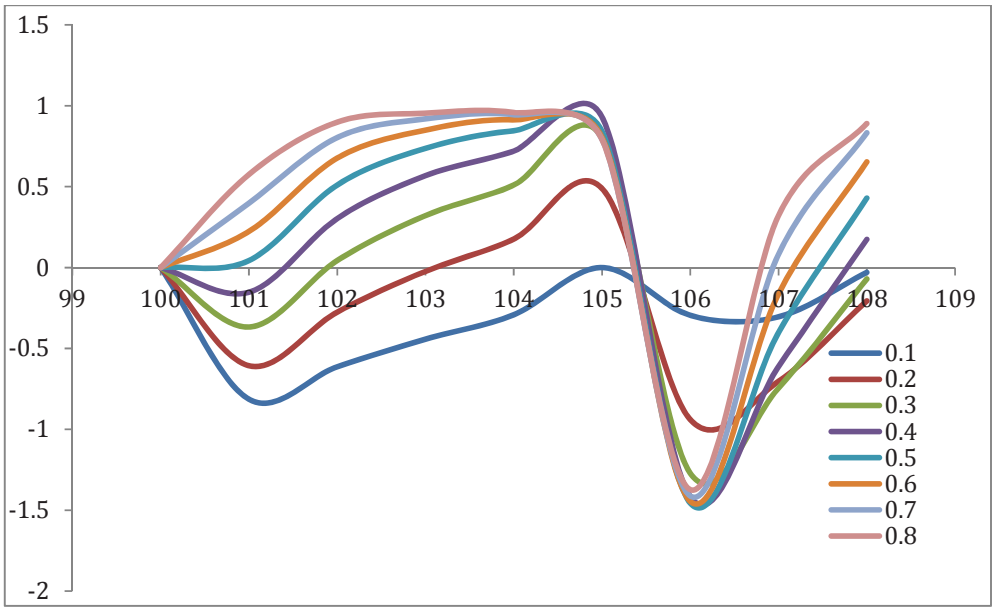


Figure 6.41: Prediction error for predicting nine future intervals (100-108) by taking CPU data with values of 1-100.

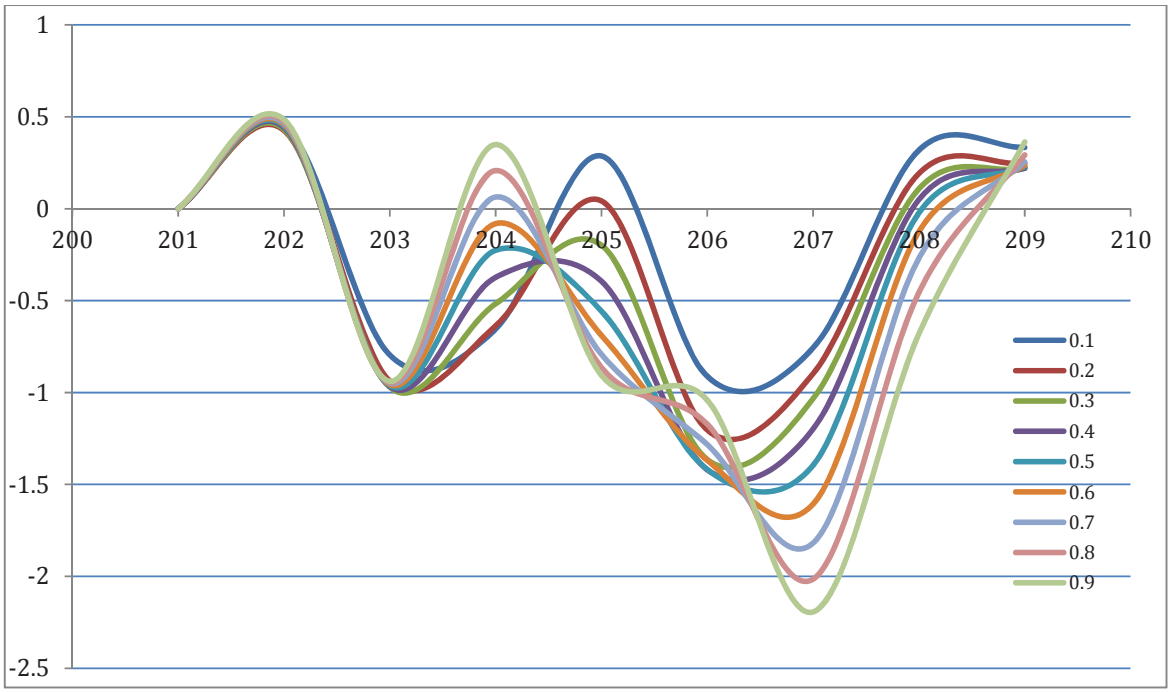


Figure 6.42: Prediction error for predicting nine future intervals (201-209) by taking CPU data with values 1-200.

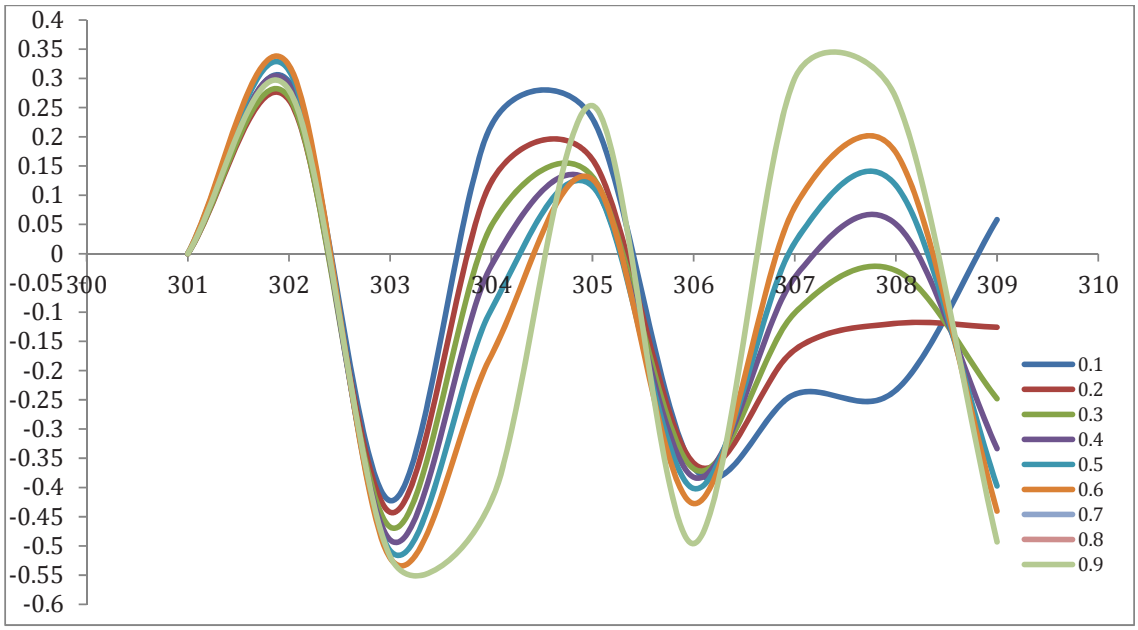


Figure 6.43: Prediction error for predicting nine future intervals (301-309) by taking CPU data with values 1-300.

Figure 6.43 shows the prediction results over the dataset with inputs 1-300. Unlike the previous two cases, it can be noted that the deviation of change in the error in time slot t2 is less than 0.5 so the observation made earlier does not hold in this case. However, it is noted that the alpha value that results in the lowest deviation in time slot t2 gives a prediction result that stays in the positive range over the predicted time slots, as shown in Figure 6.44. The curve for alpha value 0.2 shows the predicted error being in the positive range for the longest period of time over which the prediction is done.

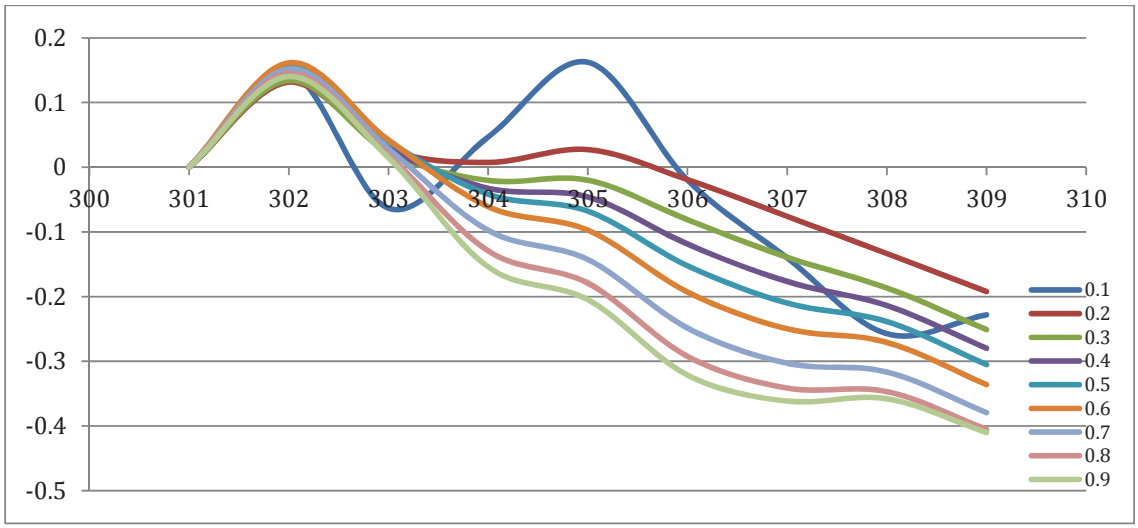


Figure 6.44: Averaged error over the predicted time slots for dataset 1-300

From the input data of 1-400 and 1-500, a common pattern is observed. When the input data is non-cyclic and the deviation is not more than 0.5%, the alpha value that gives the highest positive change will result in a prediction value that has a sustained increase in the predicted accuracy as the time slots increase, as shown in Figures 6.45 and 6.46, which illustrate the deviation of error in time slot t2 with respect to time slot t1 for datasets 1-400 and 1-500 respectively. Figures 6.47 and 6.48 show the averaged error over the predicted time slots for datasets 1-400 and 1-500, respectively.

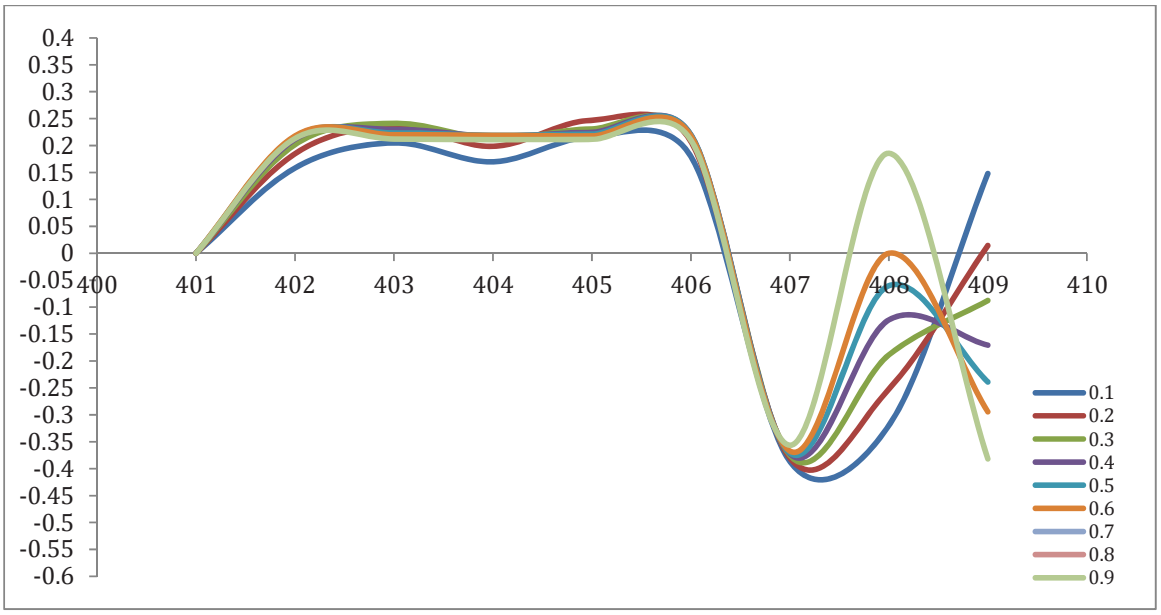


Figure 6.45: Prediction error for predicting nine future intervals (401-409) by taking CPU data with values 1-400.

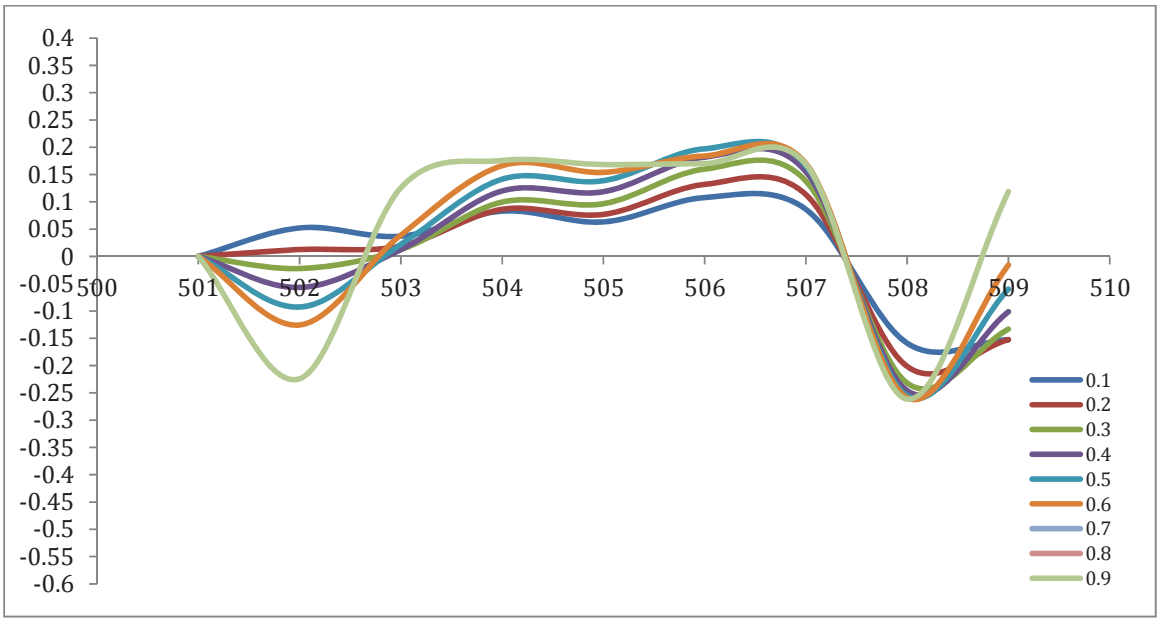


Figure 6.46: Prediction error for nine predicting future intervals (501-509) by taking CPU data with values 1-500 values

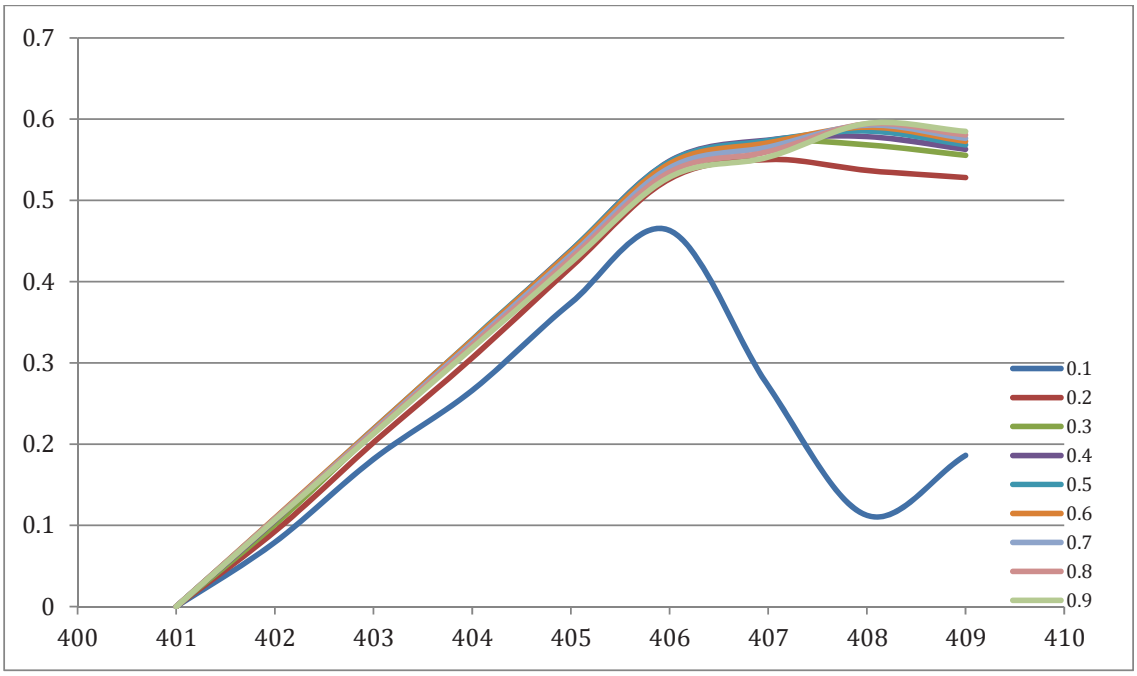


Figure 6.47: Averaged error over the predicted time slot for dataset 1-400

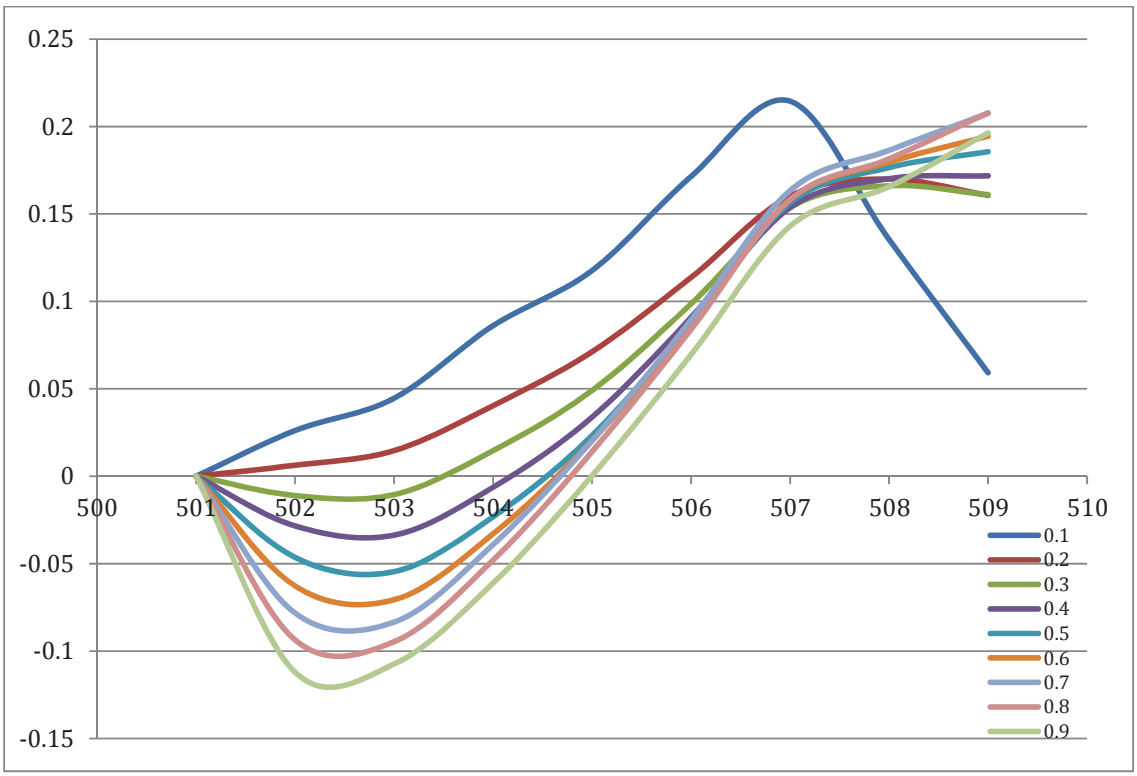


Figure 6.48: Averaged error over the predicted time slot for dataset 1-500

6.15 Conclusion

QoS prediction is an essential element in SLA management for predicting service violations. It is crucial that a service provider understands the likely behaviour of

a service consumer and must know when to take appropriate remedial action once it detects a possible service violation. Different prediction methods produce outputs with different accuracy, depending on the nature of the dataset. The right choice of a prediction method helps service providers to manage their risk and, in the case of a violation, take remedial action in quickest possible time. In this chapter, the neural network, stochastic and time series prediction methods were selected and evaluated on three QoS parameters using time series datasets from a real cloud provider. The comparative analyses showed that the ARIMA method gives the most optimal results at different time intervals.

The next chapter discusses the risk management module and describes how it helps the service provider to manage the risk of SLA violation.

7 Risk Management in the Post-interaction Time-phase

7.1 Introduction

Risk is a vital factor that cannot be ignored by any of the stakeholders in a business. When business transactions are seen from a cloud perspective, where consumers can request resources at any time and have the illusion that an infinite number of resources are dedicated to them, then the need to manage and mitigate the risks of SLA violation due to lack of resources becomes an utmost priority for small and medium providers. Risk is a ubiquitous factor associated with many business transactions that is clearly evident because of its negative occurrence due to external or internal vulnerabilities (Hussain et al. 2007). Risk has the capacity to change the results of an interaction in an adverse way; therefore, it is very crucial to manage risk before it causes negative results. Risk is a primary element to consider when deciding whether or not to continue with a transaction or terminate it. In the viable SLA management framework proposed in this research, when both parties (the service provider and the consumer) execute an SLA, then a prediction module predicts the likely resource usage behaviour of a consumer. As discussed in earlier chapters, when the predicted results exceed a threshold value, then a risk management module is activated that considers the risk attitude of a provider, the reliability of a consumer and the predicted trajectory to determine whether or not to accept a risk and the appropriate action to manage it.

In this chapter, the third section of the proposed viable SLA management framework is discussed and a risk management module (RMM) for avoiding SLA violations in the post-interaction time phase is proposed. The RMM is comprised of three modules, which consider the reliability of the consumer, the risk attitude of the provider and the predicted trajectory of deciding on an appropriate action to mitigate and manage the risk of SLA violation. The RMM notifies the service provider of a likely service violation and generates recommendations for appropriate early remedial actions to avoid SLA violations.

As discussed in Chapter 2, there are number of existing literatures that identifies the risks, vulnerabilities and necessary related actions for risk management in a cloud computing environment. However, there are still many gaps, particularly

from the cloud provider's perspective, that need to be filled. When a small or medium-sized cloud provider predicts a likely service violation, they still need a decision-making system to help the service provider to take appropriate risk management actions by considering the reliability of the consumer, the provider's attitude toward risk and the predicted trajectory of resource usage.

The chapter is organized as follows. Section 7.2 describes the proposed approach while Section 7.3 discusses using FIS for managing risks. Section 7.4 validates the proposed approach and Section 7.5.

7.2 Risk Management Module (RMM) in the Post-interaction Time Phase

RMM is the last module in the proposed viable SLA management framework. It is situated in post-interaction time phase. The RMM module works with the threshold formation module, runtime QoS monitoring module and QoS prediction module to identify, estimate and mitigate the risk of SLA violations in the post-interaction time phase. The working of modules in the post-interaction phase is presented in Figure 7.1.

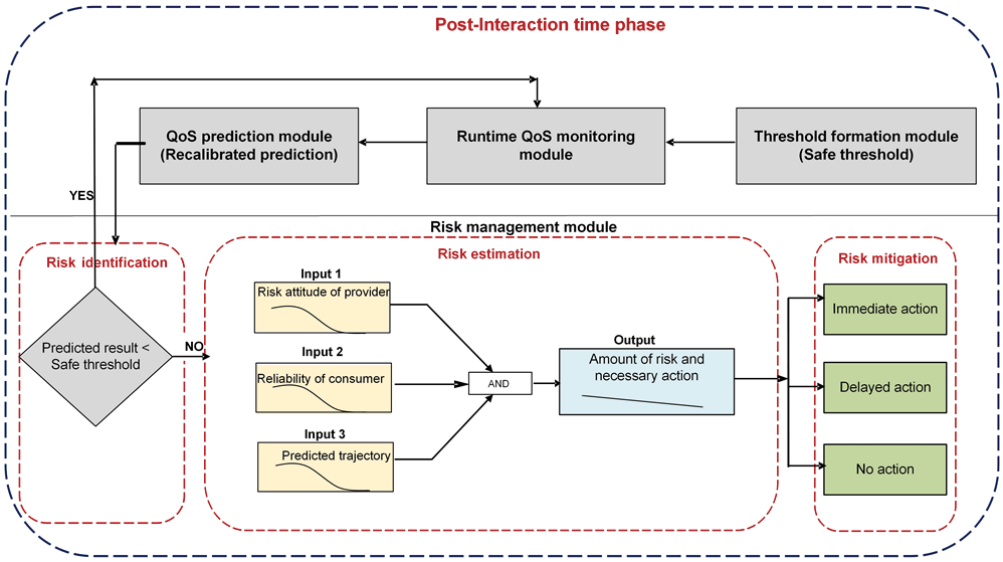


Figure 7.1: Modules in the post-interaction time phase

A description of each module is presented in the following subsections.

7.2.1 Threshold Formation Module (TFM)

The threshold formation module (TFM) is the first module in the post-interaction phase and it is responsible for forming a threshold. As discussed in earlier chapters, in this framework two thresholds were proposed for SLA violation detection and mitigation: agreed threshold (T_a) and safe threshold (T_s). These are summarized as follows.

- *Agreed threshold* (T_a): This threshold is described in the SLA and is mutually agreed on by the consumer and the provider. When both parties have finalized their SLA, they agree on certain thresholds for each SLO and QoS parameter. If a service provider does not comply with the agreed QoS parameters, it commits a service violation and is liable for violation penalties.
- *Safe threshold* (T_s): To avoid service violations and penalties, a safe threshold is proposed that is stricter than the agreed threshold. It is a customized threshold defined by the provider to alarm it, and invoke the risk management module to take the necessary action to avert risks of possible SLA violation when a runtime QoS reaches or exceeds the threshold.

The notion of T_s and T_a is clear from the following example: Let a provider and consumer agree on 80% availability of a resource (memory). The 80% availability of the memory is a T_a value, agreed by both parties, which is also defined in the SLA. However, for service management and SLA violation abatement a provider defines its customized threshold for the memory, say 90%, which is a T_s value for the provider. When the runtime availability of the memory falls below 90%, the framework alerts the service provider and activates the risk management module to manage any risk of T_a value violation.

7.2.2 Runtime QoS Monitoring Module (RQoSMM)

The runtime QoS monitoring module (RQoSMM) is responsible for monitoring the runtime QoS parameters of each agreed SLO. The module obtains the runtime QoS parameters and sends them to the QoS prediction module where they are used to recalibrate the QoS prediction results.

7.2.3 QoS Prediction Module (QoSPM)

As discussed in Chapter 6, the QoS prediction module (QoSPM) is a key module in risk management architecture that is responsible for predicting the resource usage of consumers and the QoS parameters for future intervals. The module considers each consumer and, by using an optimal prediction algorithm, predicts the likely resource usage and QoS parameters for future time intervals. The evaluation results in Chapter 6 demonstrate that an optimal prediction result is obtained by considering small intervals for prediction using an ARIMA method. The accuracy of a prediction result is enhanced by considering the value of the RQoSMM from previous intervals. For example, a prediction algorithm considers a previous observation from time interval 1 to time interval 10 (t1 – t10) to predict the time interval 11 to 14 (t11 – t14), then for the prediction of the next intervals t15 to t18, the algorithm considers the RQoSMM data up to t14. By doing so, a provider is able to obtain an optimal prediction result that is closely related to observed data with minimum deviation (Hussain, Hussain & Hussain 2016b). The pseudocode of QoSPM is presented below:

```
for (i=start limit; i <= endlimit ; i++ )
 if (RQoSMM is empty)
 input[i]= prev_observation[i];
 else
 input[i]= RQoSMM[i]+ prev_observation[i];
Pred_output= Prediction_algo(input);
```

The algorithm starts by ensuring that the runtime data is available. If a transaction has just started and runtime QoS data is not available, a prediction module considers the consumer's previous data from the identity manager module (IMM) (Hussain et al. 2015) as described in Chapter 5 of the thesis. The output of the QoSPM is compared against a T_s value. If a predicted result reaches or exceeds the T_s value, a risk management module is activated to take appropriate action to mitigate the risk of SLA violation.

7.2.4 Risk Management Module (RMM)

The RMM is activated when the recalibrated prediction results exceed the T_s value. The module is responsible for identifying possible risks, estimating the severity of a risk, and calculating the ways to manage it. The RMM is comprised

of three sub-modules: the risk identification module, the risk estimation model and the risk mitigation module. These are discussed in the following subsections.

7.2.4.1 Risk Identification Module

The risk identification module compares the QoSPM value with the T_s value obtained from the TFM. If the value of the QoSPM is less than the T_s value, it takes no action and continues to monitor the SLA; however, if the module finds that the value of the RQoSMM is greater than the T_s value, the risk estimation module is activated to evaluate and estimate possible risk.

7.2.4.2 Risk Estimation Module

The risk estimation module is responsible for estimating the impact of possible risk. The risk estimation decision depends on the risk attitude of the provider, the reputation of the consumer and the transaction trend curve for future intervals. The risk attitude of a provider is the provider's capacity to deal with risk. The risk attitude of a provider can be risk averse, risk neutral or risk taking. A provider with a risk propensity of being risk averse is more reluctant to take a risk than a provider with an attitude that is risk neutral or risk taking (Hussain et al. 2016). The reputation of a consumer is its reliability or trust value, which is calculated from previous successful transactions. A consumer is grouped into one of three classes – bronze, silver or gold – according to their reputation. The calculation of a consumer's reliability is described in Chapter 5 of this thesis. A third input is the transaction trend for future intervals. This is the prevailing tendency or inclination of resources that will be used by the consumer at a future date. This can be determined by observing the curve of the QoSPM with respect to the thresholds. When the curve of a QoSPM exceeds the T_s value, one of two possible conditions for future intervals will apply:

- The recalibrated predicted result is moving towards $T_{s;}$ or
- The recalibrated predicted result is moving away from T_s.

A risk estimation module considers the above-mentioned three inputs and uses FIS to determine the appropriate action to take to manage and mitigate the risk.

7.2.4.3 Risk Mitigation Module

The output of the risk estimation module is high risk, medium risk or low risk. Depending on the type of risk, the provider chooses an appropriate action to manage and mitigate the possible risk. When a risk is assessed as high, a module sends an alarm to the service provider for immediate action. The service provider stops taking new requests and arranges deficient resources in the fastest possible time to avoid service violations. In this case, the provider rejects the risk and tries to remove it in the fastest possible time. When the risk is estimated as medium or low, the FIS decides whether to take delayed action or no action, depending on the input values. When a module alerts a provider to a delayed action and the provider has accepted a risk but keeps it under observation, it means there is a greater chance that if the condition remains the same, it is likely to lead to a service violation. In this case, the provider takes the necessary action but within a certain time frame. The provider arranges deficient resources within a certain time period. When the risk is estimated as a low, it has no significant effect on the provider. The provider accepts the risk and does not take any action. The decision on risk mitigation depends on the output of the FIS, which is discussed in the next section of this chapter.

7.3 FIS for Managing Risk

To determine the amount of risk, a Mamdani-type FIS is used. The inputs are the risk attitude of the provider, the reputation of the consumer and the predicted trajectory, and the output is an action to mitigate a risk. Variations in these input values can change the output. The provider defines the risk acceptability (RA) value. To accept a risk, the output of the FIS should be greater than the RA value. If the output of the FIS is below the RA value, then the provider rejects the risk and takes immediate action to remove it. In terms of accepting a risk, the FIS also helps a service provider to take delayed action or no action.

The three conditions – rejecting a risk, partially accepting a risk and fully accepting a risk – can be understood by the following example. A provider defines its RA value as 50%. Let us assume there are three consumers – C1, C2 and C3. The FIS output obtained for these three consumers is 37%, 56% and 90% respectively. The FIS output for C1 (37%) is less than the RA value (50%). In this case, the RMM alerts the service provider to take immediate action to handle the risk. The output for both C2 and C3 is greater than the risk acceptability value,

therefore the module decides to accept the risk. Even though in both cases the FIS value is greater than the RA value, which is 50%, the provider is more confident that risk is less likely to occur in the case of C2. In the case of C2, however, the provider's concern is that if the condition remains the same, it may lead to a service violation. Therefore, based on the FIS output, the module decides whether to categorize the risk as a medium risk or a low risk, and to take delayed action or no action. The arrangement of inputs and outputs at the two levels in FIS is shown in Figure 7.2.

The pseudocode of the algorithm is presented below:

```
FIS-out= FIS (risk attitude of provider, reliability of consumer, predicted
trajectory);
        If   FIS-out >= RA value
             Accept risk;
             Depending on FIS-out value
             Take delayed action or no action.
        Else
             reject a risk and take immediate action.
```

The FIS and its membership function is described in Sections 7.4.1 and 7.4.2 of this chapter.

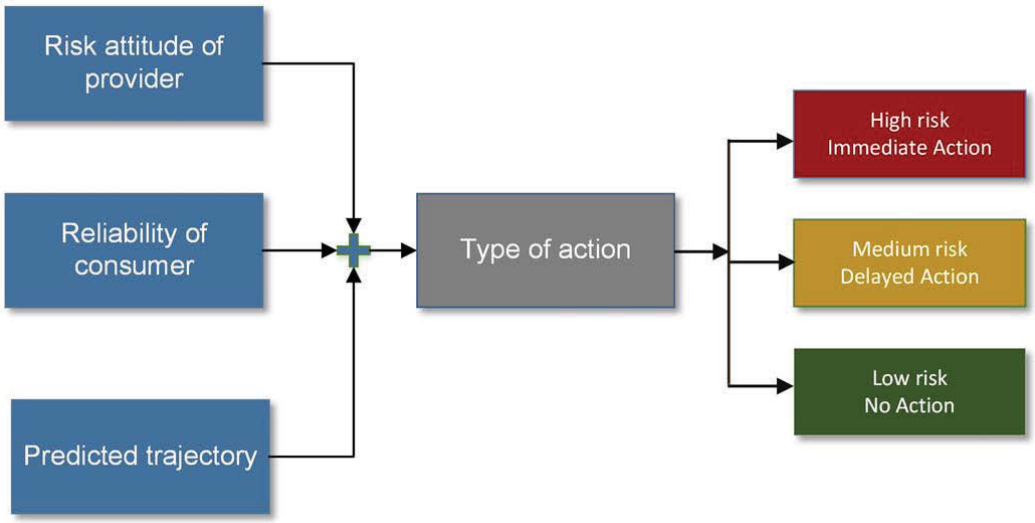


Figure 7.2: FIS for risk management

7.3.1 Fuzzy Set for First Input – Risk Attitude of the Provider

The risk attitude of the provider describes the propensity for the service provider to take a risk. Depending their risk attitude, service provider may be risk averse,

risk neutral or risk taking. A risk averse provider attempts to avoid any risk, whether it is small or large. A risk neutral provider takes the middle ground; depending on the nature of the risk, the service provider may decide to take action or to ignore the risk. A risk taking provider has a bold attitude and ignores small risks; it only takes action for risks that have a significant effect.

The numerical range for this input is 1 to 5: the risk averse range is 1 to 3; the risk neutral range is 1 to 5, and the risk taking range is 3 to 5. The membership function for the risk attitude of a provider is presented in Figure 7.3.

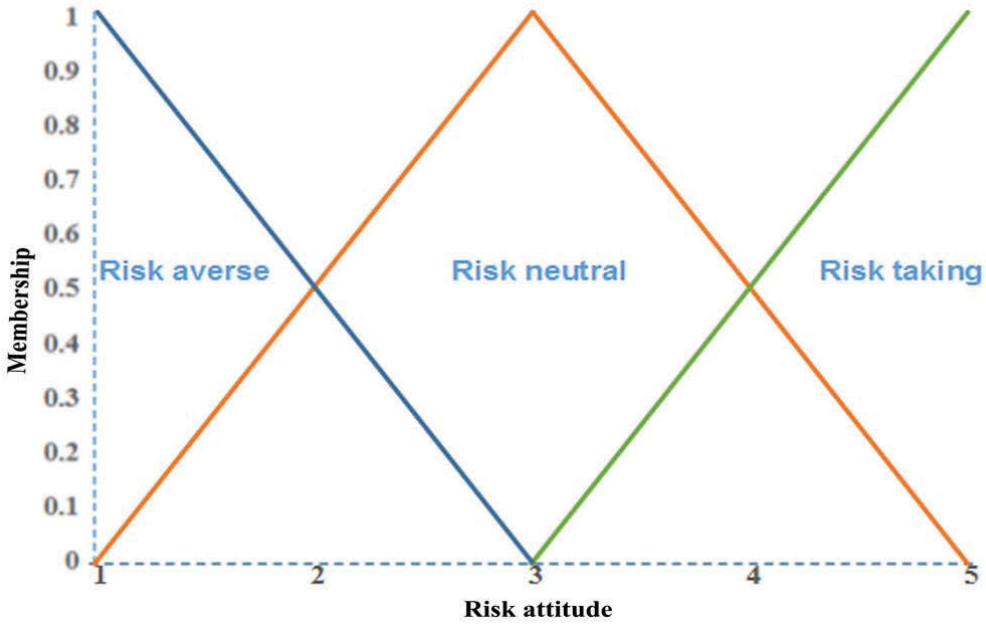


Figure 7.3: Risk attitude of the provider

7.3.2 Fuzzy Set for Second Input – Predicted Trajectory

The predicted trajectory is the trajectory of the predicted resource usage for future intervals. The values of the predicted trajectory are obtained from the QoSPM. A predicted trajectory is either "towards" (T) the T_a value or "away" (A) from the T_a value. When the predicted trajectory is defined as towards, it means that the trajectory has reached the T_s and is moving towards the T_a. When the predicted trajectory is defined as away, it means that the trajectory has exceeded the T_s value and is moving back towards the T_s. The numerical value ranges from 0 to 1. The membership function of the predicted trajectory is presented in Figure 7.4.

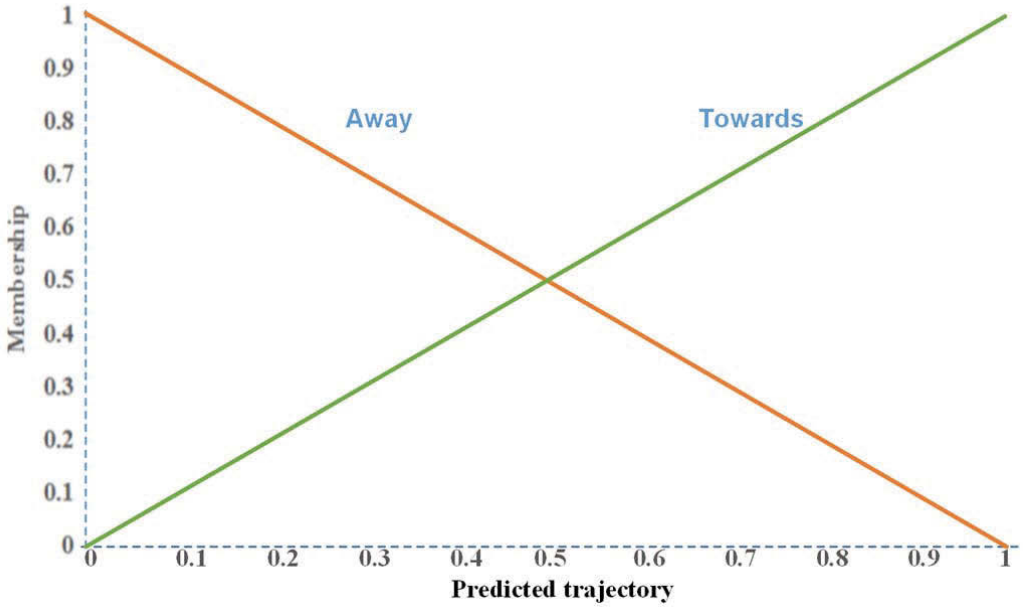


Figure 7.4: Membership function for the predicted trajectory

7.3.3 Fuzzy Set for Third Input – Reliability of the Consumer

Consumer reliability is the trustworthiness of the consumer's commitment to a service provider in previous transactions. The reliability value of a consumer is categorized as bronze, silver or gold. Consumer reliability is calculated from the transaction trend T_{trend}. For existing consumers, a T_{trend} is calculated from the consumer's previous transactions, but for a new consumer, the T_{trend} is determined from the T_{trend} of its nearest neighbours. The calculation of reliability was discussed in Chapter 5.

The numerical range for reliability is from 0 to 100: for a bronze consumer, it ranges from 0 to 45, for a silver consumer it ranges from 40 to 75, and for a gold consumer it ranges from 70 to 100. The membership function for the reliability of a consumer is presented in Figure 7.5.

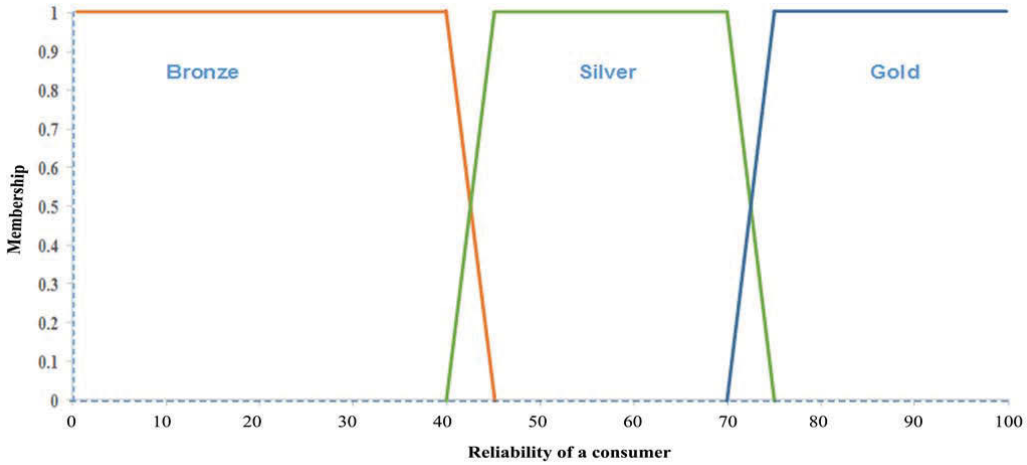


Figure 7.5: Membership function for a reliability of a consumer

7.3.4 Fuzzy Set for the Output – Type of Action

The FIS determines the appropriate action to take to manage a risk according to the risk attitude of the provider, the reputation of the consumer and the predicted trajectory. The output of the FIS is to take immediate action (IA), delayed action (DA), or no action (NA). The numerical range for the output is 0 to 1. The membership function for the type of action is presented in Figure 7.6.

7.3.5 Fuzzy Rules for a Risk Mitigation Action

The Mamdani-type FIS was used to determine the type of action to take to mitigate risk. The combination of linguistic variables for the risk attitude of the provider (risk averse (Ra), risk neutral (Rn), risk taking (Rt)), the linguistic variables for the reliability of the consumer (bronze (B), silver (S), gold (G)), and the two linguistic variables for the predicted trajectory (towards (T) and away (A)) result in a total of 18 rules, which are presented in Table 7.1.

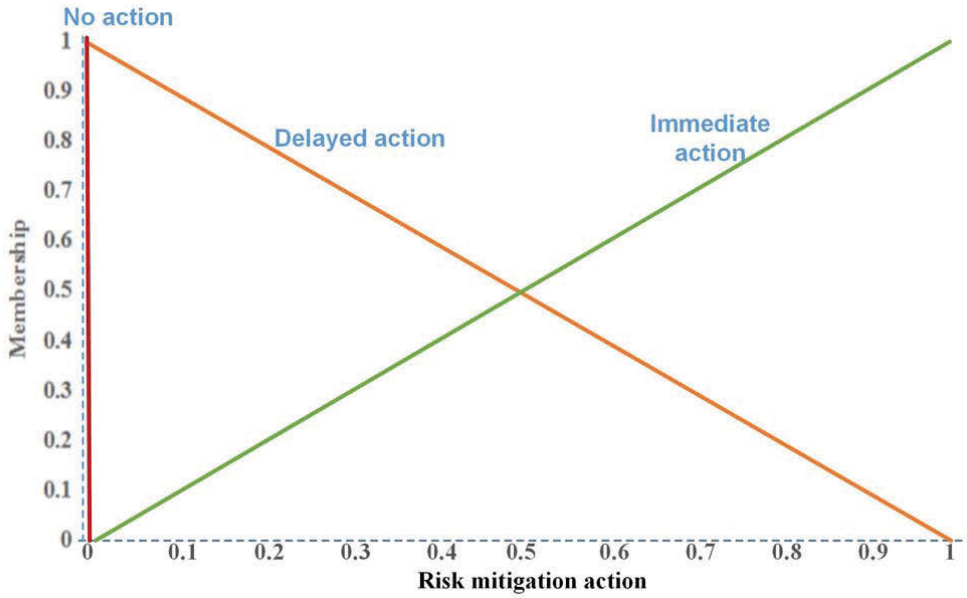


Figure 7.6: Membership function for a risk mitigation action

Table 7.1: FIS rules for risk mitigation action

Rule #		Risk attitude		Reliability		Predicted trajectory	then	Risk mitigation action
1	If	Ra	and	B	and	T		IA
2	If	Ra	and	B	and	A		IA
3	If	Ra	and	S	and	T		IA
4	If	Ra	and	S	and	A		DA
5	If	Ra	and	G	and	T		DA
6	If	Ra	and	G	and	A		DA
7	If	Rn	and	B	and	T		IA
8	If	Rn	and	B	and	A		DA
9	If	Rn	and	S	and	T		IA
10	If	Rn	and	S	and	A		NA
11	If	Rn	and	G	and	T		DA
12	If	Rn	and	G	and	A		NA
13	If	Rt	and	B	and	T		DA
14	If	Rt	and	B	and	A		NA
15	If	Rt	and	S	and	T		DA
16	If	Rt	and	S	and	A		NA
17	If	Rt	and	G	and	T		NA
18	If	Rt	and	G	and	A		NA

In the next section, the proposed module is validated using a real cloud dataset.

7.4 Validation of the Proposed Risk Management Module

To validate the proposed approach, a real cloud dataset from Amazon EC2 EU was selected (CloudClimate). The dataset was collected using PRTG service

(Monitor). The dataset is comprised of three QoS parameters: CPU, memory and I/O. A dataset was selected for three days starting from 2 March 2016 at 02:23 AM to 5 March 2016 at 02:23 AM, which is presented in Figure 7.7.

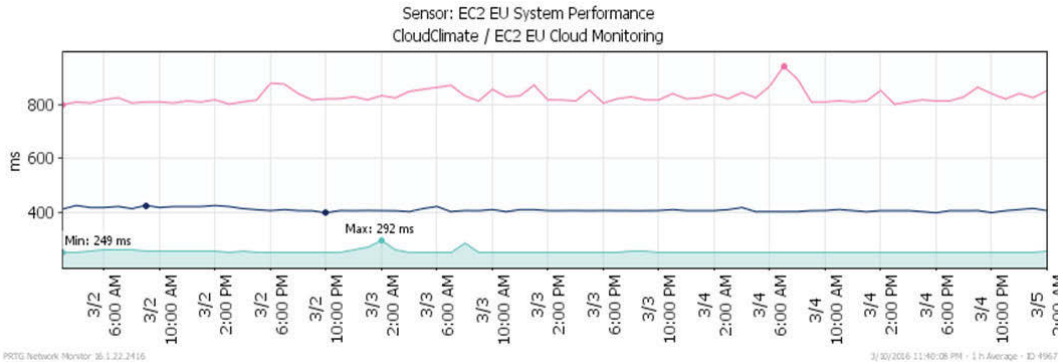


Figure 7.7: CPU, memory and I/O of EC2 – EU [22]

Only the CPU data are considered for this study. The ARIMA method is used for prediction because, as shown in in Chapter 5, the ARIMA method gives an optimal prediction result on a cloud dataset. This research therefore chooses that method. The observed and predicted CPU data for the above-mentioned period is presented in Table 7.2.

Table 7.2: Observed and predicted CPU data for the period of three days

Time slot	Observed CPU	Predicted CPU
3/2/2016 2:15:00 AM - 2:30:00 AM	249	249.0131605
3/2/2016 2:30:00 AM - 2:45:00 AM	249	249.0129859
3/2/2016 2:45:00 AM - 3:00:00 AM	249	249.0128209
3/2/2016 3:00:00 AM - 3:15:00 AM	249	249.0126573
3/2/2016 3:15:00 AM - 3:30:00 AM	249	249.0125146
3/2/2016 3:30:00 AM - 3:45:00 AM	250	250.0246918
3/2/2016 3:45:00 AM - 4:00:00 AM	249	249.0121970
3/2/2016 4:00:00 AM - 4:15:00 AM	249	249.0120531
3/2/2016 4:15:00 AM - 4:30:00 AM	248	248.0000049
3/2/2016 4:30:00 AM - 4:45:00 AM	249	249.0117674
3/2/2016 4:45:00 AM - 5:00:00 AM	249	249.0116299
3/2/2016 5:00:00 AM - 5:15:00 AM	249	249.0666662
3/2/2016 5:15:00 AM - 5:30:00 AM	255	256.0624976
3/2/2016 5:30:00 AM - 5:45:00 AM	249	249.0588231
3/2/2016 5:45:00 AM - 6:00:00 AM	246	246.9999999
3/2/2016 6:00:00 AM - 6:15:00 AM	250	250.1052613
3/2/2016 6:15:00 AM - 6:30:00 AM	249	249.0500030
3/2/2016 6:30:00 AM - 6:45:00 AM	249	249.0344840
3/2/2016 6:45:00 AM - 7:00:00 AM	249	249.0333382
3/2/2016 7:00:00 AM - 7:15:00 AM	249	249.0322609
3/2/2016 7:15:00 AM - 7:30:00 AM	249	249.0312372
3/2/2016 7:30:00 AM - 7:45:00 AM	266	266.5454552
3/2/2016 7:45:00 AM - 8:00:00 AM	249	249.0294089

Time slot	Observed CPU	Predicted CPU
3/2/2016 8:00:00 AM - 8:15:00 AM	249	249.0199938
3/2/2016 8:15:00 AM - 8:30:00 AM	249	249.0196050
3/2/2016 8:30:00 AM - 8:45:00 AM	250	250.0384646
3/2/2016 8:45:00 AM - 9:00:00 AM	251	251.0566016
3/2/2016 9:00:00 AM - 9:15:00 AM	249	249.0185186
3/2/2016 9:15:00 AM - 9:30:00 AM	249	249.0181822
3/2/2016 9:30:00 AM - 9:45:00 AM	266	266.3214221
3/2/2016 9:45:00 AM - 10:00:00 AM	249	249.0175339
3/2/2016 10:00:00 AM - 10:15:00 AM	249	249.0172422
3/2/2016 10:15:00 AM - 10:30:00 AM	249	249.0169556
3/2/2016 10:30:00 AM - 10:45:00 AM	249	249.0166670
3/2/2016 10:45:00 AM - 11:00:00 AM	249	249.0164024
3/2/2016 11:00:00 AM - 11:15:00 AM	249	249.0161107
3/2/2016 11:15:00 AM - 11:30:00 AM	249	249.0158726
3/2/2016 11:30:00 AM - 11:45:00 AM	250	250.0312437

The graph of observed and predicted CPU data is presented in Figure 7.8.

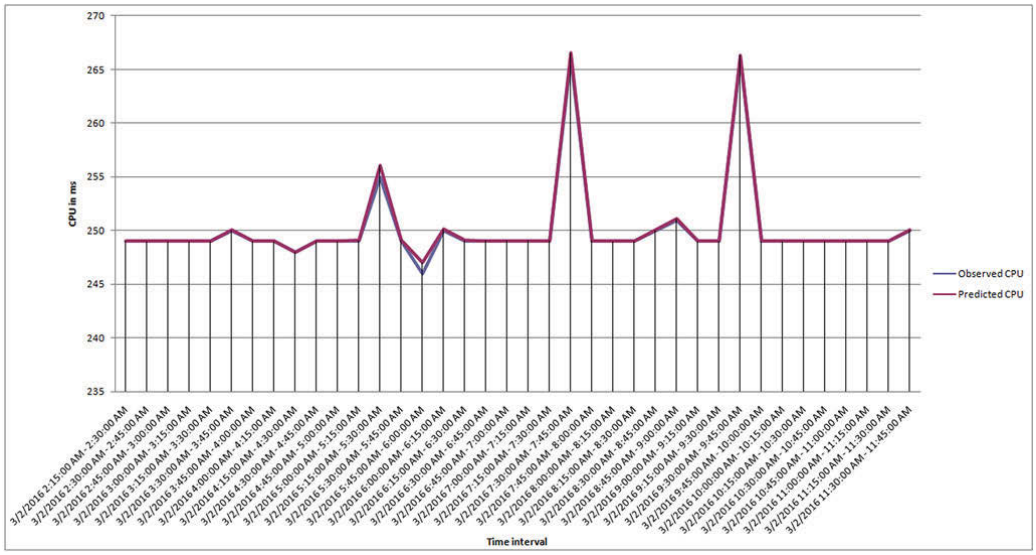


Figure 7.8: Observed and predicted CPU data

The prediction accuracy for the above-mentioned period using MAD, RMSE and MSE is presented in Table 7.3.

Table 7.3: Prediction accuracy of ARIMA method

MAD	RMSE	MSE
0.10074	0.61395	0.37693

To analyse our framework, the ARIMA method was used to predict for 12 intervals over a three-hour period on 3 February, 2016 from 11:45:00 AM to 2:30:00 PM. The prediction intervals were divided into four sections. Each section

is comprised of four time intervals separated by five minutes. Each section is evaluated using the proposed risk management framework to see what type of action the provider takes to mitigate a risk, if any. The choice of 20 minute intervals per prediction is because a system is able to request a new cloud instance approximately 12 to 15 minutes prior to arranging the required resources (Islam et al. 2012), therefore, this interval was selected for each section. There are three lines in each section. The blue line represents the prediction result using the ARIMA method for that interval, the red line represents the T_s value, and the green line represents the T_a value. It is assumed that a provider has set T_s as 260ms and T_a as 290ms. When a predicted result reaches or exceeds the red line T_s, the risk management module is activated. For each scenario, it is assumed that a consumer C1 with a reliability value of 20 is using the services of a cloud provider P1 that has a risk attitude of 3.

7.4.1 Scenario 1

The time intervals for the first scenario starts at 3 February, 2016 from 1:45AM to 12:30PM. The T_s, T_a and predicted results using the ARIMA method are presented in Table 7.4 and Figure 7.9.

Table 7.4: Predicted data for Scenario 1

Interval	Predicted result	T_s	T_a
3/2/2016 11:45:00 AM - 12:00:00 PM	249.028569ms	260ms	290ms
3/2/2016 12:00:00 PM - 12:15:00 PM	258.0277777ms	260ms	290ms
3/2/2016 12:15:00 PM - 12:30:00 PM	266.8571404ms	260ms	290ms
3/2/2016 12:30:00 PM - 12:45:00 PM	272.818178ms	260ms	290ms

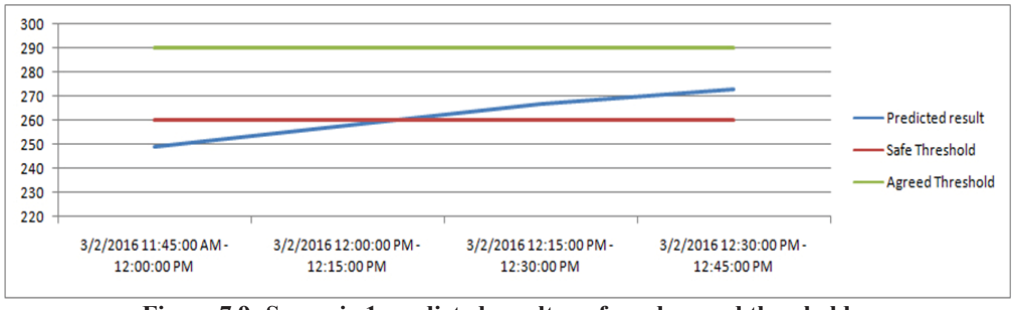


Figure 7.9: Scenario 1 predicted results, safe and agreed thresholds

In Figure 7.9, we see that at the first time interval of 11:45:00 AM to 12:00:00 PM, the predicted results are less than the T_s value, but they start to increase from

12:00:00 PM and exceed the T_s value for the next two intervals, 12:15:00 PM and 12:30:00 PM. When the predicted result exceeds the T_s value, the RMM is activated and uses the FIS to manage a risk.

The input of the FIS for the first scenario is:

- Reliability of the consumer C1 = 20
- Risk attitude of the provider P1 = 3
- Predicted trajectory = towards

When the above inputs are given to the FIS, the value generated after defuzzification is 67%, which specifies immediate action.

In the above scenario, we see that the risk attitude of the provider is risk neutral, the reliability of the consumer is bronze and the trajectory is moving towards the agreed threshold value. When the FIS obtains the inputs, Rule 7 from Table 7.1 is triggered, with values of 0.67 for immediate action and 0.33 for delayed action.

Rule 7 of Table 7.1 stipulates that when a service provider with a risk attitude of risk neutral deals with a consumer that has a reliability value of bronze, with a trajectory which has exceeded T_s and is moving towards T_a, the provider categorizes the risk as a high risk and does not accept it. Instead, the provider takes immediate action to remove the risk in the earliest possible time.

7.4.2 Scenario 2

The time interval for the second scenario starts at 3 February, 2016 from 12:45:00 PM to 1:30:00 PM. The T_s, T_a and predicted results using the ARIMA method are presented in Table 7.5 and Figure 7.10.

Table 7.5: Predicted data for Scenario 2

Interval	Predicted result	T_s	T_a
3/2/2016 12:45:00 PM - 1:00:00 PM	278.6956515ms	260ms	290ms
3/2/2016 1:00:00 PM - 1:15:00 PM	276.0416658ms	260ms	290ms
3/2/2016 1:15:00 PM - 1:30:00 PM	268.6400026ms	260ms	290ms
3/2/2016 1:30:00 PM - 1:45:00 PM	262.6923102ms	260ms	290ms

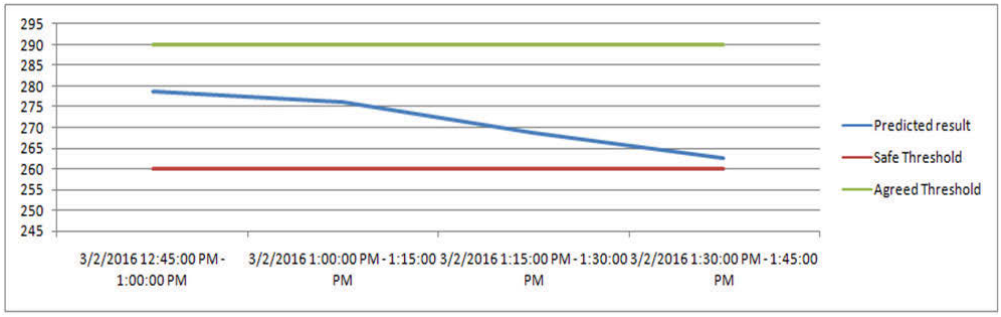


Figure 7.10: Scenario 2 predicted results, safe and agreed thresholds

In Figure 7.10, we see that from the start of the interval that runs from 12:45:00 PM to 1:00:00 PM, the predicted trajectory is above the T_s value, but it starts to decrease gradually at future intervals, from 1:00:00 PM to 1:30:00 PM. At interval 1:30:00PM, it is very close to the T_s value but is still it is above the T_s value. The inputs for the second scenario are:

- Reliability of the consumer C1 = 20
- Risk attitude of the provider P1 = 3
- Predicted trajectory = away

When the above inputs are given to the FIS, the value generated after defuzzification is 33%, which specifies delayed action. In this case, Rule 8 of Table 7.1 is triggered, which means that when the risk attitude of the provider is risk neutral, the reliability of the consumer is bronze, and the trajectory is moving away from the agreed threshold, and the action that is required is delayed action.

7.4.3 Scenario 3

The time intervals for the third scenario starts at 3 February, 2016 from 1:45:00 PM to 2:30:00 PM. The T_s, T_a and predicted results using the ARIMA method are presented in Table 7.6 and Figure 7.11.

Table 7.6: Predicted data for Scenario 3

Interval	Predicted result	T_s	T_a
3/2/2016 1:45:00 PM - 2:00:00 PM	263.000010ms	260ms	290ms
3/2/2016 2:00:00 PM - 2:15:00 PM	260.6428599ms	260ms	290ms
3/2/2016 2:15:00 PM - 2:30:00 PM	258.4864862ms	260ms	290ms
3/2/2016 2:30:00 PM - 2:45:00 PM	248.4210503ms	260ms	290ms

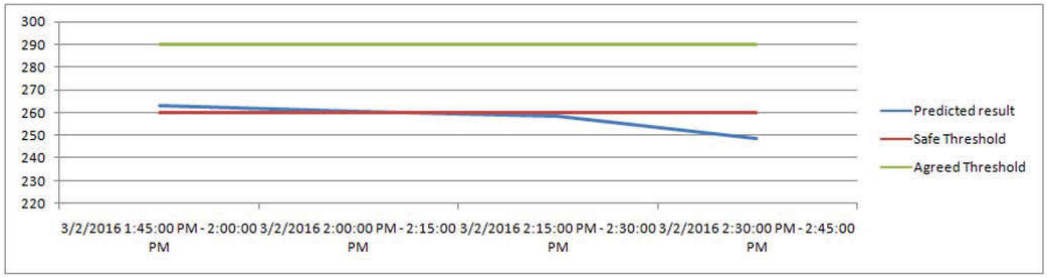


Figure 7.11: Scenario 3 predicted results, safe and agreed thresholds

From Figure 7.11, we see that at the first two intervals – 1:45:00 PM to 2:00:00 PM and 2:00:00 PM to 2:15:00 PM – the trajectory is above and equal to the T_s value; however, for the next two intervals – 2:15:00 PM to 2:30:00 PM and 2:30:00 PM to 2:45:00 PM – it moves below the T_s value. The predicted trajectory is below the T_s value and is stable, so the RMM is not activated. The system continues to monitor and takes no action.

In above three scenarios, we see that depending on the consumer's reliability, risk attitude of the provider and the trend prediction the module suggests the appropriate action to manage the risk of SLA violation. When the module finds the consumer with a lesser reliability value and the predicted trajectory is moving towards the safe threshold then there is higher chance of SLA violation in future intervals and hence the module suggests immediate possible action be taken without any delay to avoid violation; however, if the provider is risk taking and consumer has higher reliability value with the prediction trajectory moving away from the threshold value then the module estimate less chance of violation and suggests no action be taken. If the predicted trajectory moves below the threshold value and become stable then the module does not activate and lets the system execute in a normal way. In this way, the risk management module helps the service provider to manage the risk of SLA violation and suggests appropriate action to avoid SLA in the post-interaction time phase.

7.5 Conclusion

It is necessary for small and medium cloud providers in a cloud computing environment to manage their resources wisely. They need a framework that can help them to form viable SLAs by offering them the requested fixed and marginal resources wisely and manage the risks of SLA violation to avoid violation penalties. A cloud provider predicts the future QoS parameters for all SLOs.

When a provider finds a difference between the agreed and the predicted QoS parameters, it needs a framework that will take the appropriate action to mitigate risks. Our risk management framework helps service providers to take appropriate action to mitigate the risk of SLA violations when it finds discrepancies between the agreed and predicted resource usage behaviour.

8 Solution Implementation

8.1 Introduction

Previous chapters have discussed each component of the viable SLA management framework that works from the provider's side to avoid SLA violations. In this chapter, a prototype implementation of the proposed framework that gives the proof of concept as described in research methodology section in Chapter 3 is presented. Chapter 4 described the different modules and their workings. In Chapters 5, 6 and 7, the three sections of the proposed framework were discussed and their implementation and evaluation examined individually. As described in previous chapters, the provider's side viable SLA management framework has been developed in Microsoft Visual Studio, Microsoft SQL Server Management Studio and MATLAB. This chapter presents a prototype that incorporates the functionality of every individual module that resides in the three sections of the framework and provides a user interface to the developed system.

The chapter is arranged in the following way. This section introduces the thesis. Section 8.2 presents an overview of the implementation of the proposed solution implementation. Section 8.3 discusses the pre-interaction phase and Section 8.4 outlines the post-interaction phase. Section 8.5 concludes the chapter.

8.2 Overview of Solution Implementation

As described in Chapter 3, the proposed provider's side viable SLA management framework is comprised of three sections. Each section contains different components, as presented in Chapter 4. In this section, the prototype implementation of our approach is discussed.

The objective of the prototype is to combine all modules in the pre-interaction and post-interaction time phases and to help the service provider form a viable SLA, as well as monitor, predict and manage the risk of SLA violation and apply all necessary measures to avoid SLA violations and violation penalties. This chapter presents a graphical user interface that combine all modules together into a complete tool.

With reference of the three sections of the framework – viable SLA formation, QoS monitoring and prediction and risk management module that resides at two time phases (as described in Chapter 4) – there are two user interfaces. The first section (viable SLA formation) resides the in pre-interaction time phase, while the second and third sections (QoS monitoring, prediction and risk management module) reside in the post-interaction time phase. In the first section, the system takes inputs about the user to authenticate and validate the requesting consumers and validate their credentials from the repository in IMM. The IMM module authenticates the consumer and sends their profile history to VSLM to make a decision on the consumer's request.

Once the provider has executed viable SLA, then it obtains the safe and agreed threshold values from the pre-interaction phase and chooses any one of the six prediction methods at 10 different datasets. Once the system obtains the predicted results, it then compares them against the threshold value. If the results exceed the threshold value, then it suggests an appropriate action depending on the predicted trajectory, the reliability of consumer and risk attitude of the provider.

The following sections presents the graphical user interface (GUI) of the implementation and screen shots of the prototype for all modules in three sections.

8.3 Pre-interaction Phase: Viable SLA Formation

The GUI for the first section presents the formation of inputs and steps in forming a viable SLA which are presented in Figures 8.1 to 8.6. Figure 8.1 presents the first step of the prototype, when the consumer requests services from the provider. The request comes to the IMM module by the interface presented in Figure 8.1. IMM takes the user ID as input to validate the consumer.

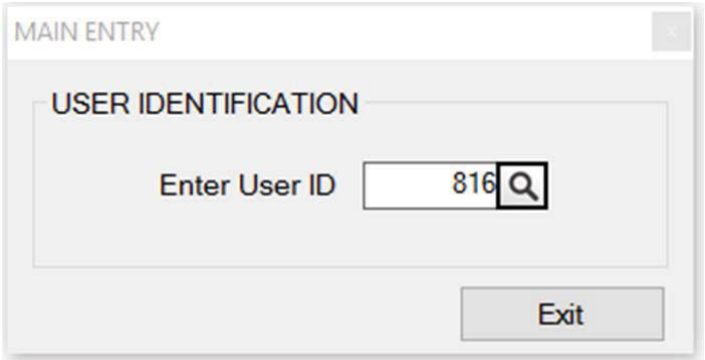


Figure 8.1: User authentication

In the case of an existing consumer, the module takes their previous transaction record and sends it to the VSLAM. If the consumer is new, the application takes a few parameters through which it finds the top-K nearest neighbours. In this study, a consumer with a user ID of 816 who does not have an existing record is considered. Therefore, the module asks consumer 816 to enter its detail as presented in Figure 8.2.

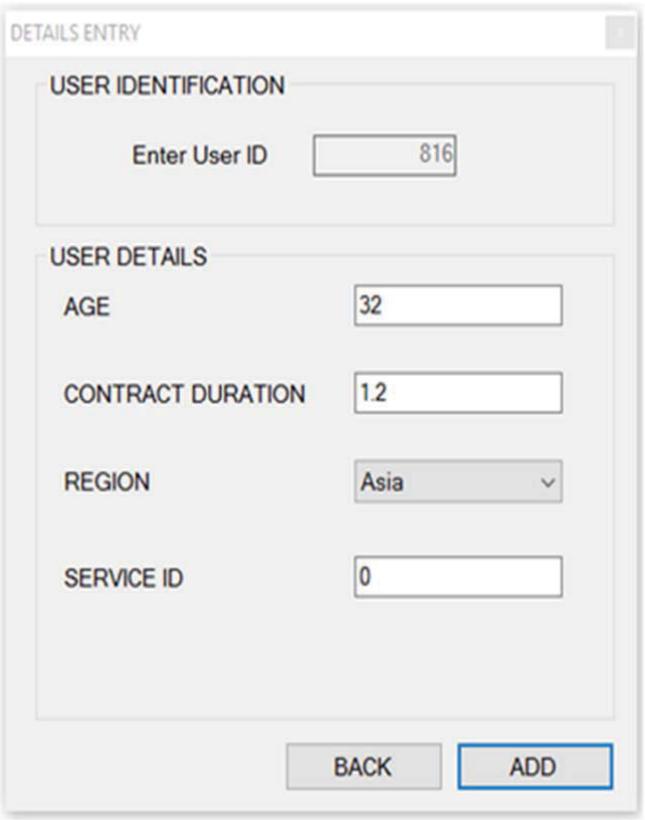


Figure 8.2: Input to find top-K nearest neighbours for the consumer ID 816

After entering the data for consumer 816, the system starts to calculate their T_{trend}. For existing consumers, the system considers the previous profile of the consumer and calculates their T_{trend}; however, for a new consumer who does not have any previous records, a module calculates the T_{trend}, which is a weighted average of the T_{trend} for the top-K nearest neighbours. The system displays the T_{trend} value of the requesting consumer and asks for a provider to enter the threshold value for their transaction. The process is presented in Figure 8.3.

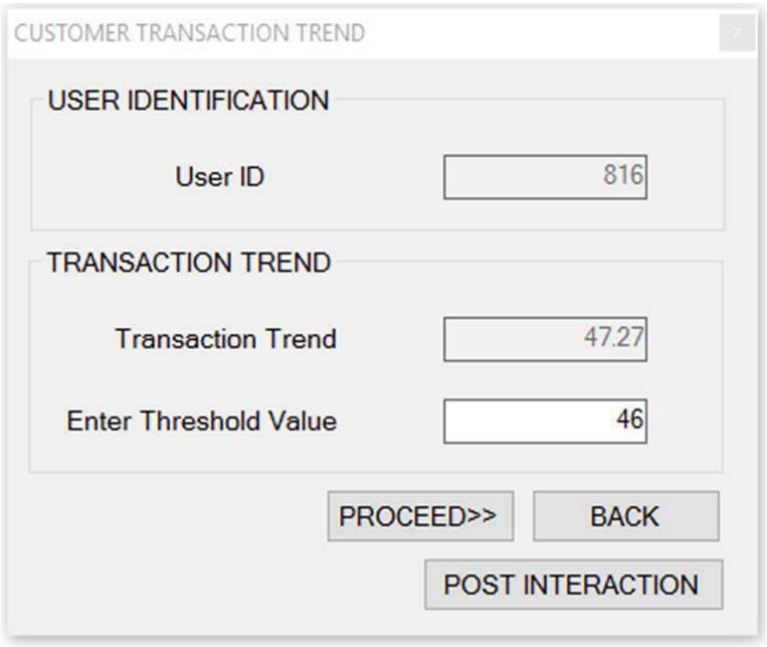


Figure 8.3: T_{trend} of a consumer 816 based on T_{trend} of its nearest neighbours and a threshold value

The system compares the threshold value with the T_{trend} of the requesting consumer. If the value of T_{trend} is greater or equal to the threshold value, the request is accepted and proceeds further, otherwise the request is rejected. In this case, the value of T_{trend} for the consumer 816 is 47.27, which is greater than the value of the threshold 46%, hence the request is accepted, as presented in Figure 8.4.

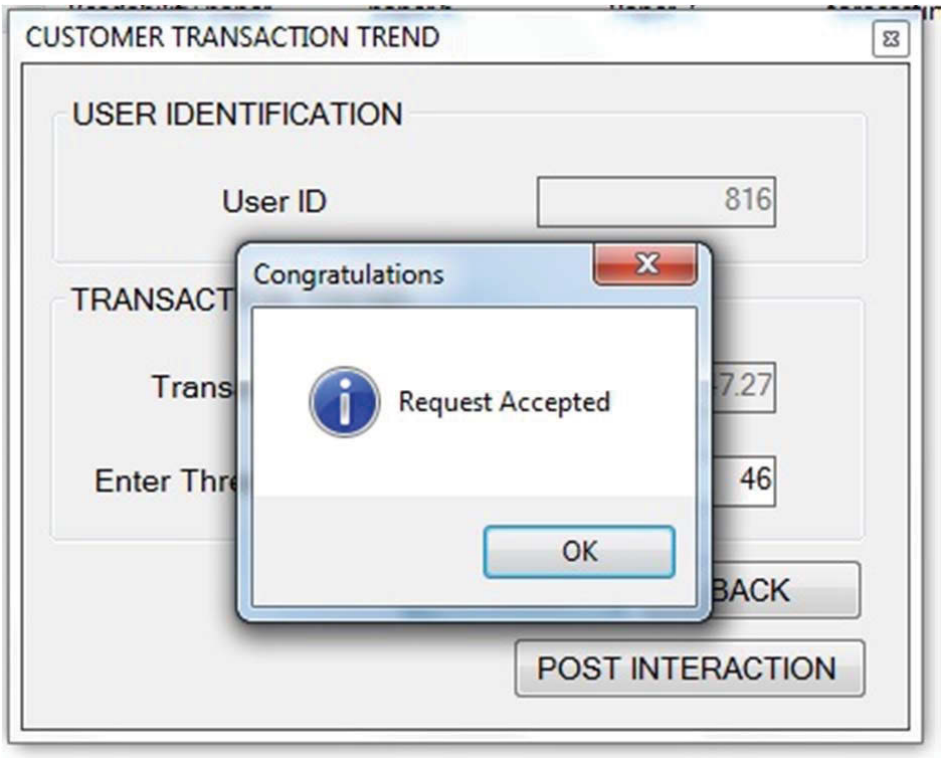


Figure 8.4: Request of consumer 816 is accepted

When the request is accepted, the system asks for the risk propensity of the provider to calculate the amount of resources to offer to the consumer. As we can see in Figure 8.5 the risk propensity of a provider is 0.5.

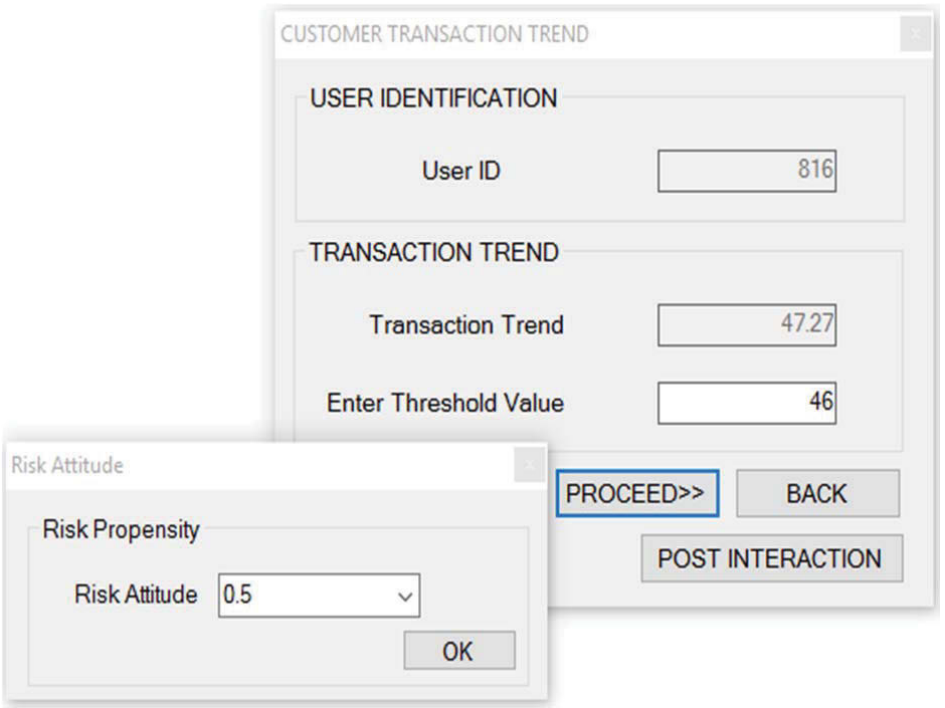
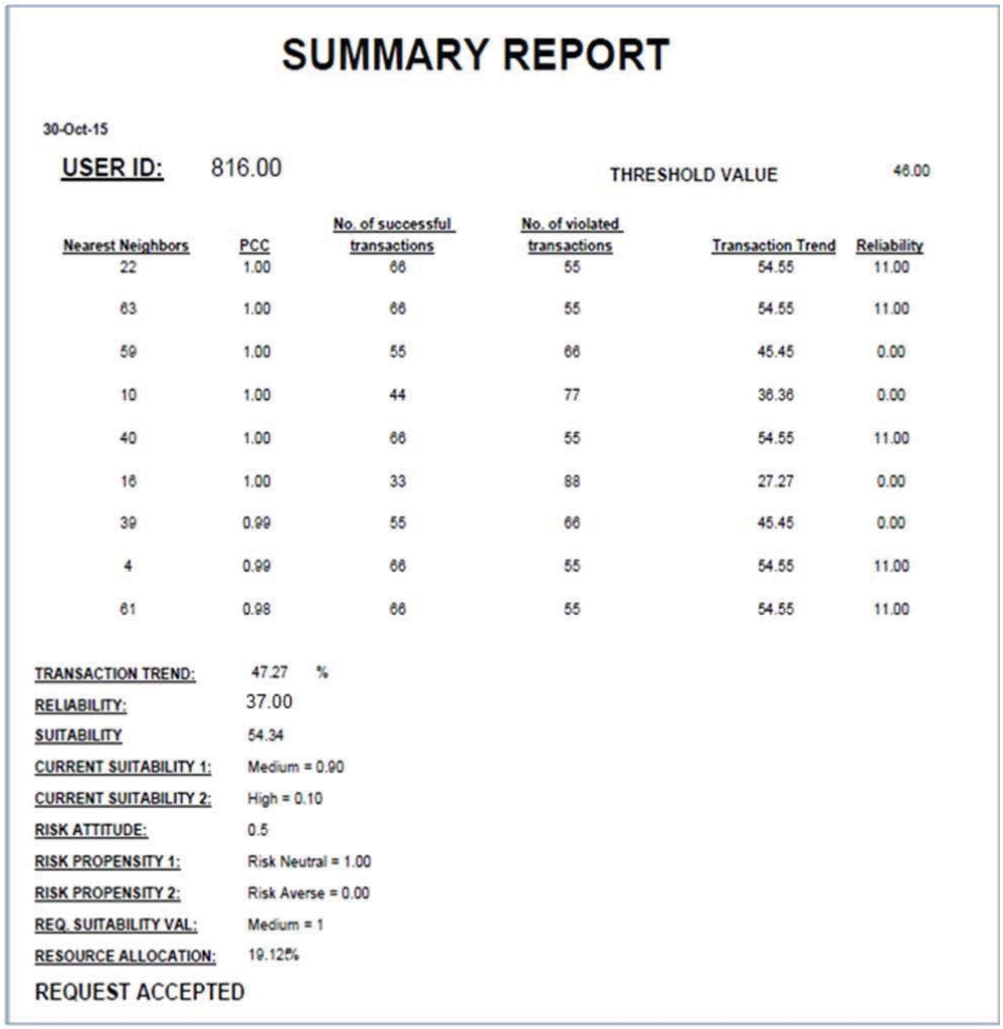


Figure 8.5: Risk propensity of the provider

This step considers the suitability of the consumer and the risk propensity of the provider. As discussed in Chapter 5, the system uses the FIS at two levels to determine the amount of resources to offer. Figure 8.6 presents a summary report on consumer 816 that displays the top-K nearest neighbours, the number of successful and violated transactions, the PCC value, the T_{trend} for each neighbour and their reliability value.



SUMMARY REPORT

30-Oct-15

USER ID: 816.00 THRESHOLD VALUE 46.00

Nearest Neighbors	PCC	No. of successful transactions	No. of violated transactions	Transaction Trend	Reliability
22	1.00	66	55	54.55	11.00
63	1.00	66	55	54.55	11.00
59	1.00	55	66	45.45	0.00
10	1.00	44	77	36.36	0.00
40	1.00	66	55	54.55	11.00
16	1.00	33	88	27.27	0.00
39	0.99	55	66	45.45	0.00
4	0.99	66	55	54.55	11.00
61	0.98	66	55	54.55	11.00

TRANSACTION TREND:	47.27 %
RELIABILITY:	37.00
SUITABILITY	54.34
CURRENT SUITABILITY 1:	Medium = 0.90
CURRENT SUITABILITY 2:	High = 0.10
RISK ATTITUDE:	0.5
RISK PROPENSITY 1:	Risk Neutral = 1.00
RISK PROPENSITY 2:	Risk Averse = 0.00
REQ. SUITABILITY VAL:	Medium = 1
RESOURCE ALLOCATION:	19.12%

REQUEST ACCEPTED

Figure 8.6: Summary report of resource allocation for consumer 816

The threshold for this transaction is 46%. The system calculates the T_{trend} as 47.27% with a reliability value of 72%. The risk attitude of the provider is risk neutral which is 0.5 and the percentage of resources to offer to the consumer is 19.12%.

Table 8.1 presents a summary of the request determination and resource allocation for consumer 816. The table shows the T_{trend}, the reliability of consumer, the

suitability value calculated by FIS, the membership functions for suitability and risk propensity, the required suitability value, the decision in relation to the consumer's request and the amount of resources to be offered to accepted consumers. By using a viable SLA model, a provider is able to choose a reliable consumer and optimally allocate the amount of resources to them.

Table 8.1: Request determination and resource allocation of requesting consumers

User ID	Transaction Trend	Reliability	Suitability of consumer	Current suitability value		Risk attitude of provider	Risk propensity		Required suitability value	Resource allocation	Decision of consumer request
816	47.27%	37	54.34	Med = 0.90	High = 0.10	0.5	RN = 1.0	RA = 0.0	Medium = 1	19.12%	Accepted

When the provider is able to form a viable SLA with the consumer, then the process of the post-interaction phase starts that monitors the runtime behaviour of the consumer and predicts the QoS parameters for future intervals. The GUI for selection of QoS prediction method by choosing its parameters is presented in Section 8.4.

8.4 Post-interaction Phase: Threshold, QoS Monitoring, Prediction and Risk Management

This section presents the threshold formation module discussed in Chapter 4, the QoS monitoring approaches that facilitate the cloud provider to choose an appropriate prediction method depending on varying dataset and the risk management module that notifies the service provider if early remedial action is required. The prediction methods are evaluated using 10 different datasets and six prediction methods with different parameters. The working of modules in the post-interaction phase is presented in Figures 8.7 to 8.12.

When the SLA is executed, the first module in the post-interaction phase is the threshold formation module. The prototype takes an input from the provider for the safe and agreed threshold. Here, we consider the QoS parameter CPU, the safe threshold is 290ms and agreed threshold as 260ms, as presented in Figure 8.7.

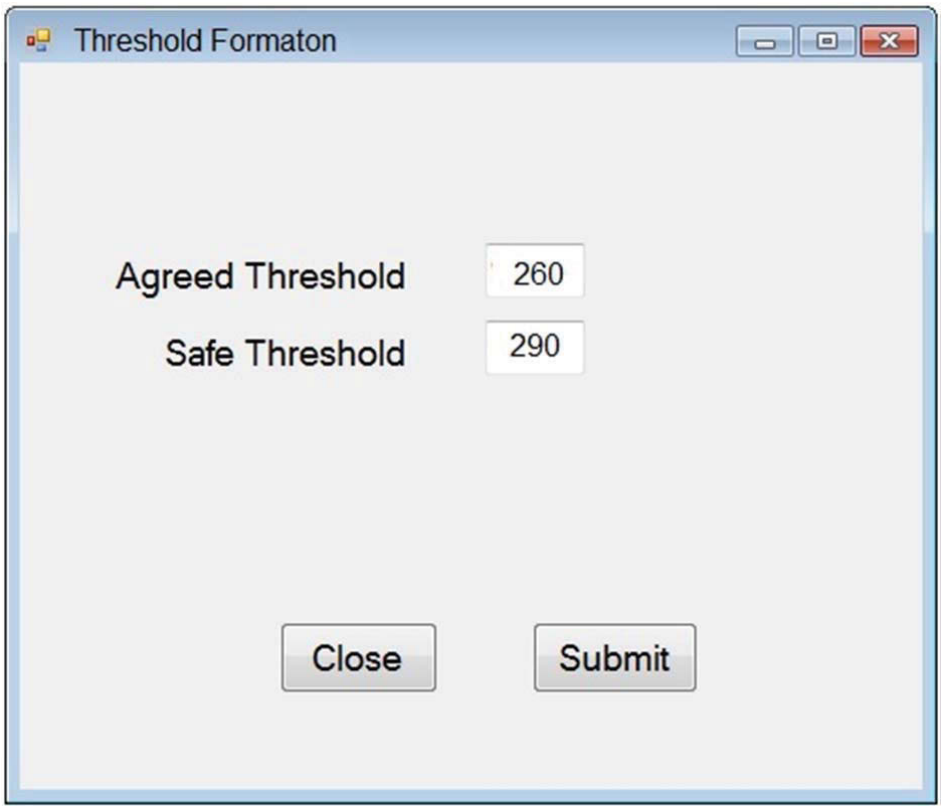


Figure 8.7: Threshold formation

After the provider has defined the safe and agreed threshold value, then the system asks for the selection of dataset. In this experiment 10 datasets were used at different time intervals. The dataset is obtained from the real cloud provider, as discussed in Chapter 6. The GUI for the dataset selection is presented in Figure 8.8.

The next step is the selection of the prediction method. The provider chooses one of the six prediction methods to predict future intervals, as presented in Figure 8.9.

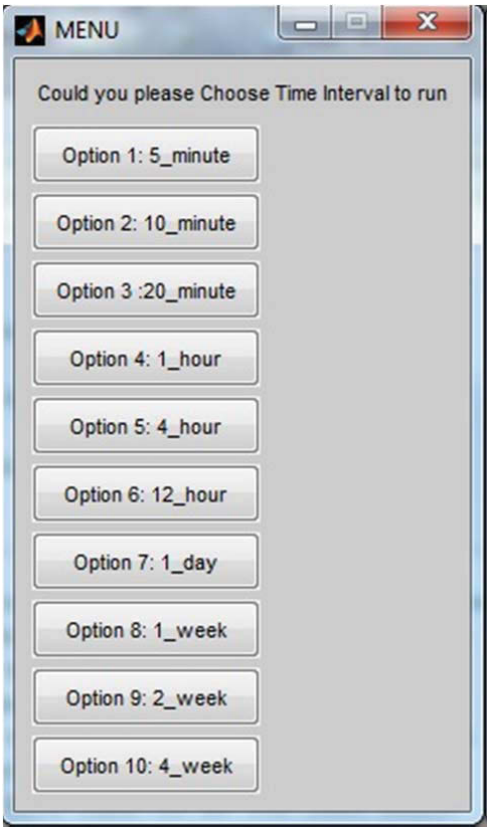


Figure 8.8: Selection of dataset at different time intervals

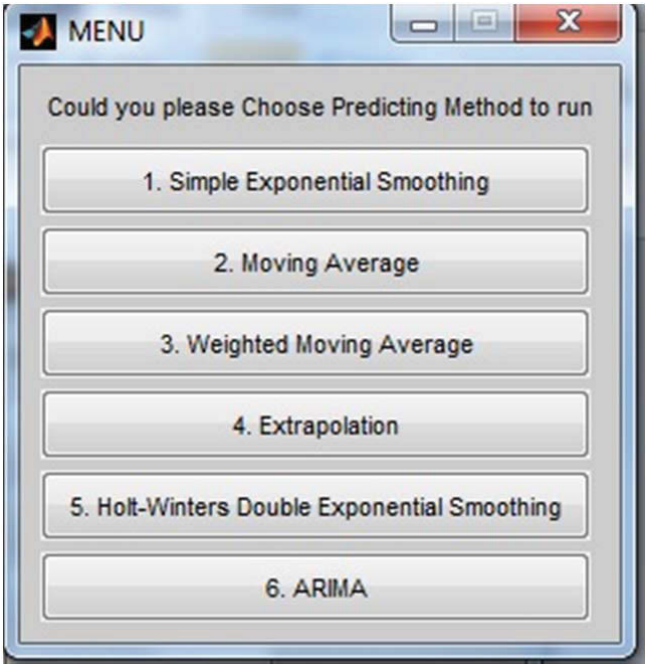


Figure 8.9: Selection of prediction methods

The difference between the agreed and the predicted QoS parameters is presented in Figure 8.10.

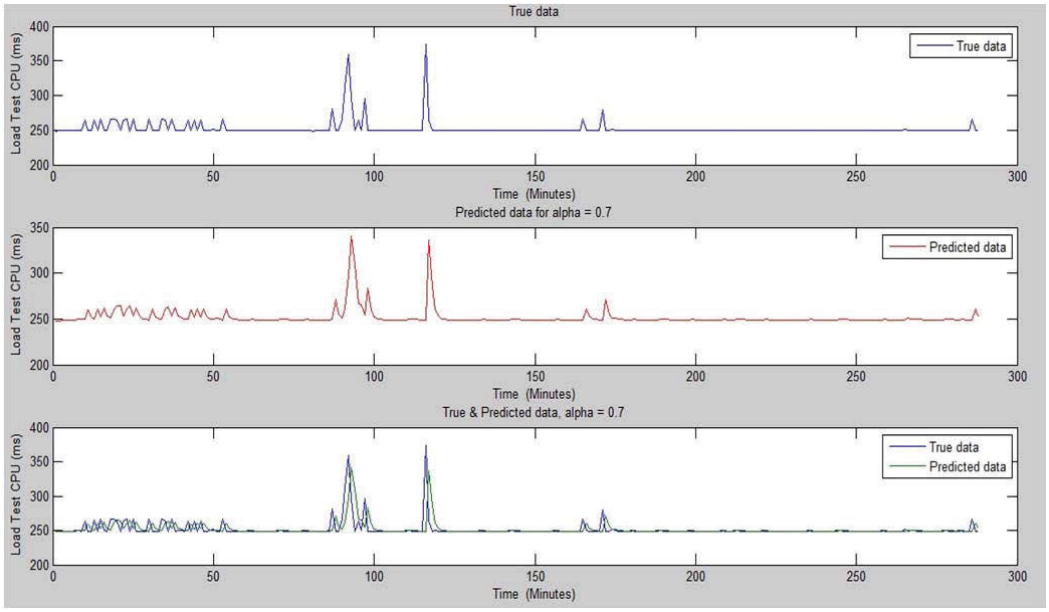


Figure 8.10: Difference between actual and predicted QoS parameters

We can see in Section 8.4 that consumer 816 has a reliability value of 37 and the risk attitude of the provider is 0.5. According to the Figures 7.3 and 7.5, the reliability of consumer is categorized as bronze and the risk attitude of the provider is identified as risk neutral.

We consider the prediction result for future four intervals as presented in Figure 8.11, in which the red line presents the safe threshold, the green line presents the agreed threshold and the blue line presents the predicted trajectory.

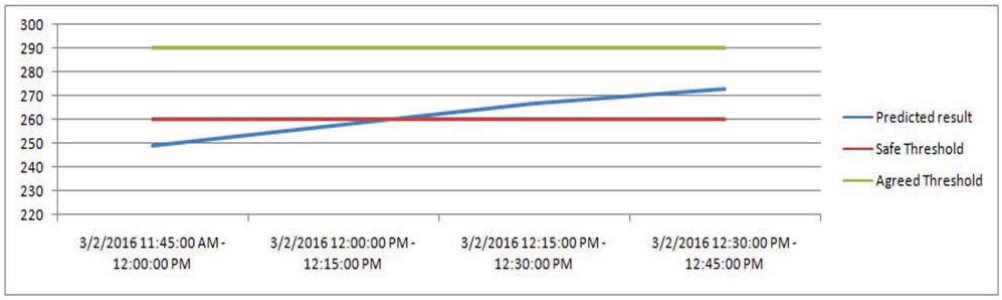


Figure 8.11: Prediction result and threshold for future intervals

As the predicted trajectory exceeds the safe threshold at the second interval, the RMM is activated. The prototype uses the FIS defined in Table 7.1 with the inputs of risk attitude as risk neutral, reliability as bronze and predicted trajectory as towards. The risk is estimated as high and the prototype recommends the cloud provider take immediate action to avoid SLA violation in future intervals.

In this way, the provider's side viable SLA management framework helps the service provider to form a viable SLA, monitor runtime QoS parameters, predict future QoS parameters, identify estimates and mitigate the risk of SLA violation to avoid actual violation and violation penalties.

8.5 Conclusion

This chapter describes the prototype implementation that integrates all modules in the pre-interaction and post-interaction time phases, which combine together to form the proposed viable SLA management framework. The system is implemented in MATLAB, Microsoft Visual Studio and Microsoft SQL Server Management Studio. The prototype has been discussed with the screen shots of the GUI for all modules to demonstrate the overall functionality of the framework.

9 Recapitulation and Future Work

9.1 Introduction

The need for and use of cloud computing platforms are increasing every day. Businesses and consumers are using cloud-computing services in a wider range of ways than ever before. This increase in the cloud market creates new challenges for cloud providers. The need for a viable SLA management architecture that ensures a service provider will have optimal resource allocation, predicts likely violations and also mitigates and arranges required resources before either the consumer or the service provider are affected is one such challenge. A number of researchers have proposed various frameworks to manage SLA in the cloud environment, as presented in Chapter 2. Additionally, it is evident from the discussion in Chapters 2 and 3 that many of these existing approaches have made significant advances in managing SLA from both the cloud consumer's and cloud provider's perspective. At the same time, however, the major shortcoming of these existing approaches is that none of these approaches can be regarded as complete for the purpose of SLA management from provider's perspective; that is, so the provider can manage SLA at both phases of the SLA management lifecycle. The reason for this is that none of these approaches monitor SLA in the pre-interaction time phase and form a viable SLA that could intelligently offer the amount of resources, monitor runtime QoS parameters, predict QoS parameters for future intervals and compare them with the customized threshold defined by the provider and, when any discrepancies are found, generate an early warning to the service provider so they can take the appropriate necessary action.

In this thesis, a provider's side viable SLA management framework that focuses on assisting small and medium cloud service providers in making a viable SLA with the consumer and managing the SLA to avoid violations and penalties is developed. The proposed framework presented in this thesis addresses the gaps in existing literature. The framework is composed of different modules that were described in detail in earlier chapters.

The chapter is organized as follows. This section introduces the chapter. In Section 9.2, the research issues that have been addressed in this thesis are recapitulated.

Section 9.3 discusses the contributions made by this thesis to address the identified research gaps in provider's side SLA management framework. Section 9.4 identifies areas of future research and Section 9.5 concludes the chapter.

9.2 Recapitulation

In any business generally and in cloud computing specifically, the primary aim of a service provider is to fulfill its commitment to consumers without SLA violations. To meet its objectives, a provider needs intelligent SLA management algorithms that initially can form a viable SLAs with a consumer then monitor them to detect and mitigate likely violations before they occur.

The existing cloud SLA management frameworks are designed to help the consumer and, in some cases, the cloud provider to manage the SLA and underlying cloud middleware. However, none of these frameworks perform SLA management from the perspective of small and medium cloud providers and none of these approaches form a viable SLA that starts in pre-interaction time phase. There are number of challenges in cloud SLA management from the provider's view point that the current SLA management frameworks are unable to answer. The existing approaches start the SLA monitoring and management once the provider and the consumer execute SLA and very few have a mechanism of early warning and recommendation mechanism to manage the risk of SLA violation.

Therefore, the object of this thesis was to develop a provider SLA management framework that assists small and medium cloud service providers in forming a viable SLA that assures the maximum utilization of resources and generates the maximum profit without any violation and penalties. The research questions that drove this research in solving this issue are:

1. Propose a viable SLA lifecycle from provider's perspective that starts the process of SLA management from pre-interaction time phase.
2. Propose a methodology to assist the cloud provider to form a viable SLA with the consumer by considering its previous profile of SLA violation.
3. Propose a method to predict the future QoS and usage of resources by the consumer based on its previous QoS history.

4. Propose a framework that helps the cloud provider to identify the risk of SLA violation, estimate it and suggest appropriate actions to manage it.
5. Validate the approach for SLA management.

9.3 Contribution of the Thesis

The main contribution of this thesis to the existing literature is that it explains and focuses the need of a viable SLA management framework for small and medium cloud providers that helps them to form a viable SLA with the consumer. The framework proposed in this research allows the cloud provider to form a viable SLA, which helps them to make timely SLA management decisions.

A very brief overview of contributions made by this thesis towards addressing the gaps in the existing literature are described in the following subsections.

9.3.1 Contribution 1: Comprehensive Survey of Present Literature

In Chapter 2, a detailed study of the existing SLA management frameworks in the cloud computing environment was conducted. To the best knowledge of the researcher, the survey carried in this study is comprehensive and extensive.

For the purpose of discussion and evaluation, the existing literature was divided into three classes based on their features and working attributes. The three classes are as follows:

1) Self-manageable case-based reasoning approach
2) Trust model-based approach
3) Proactive monitoring approach

The features and shortcomings of each of these approaches were identified. From an in-depth analysis of the literature, gaps were found with the current approaches that established the prospect of solving the research problems addressed in this thesis.

9.3.2 Contribution 2: Propose a Definition of Viable SLA Management Framework and Viable SLA Formation Lifecycle from the Small and Medium Cloud Provider's Perspective

As discussed in previous chapters, provider-side cloud SLA management is different from SLA management from the consumer's perspective, and little attention has given to this issue in the literature. Therefore, before designing the provider-side viable SLA management, a prescribed definition of provider-side viable SLA management framework was proposed that focused on small and medium cloud providers, as presented in Section 4.2 of thesis. The provider-side viable SLA management is comprised of three processes:

1) construction of a viable SLA with the requesting consumer;
2) monitoring the runtime behaviour of the consumer and predicting QoS parameters for future intervals;
3) and taking appropriate necessary action to avoid an SLA violation occurring when there are any discrepancies between the agreed performance and runtime consumer's behaviour.

The proposed viable SLA management framework is comprised of two time phases: the pre-interaction phase and the post-interaction phase. The pre-interaction phase is related to the formation of a viable SLA formation and the decision to offer a quota for the amount of marginal resources to provide to the consumer. The post-interaction phase deals with all activities after executing SLA, which includes SLA monitoring and prediction and managing risk to avoid SLA violation. In this research, a viable SLA formation lifecycle that helps the service provider to form an intelligent viable SLA with the consumer is formed.

To the best of the knowledge of the researcher, viable SLA management has not been explored from the provider's side, particularly small and medium cloud providers, which helps the service provider to monitor and manage SLA even before formation and execution of SLA with the consumer.

9.3.3 Contribution 3: Methodology for Viable SLA Formation

The third contribution of this thesis is to propose a viable SLA framework that allows the cloud provider to intelligently offer the amount of marginal resources to consumers based on its transaction trend that ensure the maximum utilization of resources and maximum profit for the provider. To achieve this objective, Chapter 5 discusses the VSLAM methodology, which considers the T_{trend} of consumers and makes a decision whether or not to accept their request for resources and the amount of resources to offer to them. The module uses FIS and the inputs of the suitability of the consumer and the risk attitude of the provider to make these decisions.

To the best of knowledge of the researcher, there is no approach in the existing literature that will help the service provider to manage and monitor SLA in the pre-interaction time phase.

9.3.4 Contribution 4: Methodology for Selecting Cloud QoS Prediction

In Chapter 6, a detailed discussion of cloud QoS prediction was discussed that considered the previous QoS records. Neural network methods, stochastic methods and time series prediction methods were analysed at different time intervals to determine their prediction accuracy. The dataset was selected from the cloud dataset at 10 different time intervals.

To the best knowledge of the researcher, this kind of empirical investigation has not been performed before and there is no literature that analyses cloud QoS parameters to assist the cloud provider in cloud SLA management.

9.3.5 Contribution 5: Risk Management Framework to Avoid SLA Violations

In Chapter 7, the risk management framework that acts as an early warning and recommender module to assist the service provider in identifying and estimating the risk of SLA violations and suggesting the appropriate necessary actions to mitigate and manage it was presented. A FIS is used that considers three inputs: the risk attitude of the provider, the reliability of consumer and the predicted trajectory for making a decision concerning the appropriate action, which could be immediate action, delayed action or no action.

To the best knowledge of the researcher, there is no approach in the existing literature that can help the service provider manage the risk of SLA violations using FIS and also recommend the appropriate action to take concerning the risk.

9.3.6 Contribution 6: Evaluation of the Proposed Framework

To evaluate the effectiveness and applicability of the proposed framework, in Chapters 5, 6 and 7 each component of the viable SLA management framework was tested and evaluated using a QoS cloud dataset. Chapter 8 presented the implementation of a prototype with graphical user interface to demonstrate the overall working of the developed framework.

9.4 Conclusion and Future Work

Although significant research has been conducted on the topic of SLA management from the perspective of small and medium cloud providers, the researcher feels that there are still several future directions that will further strengthen the proposed framework for SLA management framework. Some of the areas for future research are as follows:

1. To intelligently model the relationship between low-level and mid-level resource usages with SLO i.e. finding hidden patterns between SLOs and low-level metrics using soft computing methods and intelligently combining low-level IaaS usage metrics with SLOs to generate likely service violations.
2. Develop techniques which enable new reliable cloud consumers to bootstrap themselves for the amount of fixed and marginal resources.
3. Develop a resource management mechanism so that the service provider can manage its resources.
4. Developing a business model for provider-side SLA management.
5. Completely implement a risk management framework for the cloud consumer based on different parameters of SLOs for the same SLOs and for different SLOs.

References

27005, I.I. 2008, 'Information technology–Security techniques-Information security risk management', *ISO/IEC*, vol. 66.

Aamodt, A. & Plaza, E. 1994, 'Case-based reasoning: Foundational issues, methodological variations, and system approaches', *AI communications*, vol. 7, no. 1, pp. 39-59.

Al Falasi, A., Serhani, M.A. & Dssouli, R. 2013, 'A Model for Multi-levels SLA Monitoring in Federated Cloud Environment', *Ubiquitous Intelligence and Computing, 2013 IEEE 10th International Conference on and 10th International Conference on Autonomic and Trusted Computing (UIC/ATC)*, IEEE, pp. 363-70.

AL-Allaf, O.N.A. 2012, 'Cascade-Forward vs. Function Fitting Neural Network for Improving Image Quality and Learning Time in Image Compression System', *Proceedings of the World Congress on Engineering*, vol. 2, pp. 4-6.

Albakri, S.H., Shanmugam, B., Samy, G.N., Idris, N.B. & Ahmed, A. 2014, 'Security risk assessment framework for cloud computing environments', *Security and Communication Networks*, vol. 7, no. 11, pp. 2114-24.

Alhamad, M., Dillon, T. & Chang, E. 2010, 'Sla-based trust model for cloud computing', *Network-Based Information Systems (NBiS), 2010 13th International Conference on*, IEEE, pp. 321-4.

Amazon Web Service - Amazon EC2 Service Level Agreement, Amazon, viewed 10 September 2015 2015, <https://aws.amazon.com/ec2/sla/>.

Andrews, B.H., Dean, M.D., Swain, R. & Cole, C. 2013, 'Building ARIMA and ARIMAX Models for Predicting Long-Term Disability Benefit Application Rates in the Public/Private Sectors'.

Antonescu, A.-F., Robinson, P. & Braun, T. 2013, 'Dynamic sla management with forecasting using multi-objective optimization', *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, IEEE, pp. 457-63.

Ardagna, C., Damiani, E., Frati, F., Montalbano, G., Rebeccani, D. & Ughetti, M. 2014, 'A Competitive Scalability Approach for Cloud Architectures', *Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on*, IEEE, pp. 610-7.

Armstrong, J.S. 2001, 'Extrapolation for time-series and cross-sectional data', *Principles of forecasting*, Springer, pp. 217-43.

Armstrong, J.S. & Forecasting, L.-R. 1985, 'From crystal ball to computer', *New York ua*.

Badidi, E. 2013, 'A cloud service broker for SLA-based SaaS provisioning', *Information Society (i-Society), 2013 International Conference on*, IEEE, pp. 61-6.

Bisgaard, S. & Kulahci, M. 2009, 'Quality Quandaries*: Time series model selection and parsimony', *Quality Engineering*, vol. 21, no. 3, pp. 341-53.

Box, G.E., Jenkins, G.M. & Reinsel, G.C. 2011, *Time series analysis: forecasting and control*, vol. 734, John Wiley & Sons.

Brandic, I., Emeakaroha, V.C., Maurer, M., Dustdar, S., Acs, S., Kertesz, A. & Kecskemeti, G. 2010, 'Laysi: A layered approach for sla-violation propagation in self-manageable cloud infrastructures', *Computer Software*

and Applications Conference Workshops (COMPSACW), 2010 IEEE 34th Annual, IEEE, pp. 365-70.
Breese, J.S., Heckerman, D. & Kadie, C. 1998, 'Empirical analysis of predictive algorithms for collaborative filtering', *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, Morgan Kaufmann Publishers Inc., pp. 43-52.
Brown, R.G. 1959, 'Statistical forecasting for inventory control'.
Burstein, F. & Gregor, S. 1999, 'The systems development or engineering approach to research in information systems: An action research perspective', *Proceedings of the 10th Australasian Conference on Information Systems*, Victoria University of Wellington, New Zealand, pp. 122-34.
Cabay, S. & Jackson, L. 1976, 'A polynomial extrapolation method for finding limits and antilimits of vector sequences', *SIAM Journal on Numerical Analysis*, vol. 13, no. 5, pp. 734-52.
Cardellini, V., Casalicchio, E., Lo Presti, F. & Silvestri, L. 2011, 'Sla-aware resource management for application service providers in the cloud', *Network Cloud Computing and Applications (NCCA), 2011 First International Symposium on*, IEEE, pp. 20-7.
Chandrasekar, A., Chandrasekar, K., Mahadevan, M. & Varalakshmi, P. 2012, 'QoS monitoring and dynamic trust establishment in the cloud', *Advances in Grid and Pervasive Computing*, Springer, pp. 289-301.
Chaudhuri, A., Maity, S. & Ghosh, S.K. 2015, 'QoS prediction for network data traffic using hierarchical modified regularized least squares rough support vector regression', *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, ACM, pp. 659-61.
Chauhan, T., Chaudhary, S., Kumar, V. & Bhise, M. 2011, 'Service level agreement parameter matching in cloud computing', *Information and Communication Technologies (WICT), 2011 World Congress on*, IEEE, pp. 564-70.
Cheetham, W., Varma, A. & Goebel, K. 2001, 'Case-Based Reasoning at General Electric', *FLAIRS Conference*, pp. 93-7.
Chopra, S. & Meindl, P. 2007, *Supply chain management. Strategy, planning & operation*, Springer.
Chua, D., Li, D. & Chan, W. 2001, 'Case-based reasoning approach in bid decision making', *Journal of construction engineering and management*, vol. 127, no. 1, pp. 35-45.
Ciciani, B., Didona, D., Di Sanzo, P., Palmieri, R., Peluso, S., Quaglia, F. & Romano, P. 2012, 'Automated workload characterization in cloud-based transactional data grids', *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International*, IEEE, pp. 1525-33.
Cicotti, G., Coppolino, L., Cristaldi, R., D'Antonio, S. & Romano, L. 2011, 'QoS monitoring in a cloud services environment: the SRT-15 approach', *European Conference on Parallel Processing*, Springer, pp. 15-24.
Cicotti, G., Coppolino, L., D'Antonio, S. & Romano, L. 2015, 'How to monitor QoS in cloud infrastructures: the QoSMONaaS approach', *International Journal of Computational Science and Engineering*, vol. 11, no. 1, pp. 29-45.
CloudClimate, *Watching the Cloud*, <http://www.cloudclimate.com/>.

Computing, C. & Creeger, M. 2009, 'Cloud computing: An overview', *Distributed Computing*, vol. 7, no. 5.
Consulting, S. 2015, 'A walk in the Cloud: how cloud technologies and cloud companies have affected the economy', viewed 15 January 2016, <http://www.soliantconsulting.com/blog/2015/11/the-economy-of-the-cloud>.
Dickey, D.A. & Fuller, W.A. 1979, 'Distribution of the estimators for autoregressive time series with a unit root', *Journal of the American statistical association*, vol. 74, no. 366a, pp. 427-31.
Egea, M., Mahbub, K., Spanoudakis, G. & Vieira, M.R. 2015, 'A Certification Framework for Cloud Security Properties: The Monitoring Path', *Accountability and Security in the Cloud*, Springer, pp. 63-77.
Elman, J.L. 1990, 'Finding structure in time', *Cognitive science*, vol. 14, no. 2, pp. 179-211.
Emeakaroha, V.C., Brandic, I., Maurer, M. & Dustdar, S. 2010, 'Low level metrics to high level SLAs-LoM2HiS framework: Bridging the gap between monitored metrics and SLA parameters in cloud environments', *High Performance Computing and Simulation (HPCS), 2010 International Conference on*, IEEE, pp. 48-54.
Emeakaroha, V.C., Calheiros, R.N., Netto, M.A., Brandic, I. & De Rose, C.A. 2010, 'DeSVi: an architecture for detecting SLA violations in cloud computing infrastructures', *Proceedings of the 2nd International ICST conference on Cloud computing (CloudComp'10)*, Citeseer.
Emeakaroha, V.C., Ferreto, T.C., Netto, M.A., Brandic, I. & De Rose, C.A. 2012, 'Casvid: Application level monitoring for sla violation detection in clouds', *Computer Software and Applications Conference (COMPSAC), 2012 IEEE 36th Annual*, IEEE, pp. 499-508.
Emeakaroha, V.C., Netto, M.A., Calheiros, R.N., Brandic, I., Buyya, R. & De Rose, C.A. 2012, 'Towards autonomic detection of SLA violations in Cloud infrastructures', *Future Generation Computer Systems*, vol. 28, no. 7, pp. 1017-29.
Etro, F. 2009, 'The economic impact of cloud computing on business creation, employment and output in Europe', *Review of Business and Economics*, vol. 54, no. 2, pp. 179-208.
Fachrunnisa, O. & Hussain, F.K. 2013, 'A methodology for maintaining trust in industrial digital ecosystems', *Industrial Electronics, IEEE Transactions on*, vol. 60, no. 3, pp. 1042-58.
Fan, W. & Perros, H. 2013, 'A reliability-based trust management mechanism for cloud services', *Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on*, IEEE, pp. 1581-6.
Feng, G. & Buyya, R. 2016, 'Maximum revenue-oriented resource allocation in cloud', *International Journal of Grid and Utility Computing*, vol. 7, no. 1, pp. 12-21.
Ferretti, S., Ghini, V., Panzieri, F., Pellegrini, M. & Turrini, E. 2010, 'Qos–aware clouds', *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, IEEE, pp. 321-8.
Galliers, R.D. 1990, 'Choosing appropriate information systems research approaches: a revised taxonomy', *In Proceedings of the IFIP TC8 WG8. 2*, Citeseer.

Gao, Y. & Er, M.J. 2005, 'NARMAX time series model prediction: feedforward and recurrent fuzzy neural network approaches', *Fuzzy sets and systems*, vol. 150, no. 2, pp. 331-50.
Gardner, E.S. 1985, 'Exponential smoothing: The state of the art', *Journal of forecasting*, vol. 4, no. 1, pp. 1-28.
Gartner, I. 2016, 'Forecast: Public Cloud Services, Worldwide, 2013-2019', Gartner <http://www.gartner.com/newsroom/id/3188817>.
GoGrid – A datapipe Company, Serverpath - SLA, viewed 10 September 2015 2015, <https://www.datapipe.com/gogrid/legal/servepath_sla/>.
Goldhaber, A.S. & Nieto, M.M. 2010, 'Photon and graviton mass limits', *Reviews of Modern Physics*, vol. 82, no. 1, p. 939.
Goyal, S. & Goyal, G.K. 2012, 'Elman backpropagation single hidden layer models for estimating shelf life of kalakand', *Advances in Information Technology and Management*, vol. 1, no. 3, pp. 127-31.
Group, T.O. 2016, *News*, The Open Group, Reading, Berkshire, viewed 08 September 2016 2016, <http://www.opengroup.org/news>.
Gu, Y., Zhang, W. & Tao, J. 2012, 'A study of SLA violation compensation mechanismin complex cloud computing environment', *Instrumentation, Measurement, Computer, Communication and Control (IMCCC), 2012 Second International Conference On*, IEEE, pp. 1448-51.
Hammadi, A.M. & Hussain, O. 2012, 'A framework for SLA assurance in cloud computing', *Advanced Information Networking and Applications Workshops (WAINA), 2012 26th International Conference on*, IEEE, pp. 393-8.
Haq, I.U., Brandic, I. & Schikuta, E. 2010, 'Sla validation in layered cloud infrastructures', *Economics of Grids, Clouds, Systems, and Services*, Springer, pp. 153-64.
Haq, I.U., Paschke, A., Schikuta, E. & Boley, H. 2009, 'Rule-based workflow validation of hierarchical service level agreements', *Grid and Pervasive Computing Conference, 2009. GPC'09. Workshops at the*, IEEE, pp. 96-103.
Heizer, J.H., Render, B. & Weiss, H.J. 2004, *Operations management*, vol. 8, Pearson Prentice Hall.
Herlocker, J.L., Konstan, J.A., Borchers, A. & Riedl, J. 1999, 'An algorithmic framework for performing collaborative filtering', *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, pp. 230-7.
Hussain, O. 2010, 'A Fuzzy Approach for Transactional Risk Management in E-Business Collaborations', *e-Business Engineering (ICEBE), 2010 IEEE 7th International Conference on*, IEEE, pp. 144-51.
Hussain, O.K., Chang, E., Hussain, F.K. & Dillon, T.S. 2007, 'A methodology to quantify failure for risk-based decision support system in digital business ecosystems', *Data & knowledge engineering*, vol. 63, no. 3, pp. 597-621.
Hussain, O.K., Chang, E., Hussain, F.K. & Dillon, T.S. 2008, 'Determining the Failure Level for Risk Analysis in an e-commerce Interaction', *Advances in Web Semantics I*, Springer, pp. 290-323.
Hussain, O.K., Dillon, T., Hussain, F.K. & Chang, E. 2012, *Risk assessment and management in the networked economy*, vol. 412, Springer.

Hussain, O.K., Hussain, F.K., Singh, J., Janjua, N.K. & Chang, E. 2014, 'A User-Based Early Warning Service Management Framework in Cloud Computing', *The Computer Journal*, p. bxu064.

Hussain, W., Hussain, F., Hussain, O. & Chang, E. 2016, 'Provider-based Optimized Personalized Viable SLA (OPV-SLA) Framework to Prevent SLA Violation', *The Computer Journal* p. 24.

Hussain, W., Hussain, F.K. & Hussain, O. 2016a, 'Allocating Optimized Resources in the Cloud by a Viable SLA Model', paper presented to the *Fuzzy Systems (FUZZ-IEEE), 2016 IEEE International Conference on. IEEE, 2016*, Vancouver, Canada July 2016.

Hussain, W., Hussain, F.K. & Hussain, O. 2016b, 'QoS Prediction Methods to Avoid SLA Violation in Post-Interaction Time Phase', paper presented to the *11th IEEE Conference on Industrial Electronics and Applications (ICIEA 2016)* Hefei, China, 5 - 7 June 2016.

Hussain, W., Hussain, F.K., Hussain, O. & Chang, E. 2015, 'Profile-based viable Service Level Agreement (SLA) Violation Prediction Model in the Cloud', paper presented to the *2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, Krakow, Poland, NOVEMBER 4-6, 2015.

Hussain, W., Hussain, F.K. & Hussain, O.K. 2014, 'Maintaining Trust in Cloud Computing through SLA Monitoring', *Neural Information Processing*, Springer, pp. 690-7.

Hussain, W., Hussain, F.K. & Hussain, O.K. 2015a, 'Comparative Analysis of Consumer Profile-based Methods to Predict SLA Violation', *IEEE International Conference on Fuzzy Systems*, IEEE, Istanbul, Turkey.

Hussain, W., Hussain, F.K. & Hussain, O.K. 2015b, 'Comparative analysis of consumer profile-based methods to predict SLA violation', *Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on*, IEEE, pp. 1-8.

Hyndman, R.J. & Khandakar, Y. 2007, *Automatic time series for forecasting: the forecast package for R*, Monash University, Department of Econometrics and Business Statistics.

Hyndman, R.J., Koehler, A.B., Snyder, R.D. & Grose, S. 2002, 'A state space framework for automatic forecasting using exponential smoothing methods', *International Journal of Forecasting*, vol. 18, no. 3, pp. 439-54.

Hyndman, R.J. & Kostenko, A.V. 2007, 'Minimum sample size requirements for seasonal forecasting models', *Foresight*, vol. 6, no. Spring, pp. 12-5.

The International Foundation for Information Technology, <http://www.if4it.com/SYNTHESIZED/GLOSSARY/S/Service_Level_Agreement_SLA_Lifecycle_Management.html>.

Islam, S., Keung, J., Lee, K. & Liu, A. 2012, 'Empirical prediction models for adaptive resource provisioning in the cloud', *Future Generation Computer Systems*, vol. 28, no. 1, pp. 155-62.

Jaramillo, G.E., Ardagna, C.A. & Anisetti, M. 2015, 'A hybrid representation model for service contracts', *Information and Communication Technology Research (ICTRC), 2015 International Conference on*, IEEE, pp. 246-9.

Jeffrey D. Camm, a.J.C., Michael J. Fry , effrey W. Ohlmann, David R. Anderson 2015, *Essentials of Business Analytics*, Cengage Learning.

Kalekar, P.S. 2004, 'Time series forecasting using holt-winters exponential smoothing', *Kanwal Rekhi School of Information Technology*, vol. 4329008, pp. 1-13.
Kaplan, B. & Maxwell, J.A. 2005, 'Qualitative research methods for evaluating computer information systems', *Evaluating the organizational impact of healthcare information systems*, Springer, pp. 30-55.
Katsaros, G., Kousiouris, G., Gogouvitis, S.V., Kyriazis, D., Menychtas, A. & Varvarigou, T. 2012, 'A Self-adaptive hierarchical monitoring mechanism for Clouds', *Journal of Systems and Software*, vol. 85, no. 5, pp. 1029-41.
Kiran, M., Jiang, M., Armstrong, D.J. & Djemame, K. 2011, 'Towards a service lifecycle based methodology for risk assessment in cloud computing', *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*, IEEE, pp. 449-56.
KIŞI, Ö. 2006, 'Generalized regression neural networks for evapotranspiration modelling', *Hydrological Sciences Journal*, vol. 51, no. 6, pp. 1092-105.
Leitner, P., Michlmayr, A., Rosenberg, F. & Dustdar, S. 2010, 'Monitoring, prediction and prevention of sla violations in composite services', *Web Services (ICWS), 2010 IEEE International Conference on*, IEEE, pp. 369-76.
Leitner, P., Wetzstein, B., Rosenberg, F., Michlmayr, A., Dustdar, S. & Leymann, F. 2010, 'Runtime prediction of service level agreement violations for composite services', *Service-Oriented Computing. ICSOC/ServiceWave 2009 Workshops*, Springer, pp. 176-86.
Liu, H., Xu, D. & Miao, H.K. 2011, 'Ant colony optimization based service flow scheduling with various QoS requirements in cloud computing', *Software and Network Engineering (SSNE), 2011 First ACIS International Symposium on*, IEEE, pp. 53-8.
Liu, Z., Squillante, M.S. & Wolf, J.L. 2001, 'On maximizing service-level-agreement profits', *Proceedings of the 3rd ACM conference on Electronic Commerce*, ACM, pp. 213-23.
Lo, W., Yin, J., Li, Y. & Wu, Z. 2015, 'Efficient web service QoS prediction using local neighborhood matrix factorization', *Engineering Applications of Artificial Intelligence*, vol. 38, pp. 14-23.
Lu, K., Yahyapour, R., Wieder, P., Yaqub, E., Abdullah, M., Schloer, B. & Kotsokalis, C. 2015, 'Fault-tolerant Service Level Agreement lifecycle management in clouds using actor system', *Future Generation Computer Systems*.
Macháček, M. 2014, 'CLOUD COMPUTING AND SECURITY OF INFORMATION ASSETS', *European Scientific Journal, ESJ*, vol. 10, no. 10.
Makridakis, S. 1993, 'Accuracy measures: theoretical and practical concerns', *International Journal of Forecasting*, vol. 9, no. 4, pp. 527-9.
Mamdani, E.H. & Assilian, S. 1975, 'An experiment in linguistic synthesis with a fuzzy logic controller', *International journal of man-machine studies*, vol. 7, no. 1, pp. 1-13.
Marudhadevi, D., Dhatchayani, V.N. & Sriram, V.S. 2014, 'A Trust Evaluation Model for Cloud Computing Using Service Level Agreement', *The Computer Journal*, p. bxu129.
Mayer, R.C., Davis, J.H. & Schoorman, F.D. 1995, 'An integrative model of organizational trust', *Academy of management review*, vol. 20, no. 3, pp. 709-34.

McTavish, D.G. & Loether, H.J. 2002, *Social research: An evolving process*, Allyn and Bacon.
Mell, P. & Grance, T. 2009, 'The NIST definition of cloud computing', *National Institute of Standards and Technology*, vol. 53, no. 6, p. 50.
Mell, P. & Grance, T. 2011, 'The NIST definition of cloud computing (draft)', *NIST special publication*, vol. 800, no. 145, p. 7.
Messina, F., Pappalardo, G., Santoro, C., Rosaci, D. & Sarné, G.M. 2016, 'A multi-agent protocol for service level agreement negotiation in cloud federations', *International Journal of Grid and Utility Computing*, vol. 7, no. 2, pp. 101-12.
Microsoft Windows Azure - Service Level Agreement, Microsoft, viewed 10 September 2015 2015, <http://azure.microsoft.com/en-us/support/legal/sla/>.
Miller, R.B. & Hickman, J.C. 1973, 'Time series analysis and forecasting', *Transactions of the Society of Actuaries*, vol. 25, pp. 267-302.
Monitor, P.N., PRTG Network Monitor <https://prtg.paessler.com/>.
Morin, J.-H., Aubert, J. & Gateau, B. 2012, 'Towards cloud computing SLA risk management: issues and challenges', *System Science (HICSS), 2012 45th Hawaii International Conference on*, IEEE, pp. 5509-14.
Mosallanejad, A. & Atan, R. 2013, 'HA-SLA: A Hierarchical Autonomic SLA Model for SLA Monitoring in Cloud Computing', *Journal of Software Engineering and Applications*, vol. 6, no. 03, p. 114.
Muchahari, M.K. & Sinha, S.K. 2012, 'A new trust management architecture for cloud computing environment', *Cloud and Services Computing (ISCOS), 2012 International Symposium on*, IEEE, pp. 136-40.
Myers, D.E. 1994, 'Spatial interpolation: an overview', *Geoderma*, vol. 62, no. 1, pp. 17-28.
Noor, T.H. & Sheng, Q.Z. 2011, 'Trust as a service: a framework for trust management in cloud environments', *Web Information System Engineering–WISE 2011*, Springer, pp. 314-21.
Nunamaker Jr, J.F., Chen, M. & Purdin, T.D. 1990, 'Systems development in information systems research', *Journal of management information systems*, vol. 7, no. 3, pp. 89-106.
Ord, J.K., Koehler, A. & Snyder, R.D. 1997, 'Estimation and prediction for a class of dynamic nonlinear statistical models', *Journal of the American Statistical Association*, vol. 92, no. 440, pp. 1621-9.
Pacheco-Sanchez, S., Casale, G., Scotney, B., McClean, S., Parr, G. & Dawson, S. 2011, 'Markovian workload characterization for qos prediction in the cloud', *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, IEEE, pp. 147-54.
Pavlou, P.A. 2003, 'Consumer acceptance of electronic commerce: Integrating trust and risk with the technology acceptance model', *International journal of electronic commerce*, vol. 7, no. 3, pp. 101-34.
Qi, K., Hu, H., Song, W., Ge, J. & Lu, J. 2015, 'Personalized QoS Prediction via Matrix Factorization Integrated with Neighborhood Information', *Services Computing (SCC), 2015 IEEE International Conference on*, IEEE, pp. 186-93.
Qian, L., Luo, Z., Du, Y. & Guo, L. 2009, 'Cloud computing: an overview', *IEEE International Conference on Cloud Computing*, Springer, pp. 626-31.

Quillinan, T.B., Clark, K.P., Warnier, M., Brazier, F.M. & Rana, O. 2010, 'Negotiation and monitoring of service level agreements', *Grids and Service-Oriented Architectures for Service Level Agreements*, Springer, pp. 167-76.
Rana, O.F., Warnier, M., Quillinan, T.B., Brazier, F. & Cojocarasu, D. 2008, 'Managing violations in service level agreements', *Grid Middleware and Services*, Springer, pp. 349-58.
Rehman, Z.-u., Hussain, O.K. & Hussain, F.K. 2015, 'User-side cloud service management: state-of-the-art and future directions', *Journal of Network and Computer Applications*.
Romano, L., De Mari, D., Jerzak, Z. & Fetzer, C. 2011, 'A novel approach to QoS monitoring in the cloud', *Data Compression, Communications and Processing (CCP), 2011 First International Conference on*, IEEE, pp. 45-51.
Ron, S. & Aliko, P. 2001, 'Service level agreements', *Internet NG project*.
Sahal, R., Khafagy, M.H. & Omara, F.A. 2016, 'A Survey on SLA Management for Cloud Computing and Cloud-Hosted Big Data Analytic Applications', *International Journal of Database Theory and Application*, vol. 9, no. 4, pp. 107-18.
Sakr, S. & Liu, A. 2012, 'SLA-based and consumer-centric dynamic provisioning for cloud databases', *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, IEEE, pp. 360-7.
Saunders, H.H. 1985, 'We need a larger theory of negotiation: The importance of pre-negotiating phases', *Negotiation journal*, vol. 1, no. 3, pp. 249-62.
Schmieders, E., Micsik, A., Oriol, M., Mahbub, K. & Kazhamiakin, R. 2011, 'Combining SLA prediction and cross layer adaptation for preventing SLA violations'.
Schroeder, R.G., Goldstein, S.M. & Rungtusanatham, M.J. 2013, *Operations management in the supply chain: Decisions and cases*.
Subashini, S. & Kavitha, V. 2011, 'A survey on security issues in service delivery models of cloud computing', *Journal of network and computer applications*, vol. 34, no. 1, pp. 1-11.
Sun, L., Dong, H., Hussain, F.K., Hussain, O.K. & Chang, E. 2014, 'Cloud service selection: State-of-the-art and future research directions', *Journal of Network and Computer Applications*, vol. 45, pp. 134-50.
Sun, X., Chang, G. & Li, F. 2011, 'A trust management model to enhance security of cloud computing environments', *Networking and Distributed Computing (ICNDC), 2011 Second International Conference on*, IEEE, pp. 244-8.
Sun, Y., Tan, W., Li, L., Lu, G. & Tang, A. 2013, 'SLA detective control model for workflow composition of cloud services', *Computer Supported Cooperative Work in Design (CSCWD), 2013 IEEE 17th International Conference on*, IEEE, pp. 165-71.
Walayat Hussain, F.K.H., Omar Hussain 2015, 'Comparative analysis of consumer profile-based methods to predict SLA violation', paper presented to the *Fuzz- IEEE*, Turkey
Wang, H., Liu, F. & Liu, H. 2012, 'A method of the cloud computing security management risk assessment', *Advances in Computer Science and Engineering*, Springer, pp. 609-18.
Wang, J., De Vries, A.P. & Reinders, M.J. 2006, 'Unifying user-based and item-based collaborative filtering approaches by similarity fusion', *Proceedings*

of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, ACM, pp. 501-8.
Wang, M., Wu, X., Zhang, W., Ding, F., Zhou, J. & Pei, G. 2011, 'A conceptual platform of SLA in cloud computing', *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*, IEEE, pp. 1131-5.
Wang, P., Lin, W.-H., Kuo, P.-T., Lin, H.-T. & Wang, T.C. 2012, 'Threat risk analysis for cloud security based on Attack-Defense Trees', *Computing Technology and Information Management (ICCM), 2012 8th International Conference on*, vol. 1, IEEE, pp. 106-11.
Waters, D. 2011, *Supply chain risk management: vulnerability and resilience in logistics*, Kogan Page Publishers.
Williams, C.A. & Heins, R.M. 1985, *Risk management and insurance*, McGraw-Hill Companies.
Wood, T., Shenoy, P., Venkataramani, A. & Yousif, M. 2009, 'Sandpiper: Black-box and gray-box resource management for virtual machines', *Computer Networks*, vol. 53, no. 17, pp. 2923-38.
Wu, L. & Buyya, R. 2012, 'Service level agreement (sla) in utility computing systems', *IGI Global*.
Wu, L., Garg, S.K. & Buyya, R. 2011, 'SLA-based resource allocation for software as a service provider (SaaS) in cloud computing environments', *Cluster, Cloud and Grid Computing (CCGrid), 2011 11th IEEE/ACM International Symposium on*, IEEE, pp. 195-204.
Wustenhoff, E. & BluePrints, S. 2002, 'Service level agreement in the data center', *Sun BluePrints*.
Yun, S. 2013, 'Guaranteed QoS Resource Scheduling Scheme Based on Improved Electromagnetism-Like Mechanism Algorithm in Cloud Environment', *Intelligent Networking and Collaborative Systems (INCoS), 2013 5th International Conference on*, IEEE, pp. 353-7.
Zadeh, L.A. 1965, 'Fuzzy sets', *Information and control*, vol. 8, no. 3, pp. 338-53.
Zadeh, L.A. 1973, 'Outline of a new approach to the analysis of complex systems and decision processes', *Systems, Man and Cybernetics, IEEE Transactions on*, no. 1, pp. 28-44.
Zhang, X., Wuwong, N., Li, H. & Zhang, X. 2010, 'Information security risk management framework for the cloud computing environments', *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, IEEE, pp. 1328-34.
Zhang, Y., Zheng, Z. & Lyu, M.R. 2011a, 'Exploring latent features for memory-based QoS prediction in cloud computing', *Reliable Distributed Systems (SRDS), 2011 30th IEEE Symposium on*, IEEE, pp. 1-10.
Zhang, Y., Zheng, Z. & Lyu, M.R. 2011b, 'WSPred: A time-aware personalized QoS prediction framework for Web services', *Software Reliability Engineering (ISSRE), 2011 IEEE 22nd International Symposium on*, IEEE, pp. 210-9.
Zhu, C., Nicanfar, H., Leung, V.C. & Yang, L.T. 2015, 'An Authenticated Trust and Reputation Calculation and Management System for Cloud and Sensor Networks Integration', *Information Forensics and Security, IEEE Transactions on*, vol. 10, no. 1, pp. 118-31.
Ziegel, E.R. 1994, 'Forecasting and time series: An applied approach', *Technometrics*, vol. 36, no. 4, pp. 434-.

Appendix A – Prediction Results Using SES Method

Simple Exponential Smoothing for CPU

MSE, RMSE and MAD with nine possible values of alpha are presented in Table A-1. From the Table A-1, we see that for each input series of different time intervals, different values for the smoothing constant generates an optimal result at different time intervals.

Table A1: Forecasting by Simple Exponential Smoothing method for CPU values

Time interval		Method 1:Simple Exponential Smoothing – CPU								
5 minutes	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	5,814.78	6,127.30	6,477.62	6,871.47	7,317.54	7,827.20	8,415.24	9,101.31	9,912.08
	RMSE	76.25	78.28	80.48	82.89	85.54	88.47	91.73	95.40	99.56
	MAD	9.92	9.93	9.98	10.05	10.13	10.20	10.25	10.30	10.35
10 minutes	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	43.57	43.67	44.80	46.53	48.78	51.55	54.87	58.84	63.57
	RMSE	6.60	6.61	6.69	6.82	6.98	7.18	7.41	7.67	7.97
	MAD	4.59	4.57	4.57	4.58	4.62	4.67	4.75	4.82	4.90

Time interval		Method 1:Simple Exponential Smoothing – CPU								
20 minutes	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	27.99	27.15	27.34	27.98	28.94	30.22	31.82	33.80	36.23
	RMSE	5.29	5.21	5.23	5.29	5.38	5.50	5.64	5.81	6.02
	MAD	3.81	3.72	3.70	3.69	3.71	3.77	3.83	3.91	4.00
1 hour	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	19.81	16.19	15.43	15.39	15.68	16.22	16.97	17.95	19.19
	RMSE	4.45	4.02	3.93	3.92	3.96	4.03	4.12	4.24	4.38
	MAD	3.22	2.87	2.81	2.81	2.85	2.90	2.97	3.04	3.15
4 hours	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	420.83	413.31	424.71	445.18	472.04	504.88	544.17	590.98	646.94
	RMSE	20.51	20.33	20.61	21.10	21.73	22.47	23.33	24.31	25.44
	MAD	8.10	6.76	6.11	5.80	5.67	5.58	5.53	5.50	5.50
12 hours	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	238.79	216.12	206.02	201.68	201.89	206.24	214.49	226.63	242.95
	RMSE	15.45	14.70	14.35	14.20	14.21	14.36	14.65	15.05	15.59
	MAD	9.96	8.69	7.87	7.28	6.90	6.59	6.38	6.21	6.21
	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90

Time interval		Method 1:Simple Exponential Smoothing – CPU								
1 day	MSE	90,26.24	94,89.07	101,39.25	108,80.19	116,93.45	125,74.30	135,25.61	145,51.94	156,63.40
	RMSE	30.43	30.04	31.42	32.85	34.95	35.61	36.77	38.47	39.77
	MAD	15.65	17.75	18.69	19.51	20.24	21.61	22.76	24.72	25.79
1 week	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	12,924.12	7,157.09	4,981.36	3,898.21	3,263.83	2,856.92	2,581.83	2,390.80	2,258.07
	RMSE	113.68	84.60	70.58	62.44	57.13	53.45	50.81	48.90	47.52
	MAD	80.72	54.21	41.05	33.92	29.77	26.90	24.82	23.39	22.42
2 weeks	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	11,404.40	6,619.06	4,561.19	3,494.42	2,851.81	2,425.29	2,125.19	1,907.26	1,747.38
	RMSE	106.79	81.36	67.54	59.11	53.40	49.25	46.10	43.67	41.80
	MAD	80.53	50.26	36.35	28.72	24.57	21.90	20.12	18.77	17.86
4 weeks	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	16,357.11	11,822.96	8,864.75	7,004.97	5,809.26	5,005.89	4,443.18	4,036.18	3,736.51
	RMSE	127.89	108.73	94.15	83.70	76.22	70.75	66.66	63.53	61.13
	MAD	109.07	81.40	62.50	49.87	41.57	35.57	31.29	28.07	25.84

Simple Exponential Smoothing for Memory

The values of MAD, MSE and RMSE for the 10 time intervals for the parameter memory are presented in Table A-2.

Table A2: Simple Exponential Smoothing method for memory values

Time interval		Method 1: Simple Exponential Smoothing – Memory								
5 minutes	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	12237648.40	12693561.57	13308104.48	14051626.68	14919796.91	15926276.58	17098431.06	18476282.73	20114942.86
	RMSE	3,498.24	3,562.80	3,648.03	3,748.55	3,862.62	3,990.77	4,135.02	4,298.40	4,484.97
	MAD	694.30	647.55	628.59	622.50	618.02	616.33	616.20	617.58	620.53
10 minutes	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	7008225.09	7209967.45	7477853.75	7821441.69	8246062.59	8759249.94	9371647.72	10098380.18	10961079.87
	RMSE	2,647.31	2,685.14	2,734.57	2,796.68	2,871.60	2,959.60	3,061.31	3,177.79	3,310.75
	MAD	748.76	704.26	679.03	664.91	654.71	652.48	653.68	655.24	658.07
20 minutes	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	4395116.12	4549645.36	4708783.10	4896205.28	5123693.10	5400615.01	5738191.27	6151390.23	6660605.38
	RMSE	2,096.45	2,132.99	2,169.97	2,212.74	2,263.56	2,323.92	2,395.45	2,480.20	2,580.81
	MAD	801.46	769.94	743.50	727.21	718.48	712.96	709.20	706.99	709.55

Time interval		Method 1: Simple Exponential Smoothing – Memory								
1 hour	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	1952440.66	2082603.56	2215721.48	2354716.98	2503486.98	2666616.73	2850059.82	3061662.53	3311580.33
	RMSE	1,397.30	1,443.12	1,488.53	1,534.51	1,582.24	1,632.98	1,688.21	1,749.76	1,819.77
	MAD	763.71	785.54	793.91	798.37	798.79	797.11	796.79	795.91	795.96
4 hours	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	466,636.37	495,761.66	532,284.34	575,730.29	625,461.58	681,192.04	743,158.15	812,166.84	889,623.03
	RMSE	683.11	704.10	729.58	758.77	790.86	825.34	862.07	901.20	943.20
	MAD	481.46	503.68	524.97	547.03	569.36	591.10	614.37	636.38	655.35
12 hours	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	126,211.77	126,530.73	131,598.83	138,403.25	146,962.66	157,762.50	171,458.63	188,869.44	211,054.37
	RMSE	355.26	355.71	362.77	372.03	383.36	397.19	414.08	434.59	459.41
	MAD	239.25	252.70	263.10	274.39	284.24	293.78	304.15	317.42	335.14
1 day	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	36,542.23	37,717.02	39,012.72	40,645.79	42,651.31	45,045.17	47,859.67	51,159.36	55,053.47
	RMSE	191.16	194.21	197.52	201.61	206.52	212.24	218.77	226.18	234.63
	MAD	89.91	95.74	102.01	106.17	108.87	111.19	112.83	113.81	114.49
	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90

Time interval		Method 1: Simple Exponential Smoothing – Memory								
1 week	MSE	77,962.25	57,455.82	49,561.98	46,850.06	46,576.34	47,677.22	49,718.34	52,557.22	56,239.68
	RMSE	279.22	239.70	222.63	216.45	215.82	218.35	222.98	229.25	237.15
	MAD	208.78	176.51	168.22	164.48	162.80	162.57	163.27	164.50	166.41
2 weeks	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	36,344.13	37,241.95	28,843.48	23,993.54	21,503.38	20,389.44	20,191.53	20,700.69	21,846.50
	RMSE	239.88	195.56	169.83	154.90	146.64	142.79	142.10	143.88	147.81
	MAD	184.58	149.07	129.89	118.31	111.49	110.22	111.27	113.65	117.33
4 weeks	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	73,315.65	56,055.52	44,486.39	36,512.09	30,909.62	26,926.83	24,092.28	22,099.93	20,753.18
	RMSE	270.77	236.76	210.92	191.08	175.81	164.09	155.22	148.66	144.06
	MAD	225.14	181.44	156.83	137.15	122.27	114.01	106.76	100.99	97.09

Simple Exponential Smoothing for I/O

The values of MAD, MSE and RMSE for the 10 time intervals for the parameter I/O are presented in Table 6.5.

Table A3: Simple Exponential Smoothing method for I/O values

Time Interval interval		Method 1: Simple Exponential Smoothing – I/O								
5 minutes	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	35019.96	35725.05	37024.43	38637.74	40536.73	42749.33	45332.83	48372.44	51988.51
	RMSE	187.14	189.01	192.42	196.56	201.34	206.76	212.92	219.94	228.01
	MAD	66.78	65.78	66.07	66.56	67.34	68.40	69.47	70.72	72.18
10 minutes	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	22113.70	21917.42	22447.51	23302.51	24389.04	25693.38	27234.87	29055.71	31221.95
	RMSE	148.71	148.05	149.82	152.65	156.17	160.29	165.03	170.46	176.70
	MAD	66.62	62.86	62.00	62.11	62.67	63.18	64.08	65.13	66.24
20 minutes	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	15781.12	15148.11	15225.24	15665.98	16364.98	17290.25	18445.87	19859.94	21583.16
	RMSE	125.62	123.08	123.39	125.16	127.93	131.49	135.82	140.93	146.91
	MAD	67.30	62.39	60.47	59.62	59.50	59.93	60.80	61.97	63.44
	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	10406.95	9026.55	8537.93	8334.35	8274.60	8323.02	8471.16	8720.20	9077.75

Time Interval interval		Method 1: Simple Exponential Smoothing – I/O								
1 hour	RMSE	102.01	95.01	92.40	91.29	90.96	91.23	92.04	93.38	95.28
	MAD	71.78	63.40	60.68	59.42	58.78	58.23	57.99	57.91	58.04
4 hours	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	.80	0.90
	MSE	78195.80	78519.58	81090.08	85256.75	90700.52	97250.03	104864.71	113650.49	123903.50
	RMSE	279.64	280.21	284.76	291.99	301.17	311.85	323.83	337.12	352.00
	MAD	109.42	104.46	101.31	100.13	99.57	100.00	100.58	101.53	103.79
12 hours	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	32481.88	31207.66	30755.31	30869.58	31483.38	32617.33	34343.00	36773.74	40070.03
	RMSE	180.23	176.66	175.37	175.70	177.44	180.60	185.32	191.76	200.17
	MAD	94.85	98.28	96.71	95.25	94.73	94.40	94.76	95.35	96.53
1 day	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	105.92	56.00	39.53	31.50	26.88	24.00	22.16	21.00	20.36
	RMSE	10.29	7.48	6.29	5.61	5.18	4.90	4.71	4.58	4.51
	MAD	7.72	4.03	2.69	2.02	1.61	1.34	1.15	1.01	0.90
	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90

Time Interval interval		Method 1: Simple Exponential Smoothing – I/O								
1 week	MSE	18485.61	19604.21	20882.09	22377.86	24089.96	25992.25	28058.75	30282.19	32691.46
	RMSE	135.96	140.02	144.51	149.59	155.21	161.22	167.51	174.02	180.81
	MAD	97.02	99.35	101.44	106.19	111.51	115.78	120.08	123.79	126.13
2 weeks	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	8965.55	8269.53	7954.97	7890.94	8032.84	8380.42	8957.33	9803.70	10980.47
	RMSE	94.69	90.94	89.19	88.83	89.63	91.54	94.64	99.01	104.79
	MAD	76.50	74.65	72.38	70.53	69.76	70.39	72.14	75.30	79.98
4 weeks	Alpha (α)	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
	MSE	6047.16	5341.67	4976.40	4695.40	4472.69	4295.93	4149.34	4019.77	3900.34
	RMSE	77.76	73.09	70.54	68.52	66.88	65.54	64.42	63.40	62.45
	MAD	63.83	61.16	60.42	59.13	58.37	57.66	57.44	56.87	55.88

Appendix B – Prediction Results Using SMA Method

Simple Moving Average for CPU

CPU data is analysed with different values of *k* in 10 time intervals. Table B1 shows the MAD, MSE and RMSE values for CPU prediction for the 10 time intervals with different values of *k*. From the table, we see that when the value of *k* increases, prediction accuracy decreases.

Table B1: Simple moving average for CPU at 10 time intervals with different values of *k*

Time interval		Method 2: Simple moving average method – CPU											
5 minutes	*k*	2	193	384	575	766	957	1148	1339	1530	1721	1912	
	MSE	3669.63	5486.13	5511.18	5514.30	5516.05	5519.17	5521.13	5521.29	5522.52	5522.39	5522.50	
	RMSE	60.58	74.07	74.24	74.26	74.27	74.29	74.30	74.31	74.31	74.31	74.31	
	MAD	6.89	10.46	10.89	10.80	10.78	10.79	10.78	10.76	10.75	10.74	10.74	
10 minutes	*k*	2	98	194	290	386	482	578	674	770	866		
	MSE	24.57	49.25	58.59	66.89	73.59	78.85	82.00	83.95	85.45	86.52		
	RMSE	4.96	7.02	7.65	8.18	8.58	8.88	9.06	9.16	9.24	9.30		
	MAD	3.22	5.13	5.90	6.37	6.67	6.86	6.98	7.04	7.09	7.13		

Time interval		Method 2: Simple moving average method – CPU											
20 minutes	*k*	2	50	98	146	194	242	290	338	386	434		
	MSE	14.41	32.44	41.59	49.72	56.46	61.70	64.80	66.76	68.25	69.28		
	RMSE	3.80	5.70	6.45	7.05	7.51	7.86	8.05	8.17	8.26	8.32		
	MAD	2.59	4.15	4.86	5.32	5.61	5.81	5.93	6.00	6.05	6.08		
1 hour	*k*	2	18	34	50	66	82	98	114	130	146		
	MSE	7.65	19.17	28.30	36.10	42.79	47.78	50.63	52.63	54.06	55.09		
	RMSE	2.77	4.38	5.32	6.01	6.54	6.91	7.12	7.25	7.35	7.42		
	MAD	2.01	3.04	3.78	4.23	4.53	4.73	4.82	4.89	4.94	4.98		
4 hours	*k*	2	20	38	56	74	92	110	128	146	164	182	
	MSE	231.35	409.87	439.66	443.32	460.68	473.80	487.32	488.28	492.66	494.30	493.68	
	RMSE	15.21	20.25	20.97	21.06	21.46	21.77	22.08	22.10	22.20	22.23	22.22	
	MAD	3.84	8.59	9.62	10.41	11.21	12.05	12.95	13.12	13.32	13.39	13.34	
12 hours	*k*	2	8	14	20	26	32	38	44	50	56		
	MSE	93.42	193.83	207.02	224.08	234.69	247.27	256.59	259.14	263.17	263.29		
	RMSE	9.67	13.92	14.39	14.97	15.32	15.72	16.02	16.10	16.22	16.23		
	MAD	4.33	8.12	8.97	10.04	10.80	11.67	12.44	12.55	12.73	12.76		
	k	2	5	8	11	14	17	20	23	26	29		

Time interval		Method 2: Simple moving average method – CPU											
1 day	MSE	63670.95	66687.11	73782.67	69522.18	70676.63	69877.87	70270.89	70131.15	70191.73	70173.93		
	RMSE	252.33	258.24	271.63	263.67	265.85	264.34	265.09	264.82	264.94	264.90		
	MAD	152.96	155.50	156.29	145.37	147.34	147.11	150.36	148.85	149.32	148.99		
1 week	*k*	2	4	6	8	10	12	14	16	18	20	22	24
	MSE	1359.15	2989.67	4749.10	6522.72	8352.68	10058.03	11736.31	13191.20	14320.20	15130.10	15643.45	15901.79
	RMSE	36.87	54.68	68.91	80.76	91.39	100.29	108.33	114.85	119.67	123.00	125.07	126.10
	MAD	17.71	27.04	38.25	47.96	58.30	67.08	75.85	82.29	86.87	90.00	91.96	92.98
2 weeks	*k*	2	12	22	32	42	52	62	72	82	92	102	
	MSE	1120.64	9354.04	15450.27	17891.26	17376.50	14254.08	13482.06	13978.80	14527.24	14810.67	14411.74	
	RMSE	33.48	96.72	124.30	133.76	131.82	119.39	116.11	118.23	120.53	121.70	120.05	
	MAD	13.81	61.54	95.35	111.81	115.29	105.62	101.71	103.06	105.22	107.00	106.04	
4 weeks	*k*	2	7	12	17	22	27	32	37	42	47	52	
	MSE	2367.97	10155.94	15324.11	17370.00	15914.55	13084.02	12931.61	13501.46	14055.19	14161.09	13799.12	
	RMSE	48.66	100.78	123.79	131.80	126.15	114.39	113.72	116.20	118.55	119.00	117.47	
	MAD	22.35	68.63	97.97	111.98	109.86	100.06	98.84	100.60	102.95	104.08	103.10	

Simple Moving Average for Memory

The memory data are analysed with different values of *k*. The values of MAD, MSE and RMSE at the 10 time intervals for memory are presented in Table B2.

Table B2: Simple moving average for memory at 10 time intervals with different values of *k*

Time interval		Method 2: Simple moving average – memory											
5 minutes	*k*	2	193	384	575	766	957	1148	1339	1530	1721	1912	
	MSE	7432322.5	11842225.5	11845699.5	11844949.2	11844562.0	11845075.5	11846606.1	11848788.7	11847002.4	11848379.1	11847224.3	
	RMSE	2726.2	3441.3	3441.8	3441.6	3441.6	3441.7	3441.9	3442.2	3441.9	3442.1	3442.0	
	MAD	414.9	777.2	780.4	773.7	770.7	769.9	768.7	767.5	765.8	765.4	763.9	
10 minutes	*k*	2	98	194	290	386	482	578	674	770	866		
	MSE	4156421.1	6710051.9	6719369.3	6708553.3	6713690.5	6715133.9	6715552.8	6720013.0	6717607.5	6718042.2		
	RMSE	2038.7	2590.4	2592.2	2590.1	2591.1	2591.4	2591.4	2592.3	2591.8	2591.9		
	MAD	447.6	783.7	786.8	775.6	772.2	770.8	769.2	767.9	765.8	765.0		
	k	2	50	98	146	194	242	290	338	386	434		
	MSE	2478822.1	4125006.9	4150012.8	4133217.8	4141057.8	4141024.3	4140918.3	4144407.7	4144217.9	4144736.2		

Time interval		Method 2: Simple moving average – memory											
20 minutes	RMSE	1574.4	2031.0	2037.2	2033.0	2035.0	2035.0	2034.9	2035.8	2035.7	2035.9		
	MAD	474.4	797.6	800.1	782.9	778.3	775.4	772.5	770.8	768.3	766.9		
1 hour	*k*	2	18	34	50	66	82	98	114	130	146		
	MSE	1268128.0	1756327.6	1763735.8	1765507.3	1770026.5	1773348.3	1769840.7	1772522.3	1775691.9	1775233.3		
	RMSE	1126.1	1325.3	1328.1	1328.7	1330.4	1331.7	1330.4	1331.4	1332.6	1332.4		
	MAD	533.7	726.4	726.3	712.1	707.6	706.9	703.8	702.1	699.9	698.5		
4 hours	*k*	2	20	38	56	74	92	110	128	146	164	182	
	MSE	333686.6	415307.8	420234.3	424898.6	421423.3	422565.7	422261.2	422297.8	422195.9	422040.4	422054.8	
	RMSE	577.7	644.4	648.3	651.8	649.2	650.1	649.8	649.8	649.8	649.6	649.7	
	MAD	418.2	457.3	457.0	460.6	457.9	458.7	459.0	459.8	459.2	458.8	458.8	
12 hours	*k*	2	8	14	20	26	32	38	44	50	56		
	MSE	60694.9	94019.1	97338.8	99157.9	100245.1	100181.1	100405.4	100075.4	99968.7	99944.9		
	RMSE	246.4	306.6	312.0	314.9	316.6	316.5	316.9	316.3	316.2	316.1		
	MAD	184.7	213.5	216.4	219.1	215.3	213.1	212.7	212.9	213.4	213.6		

Time interval		Method 2: Simple moving average – memory											
1 day	*k*	2	5	8	11	14	17	20	23	26	29		
	MSE	21788.1	28173.7	31788.4	32452.3	32830.9	31935.9	32273.6	31775.0	31612.5	31520.3		
	RMSE	147.6	167.9	178.3	180.1	181.2	178.7	179.6	178.3	177.8	177.5		
	MAD	76.8	86.0	85.8	86.3	89.9	87.8	88.0	85.5	84.6	84.3		
1 week	*k*	2	4	6	8	10	12	14	16	18	20	22	24
	MSE	21834.3	25614.6	34403.4	42171.9	49851.7	60052.9	68631.4	76760.8	82348.2	86969.4	89931.7	91480.0
	RMSE	147.8	160.0	185.5	205.4	223.3	245.1	262.0	277.1	287.0	294.9	299.9	302.5
	MAD	105.7	117.7	148.9	159.9	173.2	186.7	206.3	221.7	233.8	241.8	247.0	249.4
2 weeks	*k*	2	12	22	32	42	52	62	72	82	92	102	
	MSE	8325.9	54371.8	71204.9	74795.4	77754.9	65605.6	63228.9	65203.4	66054.6	68005.4	66358.4	
	RMSE	91.2	233.2	266.8	273.5	278.8	256.1	251.5	255.3	257.0	260.8	257.6	
	MAD	68.1	178.7	208.7	224.4	231.7	213.9	207.3	210.8	212.6	216.7	214.7	
4 weeks	*k*	2	7	12	17	22	27	32	37	42	47	52	
	MSE	12359.2	52187.9	63495.9	70194.3	68340.5	56820.4	57368.0	58768.2	60335.7	61393.1	59862.6	
	RMSE	111.2	228.4	252.0	264.9	261.4	238.4	239.5	242.4	245.6	247.8	244.7	
	MAD	76.2	172.3	197.7	221.3	219.8	200.4	199.2	202.1	205.5	208.2	206.2	

Simple Moving Average for I/O

The values of MAD, MSE and RMSE at 10 time intervals for I/O are presented in Table B3.

Table B3: Simple moving average for I/O at 10 time intervals with different values of *k*

Time interval		Method 2: Simple moving average – I/O											
5 minutes	*k*	2	193	384	575	766	957	1148	1339	1530	1721	1912	
	MSE	20609.07	39825.54	42941.17	43409.40	42868.96	41832.09	41830.64	42203.56	42311.62	42175.57	42123.74	
	RMSE	143.56	199.56	207.22	208.35	207.05	204.53	204.53	205.44	205.70	205.37	205.24	
	MAD	47.05	88.21	103.86	107.39	104.51	99.92	99.35	100.49	100.52	99.78	99.48	
10 minutes	*k*	2	98	194	290	386	482	578	674	770	866		
	MSE	12400.27	26001.38	29143.63	29640.45	29075.47	28063.98	28078.72	28447.29	28546.53	28416.77		
	RMSE	111.36	161.25	170.72	172.16	170.52	167.52	167.57	168.66	168.96	168.57		
	MAD	44.03	85.18	101.12	104.41	101.45	97.01	96.53	97.64	97.64	96.93		
20 minutes	*k*	2	50	98	146	194	242	290	338	386	434		
	MSE	7956.57	18380.92	21459.92	21964.14	21383.34	20398.31	20430.31	20797.43	20888.98	20763.02		
	RMSE	89.20	135.58	146.49	148.20	146.23	142.82	142.93	144.21	144.53	144.09		

Time interval		Method 2: Simple moving average – I/O											
	MAD	40.40	79.85	95.85	98.83	96.05	91.64	91.24	92.34	92.31	91.65		
1 hour	*k*	2	18	34	50	66	82	98	114	130	146		
	MSE	4087.48	10631.85	13516.11	13932.21	13274.74	12421.36	12504.68	12844.07	12913.18	12793.22		
	RMSE	63.93	103.11	116.26	118.03	115.22	111.45	111.82	113.33	113.64	113.11		
	MAD	40.53	72.00	86.98	89.45	86.07	82.08	81.95	83.06	82.95	82.26		
4 hours	*k*	2	20	38	56	74	92	110	128	146	164	182	
	MSE	45471.88	74950.15	78971.67	81228.47	81526.88	81293.13	80811.20	80876.51	80459.84	80258.86	80180.23	
	RMSE	213.24	273.77	281.02	285.01	285.53	285.12	284.27	284.39	283.65	283.30	283.16	
	MAD	70.33	112.08	111.03	109.01	109.20	109.15	107.65	107.66	106.19	105.49	105.22	
12 hours	*k*	2	8	14	20	26	32	38	44	50	56		
	MSE	13498.48	27190.01	30643.62	32233.66	32867.71	32516.26	32406.70	31951.21	31665.08	31495.70		
	RMSE	116.18	164.89	175.05	179.54	181.29	180.32	180.02	178.75	177.95	177.47		
	MAD	63.74	92.43	94.00	94.25	95.78	93.09	92.15	91.08	90.41	89.92		
	k	2	5	8	11	14	17	20	23	26	29		
	MSE	7.28	9.91	10.88	11.39	11.70	11.91	12.07	12.18	12.27	12.34		

Time interval		Method 2: Simple moving average – I/O											
1 day	RMSE	2.70	3.15	3.30	3.38	3.42	3.45	3.47	3.49	3.50	3.51		
	MAD	0.67	1.17	1.47	1.70	1.87	2.01	2.13	2.24	2.33	2.42		
1 week	*k*	2	4	6	8	10	12	14	16	18	20	22	24
	MSE	13251.44	10647.83	12896.84	13042.75	13456.06	13912.17	13361.16	13836.17	13393.52	13751.02	13642.66	13733.57
	RMSE	115.11	103.19	113.56	114.20	116.00	117.95	115.59	117.63	115.73	117.26	116.80	117.19
	MAD	84.40	76.22	84.03	87.36	88.85	93.41	91.82	93.77	92.18	93.23	93.06	93.56
2 weeks	*k*	2	12	22	32	42	52	62	72	82	92	102	102
	MSE	2867.03	7825.61	7926.93	7451.81	8118.09	7769.20	7786.76	7919.19	7807.97	7906.09	7867.09	7807.97
	RMSE	53.54	88.46	89.03	86.32	90.10	88.14	88.24	88.99	88.36	88.92	88.70	88.36
	MAD	42.06	70.88	68.68	67.98	70.23	68.83	69.05	69.26	69.04	69.42	69.21	69.04
4 weeks	*k*	2	7	12	17	22	27	32	37	42	47	52	
	MSE	2312.63	4522.49	4077.00	3994.78	4412.13	4045.68	4178.76	4213.70	4175.12	4263.33	4223.39	
	RMSE	48.09	67.25	63.85	63.20	66.42	63.61	64.64	64.91	64.62	65.29	64.99	
	MAD	42.50	55.54	52.28	52.62	54.54	52.43	53.23	53.40	53.29	53.68	53.50	

Appendix C – Prediction Results Using WMA Method

Table C1: Prediction using weighted moving average for different time periods

Time interval			CPU test			Memory test			I/O test		
	k	α	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
5 minutes	2	0.5	6311.04	79.44	8.92	12774721.50	3574.17	543.73	36220.42	190.32	61.91
	2	1.5	2312.15	48.08	5.44	4684977.79	2164.48	325.53	12773.97	113.02	37.06
	2	2	1566.36	39.58	4.43	3174781.89	1781.79	264.51	8553.48	92.49	30.20
	2	5	352.28	18.77	2.01	714546.05	845.31	119.68	515.02	22.69	7.18
	2	10	98.23	9.91	1.03	199295.26	446.42	61.53	1869.15	43.23	13.85
	900	0.5	5578.80	74.69	13.07	14973518.27	3869.56	671.04	47574.63	218.12	93.25
	900	1.5	2919.58	54.03	4.71	3147991.85	1774.26	221.91	11588.11	107.65	24.68
	900	2	1820.03	42.66	3.58	1952605.32	1397.36	163.71	7062.94	84.04	18.81
	900	5	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	900	10	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	1910	0.5	5599.86	74.83	15.33	12039412.91	3469.79	430.82	49081.23	221.54	91.73
	1910	1.2	0.0014	0.0369	0.0033	0.36	0.60	0.04	30.00	5.48	0.18
	1910	2	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	1910	5	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	1910	10	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Time interval			CPU test			Memory test			I/O test		
	k	α	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
10 minutes	2	0.5	30.78	5.55	3.34	5329348.86	2308.54	440.42	15036.31	122.62	45.26
	2	1.5	11.08	3.33	2.01	1918565.59	1385.12	264.25	5413.07	73.57	27.16
	2	2	7.70	2.77	1.67	1332337.21	1154.27	220.21	3759.08	61.31	22.63
	2	5	1.92	1.39	0.84	333084.30	577.13	110.10	939.77	30.66	11.32
	2	10	0.57	0.76	0.46	99099.46	314.80	60.06	279.60	16.72	6.17
	480	0.5	127.86	11.31	8.62	8261169.04	2874.22	653.31	33271.30	182.40	91.06
	480	1.5	8.76	2.96	1.35	1344374.70	1159.47	203.59	7003.64	83.69	22.57
	480	2	5.37	2.32	1.03	813530.58	901.96	147.66	4270.24	65.35	17.19
	480	5	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	480	10	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	950	0.5	150.65	12.27	9.49	6923553.40	2631.26	438.42	34638.38	186.11	87.98
	950	1.2	0.00283	0.05	0.00705	12.19	3.49	0.64	42.66	6.53	0.38
	950	2	0.00092	0.03	0.00257	0.19	0.44	0.04	14.50	3.81	0.16
	950	5	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	950	10	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
20 minutes	2	0.5	158.37	12.58	9.79	4272631.52	2067.03	619.39	13664.65	116.90	52.31
	2	1.5	15.31930	3.91	2.53640	1559267.79	1248.71	373.30	5017.94	70.84	32.00
	2	2	10.30230	3.21	2.06870	1055095.41	1027.18	303.51	3401.49	58.32	26.18
	2	5	2.27580	1.51	0.94321	236629.83	486.45	136.83	766.07	27.68	12.09
	2	10	0.62993	0.79	0.48701	65904.27	256.72	70.39	213.71	14.62	6.30

Time interval			CPU test			Memory test			I/O test		
	k	α	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
	240	0.5	114.60	10.71	8.29	4831640.19	2198.10	650.13	23292.68	152.62	84.85
	240	1.5	14.190	3.767	2.156	1032723.49	1016.23	243.05	4532.16	67.32	21.63
	240	2	8.609	2.934	1.626	616540.78	785.20	175.41	2767.77	52.61	15.94
	240	5	1.67	1.29	0.67	116333.39	341.08	67.95	544.37	23.33	6.62
	240	10	0.45	0.67	0.34	31290.04	176.89	33.99	147.94	12.16	3.38
	480	0.5	127.86	11.31	8.62	4372559.71	2091.07	457.93	26662.04	163.29	85.45
	480	1.2	12.92	3.59	1.69	114.26	10.69	480.00	35.03	5.92	0.52
	480	2	5.37	2.32	1.03	0.09	0.29	0.02	12.73	3.57	0.18
	480	5	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	480	10	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
1 hour	2	0.5	9.22	3.04	2.19	1605775.89	1267.19	531.89	4248.27	65.18	38.85
	2	1.5	3.32	1.82	1.31	578079.32	760.32	319.13	1529.38	39.11	23.31
	2	2	2.31	1.52	1.09	401443.97	633.60	265.94	1062.07	32.59	19.43
	2	5	0.58	0.76	0.55	100360.99	316.80	132.97	265.52	16.29	9.71
	2	10	0.17	0.41	0.30	29859.47	172.80	72.53	79.00	8.89	5.30
	80	0.5	78.61	8.87	5.97	2177578.12	1475.66	628.66	12551.67	112.03	72.94
	80	1.5	3.40	1.84	0.95169	658035.99	811.19	308.98	2146.81	46.33	22.36
	80	2	1.84	1.36	0.70180	408873.39	639.43	232.91	1206.85	34.74	16.29
	80	5	0.33	0.58	0.29277	81692.82	285.82	93.20	213.47	14.61	6.48
	80	10	0.09	0.30	0.15054	22368.45	149.56	46.71	56.30	7.50	3.26

Time interval			CPU test			Memory test			I/O test		
	k	α	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
	155	0.5	83.23	9.12	6.22340	2003632.21	1415.50	454.96	18573.78	136.29	82.65
	155	1.2	0.16683	0.41	0.10635	20927.8731	144.66	32.2856	129.30	11.37	3.32
	155	2	0.07010	0.26	0.04304	9199.39	95.91	13.12	45.42	6.74	1.29
	155	5	0.01440	0.12	0.01635	1828.86	42.77	5.27	8.01	2.83	0.47
	155	10	0.00400	0.06	0.00810	497.57	22.31	2.64	2.05	1.43	0.23
4 hours	2	0.5	393.25	19.83	5.13	575896.54	758.88	533.54	77988.75	279.26	92.82
	2	1.5	147.02	12.13	2.98	209487.16	758.88	332.18	28707.86	169.43	54.89
	2	2	100.19	10.01	2.40	141631.62	376.34	272.25	19475.45	139.55	44.41
	2	5	22.86	4.78	1.07	31704.22	178.06	125.65	4395.06	66.30	19.86
	2	10	6.41	2.53	0.54	8823.85	93.94	65.10	1227.16	35.03	10.30
	90	0.5	860.07	29.33	17.66	559801.49	748.20	486.57	115826.19	340.33	143.66
	90	1.5	6.23	2.50	0.82	93541.65	305.85	182.94	2089.75	45.71	23.93
	90	2	3.35	1.83	0.62	60932.74	246.85	146.15	1146.88	33.87	16.93
	90	5	0.51	0.71	0.24	12904.02	113.60	64.54	192.49	13.87	6.64
	90	10	0.13	0.36	0.12	3563.23	59.69	33.05	49.77	7.05	3.35
	180	0.5	517.48	22.75	11.43	465316.86	682.14	427.73	82173.66	286.66	110.23
	180	1.2	2.4145	1.5539	0.3149	10669.502	103.293	22.846	168.302	12.973	3.878
	180	2	0.1839	0.4289	0.0715	4441.5515	66.6450	10.1793	13.003	3.606	0.646
	180	5	0.0282	0.1678	0.0241	920.0587	30.3325	4.3296	2.888	1.699	0.278
	180	10	0.0087	0.0934	0.0114	252.5492	15.8918	2.2087	0.812	0.901	0.148

Time interval			CPU test			Memory test			I/O test		
	k	α	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
12 hours	2	0.5	162.75	12.76	5.77	94509.54	307.42	226.06	22495.73	149.99	82.83
	2	1.5	58.27	7.63	3.34	40484.62	201.21	149.40	8685.50	93.20	49.79
	2	2	39.20	6.26	2.68	28626.23	169.19	124.44	5974.71	77.30	40.38
	2	5	8.67	2.94	1.19	7096.47	84.24	60.84	1393.48	37.33	18.31
	2	10	2.40	1.55	0.61	2053.53	45.32	32.65	394.19	19.85	9.47
	31	0.5	511.16	22.61	16.43	158732.52	398.41	255.05	52301.76	228.70	127.62
	31	1.5	8.07	2.84	1.14	14153.89	118.97	81.42	1828.21	42.76	28.35
	31	2	4.40	2.10	0.69	9060.03	95.18	61.76	932.73	30.54	18.19
	31	5	0.82	0.91	0.29	1962.95	44.31	27.35	150.78	12.28	6.84
	31	10	0.22	0.47	0.15	551.46	23.48	14.32	38.6411	6.22	3.43
	60	0.5	277.21	16.65	11.57	161139.11	401.42	262.13	33827.09	183.92	95.35
	60	1.2	8.69	2.95	0.66	25104.38	158.44	158.44	763.6462	27.63	10.78
	60	2	2.82	1.68	0.28	1188.96	34.48	9.08	118.76	10.90	2.77
	60	5	0.51	0.72	0.13	275.12	16.59	3.78	5.22	2.28	0.44
	60	10	0.14	0.37	0.07	79.68	8.93	1.99	0.73	0.85	0.14
1 day	2	0.5	75023.71	273.90	178.41	26539.15	162.91	76.95	8.96	2.99	0.53763
	2	1.5	27008.54	164.34	107.05	9554.10	97.75	46.17	3.23	1.80	0.32
	2	2	18755.93	136.95	89.20	6634.79	81.45	38.47	2.24	1.50	0.27
	2	5	4688.98	68.48	44.60	1658.70	40.73	19.24	0.56	0.75	0.13
	2	10	1395.07	37.35	24.33	493.50	22.21	10.49	0.17	0.41	0.07

Time interval			CPU test			Memory test			I/O test		
	k	α	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
	16	0.5	99642.02	315.66	196.45	49632.18	222.78	126.73	75.61	8.70	6.05
	16	1.5	8648.52	93.00	58.32	2806.84	52.98	36.29	0.00018	0.013	0.009
	16	2	4494.88	67.04	39.59	1243.77	35.27	23.12	7.04E-08	2.65E-04	1.85E-04
	16	5	882.02	29.70	17.41	200.33	14.15	7.78	2.08E-19	4.56E-10	3.17E-10
	16	10	239.47	15.47	9.23	52.62	7.25	3.81	3.52E-28	1.88E-14	1.74E-14
	30	0.5	76458.75	276.51	155.14	35176.65	187.55	86.40	146.17	12.09	11.69
	30	1.2	1045.17	32.33	19.36	1405.88	37.50	17.90	4.19E-04	2.05E-02	1.98E-02
	30	2	177.49	13.32	4.02	436.68	20.90	6.56	5.07E-16	2.25E-08	2.18E-08
	30	5	42.20	6.50	1.62	79.52	8.92	2.33	3.68E-27	6.07E-14	5.91E-14
	30	10	12.36	3.52	0.85	21.62	4.65	1.09	1.69E-28	1.30E-14	1.19E-14
1 week	2	0.5	964.23	31.05	14.50	27105.48	164.64	115.83	15720.73	125.38	85.90
	2	1.5	347.12	18.63	8.70	9757.97	98.78	69.50	5659.46	75.229	51.541
	2	2	241.06	15.53	7.25	6776.37	82.32	57.91	3930.18	62.691	42.951
	2	5	60.26	7.76	3.62	1694.09	41.16	28.96	982.55	31.346	21.475
	2	10	17.93	4.23	1.98	504.03	22.45	15.79	292.33	17.098	11.714
	13	0.5	27094.25	164.60	116.35	150992.94	388.58	294.88	21159.71	145.464	113.313
	13	1.5	1924.73	43.87	21.73	9638.37	98.18	65.89	2193.51	46.83	31.39
	13	2	774.49	27.83	11.86	4355.01	65.99	41.41	1297.17	36.02	23.15
	13	5	87.30	9.34	3.31	658.92	25.67	14.59	254.03	15.94	9.30
	13	10	20.31	4.51	1.55	171.87	13.11	7.29	68.57	8.28	4.60

Time interval			CPU test			Memory test			I/O test		
	k	α	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
	26	0.5	33084.78	181.89	132.98	156791.09	395.97	313.81	15539.88	124.66	99.20
	26	1.2	281.29	16.77	11.79	2279.99	47.75	30.72	867.25	29.45	16.33
	26	2	0.78	0.88	0.31	63.73	7.98	3.24	107.97	10.39	3.93
	26	5	1.32E-02	1.15E-01	3.41E-02	1.65	1.28	0.40	5.14	2.27	0.69
	26	10	5.41E-04	2.33E-02	5.90E-03	0.10	0.32	0.07	0.56	0.75	0.21
2 weeks	2	0.5	725.24	26.93	11.91	10512.10	102.53	81.69	5591.43	74.78	57.30
	2	1.5	261.08	16.16	7.15	3784.36	61.52	49.01	2012.91	44.87	34.3781
	2	2	181.31	13.47	5.96	2628.03	51.26	40.84	1397.86	37.39	28.6484
	2	5	45.33	6.73	2.98	657.01	25.63	20.42	349.46	18.69	14.32
	2	10	13.49	3.67	1.62	195.47	13.98	11.14	103.97	10.20	7.81
	50	0.5	26996.51	164.31	111.47	126048.34	355.03	266.15	15633.65	125.03	100.19
	50	1.5	961.07	31.00	12.53	6296.20	79.35	46.44	1882.98	43.39	26.29
	50	2	367.41	19.17	6.41	2795.05	52.87	29.78	1070.64	32.72	18.97
	50	5	40.27	6.35	1.94	432.94	20.81	11.91	207.08	14.39	8.00
	50	10	9.32	3.05	0.92	114.21	10.69	6.15	57.83	7.60	4.24
	104	0.5	45427.43	213.14	177.00	204923.89	452.69	377.92	18127.21	134.64	104.64
	104	1.2	285.09	16.88	5.09	1538.60	39.23	12.15	248.99	15.78	4.49
	104	2	0.2617	0.5115	0.1242	38.40	6.20	1.07	16.46	4.06	0.79
	104	5	0.0413	0.2031	0.0296	9.30	3.05	0.40	0.31	0.55	0.09
	104	10	0.0114	0.1068	0.0150	2.71	1.65	0.21	0.04	0.20	0.03

Time interval			CPU test			Memory test			I/O test		
	k	α	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
4 weeks	2	0.5	1562.79	39.5321	16.44	8859.56	94.13	64.55	1684.52	41.04	36.28
	2	1.5	562.60	23.7193	9.86	3189.44	56.48	38.73	606.43	24.63	21.77
	2	2	390.70	19.7661	8.22	2214.89	47.06	32.27	421.13	20.52	18.14
	2	5	97.67	9.8830	4.11	553.72	23.53	16.14	105.28	10.26	9.07
	2	10	29.06	5.3907	2.24	164.74	12.84	8.80	31.32	5.60	4.95
	26	0.5	23232.43	152.4219	105.86	102738.73	320.53	236.15	6732.42	82.05	67.61
	26	1.5	562.60	23.7193	9.86	10228.00	101.13	60.71	1217.33	34.89	23.13
	26	2	747.52	27.3408	11.69	4036.87	63.54	34.47	622.94	24.96	16.50
	26	5	81.75	9.0416	2.85	454.78	21.33	10.57	87.44	9.35	6.22
	26	10	18.96	4.3543	1.31	107.26	10.36	5.05	20.98	4.58	3.01
	52	0.5	40294.89	200.7359	168.86	172776.65	415.66	348.10	8754.56	93.57	75.84
	52	1.2	1278.71	35.7591	12.39	5899.38	76.81	27.86	504.44	22.46	7.87
	52	2	18.64	4.3178	1.27	137.79	11.74	3.48	93.89	9.69	2.24
	52	5	4.02E-02	2.01E-01	4.89E-02	1.64E+00	1.28	2.73E-01	9.71	3.12	0.59
	52	10	9.00E-03	9.48E-02	1.78E-02	2.71E-01	0.52	1.04E-01	2.11	1.45	0.25

As discussed earlier, due to the larger input for *k* and the large value of the increasing factor, a smaller weight factor is generated which is smaller than the smallest non-zero floating point value, therefore the system did not produce a value, as indicated by N/A in Table C1. From Table C1, we observe that with a higher number of observations and a higher value of alpha, we obtain better prediction accuracy at every time interval.

Appendix D – Prediction Results Using Extrapolation Method

Table D1: Extrapolation method for different time periods

Time interval	CPU test			Memory test			I/O test		
	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
5 minutes	32,468.23	180.19	20.21	70198694.17	8378.466	1500.6326	4199716.751	2049.32	1994.8405
10 minutes	201.13	14.18	9.18	40582824.49	6370.465	1568.985	413970.5677	2032.7249	1994.1052
20 minutes	111.68	10.57	7.1	25193532.05	5019.3159	1643.5988	4078063.168	2019.4215	1988.8025
1 hour	58.92	7.68	5.74	12115868.43	3480.785	17250.5776	4006667.12	2001.6661	1973.1863
4 hours	2,174.31	46.63	10.55	35566415.51	1888.495	1370.484	5113263.77	2261.2527	2148.64
12 hours	767.32	27.7	11.26	1419041.564	1191.2353	938.564	4766165.43	2183.154	2108.7258
1 day	546.4667	23.3766	11.2667	1366942.3	1169.1631	954.833	4669856.767	2160.9851	2041.9
1 week	3646.6264	60.387	36.33	4174309	2043.11	1777.6	6297784.43	2509.539	2377.028
2 weeks	2,298.49	47.94	27.45	4705600.319	2169.239	2012.755	6806946.164	2609.0125	2561.5201
4 weeks	5166.0665	71.875	37.168	4651048.301	2156.628	1985.936	6695974.526	2587.6581	2519.9319

Appendix E – Prediction Results Using Holt-Winter Double Exponential Smoothing Method

Table E1: Holt-Winters method with all possible values of alpha and beta for different time periods

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
5 minutes	0.1	0.1	4961.90	70.44	10.69	10615404.51	3258.13	843.83	30281.04	174.01	67.72
5 minutes	0.1	0.2	5220.43	72.25	11.69	11250389.13	3354.16	962.53	32021.44	178.95	73.59
5 minutes	0.1	0.3	5479.12	74.02	12.69	11880022.66	3446.74	1071.25	33862.01	184.02	78.87
5 minutes	0.1	0.4	5716.66	75.61	13.42	12499339.54	3535.44	1165.74	35507.56	188.43	83.46
5 minutes	0.1	0.5	5994.37	77.42	14.18	13052750.52	3612.86	1242.77	37078.64	192.56	86.97
5 minutes	0.1	0.6	6261.28	79.13	15.00	13553233.32	3681.47	1309.06	38981.22	197.44	90.78
5 minutes	0.1	0.7	6486.90	80.54	15.54	14029038.58	3745.54	1361.65	41103.93	202.74	94.48
5 minutes	0.1	0.8	6732.35	82.05	15.98	14483454.90	3805.71	1405.82	43066.75	207.53	98.02
5 minutes	0.1	0.9	7028.84	83.84	16.58	14925318.27	3863.33	1445.24	44841.47	211.76	100.73
5 minutes	0.2	0.1	4144.52	64.38	8.89	8635225.48	2938.58	634.36	24383.07	156.15	57.23
5 minutes	0.2	0.2	4369.92	66.11	9.59	9113842.81	3018.91	704.64	25881.56	160.88	61.22
5 minutes	0.2	0.3	4597.91	67.81	10.20	9563411.35	3092.48	762.21	27404.29	165.54	64.54
5 minutes	0.2	0.4	4828.85	69.49	10.74	9991768.94	3160.98	805.46	28961.49	170.18	67.65
5 minutes	0.2	0.5	5064.51	71.17	11.27	10411555.88	3226.69	841.02	30516.93	174.69	70.48
5 minutes	0.2	0.6	5303.60	72.83	11.77	10833018.77	3291.36	871.43	32074.55	179.09	73.15
5 minutes	0.2	0.7	5546.96	74.48	12.21	11268079.26	3356.80	897.90	33644.01	183.42	75.68
5 minutes	0.2	0.8	5793.25	76.11	12.69	11725198.45	3424.21	923.05	35237.34	187.72	78.16

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
5 minutes	0.2	0.9	6040.87	77.72	13.10	12204024.42	3493.43	948.39	36854.62	191.98	80.70
5 minutes	0.3	0.1	3365.64	58.01	7.65	6926714.88	2631.87	517.53	19383.87	139.23	49.38
5 minutes	0.3	0.2	3560.80	59.67	8.14	7316507.49	2704.90	562.66	20633.33	143.64	52.33
5 minutes	0.3	0.3	3759.95	61.32	8.59	7700735.05	2775.02	597.93	21901.74	147.99	54.93
5 minutes	0.3	0.4	3963.27	62.95	9.02	8090482.93	2844.38	627.24	23182.25	152.26	57.32
5 minutes	0.3	0.5	4170.45	64.58	9.43	8494179.60	2914.48	650.37	24473.45	156.44	59.72
5 minutes	0.3	0.6	4381.07	66.19	9.84	8915800.45	2985.93	676.03	25772.87	160.54	61.94
5 minutes	0.3	0.7	4594.88	67.79	10.18	9355651.45	3058.70	701.32	27074.23	164.54	63.98
5 minutes	0.3	0.8	4811.88	69.37	10.53	9812718.71	3132.53	727.11	28378.02	168.46	66.13
5 minutes	0.3	0.9	5032.23	70.94	10.93	10285898.03	3207.16	757.42	29696.26	172.33	68.26
5 minutes	0.4	0.1	2632.63	51.31	6.51	5385960.81	2320.77	428.38	14897.49	122.06	42.28
5 minutes	0.4	0.2	2795.68	52.87	6.89	5711497.15	2389.87	459.17	15899.28	126.09	44.62
5 minutes	0.4	0.3	2963.06	54.43	7.26	6043885.09	2458.43	482.24	16917.93	130.07	46.88
5 minutes	0.4	0.4	3134.72	55.99	7.60	6389101.40	2527.67	504.80	17951.62	133.98	48.89
5 minutes	0.4	0.5	3310.57	57.54	7.92	6749574.59	2597.99	528.37	19000.50	137.84	50.65
5 minutes	0.4	0.6	3490.68	59.08	8.25	7125368.12	2669.34	552.15	20067.15	141.66	52.56
5 minutes	0.4	0.7	3675.29	60.62	8.51	7515710.91	2741.48	573.53	21157.50	145.46	54.34
5 minutes	0.4	0.8	3864.73	62.17	8.76	7919674.52	2814.19	595.70	22277.85	149.26	56.00
5 minutes	0.4	0.9	4059.38	63.71	9.07	8336162.79	2887.24	621.52	23431.63	153.07	57.90
5 minutes	0.5	0.1	1954.88	44.21	5.42	3986180.97	1996.54	348.35	10885.32	104.33	35.53
5 minutes	0.5	0.2	2084.66	45.66	5.73	4247288.65	2060.90	371.64	11652.85	107.95	37.48
5 minutes	0.5	0.3	2218.89	47.11	6.01	4518322.86	2125.63	389.86	12438.68	111.53	39.10
5 minutes	0.5	0.4	2357.68	48.56	6.28	4801641.13	2191.26	408.78	13243.67	115.08	40.65
5 minutes	0.5	0.5	2501.23	50.01	6.51	5097583.05	2257.78	425.81	14070.24	118.62	42.27

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
5 minutes	0.5	0.6	2649.85	51.48	6.78	5405578.59	2324.99	446.98	14921.53	122.15	43.90
5 minutes	0.5	0.7	2803.89	52.95	7.03	5724753.47	2392.65	465.73	15799.81	125.70	45.49
5 minutes	0.5	0.8	2963.74	54.44	7.26	6054108.98	2460.51	481.15	16705.49	129.25	46.99
5 minutes	0.5	0.9	3129.76	55.94	7.48	6392652.67	2528.37	496.41	17637.34	132.81	48.43
5 minutes	0.6	0.1	1344.56	36.67	4.34	2735647.04	1653.98	274.41	7372.56	85.86	28.76
5 minutes	0.6	0.2	1440.81	37.96	4.59	2929879.29	1711.69	292.42	7921.81	89.00	30.14
5 minutes	0.6	0.3	1541.33	39.26	4.80	3133437.51	1770.15	306.00	8489.45	92.14	31.47
5 minutes	0.6	0.4	1646.38	40.58	5.02	3347209.34	1829.54	321.09	9077.08	95.27	32.90
5 minutes	0.6	0.5	1756.30	41.91	5.22	3571202.41	1889.76	334.85	9686.64	98.42	34.23
5 minutes	0.6	0.6	1871.47	43.26	5.39	3805175.28	1950.69	346.80	10319.77	101.59	35.49
5 minutes	0.6	0.7	1992.30	44.64	5.58	4048958.98	2012.20	358.58	10977.44	104.77	36.70
5 minutes	0.6	0.8	2119.24	46.04	5.82	4302652.02	2074.28	372.39	11660.38	107.98	37.99
5 minutes	0.6	0.9	2252.74	47.46	6.06	4566780.04	2137.00	387.11	12369.78	111.22	39.36
5 minutes	0.7	0.1	817.59	28.59	3.27	1660905.23	1288.76	203.73	4416.40	66.46	21.87
5 minutes	0.7	0.2	881.15	29.68	3.44	1789161.59	1337.60	216.69	4767.09	69.04	22.89
5 minutes	0.7	0.3	948.34	30.80	3.61	1924904.14	1387.41	227.46	5133.60	71.65	24.00
5 minutes	0.7	0.4	1019.46	31.93	3.76	2068683.66	1438.29	235.67	5517.47	74.28	25.04
5 minutes	0.7	0.5	1094.89	33.09	3.92	2220825.28	1490.24	246.07	5920.35	76.94	26.07
5 minutes	0.7	0.6	1175.06	34.28	4.11	2381756.44	1543.29	256.73	6343.86	79.65	27.14
5 minutes	0.7	0.7	1260.43	35.50	4.29	2552179.72	1597.55	266.51	6789.83	82.40	28.21
5 minutes	0.7	0.8	1351.53	36.76	4.45	2733166.34	1653.23	275.57	7260.53	85.21	29.22
5 minutes	0.7	0.9	1448.98	38.07	4.61	2926192.89	1710.61	283.85	7758.96	88.08	30.21
5 minutes	0.8	0.1	395.54	19.89	2.19	802757.60	895.97	135.31	2105.40	45.88	14.84
5 minutes	0.8	0.2	429.22	20.72	2.30	870703.66	933.12	143.63	2285.46	47.81	15.54

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
5 minutes	0.8	0.3	465.34	21.57	2.42	943573.07	971.38	150.34	2476.26	49.76	16.30
5 minutes	0.8	0.4	504.18	22.45	2.55	1021863.12	1010.87	157.24	2679.07	51.76	17.05
5 minutes	0.8	0.5	546.07	23.37	2.67	1106113.90	1051.72	163.86	2895.33	53.81	17.84
5 minutes	0.8	0.6	591.38	24.32	2.79	1197039.87	1094.09	169.55	3126.73	55.92	18.58
5 minutes	0.8	0.7	640.57	25.31	2.89	1295592.39	1138.24	174.26	3375.28	58.10	19.29
5 minutes	0.8	0.8	694.17	26.35	2.99	1402984.78	1184.48	178.36	3643.48	60.36	20.00
5 minutes	0.8	0.9	752.80	27.44	3.11	1520698.14	1233.17	186.34	3934.28	62.72	20.82
5 minutes	0.9	0.1	108.53	10.42	1.10	220176.23	469.23	67.65	569.37	23.86	7.58
5 minutes	0.9	0.2	118.75	10.90	1.17	240824.47	490.74	71.91	622.47	24.95	7.95
5 minutes	0.9	0.3	129.92	11.40	1.24	263362.19	513.19	75.32	679.74	26.07	8.36
5 minutes	0.9	0.4	142.15	11.92	1.30	288058.38	536.71	78.45	741.78	27.24	8.78
5 minutes	0.9	0.5	155.63	12.48	1.36	315241.29	561.46	81.75	809.36	28.45	9.22
5 minutes	0.9	0.6	170.55	13.06	1.45	345331.69	587.65	86.05	883.38	29.72	9.69
5 minutes	0.9	0.7	187.16	13.68	1.54	378863.18	615.52	90.63	964.99	31.06	10.20
5 minutes	0.9	0.8	205.77	14.34	1.63	416501.07	645.37	95.79	1055.60	32.49	10.74
5 minutes	0.9	0.9	226.77	15.06	1.73	459069.49	677.55	101.65	1156.99	34.01	11.32
10 minutes	0.1	0.1	38.23	6.18	4.40	6137253.33	2477.35	846.12	19657.00	140.20	67.83
10 minutes	0.1	0.2	41.09	6.41	4.69	6634752.40	2575.80	1010.78	20622.42	143.61	74.53
10 minutes	0.1	0.3	44.09	6.64	4.86	7095910.31	2663.81	1156.70	21610.97	147.01	78.59
10 minutes	0.1	0.4	46.22	6.80	4.91	7397287.80	2719.80	1224.82	22429.46	149.76	80.98
10 minutes	0.1	0.5	48.27	6.95	4.98	7732370.75	2780.71	1269.35	23205.79	152.33	83.76
10 minutes	0.1	0.6	49.93	7.07	5.05	8140987.05	2853.24	1330.99	24052.58	155.09	86.08
10 minutes	0.1	0.7	51.62	7.18	5.11	8549887.33	2924.02	1395.43	25011.77	158.15	88.71
10 minutes	0.1	0.8	54.30	7.37	5.26	8937742.13	2989.61	1460.31	26138.32	161.67	91.66

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
10 minutes	0.1	0.9	58.24	7.63	5.48	9364990.58	3060.23	1529.59	27452.51	165.69	95.11
10 minutes	0.2	0.1	30.05	5.48	3.87	4958008.49	2226.66	686.97	15003.73	122.49	54.95
10 minutes	0.2	0.2	32.01	5.66	3.98	5279024.28	2297.61	771.66	15819.61	125.78	58.36
10 minutes	0.2	0.3	33.86	5.82	4.07	5587707.48	2363.83	831.85	16646.41	129.02	61.39
10 minutes	0.2	0.4	35.81	5.98	4.18	5901940.80	2429.39	888.91	17529.52	132.40	64.54
10 minutes	0.2	0.5	37.82	6.15	4.31	6216310.61	2493.25	945.36	18446.44	135.82	67.04
10 minutes	0.2	0.6	39.64	6.30	4.45	6526556.33	2554.71	994.38	19339.14	139.07	69.84
10 minutes	0.2	0.7	41.21	6.42	4.59	6832013.76	2613.81	1044.46	20172.39	142.03	72.22
10 minutes	0.2	0.8	42.66	6.53	4.70	7128869.17	2669.99	1088.57	20963.85	144.79	74.45
10 minutes	0.2	0.9	44.11	6.64	4.80	7411701.65	2722.44	1128.42	21771.31	147.55	76.66
10 minutes	0.3	0.1	23.52	4.85	3.34	3922583.53	1980.55	559.65	11741.26	108.36	46.66
10 minutes	0.3	0.2	25.01	5.00	3.44	4170645.32	2042.22	609.92	12442.88	111.55	49.52
10 minutes	0.3	0.3	26.48	5.15	3.54	4415439.82	2101.29	654.18	13163.60	114.73	51.88
10 minutes	0.3	0.4	27.90	5.28	3.66	4657134.71	2158.04	695.37	13892.13	117.86	54.08
10 minutes	0.3	0.5	29.24	5.41	3.77	4893456.56	2212.12	732.09	14618.85	120.91	56.23
10 minutes	0.3	0.6	30.54	5.53	3.86	5123178.26	2263.44	767.21	15356.43	123.92	58.27
10 minutes	0.3	0.7	31.84	5.64	3.93	5346714.22	2312.30	795.12	16120.01	126.96	60.16
10 minutes	0.3	0.8	33.14	5.76	3.99	5565620.22	2359.16	819.97	16908.29	130.03	61.71
10 minutes	0.3	0.9	34.46	5.87	4.05	5781591.38	2404.49	844.84	17708.46	133.07	63.58
10 minutes	0.4	0.1	17.93	4.23	2.85	3012181.73	1735.56	456.52	8970.23	94.71	39.77
10 minutes	0.4	0.2	19.08	4.37	2.95	3203818.81	1789.92	491.84	9543.66	97.69	41.87
10 minutes	0.4	0.3	20.21	4.50	3.05	3393163.24	1842.05	523.11	10130.13	100.65	43.73
10 minutes	0.4	0.4	21.31	4.62	3.14	3580083.62	1892.11	552.56	10727.65	103.57	45.44
10 minutes	0.4	0.5	22.41	4.73	3.22	3764703.50	1940.28	579.19	11339.34	106.49	47.12

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
10 minutes	0.4	0.6	23.52	4.85	3.29	3948110.54	1986.99	602.73	11967.16	109.39	48.74
10 minutes	0.4	0.7	24.67	4.97	3.36	4131824.07	2032.69	619.61	12608.81	112.29	50.37
10 minutes	0.4	0.8	25.84	5.08	3.45	4317350.31	2077.82	636.67	13262.29	115.16	51.78
10 minutes	0.4	0.9	27.07	5.20	3.54	4506091.47	2122.76	656.93	13928.25	118.02	53.17
10 minutes	0.5	0.1	13.08	3.62	2.40	2208315.60	1486.04	369.16	6539.02	80.86	33.14
10 minutes	0.5	0.2	13.94	3.73	2.48	2353747.62	1534.19	396.69	6984.40	83.57	34.77
10 minutes	0.5	0.3	14.81	3.85	2.56	2499248.28	1580.90	417.96	7442.87	86.27	36.26
10 minutes	0.5	0.4	15.69	3.96	2.64	2645500.18	1626.50	439.70	7914.94	88.97	37.78
10 minutes	0.5	0.5	16.59	4.07	2.72	2793563.51	1671.40	455.54	8401.60	91.66	39.18
10 minutes	0.5	0.6	17.53	4.19	2.80	2944809.02	1716.04	472.89	8903.10	94.36	40.50
10 minutes	0.5	0.7	18.50	4.30	2.88	3100642.79	1760.86	488.08	9419.96	97.06	41.82
10 minutes	0.5	0.8	19.51	4.42	2.96	3262404.44	1806.21	500.37	9953.54	99.77	43.04
10 minutes	0.5	0.9	20.57	4.54	3.04	3431316.46	1852.38	512.49	10505.61	102.50	44.21
10 minutes	0.6	0.1	8.87	2.98	1.94	1505963.54	1227.18	290.85	4425.37	66.52	26.71
10 minutes	0.6	0.2	9.50	3.08	2.02	1611464.47	1269.43	310.72	4747.53	68.90	27.94
10 minutes	0.6	0.3	10.14	3.18	2.09	1719117.41	1311.15	325.83	5082.13	71.29	29.22
10 minutes	0.6	0.4	10.81	3.29	2.16	1829864.73	1352.72	340.63	5429.95	73.69	30.47
10 minutes	0.6	0.5	11.50	3.39	2.23	1944810.53	1394.56	353.33	5791.86	76.10	31.59
10 minutes	0.6	0.6	12.23	3.50	2.31	2065111.95	1437.05	364.21	6168.80	78.54	32.68
10 minutes	0.6	0.7	12.99	3.60	2.38	2191867.44	1480.50	376.33	6561.97	81.01	33.71
10 minutes	0.6	0.8	13.80	3.71	2.46	2326076.67	1525.15	390.92	6972.62	83.50	34.84
10 minutes	0.6	0.9	14.65	3.83	2.55	2468644.32	1571.19	405.26	7401.91	86.03	36.04
10 minutes	0.7	0.1	5.34	2.31	1.48	910597.53	954.25	216.16	2650.64	51.48	20.23
10 minutes	0.7	0.2	5.74	2.40	1.54	979849.84	989.87	230.64	2858.06	53.46	21.20

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
10 minutes	0.7	0.3	6.17	2.48	1.61	1052065.95	1025.70	242.42	3075.83	55.46	22.19
10 minutes	0.7	0.4	6.62	2.57	1.67	1128057.23	1062.10	251.90	3304.79	57.49	23.12
10 minutes	0.7	0.5	7.10	2.66	1.73	1208683.10	1099.40	262.06	3545.94	59.55	24.01
10 minutes	0.7	0.6	7.60	2.76	1.80	1294801.66	1137.89	272.95	3800.39	61.65	24.98
10 minutes	0.7	0.7	8.14	2.85	1.87	1387245.81	1177.81	283.11	4069.37	63.79	25.93
10 minutes	0.7	0.8	8.72	2.95	1.94	1486838.94	1219.36	291.95	4354.26	65.99	26.88
10 minutes	0.7	0.9	9.34	3.06	2.01	1594431.29	1262.71	301.67	4656.63	68.24	27.83
10 minutes	0.8	0.1	2.56	1.60	1.01	438785.38	662.41	143.66	1263.73	35.55	13.70
10 minutes	0.8	0.2	2.77	1.66	1.05	475517.20	689.58	153.33	1370.87	37.03	14.36
10 minutes	0.8	0.3	3.00	1.73	1.10	514657.52	717.40	161.69	1484.85	38.53	15.02
10 minutes	0.8	0.4	3.24	1.80	1.15	556729.72	746.14	169.25	1606.41	40.08	15.73
10 minutes	0.8	0.5	3.51	1.87	1.20	602282.16	776.07	175.28	1736.41	41.67	16.44
10 minutes	0.8	0.6	3.79	1.95	1.26	651882.66	807.39	181.25	1875.88	43.31	17.12
10 minutes	0.8	0.7	4.10	2.03	1.31	706128.25	840.31	186.87	2025.98	45.01	17.77
10 minutes	0.8	0.8	4.44	2.11	1.37	765667.84	875.02	192.28	2188.13	46.78	18.44
10 minutes	0.8	0.9	4.81	2.19	1.42	831228.85	911.72	201.19	2364.00	48.62	19.19
10 minutes	0.9	0.1	0.70	0.83	0.51	119974.63	346.37	71.79	341.76	18.49	6.97
10 minutes	0.9	0.2	0.76	0.87	0.54	131140.70	362.13	76.90	373.47	19.33	7.34
10 minutes	0.9	0.3	0.83	0.91	0.57	143298.31	378.55	81.08	407.77	20.19	7.72
10 minutes	0.9	0.4	0.91	0.95	0.60	156646.15	395.79	84.47	445.02	21.10	8.10
10 minutes	0.9	0.5	0.99	1.00	0.63	171401.34	414.01	88.26	485.67	22.04	8.49
10 minutes	0.9	0.6	1.08	1.04	0.67	187805.05	433.36	93.09	530.24	23.03	8.93
10 minutes	0.9	0.7	1.19	1.09	0.70	206132.63	454.02	97.85	579.40	24.07	9.39
10 minutes	0.9	0.8	1.30	1.14	0.74	226707.94	476.14	102.90	633.97	25.18	9.92

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
10 minutes	0.9	0.9	1.43	1.20	0.78	249921.50	499.92	108.84	694.97	26.36	10.46
20 minutes	0.1	0.1	24.48	4.95	3.51	3813785.42	1952.89	810.31	14149.23	118.95	65.64
20 minutes	0.1	0.2	27.13	5.21	3.74	4088234.64	2021.94	907.88	15693.43	125.27	70.92
20 minutes	0.1	0.3	29.07	5.39	3.91	4398498.37	2097.26	997.29	17051.43	130.58	76.44
20 minutes	0.1	0.4	29.47	5.43	4.00	4657090.03	2158.03	1076.06	18135.40	134.67	82.11
20 minutes	0.1	0.5	30.21	5.50	4.14	4887957.93	2210.87	1130.40	18452.21	135.84	85.68
20 minutes	0.1	0.6	31.76	5.64	4.30	5187676.99	2277.65	1201.51	18356.89	135.49	87.52
20 minutes	0.1	0.7	34.07	5.84	4.50	5615287.37	2369.66	1302.13	18508.43	136.05	89.16
20 minutes	0.1	0.8	37.11	6.09	4.72	6152621.05	2480.45	1428.57	19145.52	138.37	92.05
20 minutes	0.1	0.9	40.60	6.37	4.92	6734459.54	2595.08	1573.28	20094.08	141.75	95.98
20 minutes	0.2	0.1	18.81	4.34	3.07	3135276.49	1770.67	699.05	10539.01	102.66	54.03
20 minutes	0.2	0.2	20.09	4.48	3.25	3368909.36	1835.46	776.72	11211.16	105.88	58.61
20 minutes	0.2	0.3	21.36	4.62	3.42	3613390.00	1900.89	858.31	11682.08	108.08	62.04
20 minutes	0.2	0.4	22.79	4.77	3.55	3872953.09	1967.98	943.71	12126.07	110.12	65.22
20 minutes	0.2	0.5	24.17	4.92	3.61	4111083.87	2027.58	1019.88	12604.51	112.27	67.29
20 minutes	0.2	0.6	25.26	5.03	3.67	4295079.14	2072.46	1071.94	13059.88	114.28	69.13
20 minutes	0.2	0.7	26.16	5.11	3.72	4439380.18	2106.98	1106.12	13481.23	116.11	70.02
20 minutes	0.2	0.8	27.06	5.20	3.79	4580639.10	2140.24	1130.35	13894.01	117.87	71.17
20 minutes	0.2	0.9	27.98	5.29	3.87	4740297.95	2177.22	1152.51	14314.93	119.65	72.77
20 minutes	0.3	0.1	14.44	3.80	2.69	2483656.53	1575.96	590.40	8001.56	89.45	45.41
20 minutes	0.3	0.2	15.43	3.93	2.81	2662903.58	1631.84	655.88	8440.65	91.87	48.39
20 minutes	0.3	0.3	16.40	4.05	2.90	2838091.25	1684.66	715.27	8835.38	94.00	50.86
20 minutes	0.3	0.4	17.32	4.16	2.99	2999072.95	1731.78	762.63	9225.66	96.05	52.52
20 minutes	0.3	0.5	18.17	4.26	3.08	3148852.37	1774.50	799.25	9615.25	98.06	54.01

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
20 minutes	0.3	0.6	19.00	4.36	3.15	3300043.19	1816.60	832.34	10012.64	100.06	55.70
20 minutes	0.3	0.7	19.85	4.46	3.22	3459596.33	1860.00	859.30	10426.39	102.11	57.38
20 minutes	0.3	0.8	20.75	4.56	3.30	3627497.57	1904.60	889.28	10860.66	104.21	59.19
20 minutes	0.3	0.9	21.74	4.66	3.38	3801450.75	1949.73	925.51	11316.46	106.38	60.92
20 minutes	0.4	0.1	10.84	3.29	2.31	1895927.86	1376.93	494.25	6024.88	77.62	37.91
20 minutes	0.4	0.2	11.58	3.40	2.40	2028497.95	1424.25	541.52	6365.27	79.78	40.15
20 minutes	0.4	0.3	12.30	3.51	2.49	2156884.28	1468.63	580.21	6695.10	81.82	41.89
20 minutes	0.4	0.4	13.00	3.61	2.57	2282174.17	1510.69	612.28	7030.10	83.85	43.49
20 minutes	0.4	0.5	13.71	3.70	2.65	2408976.39	1552.09	639.73	7376.05	85.88	45.09
20 minutes	0.4	0.6	14.44	3.80	2.72	2539548.69	1593.60	667.72	7735.81	87.95	46.79
20 minutes	0.4	0.7	15.21	3.90	2.80	2673354.62	1635.04	692.82	8108.86	90.05	48.32
20 minutes	0.4	0.8	16.00	4.00	2.87	2808920.11	1675.98	716.91	8491.97	92.15	49.68
20 minutes	0.4	0.9	16.79	4.10	2.95	2944942.18	1716.08	744.85	8880.63	94.24	51.13
20 minutes	0.5	0.1	7.79	2.79	1.94	1378387.41	1174.05	402.72	4373.23	66.13	31.47
20 minutes	0.5	0.2	8.33	2.89	2.02	1475497.51	1214.70	438.56	4638.74	68.11	33.14
20 minutes	0.5	0.3	8.87	2.98	2.09	1571749.61	1253.69	463.90	4905.94	70.04	34.61
20 minutes	0.5	0.4	9.41	3.07	2.16	1668709.06	1291.79	487.37	5182.09	71.99	36.02
20 minutes	0.5	0.5	9.97	3.16	2.23	1767732.74	1329.56	506.18	5469.10	73.95	37.35
20 minutes	0.5	0.6	10.54	3.25	2.30	1868881.50	1367.07	529.25	5766.60	75.94	38.64
20 minutes	0.5	0.7	11.13	3.34	2.36	1971726.54	1404.18	551.10	6073.36	77.93	39.81
20 minutes	0.5	0.8	11.71	3.42	2.42	2076042.43	1440.85	569.99	6388.69	79.93	40.83
20 minutes	0.5	0.9	12.30	3.51	2.49	2181945.66	1477.14	588.54	6713.50	81.94	41.76
20 minutes	0.6	0.1	5.21	2.28	1.57	931556.61	965.17	317.11	2966.18	54.46	25.29
20 minutes	0.6	0.2	5.59	2.36	1.64	999543.21	999.77	342.06	3162.70	56.24	26.56

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
20 minutes	0.6	0.3	5.98	2.44	1.70	1068263.90	1033.57	359.62	3365.13	58.01	27.69
20 minutes	0.6	0.4	6.37	2.52	1.76	1138511.77	1067.01	376.74	3576.73	59.81	28.86
20 minutes	0.6	0.5	6.77	2.60	1.82	1210718.36	1100.33	393.70	3798.33	61.63	29.88
20 minutes	0.6	0.6	7.19	2.68	1.88	1284961.09	1133.56	408.31	4030.27	63.48	30.69
20 minutes	0.6	0.7	7.62	2.76	1.93	1361325.39	1166.76	422.39	4273.28	65.37	31.53
20 minutes	0.6	0.8	8.05	2.84	2.00	1440025.11	1200.01	438.87	4528.81	67.30	32.41
20 minutes	0.6	0.9	8.51	2.92	2.06	1521364.30	1233.44	455.84	4798.78	69.27	33.43
20 minutes	0.7	0.1	3.10	1.76	1.20	558587.30	747.39	234.39	1788.52	42.29	19.20
20 minutes	0.7	0.2	3.34	1.83	1.26	601723.52	775.71	251.75	1919.09	43.81	20.09
20 minutes	0.7	0.3	3.58	1.89	1.31	646118.38	803.81	266.09	2056.15	45.34	20.96
20 minutes	0.7	0.4	3.84	1.96	1.36	692194.01	831.98	277.13	2201.41	46.92	21.77
20 minutes	0.7	0.5	4.11	2.03	1.40	740238.03	860.37	288.48	2355.76	48.54	22.49
20 minutes	0.7	0.6	4.38	2.09	1.45	790486.36	889.09	300.37	2520.16	50.20	23.25
20 minutes	0.7	0.7	4.68	2.16	1.50	843234.33	918.28	312.61	2695.84	51.92	24.05
20 minutes	0.7	0.8	4.98	2.23	1.55	898862.86	948.08	323.49	2884.23	53.71	24.81
20 minutes	0.7	0.9	5.31	2.30	1.60	957840.66	978.69	333.54	3086.88	55.56	25.57
20 minutes	0.8	0.1	1.47	1.21	0.82	267447.72	517.15	154.97	861.25	29.35	13.00
20 minutes	0.8	0.2	1.59	1.26	0.86	289754.29	538.29	166.59	931.16	30.51	13.62
20 minutes	0.8	0.3	1.72	1.31	0.90	313183.85	559.63	175.91	1005.94	31.72	14.21
20 minutes	0.8	0.4	1.86	1.36	0.93	337994.55	581.37	183.87	1086.58	32.96	14.82
20 minutes	0.8	0.5	2.00	1.41	0.97	364431.56	603.68	191.33	1173.87	34.26	15.41
20 minutes	0.8	0.6	2.15	1.47	1.01	392767.38	626.71	198.76	1268.72	35.62	15.95
20 minutes	0.8	0.7	2.32	1.52	1.04	423337.03	650.64	205.23	1372.23	37.04	16.48
20 minutes	0.8	0.8	2.50	1.58	1.08	456553.04	675.69	210.17	1485.65	38.54	17.05

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
20 minutes	0.8	0.9	2.70	1.64	1.12	492917.29	702.08	218.56	1610.46	40.13	17.72
20 minutes	0.9	0.1	0.40	0.63	0.42	72892.21	269.99	77.45	235.86	15.36	6.66
20 minutes	0.9	0.2	0.43	0.66	0.44	79596.79	282.13	83.33	257.34	16.04	7.00
20 minutes	0.9	0.3	0.47	0.69	0.46	86812.66	294.64	88.03	280.81	16.76	7.34
20 minutes	0.9	0.4	0.51	0.72	0.48	94652.28	307.66	92.11	306.65	17.51	7.66
20 minutes	0.9	0.5	0.56	0.75	0.51	103242.52	321.31	96.29	335.25	18.31	7.98
20 minutes	0.9	0.6	0.61	0.78	0.53	112737.02	335.76	101.27	367.08	19.16	8.36
20 minutes	0.9	0.7	0.67	0.82	0.55	123327.53	351.18	106.06	402.73	20.07	8.82
20 minutes	0.9	0.8	0.73	0.85	0.58	135254.43	367.77	111.22	442.89	21.04	9.33
20 minutes	0.9	0.9	0.80	0.89	0.60	148820.08	385.77	117.09	488.46	22.10	9.87
1 hour	0.1	0.1	15.49	3.94	2.89	1637323.53	1279.58	716.99	9718.88	98.58	66.98
1 hour	0.1	0.2	16.67	4.08	3.03	1697844.86	1303.01	724.84	10258.37	101.28	72.74
1 hour	0.1	0.3	16.99	4.12	3.05	1767374.73	1329.43	753.36	10453.97	102.24	74.90
1 hour	0.1	0.4	17.50	4.18	3.11	1842536.69	1357.40	780.85	10185.92	100.93	73.51
1 hour	0.1	0.5	17.98	4.24	3.13	1925454.33	1387.61	815.52	10103.13	100.51	72.90
1 hour	0.1	0.6	18.20	4.27	3.18	2021583.43	1421.82	862.87	10401.25	101.99	74.30
1 hour	0.1	0.7	18.14	4.26	3.23	2130424.87	1459.60	921.57	11040.18	105.07	77.68
1 hour	0.1	0.8	18.16	4.26	3.26	2245569.57	1498.52	972.76	11916.34	109.16	81.59
1 hour	0.1	0.9	18.42	4.29	3.31	2362359.20	1537.00	1014.23	12949.41	113.80	84.46
1 hour	0.2	0.1	10.76	3.28	2.37	1399467.41	1182.99	645.99	6308.54	79.43	53.32
1 hour	0.2	0.2	11.26	3.36	2.44	1473655.67	1213.94	675.13	6609.85	81.30	56.78
1 hour	0.2	0.3	11.73	3.42	2.50	1553629.13	1246.45	713.05	6977.17	83.53	59.47
1 hour	0.2	0.4	12.23	3.50	2.58	1635735.96	1278.96	743.76	7504.12	86.63	61.56
1 hour	0.2	0.5	12.91	3.59	2.66	1714058.62	1309.22	767.12	8123.81	90.13	63.08

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
1 hour	0.2	0.6	13.80	3.71	2.73	1787658.09	1337.03	793.29	8753.13	93.56	64.34
1 hour	0.2	0.7	14.88	3.86	2.86	1864694.84	1365.54	824.21	9349.19	96.69	66.00
1 hour	0.2	0.8	16.09	4.01	3.00	1954884.29	1398.17	858.27	9895.92	99.48	67.36
1 hour	0.2	0.9	17.33	4.16	3.15	2061704.05	1435.86	903.29	10396.52	101.96	67.80
1 hour	0.3	0.1	7.96	2.82	2.01	1149620.57	1072.20	577.82	4542.59	67.40	45.09
1 hour	0.3	0.2	8.42	2.90	2.07	1218404.58	1103.81	610.23	4864.88	69.75	47.29
1 hour	0.3	0.3	8.93	2.99	2.16	1289636.46	1135.62	642.06	5243.93	72.41	48.56
1 hour	0.3	0.4	9.52	3.09	2.25	1362592.00	1167.30	672.70	5649.60	75.16	50.52
1 hour	0.3	0.5	10.17	3.19	2.34	1439630.05	1199.85	704.28	6046.33	77.76	51.99
1 hour	0.3	0.6	10.78	3.28	2.43	1522337.40	1233.83	734.37	6425.48	80.16	53.53
1 hour	0.3	0.7	11.27	3.36	2.52	1608519.69	1268.27	762.42	6793.28	82.42	55.50
1 hour	0.3	0.8	11.61	3.41	2.59	1694120.42	1301.58	788.27	7151.07	84.56	57.81
1 hour	0.3	0.9	11.82	3.44	2.63	1776499.05	1332.85	815.70	7484.46	86.51	59.95
1 hour	0.4	0.1	5.88	2.42	1.72	903191.77	950.36	503.63	3258.40	57.08	37.31
1 hour	0.4	0.2	6.26	2.50	1.79	962181.74	980.91	533.75	3509.81	59.24	38.88
1 hour	0.4	0.3	6.66	2.58	1.88	1023781.57	1011.82	561.60	3776.03	61.45	40.85
1 hour	0.4	0.4	7.07	2.66	1.95	1088249.07	1043.19	588.09	4037.96	63.54	42.55
1 hour	0.4	0.5	7.44	2.73	2.02	1155466.99	1074.93	614.47	4285.68	65.47	44.48
1 hour	0.4	0.6	7.76	2.79	2.08	1224374.83	1106.51	639.15	4515.01	67.19	46.28
1 hour	0.4	0.7	8.04	2.84	2.14	1294365.29	1137.70	663.80	4720.30	68.70	47.58
1 hour	0.4	0.8	8.32	2.88	2.19	1366174.64	1168.83	687.09	4896.96	69.98	48.72
1 hour	0.4	0.9	8.63	2.94	2.23	1441597.73	1200.67	716.35	5046.95	71.04	50.05
1 hour	0.5	0.1	4.18	2.05	1.46	670425.75	818.80	422.35	2245.19	47.38	30.67
1 hour	0.5	0.2	4.46	2.11	1.53	717696.72	847.17	447.65	2417.67	49.17	32.36

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
1 hour	0.5	0.3	4.75	2.18	1.59	767326.05	875.97	471.80	2590.42	50.90	33.89
1 hour	0.5	0.4	5.02	2.24	1.65	819281.00	905.14	494.95	2754.82	52.49	35.38
1 hour	0.5	0.5	5.29	2.30	1.71	873431.98	934.58	515.83	2907.09	53.92	36.60
1 hour	0.5	0.6	5.55	2.36	1.76	929908.00	964.32	539.97	3046.42	55.19	37.80
1 hour	0.5	0.7	5.82	2.41	1.81	989305.01	994.64	565.89	3174.83	56.35	39.13
1 hour	0.5	0.8	6.11	2.47	1.86	1052310.53	1025.82	591.39	3297.55	57.42	40.32
1 hour	0.5	0.9	6.43	2.54	1.91	1119183.02	1057.91	616.27	3421.09	58.49	41.39
1 hour	0.6	0.1	2.78	1.67	1.20	459330.74	677.74	338.31	1444.56	38.01	24.47
1 hour	0.6	0.2	2.97	1.72	1.25	494036.86	702.88	359.34	1554.42	39.43	25.72
1 hour	0.6	0.3	3.17	1.78	1.30	530628.46	728.44	378.91	1662.21	40.77	26.87
1 hour	0.6	0.4	3.37	1.84	1.35	569111.43	754.39	398.98	1764.95	42.01	27.99
1 hour	0.6	0.5	3.57	1.89	1.39	609545.10	780.73	418.50	1862.55	43.16	29.02
1 hour	0.6	0.6	3.78	1.94	1.44	652059.02	807.50	437.58	1956.81	44.24	30.05
1 hour	0.6	0.7	4.00	2.00	1.49	696689.14	834.68	456.42	2050.63	45.28	31.00
1 hour	0.6	0.8	4.24	2.06	1.53	743187.63	862.08	478.42	2146.77	46.33	31.75
1 hour	0.6	0.9	4.49	2.12	1.58	791010.94	889.39	499.61	2246.90	47.40	32.55
1 hour	0.7	0.1	1.64	1.28	0.93	277556.98	526.84	254.11	827.51	28.77	18.22
1 hour	0.7	0.2	1.77	1.33	0.97	299991.62	547.71	270.29	891.22	29.85	19.16
1 hour	0.7	0.3	1.89	1.38	1.01	323762.07	569.00	286.21	954.09	30.89	20.04
1 hour	0.7	0.4	2.02	1.42	1.04	348884.71	590.66	300.44	1015.60	31.87	20.92
1 hour	0.7	0.5	2.16	1.47	1.09	375377.54	612.68	315.50	1076.57	32.81	21.67
1 hour	0.7	0.6	2.30	1.52	1.13	403221.66	635.00	331.40	1138.36	33.74	22.36
1 hour	0.7	0.7	2.46	1.57	1.17	432325.43	657.51	346.94	1202.36	34.68	23.07
1 hour	0.7	0.8	2.62	1.62	1.22	462562.31	680.12	360.25	1269.55	35.63	23.67

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
1 hour	0.7	0.9	2.79	1.67	1.26	493877.26	702.76	371.56	1340.63	36.61	24.14
1 hour	0.8	0.1	0.78	0.88	0.64	133270.13	365.06	169.61	379.43	19.48	12.12
1 hour	0.8	0.2	0.84	0.92	0.67	144857.84	380.60	180.80	409.89	20.25	12.75
1 hour	0.8	0.3	0.91	0.95	0.70	157234.99	396.53	191.67	440.56	20.99	13.38
1 hour	0.8	0.4	0.98	0.99	0.73	170431.64	412.83	202.19	471.61	21.72	13.97
1 hour	0.8	0.5	1.05	1.03	0.76	184482.71	429.51	212.10	503.60	22.44	14.51
1 hour	0.8	0.6	1.13	1.06	0.79	199429.42	446.58	220.85	537.20	23.18	14.96
1 hour	0.8	0.7	1.22	1.10	0.82	215339.41	464.05	228.82	573.02	23.94	15.32
1 hour	0.8	0.8	1.31	1.15	0.85	232343.56	482.02	235.71	611.63	24.73	15.62
1 hour	0.8	0.9	1.41	1.19	0.88	250667.01	500.67	245.15	653.58	25.57	15.97
1 hour	0.9	0.1	0.21	0.46	0.33	36283.73	190.48	85.07	99.15	9.96	6.08
1 hour	0.9	0.2	0.23	0.48	0.35	39720.99	199.30	90.80	107.68	10.38	6.42
1 hour	0.9	0.3	0.25	0.50	0.37	43445.37	208.44	96.29	116.51	10.79	6.76
1 hour	0.9	0.4	0.27	0.52	0.39	47485.52	217.91	101.37	125.76	11.21	7.04
1 hour	0.9	0.5	0.29	0.54	0.40	51880.44	227.77	106.57	135.64	11.65	7.28
1 hour	0.9	0.6	0.32	0.57	0.42	56684.94	238.09	112.97	146.35	12.10	7.50
1 hour	0.9	0.7	0.35	0.59	0.44	61977.56	248.95	119.70	158.11	12.57	7.74
1 hour	0.9	0.8	0.38	0.62	0.46	67867.94	260.51	126.22	171.16	13.08	8.06
1 hour	0.9	0.9	0.42	0.65	0.48	74501.12	272.95	133.28	185.78	13.63	8.43
4 hours	0.1	0.1	379.11	19.47	9.00	395726.64	629.07	450.73	69197.97	263.06	113.14
4 hours	0.1	0.2	408.22	20.20	10.00	414642.24	643.93	458.99	74019.73	272.07	122.22
4 hours	0.1	0.3	408.38	20.21	10.33	439097.88	662.64	471.55	79456.38	281.88	134.94
4 hours	0.1	0.4	404.79	20.12	10.13	461445.26	679.30	488.51	85095.05	291.71	146.84
4 hours	0.1	0.5	415.15	20.38	10.22	471316.41	686.52	499.80	89205.04	298.67	154.57

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
4 hours	0.1	0.6	431.29	20.77	10.78	478057.45	691.42	505.03	92708.58	304.48	159.85
4 hours	0.1	0.7	440.63	20.99	11.18	492592.97	701.85	513.36	97069.81	311.56	166.40
4 hours	0.1	0.8	442.49	21.04	11.06	513177.01	716.36	527.76	102430.52	320.05	177.78
4 hours	0.1	0.9	446.42	21.13	10.91	529497.75	727.67	543.38	106756.83	326.74	184.39
4 hours	0.2	0.1	281.71	16.78	6.20	331762.21	575.99	413.97	53901.84	232.17	91.27
4 hours	0.2	0.2	291.60	17.08	6.57	345572.36	587.85	422.22	57194.08	239.15	99.86
4 hours	0.2	0.3	301.77	17.37	6.75	356953.94	597.46	433.09	60067.04	245.09	107.61
4 hours	0.2	0.4	313.16	17.70	6.92	366930.44	605.75	444.04	62481.00	249.96	111.82
4 hours	0.2	0.5	326.02	18.06	6.99	375151.18	612.50	451.65	64270.65	253.52	115.53
4 hours	0.2	0.6	341.02	18.47	7.16	381824.27	617.92	455.81	65525.80	255.98	117.89
4 hours	0.2	0.7	357.50	18.91	7.28	388623.78	623.40	459.80	66675.01	258.22	119.06
4 hours	0.2	0.8	374.69	19.36	7.43	396028.23	629.31	464.69	67953.25	260.68	121.14
4 hours	0.2	0.9	391.95	19.80	7.65	403408.06	635.14	469.89	69330.52	263.31	122.37
4 hours	0.3	0.1	220.11	14.84	4.77	272683.06	522.19	375.31	42235.93	205.51	76.04
4 hours	0.3	0.2	230.35	15.18	5.02	283593.06	532.53	383.91	44402.36	210.72	81.97
4 hours	0.3	0.3	241.40	15.54	5.16	293521.97	541.78	393.46	46232.45	215.02	85.53
4 hours	0.3	0.4	253.49	15.92	5.29	302960.21	550.42	401.61	47798.56	218.63	89.12
4 hours	0.3	0.5	266.31	16.32	5.42	312468.07	558.99	407.35	49262.21	221.95	91.28
4 hours	0.3	0.6	279.35	16.71	5.55	322423.82	567.82	412.15	50733.70	225.24	93.62
4 hours	0.3	0.7	292.20	17.09	5.71	333030.34	577.09	417.14	52233.44	228.55	95.89
4 hours	0.3	0.8	304.62	17.45	5.87	344531.64	586.97	422.00	53764.12	231.87	97.71
4 hours	0.3	0.9	316.49	17.79	6.01	357119.56	597.59	427.79	55344.64	235.25	98.97
4 hours	0.4	0.1	169.84	13.03	3.80	217670.70	466.55	335.95	32554.89	180.43	63.46
4 hours	0.4	0.2	179.06	13.38	3.98	227781.28	477.26	344.53	34223.15	184.99	67.14

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
4 hours	0.4	0.3	188.84	13.74	4.07	237884.85	487.73	352.73	35761.80	189.11	71.12
4 hours	0.4	0.4	199.04	14.11	4.16	248415.93	498.41	359.66	37267.83	193.05	73.98
4 hours	0.4	0.5	209.38	14.47	4.28	259704.17	509.61	366.14	38815.97	197.02	75.82
4 hours	0.4	0.6	219.68	14.82	4.41	271973.83	521.51	372.61	40444.51	201.11	77.70
4 hours	0.4	0.7	229.92	15.16	4.47	285383.22	534.21	379.67	42182.34	205.38	79.20
4 hours	0.4	0.8	240.20	15.50	4.61	300037.43	547.76	387.64	44059.61	209.90	80.60
4 hours	0.4	0.9	250.72	15.83	4.75	316018.54	562.16	397.48	46104.77	214.72	82.50
4 hours	0.5	0.1	125.56	11.21	3.04	165408.60	406.70	292.53	24117.83	155.30	52.26
4 hours	0.5	0.2	133.15	11.54	3.14	174662.07	417.93	300.49	25493.72	159.67	55.74
4 hours	0.5	0.3	141.09	11.88	3.20	184411.28	429.43	307.99	26867.34	163.91	58.44
4 hours	0.5	0.4	149.30	12.22	3.30	194940.82	441.52	315.12	28300.63	168.23	60.36
4 hours	0.5	0.5	157.69	12.56	3.41	206448.50	454.37	322.50	29835.40	172.73	62.30
4 hours	0.5	0.6	166.30	12.90	3.52	219069.33	468.05	330.73	31500.68	177.48	63.78
4 hours	0.5	0.7	175.21	13.24	3.64	232902.64	482.60	340.94	33320.50	182.54	65.56
4 hours	0.5	0.8	184.53	13.58	3.74	248031.59	498.03	352.04	35314.76	187.92	67.58
4 hours	0.5	0.9	194.36	13.94	3.83	264530.84	514.33	363.69	37499.21	193.65	69.40
4 hours	0.6	0.1	86.39	9.29	2.36	116259.86	340.97	243.94	16641.88	129.00	41.95
4 hours	0.6	0.2	92.16	9.60	2.43	123955.75	352.07	251.05	17729.98	133.15	44.39
4 hours	0.6	0.3	98.22	9.91	2.52	132277.09	363.70	258.17	18869.03	137.36	46.48
4 hours	0.6	0.4	104.55	10.23	2.62	141390.09	376.02	265.58	20095.22	141.76	48.31
4 hours	0.6	0.5	111.18	10.54	2.69	151406.15	389.11	274.26	21433.23	146.40	49.81
4 hours	0.6	0.6	118.15	10.87	2.78	162401.15	402.99	283.82	22900.90	151.33	51.40
4 hours	0.6	0.7	125.53	11.20	2.88	174428.23	417.65	294.08	24511.82	156.56	53.54
4 hours	0.6	0.8	133.38	11.55	2.99	187521.12	433.04	305.63	26276.17	162.10	55.87

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
4 hours	0.6	0.9	141.74	11.91	3.08	201691.59	449.10	318.13	28201.28	167.93	58.12
4 hours	0.7	0.1	52.70	7.26	1.74	71965.83	268.26	190.41	10163.78	100.82	31.64
4 hours	0.7	0.2	56.60	7.52	1.81	77460.04	278.32	196.62	10921.41	104.51	33.37
4 hours	0.7	0.3	60.76	7.79	1.89	83479.45	288.93	203.12	11735.81	108.33	34.95
4 hours	0.7	0.4	65.17	8.07	1.96	90109.45	300.18	210.37	12625.54	112.36	36.34
4 hours	0.7	0.5	69.88	8.36	2.04	97405.46	312.10	218.35	13603.05	116.63	38.01
4 hours	0.7	0.6	74.93	8.66	2.12	105403.05	324.66	227.50	14676.81	121.15	39.70
4 hours	0.7	0.7	80.37	8.96	2.19	114123.31	337.82	237.42	15852.40	125.91	41.41
4 hours	0.7	0.8	86.23	9.29	2.27	123575.79	351.53	247.45	17132.94	130.89	43.09
4 hours	0.7	0.9	92.56	9.62	2.34	133762.23	365.74	258.03	18519.17	136.09	44.74
4 hours	0.8	0.1	25.62	5.06	1.15	35262.59	187.78	131.85	4932.96	70.24	21.24
4 hours	0.8	0.2	27.74	5.27	1.21	38306.48	195.72	136.51	5346.96	73.12	22.33
4 hours	0.8	0.3	30.03	5.48	1.27	41669.43	204.13	141.74	5798.90	76.15	23.55
4 hours	0.8	0.4	32.51	5.70	1.34	45389.38	213.05	147.99	6296.39	79.35	24.73
4 hours	0.8	0.5	35.21	5.93	1.39	49492.93	222.47	154.80	6844.45	82.73	25.96
4 hours	0.8	0.6	38.15	6.18	1.43	54000.74	232.38	162.18	7446.32	86.29	27.12
4 hours	0.8	0.7	41.38	6.43	1.51	58931.73	242.76	169.49	8104.07	90.02	28.33
4 hours	0.8	0.8	44.93	6.70	1.59	64307.71	253.59	177.17	8818.89	93.91	29.41
4 hours	0.8	0.9	48.84	6.99	1.67	70159.07	264.88	186.62	9591.56	97.94	30.55
4 hours	0.9	0.1	7.07	2.66	0.58	9743.62	98.71	68.08	1355.68	36.82	10.86
4 hours	0.9	0.2	7.73	2.78	0.62	10683.82	103.36	71.05	1482.88	38.51	11.48
4 hours	0.9	0.3	8.45	2.91	0.66	11731.45	108.31	74.47	1623.43	40.29	12.14
4 hours	0.9	0.4	9.25	3.04	0.70	12898.64	113.57	78.39	1779.42	42.18	12.76
4 hours	0.9	0.5	10.14	3.18	0.73	14196.42	119.15	83.13	1952.41	44.19	13.38

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
4 hours	0.9	0.6	11.13	3.34	0.78	15636.68	125.05	88.13	2143.84	46.30	14.16
4 hours	0.9	0.7	12.25	3.50	0.82	17234.25	131.28	93.42	2355.26	48.53	14.93
4 hours	0.9	0.8	13.50	3.67	0.86	19008.93	137.87	99.12	2588.75	50.88	15.69
4 hours	0.9	0.9	14.92	3.86	0.90	20987.59	144.87	105.61	2847.34	53.36	16.44
12 hours	0.1	0.1	225.84	15.03	10.52	105358.34	324.59	244.68	29562.70	171.94	100.57
12 hours	0.1	0.2	231.28	15.21	10.95	107081.14	327.23	247.20	33041.14	181.77	109.27
12 hours	0.1	0.3	236.04	15.36	11.43	109717.60	331.24	243.42	36074.21	189.93	121.70
12 hours	0.1	0.4	247.23	15.72	12.24	114798.83	338.82	241.30	38674.19	196.66	128.03
12 hours	0.1	0.5	264.32	16.26	12.67	122322.44	349.75	244.06	40734.21	201.83	132.11
12 hours	0.1	0.6	287.88	16.97	13.07	131703.22	362.91	252.42	42274.22	205.61	136.70
12 hours	0.1	0.7	316.54	17.79	13.63	142182.97	377.07	266.01	43911.60	209.55	141.36
12 hours	0.1	0.8	347.93	18.65	14.41	152794.16	390.89	281.72	45975.74	214.42	145.66
12 hours	0.1	0.9	380.44	19.50	15.08	162156.40	402.69	293.54	48282.86	219.73	153.06
12 hours	0.2	0.1	152.63	12.35	7.75	84843.49	291.28	216.17	22106.74	148.68	87.20
12 hours	0.2	0.2	164.03	12.81	8.62	89578.68	299.30	216.94	24015.74	154.97	93.02
12 hours	0.2	0.3	179.90	13.41	9.04	95428.56	308.92	221.10	25572.52	159.91	98.41
12 hours	0.2	0.4	197.81	14.06	9.62	101530.89	318.64	228.81	27038.49	164.43	103.63
12 hours	0.2	0.5	215.98	14.70	10.05	107016.39	327.13	238.78	28522.22	168.89	108.62
12 hours	0.2	0.6	233.48	15.28	10.34	111655.28	334.15	246.44	29858.88	172.80	109.76
12 hours	0.2	0.7	248.91	15.78	10.54	115942.84	340.50	253.45	30927.31	175.86	110.28
12 hours	0.2	0.8	260.30	16.13	10.95	120732.72	347.47	261.08	31822.12	178.39	111.52
12 hours	0.2	0.9	266.15	16.31	11.42	126790.07	356.08	267.99	32790.50	181.08	112.79
12 hours	0.3	0.1	110.63	10.52	6.20	68260.76	261.27	194.12	16457.74	128.29	73.59
12 hours	0.3	0.2	120.05	10.96	6.66	72558.50	269.37	197.22	17677.85	132.96	76.15

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
12 hours	0.3	0.3	130.00	11.40	7.03	77037.58	277.56	202.17	18770.15	137.00	80.24
12 hours	0.3	0.4	138.83	11.78	7.29	81426.98	285.35	208.38	19796.70	140.70	83.49
12 hours	0.3	0.5	145.45	12.06	7.51	85904.61	293.09	217.56	20756.49	144.07	85.15
12 hours	0.3	0.6	149.25	12.22	7.72	90787.38	301.31	225.23	21715.35	147.36	87.00
12 hours	0.3	0.7	150.48	12.27	7.91	96135.27	310.06	233.12	22747.25	150.82	90.69
12 hours	0.3	0.8	150.20	12.26	7.97	101650.00	318.83	240.90	23847.19	154.43	94.65
12 hours	0.3	0.9	149.67	12.23	7.86	106833.20	326.85	247.20	24945.57	157.94	97.45
12 hours	0.4	0.1	78.94	8.88	4.88	52955.77	230.12	171.54	12041.27	109.73	61.27
12 hours	0.4	0.2	84.78	9.21	5.16	56321.88	237.32	175.89	12874.37	113.47	63.31
12 hours	0.4	0.3	89.97	9.49	5.35	59753.11	244.44	182.14	13643.92	116.81	65.96
12 hours	0.4	0.4	93.87	9.69	5.57	63231.05	251.46	189.70	14383.38	119.93	68.81
12 hours	0.4	0.5	96.47	9.82	5.75	66776.26	258.41	196.46	15117.08	122.95	72.14
12 hours	0.4	0.6	98.28	9.91	5.83	70273.52	265.09	201.88	15852.01	125.90	74.59
12 hours	0.4	0.7	99.97	10.00	5.89	73527.20	271.16	206.82	16568.08	128.72	76.99
12 hours	0.4	0.8	102.02	10.10	5.99	76417.66	276.44	210.57	17243.51	131.31	79.56
12 hours	0.4	0.9	104.58	10.23	6.12	78985.90	281.04	215.60	17874.70	133.70	81.66
12 hours	0.5	0.1	54.50	7.38	3.78	39136.77	197.83	147.79	8490.47	92.14	50.06
12 hours	0.5	0.2	58.07	7.62	3.95	41641.52	204.06	152.76	9058.90	95.18	51.77
12 hours	0.5	0.3	61.10	7.82	4.16	44169.94	210.17	158.10	9596.33	97.96	54.45
12 hours	0.5	0.4	63.54	7.97	4.29	46689.14	216.08	163.25	10119.57	100.60	56.73
12 hours	0.5	0.5	65.64	8.10	4.43	49151.09	221.70	167.21	10632.77	103.12	58.94
12 hours	0.5	0.6	67.74	8.23	4.50	51502.87	226.94	171.57	11131.28	105.50	60.84
12 hours	0.5	0.7	70.02	8.37	4.58	53743.88	231.83	176.52	11611.40	107.76	62.85
12 hours	0.5	0.8	72.51	8.52	4.67	55929.52	236.49	181.02	12075.20	109.89	64.68

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
12 hours	0.5	0.9	75.16	8.67	4.74	58124.70	241.09	185.10	12526.55	111.92	66.40
12 hours	0.6	0.1	35.53	5.96	2.84	26960.83	164.20	122.48	5619.88	74.97	39.63
12 hours	0.6	0.2	37.79	6.15	3.03	28737.34	169.52	126.36	5996.47	77.44	41.20
12 hours	0.6	0.3	39.82	6.31	3.16	30528.48	174.72	130.07	6359.66	79.75	42.98
12 hours	0.6	0.4	41.68	6.46	3.29	32315.20	179.76	134.12	6716.43	81.95	44.83
12 hours	0.6	0.5	43.53	6.60	3.37	34086.59	184.63	139.18	7067.80	84.07	46.29
12 hours	0.6	0.6	45.50	6.75	3.41	35850.04	189.34	143.53	7413.50	86.10	48.24
12 hours	0.6	0.7	47.62	6.90	3.47	37630.90	193.99	147.58	7754.42	88.06	49.86
12 hours	0.6	0.8	49.92	7.07	3.57	39455.33	198.63	152.03	8091.85	89.95	51.04
12 hours	0.6	0.9	52.41	7.24	3.68	41340.16	203.32	155.97	8426.85	91.80	52.12
12 hours	0.7	0.1	20.81	4.56	2.06	16555.06	128.67	95.07	3331.86	57.72	29.77
12 hours	0.7	0.2	22.21	4.71	2.20	17722.42	133.13	98.36	3564.60	59.70	30.89
12 hours	0.7	0.3	23.55	4.85	2.31	18918.71	137.55	101.79	3794.89	61.60	32.33
12 hours	0.7	0.4	24.90	4.99	2.39	20143.70	141.93	105.97	4026.24	63.45	33.66
12 hours	0.7	0.5	26.33	5.13	2.43	21405.97	146.31	109.76	4260.28	65.27	35.08
12 hours	0.7	0.6	27.88	5.28	2.51	22722.36	150.74	113.80	4498.75	67.07	36.16
12 hours	0.7	0.7	29.57	5.44	2.60	24114.10	155.29	117.56	4744.21	68.88	37.02
12 hours	0.7	0.8	31.44	5.61	2.66	25604.00	160.01	121.66	5000.21	70.71	38.06
12 hours	0.7	0.9	33.49	5.79	2.72	27218.17	164.98	125.77	5271.70	72.61	39.05
12 hours	0.8	0.1	9.81	3.13	1.35	8160.77	90.34	66.18	1592.13	39.90	19.99
12 hours	0.8	0.2	10.55	3.25	1.45	8802.11	93.82	68.75	1713.38	41.39	20.87
12 hours	0.8	0.3	11.29	3.36	1.50	9478.85	97.36	71.66	1837.61	42.87	21.90
12 hours	0.8	0.4	12.08	3.48	1.55	10198.14	100.99	74.61	1967.04	44.35	22.87
12 hours	0.8	0.5	12.94	3.60	1.61	10971.84	104.75	77.89	2103.63	45.87	23.63

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
12 hours	0.8	0.6	13.90	3.73	1.67	11815.65	108.70	81.51	2249.85	47.43	24.33
12 hours	0.8	0.7	14.95	3.87	1.70	12748.47	112.91	85.16	2408.93	49.08	25.15
12 hours	0.8	0.8	16.14	4.02	1.74	13792.56	117.44	88.78	2584.99	50.84	26.20
12 hours	0.8	0.9	17.46	4.18	1.78	14974.30	122.37	92.34	2783.04	52.75	27.25
12 hours	0.9	0.1	2.65	1.63	0.68	2302.08	47.98	34.98	436.95	20.90	10.13
12 hours	0.9	0.2	2.87	1.70	0.72	2510.44	50.10	36.45	474.75	21.79	10.65
12 hours	0.9	0.3	3.11	1.76	0.74	2738.22	52.33	38.27	515.12	22.70	11.13
12 hours	0.9	0.4	3.38	1.84	0.78	2990.17	54.68	40.38	559.06	23.64	11.58
12 hours	0.9	0.5	3.67	1.92	0.81	3272.71	57.21	42.51	607.68	24.65	12.21
12 hours	0.9	0.6	4.00	2.00	0.84	3593.94	59.95	44.69	662.37	25.74	12.85
12 hours	0.9	0.7	4.38	2.09	0.88	3963.77	62.96	46.90	724.92	26.92	13.51
12 hours	0.9	0.8	4.80	2.19	0.92	4394.34	66.29	49.14	797.50	28.24	14.14
12 hours	0.9	0.9	5.29	2.30	0.96	4900.66	70.00	51.85	882.80	29.71	14.75
1 day	0.1	0.1	75836.96	275.39	157.74	31625.90	177.84	89.04	80.10	8.95	7.19
1 day	0.1	0.2	77670.23	278.69	163.91	33538.08	183.13	95.06	83.41	9.13	7.55
1 day	0.1	0.3	78967.61	281.01	170.87	35318.67	187.93	99.61	81.55	9.03	7.15
1 day	0.1	0.4	80584.36	283.87	178.91	37232.97	192.96	102.25	82.55	9.09	7.32
1 day	0.1	0.5	82387.20	287.03	186.29	39486.16	198.71	108.27	84.00	9.17	7.53
1 day	0.1	0.6	84106.85	290.01	190.13	42055.56	205.07	115.84	83.98	9.16	7.47
1 day	0.1	0.7	85685.94	292.72	192.41	44736.17	211.51	123.32	83.33	9.13	7.33
1 day	0.1	0.8	87233.12	295.35	193.33	47282.80	217.45	131.03	83.25	9.12	7.32
1 day	0.1	0.9	88903.63	298.17	193.57	49532.63	222.56	137.00	83.93	9.16	7.39
1 day	0.2	0.1	63152.90	251.30	149.24	25860.61	160.81	83.74	35.98	6.00	4.23
1 day	0.2	0.2	65358.91	255.65	157.70	27609.73	166.16	89.41	36.15	6.01	4.05

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
1 day	0.2	0.3	67714.06	260.22	163.24	29530.04	171.84	96.71	36.42	6.03	4.09
1 day	0.2	0.4	70187.95	264.93	165.35	31435.09	177.30	104.12	36.58	6.05	4.03
1 day	0.2	0.5	72908.81	270.02	166.53	33147.71	182.07	111.50	36.82	6.07	4.08
1 day	0.2	0.6	75985.22	275.65	169.57	34677.48	186.22	119.05	37.04	6.09	4.05
1 day	0.2	0.7	79411.27	281.80	176.41	36083.83	189.96	122.10	37.23	6.10	4.04
1 day	0.2	0.8	83101.17	288.27	183.42	37365.03	193.30	122.29	37.46	6.12	4.09
1 day	0.2	0.9	86947.94	294.87	189.19	38488.90	196.19	122.26	37.69	6.14	4.12
1 day	0.3	0.1	52045.56	228.13	136.15	20465.55	143.06	76.77	19.54	4.42	2.67
1 day	0.3	0.2	54476.13	233.40	141.19	21850.58	147.82	83.47	19.72	4.44	2.53
1 day	0.3	0.3	57095.78	238.95	144.18	23214.50	152.36	89.56	19.90	4.46	2.48
1 day	0.3	0.4	59942.85	244.83	148.98	24468.69	156.42	92.77	20.08	4.48	2.49
1 day	0.3	0.5	62998.87	251.00	154.47	25610.11	160.03	93.74	20.26	4.50	2.50
1 day	0.3	0.6	66186.15	257.27	160.72	26656.07	163.27	95.82	20.45	4.52	2.49
1 day	0.3	0.7	69425.49	263.49	165.87	27648.00	166.28	98.83	20.65	4.54	2.47
1 day	0.3	0.8	72650.75	269.54	169.85	28650.32	169.26	102.45	20.84	4.57	2.47
1 day	0.3	0.9	75796.43	275.31	173.40	29713.13	172.37	106.75	21.04	4.59	2.51
1 day	0.4	0.1	41362.09	203.38	121.27	15661.82	125.15	68.55	11.48	3.39	1.82
1 day	0.4	0.2	43680.77	209.00	124.88	16697.08	129.22	73.59	11.63	3.41	1.68
1 day	0.4	0.3	46166.20	214.86	130.04	17697.47	133.03	76.18	11.78	3.43	1.65
1 day	0.4	0.4	48796.65	220.90	135.62	18643.42	136.54	77.62	11.94	3.46	1.62
1 day	0.4	0.5	51524.12	226.99	141.06	19556.06	139.84	79.90	12.10	3.48	1.62
1 day	0.4	0.6	54305.87	233.04	145.97	20473.50	143.09	82.31	12.26	3.50	1.64
1 day	0.4	0.7	57118.84	239.00	150.41	21426.82	146.38	85.93	12.43	3.53	1.66
1 day	0.4	0.8	59969.24	244.89	156.12	22424.25	149.75	89.63	12.60	3.55	1.67

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
1 day	0.4	0.9	62900.82	250.80	161.81	23456.13	153.15	91.89	12.78	3.57	1.67
1 day	0.5	0.1	31111.85	176.39	106.51	11429.42	106.91	58.49	6.83	2.61	1.26
1 day	0.5	0.2	33115.73	181.98	111.73	12201.53	110.46	61.56	6.95	2.64	1.18
1 day	0.5	0.3	35254.14	187.76	116.64	12963.61	113.86	63.11	7.07	2.66	1.12
1 day	0.5	0.4	37510.28	193.68	121.38	13718.39	117.13	64.81	7.20	2.68	1.12
1 day	0.5	0.5	39873.21	199.68	126.32	14485.87	120.36	66.83	7.33	2.71	1.12
1 day	0.5	0.6	42352.48	205.80	131.72	15282.95	123.62	69.64	7.47	2.73	1.12
1 day	0.5	0.7	44981.44	212.09	136.56	16115.03	126.94	72.69	7.61	2.76	1.12
1 day	0.5	0.8	47811.94	218.66	140.79	16981.26	130.31	75.25	7.75	2.78	1.12
1 day	0.5	0.9	50902.43	225.62	146.83	17883.19	133.73	77.73	7.91	2.81	1.11
1 day	0.6	0.1	21578.02	146.89	89.90	7749.67	88.03	47.34	3.92	1.98	0.88
1 day	0.6	0.2	23145.94	152.14	94.49	8303.63	91.12	49.13	4.01	2.00	0.83
1 day	0.6	0.3	24829.16	157.57	99.13	8865.67	94.16	50.59	4.10	2.03	0.79
1 day	0.6	0.4	26628.71	163.18	103.83	9441.57	97.17	51.84	4.20	2.05	0.77
1 day	0.6	0.5	28556.29	168.99	109.43	10041.94	100.21	54.53	4.30	2.07	0.76
1 day	0.6	0.6	30635.50	175.03	115.05	10673.75	103.31	56.91	4.41	2.10	0.74
1 day	0.6	0.7	32895.40	181.37	120.59	11341.03	106.49	58.87	4.52	2.13	0.74
1 day	0.6	0.8	35361.41	188.05	126.09	12048.33	109.76	60.53	4.63	2.15	0.76
1 day	0.6	0.9	38047.29	195.06	131.61	12801.60	113.14	62.65	4.76	2.18	0.79
1 day	0.7	0.1	13155.18	114.70	72.07	4652.46	68.21	35.75	2.05	1.43	0.58
1 day	0.7	0.2	14219.87	119.25	76.21	5010.99	70.79	37.11	2.11	1.45	0.56
1 day	0.7	0.3	15373.42	123.99	80.70	5383.47	73.37	38.11	2.17	1.47	0.53
1 day	0.7	0.4	16622.34	128.93	85.33	5774.05	75.99	39.69	2.23	1.49	0.50
1 day	0.7	0.5	17977.68	134.08	89.96	6188.14	78.66	41.22	2.30	1.52	0.50

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
1 day	0.7	0.6	19451.88	139.47	94.52	6629.86	81.42	43.08	2.38	1.54	0.51
1 day	0.7	0.7	21053.99	145.10	98.99	7102.61	84.28	44.97	2.46	1.57	0.52
1 day	0.7	0.8	22786.42	150.95	103.34	7609.55	87.23	46.64	2.54	1.59	0.54
1 day	0.7	0.9	24644.67	156.99	107.55	8153.30	90.30	48.00	2.63	1.62	0.55
1 day	0.8	0.1	6337.38	79.61	51.44	2222.92	47.15	23.96	0.87	0.93	0.35
1 day	0.8	0.2	6902.03	83.08	54.55	2409.58	49.09	24.86	0.90	0.95	0.34
1 day	0.8	0.3	7518.68	86.71	57.78	2607.46	51.06	25.69	0.93	0.97	0.32
1 day	0.8	0.4	8191.65	90.51	61.02	2818.91	53.09	26.83	0.97	0.98	0.32
1 day	0.8	0.5	8925.91	94.48	64.19	3046.59	55.20	28.09	1.01	1.00	0.32
1 day	0.8	0.6	9725.63	98.62	67.23	3292.86	57.38	29.17	1.05	1.02	0.33
1 day	0.8	0.7	10593.24	102.92	70.11	3559.92	59.67	30.02	1.10	1.05	0.34
1 day	0.8	0.8	11529.99	107.38	72.77	3849.97	62.05	30.61	1.15	1.07	0.34
1 day	0.8	0.9	12537.63	111.97	75.17	4165.41	64.54	31.35	1.20	1.10	0.35
1 day	0.9	0.1	1718.08	41.45	27.15	602.19	24.54	12.04	0.21	0.46	0.16
1 day	0.9	0.2	1885.12	43.42	28.79	657.79	25.65	12.48	0.22	0.47	0.16
1 day	0.9	0.3	2068.85	45.48	30.43	717.92	26.79	13.01	0.23	0.48	0.16
1 day	0.9	0.4	2270.67	47.65	32.02	783.41	27.99	13.50	0.24	0.49	0.16
1 day	0.9	0.5	2492.07	49.92	33.51	855.23	29.24	13.95	0.26	0.51	0.16
1 day	0.9	0.6	2734.42	52.29	34.86	934.37	30.57	14.52	0.27	0.52	0.16
1 day	0.9	0.7	2999.13	54.76	36.12	1021.95	31.97	15.41	0.29	0.53	0.16
1 day	0.9	0.8	3287.96	57.34	37.29	1119.33	33.46	16.39	0.30	0.55	0.16
1 day	0.9	0.9	3603.29	60.03	39.06	1228.21	35.05	17.40	0.32	0.57	0.17
1 week	0.1	0.1	19607.61	140.03	90.71	86865.99	294.73	222.52	220742.73	469.83	255.83
1 week	0.1	0.2	14235.72	119.31	77.68	104665.46	323.52	237.45	202361.96	449.85	247.22

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
1 week	0.1	0.3	12180.46	110.37	73.28	117387.80	342.62	257.95	189438.09	435.24	240.68
1 week	0.1	0.4	11703.78	108.18	73.65	121052.67	347.93	268.92	182202.03	426.85	236.32
1 week	0.1	0.5	12038.56	109.72	75.73	116527.66	341.36	270.40	179040.44	423.13	234.42
1 week	0.1	0.6	12840.87	113.32	78.23	107291.55	327.55	259.76	178046.14	421.96	235.70
1 week	0.1	0.7	13941.15	118.07	81.62	96772.32	311.08	250.75	177624.43	421.46	236.41
1 week	0.1	0.8	15234.08	123.43	85.83	87076.63	295.09	238.33	176694.20	420.35	236.02
1 week	0.1	0.9	16634.03	128.97	89.41	78976.88	281.03	227.62	174704.14	417.98	233.50
1 week	0.2	0.1	9641.84	98.19	59.16	56348.99	237.38	176.75	123569.02	351.52	192.84
1 week	0.2	0.2	8778.01	93.69	59.01	58919.19	242.73	182.00	114587.40	338.51	183.83
1 week	0.2	0.3	9204.19	95.94	61.34	55189.76	234.93	178.80	109988.63	331.65	178.48
1 week	0.2	0.4	10065.40	100.33	64.03	50005.03	223.62	171.26	106884.89	326.93	177.47
1 week	0.2	0.5	11023.79	104.99	66.82	45887.50	214.21	163.67	103973.25	322.45	178.38
1 week	0.2	0.6	11902.75	109.10	68.96	43216.18	207.89	162.67	101182.35	318.09	177.17
1 week	0.2	0.7	12620.56	112.34	70.08	41776.53	204.39	163.55	98860.74	314.42	179.32
1 week	0.2	0.8	13164.67	114.74	71.77	41339.86	203.32	165.20	97298.86	311.93	180.80
1 week	0.2	0.9	13558.91	116.44	73.78	41713.05	204.24	167.18	96617.91	310.83	181.12
1 week	0.3	0.1	6250.22	79.06	46.25	35095.04	187.34	136.54	74356.81	272.68	146.37
1 week	0.3	0.2	6302.85	79.39	48.77	34478.46	185.68	135.53	70148.33	264.86	141.33
1 week	0.3	0.3	6793.10	82.42	52.06	32710.54	180.86	134.80	68126.19	261.01	142.05
1 week	0.3	0.4	7321.20	85.56	53.62	31684.66	178.00	138.64	67153.67	259.14	146.19
1 week	0.3	0.5	7767.05	88.13	54.89	31610.57	177.79	141.63	67043.67	258.93	149.46
1 week	0.3	0.6	8108.72	90.05	55.64	32321.31	179.78	144.49	67746.46	260.28	151.08
1 week	0.3	0.7	8352.41	91.39	56.27	33644.91	183.43	147.08	69135.98	262.94	152.74
1 week	0.3	0.8	8505.50	92.23	56.67	35448.30	188.28	149.86	71030.98	266.52	157.29

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
1 week	0.3	0.9	8578.51	92.62	55.81	37626.41	193.98	153.25	73225.59	270.60	161.11
1 week	0.4	0.1	4148.04	64.41	36.87	23052.56	151.83	111.34	46376.84	215.35	114.83
1 week	0.4	0.2	4318.77	65.72	39.92	22950.98	151.50	112.15	44915.51	211.93	116.03
1 week	0.4	0.3	4630.18	68.05	41.79	22894.86	151.31	115.78	44874.67	211.84	120.57
1 week	0.4	0.4	4918.72	70.13	42.28	23392.99	152.95	119.16	45656.46	213.67	123.00
1 week	0.4	0.5	5151.27	71.77	42.03	24374.84	156.12	122.58	46982.04	216.75	127.86
1 week	0.4	0.6	5326.42	72.98	41.85	25699.72	160.31	125.40	48620.78	220.50	132.48
1 week	0.4	0.7	5454.43	73.85	41.69	27230.17	165.02	128.26	50372.93	224.44	136.04
1 week	0.4	0.8	5555.09	74.53	41.29	28828.37	169.79	131.47	52085.26	228.22	138.80
1 week	0.4	0.9	5652.22	75.18	41.93	30356.84	174.23	133.95	53664.75	231.66	140.94
1 week	0.5	0.1	2669.09	51.66	29.10	15350.22	123.90	91.88	29033.66	170.39	91.36
1 week	0.5	0.2	2812.78	53.04	31.26	15674.82	125.20	93.95	28931.39	170.09	95.42
1 week	0.5	0.3	3002.64	54.80	31.86	16162.64	127.13	97.11	29641.68	172.17	99.76
1 week	0.5	0.4	3171.62	56.32	31.47	16924.25	130.09	100.31	30747.34	175.35	104.73
1 week	0.5	0.5	3309.66	57.53	31.37	17870.61	133.68	102.86	32014.44	178.93	108.25
1 week	0.5	0.6	3421.98	58.50	31.09	18896.97	137.47	104.75	33294.77	182.47	110.76
1 week	0.5	0.7	3519.65	59.33	31.98	19911.95	141.11	106.44	34509.77	185.77	112.59
1 week	0.5	0.8	3612.73	60.11	32.57	20853.30	144.41	107.76	35639.45	188.78	113.81
1 week	0.5	0.9	3705.51	60.87	32.96	21702.04	147.32	109.38	36706.21	191.59	115.08
1 week	0.6	0.1	1611.66	40.15	22.01	9800.62	99.00	73.76	17467.12	132.16	71.75
1 week	0.6	0.2	1707.93	41.33	22.99	10206.46	101.03	75.40	17803.83	133.43	76.02
1 week	0.6	0.3	1818.11	42.64	23.16	10713.05	103.50	78.14	18524.10	136.10	80.47
1 week	0.6	0.4	1914.80	43.76	22.94	11328.54	106.44	80.04	19381.97	139.22	83.47
1 week	0.6	0.5	1995.84	44.67	23.21	11997.37	109.53	81.24	20259.55	142.34	85.72

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
1 week	0.6	0.6	2065.28	45.45	23.66	12673.21	112.58	81.81	21110.74	145.30	87.27
1 week	0.6	0.7	2127.54	46.13	23.79	13334.08	115.47	83.32	21935.89	148.11	90.19
1 week	0.6	0.8	2184.82	46.74	23.61	13984.82	118.26	85.35	22761.49	150.87	93.20
1 week	0.6	0.9	2237.44	47.30	23.77	14649.47	121.03	86.81	23623.41	153.70	95.99
1 week	0.7	0.1	868.59	29.47	15.43	5631.86	75.05	55.44	9505.50	97.50	53.10
1 week	0.7	0.2	922.78	30.38	15.91	5946.03	77.11	56.93	9845.05	99.22	57.16
1 week	0.7	0.3	980.14	31.31	15.94	6306.88	79.42	58.49	10337.64	101.67	59.94
1 week	0.7	0.4	1030.46	32.10	16.02	6708.42	81.90	59.41	10871.59	104.27	61.73
1 week	0.7	0.5	1073.80	32.77	16.23	7130.00	84.44	60.54	11408.03	106.81	65.10
1 week	0.7	0.6	1112.28	33.35	16.41	7561.97	86.96	62.00	11944.81	109.29	67.89
1 week	0.7	0.7	1147.83	33.88	17.16	8007.37	89.48	63.87	12496.48	111.79	70.53
1 week	0.7	0.8	1181.82	34.38	17.98	8476.88	92.07	66.08	13081.12	114.37	72.92
1 week	0.7	0.9	1215.45	34.86	18.77	8982.02	94.77	68.31	13712.43	117.10	75.08
1 week	0.8	0.1	375.58	19.38	9.73	2602.36	51.01	37.11	4177.20	64.63	35.34
1 week	0.8	0.2	399.66	19.99	9.84	2776.90	52.70	38.21	4377.87	66.17	37.59
1 week	0.8	0.3	424.23	20.60	9.70	2970.44	54.50	39.31	4629.39	68.04	40.21
1 week	0.8	0.4	446.34	21.13	10.15	3180.69	56.40	40.38	4895.38	69.97	42.82
1 week	0.8	0.5	466.42	21.60	10.78	3403.59	58.34	41.29	5168.40	71.89	45.03
1 week	0.8	0.6	485.51	22.03	11.32	3639.58	60.33	42.68	5452.68	73.84	46.85
1 week	0.8	0.7	504.53	22.46	11.85	3892.26	62.39	44.31	5755.10	75.86	48.38
1 week	0.8	0.8	524.19	22.90	12.49	4165.87	64.54	46.54	6080.85	77.98	49.70
1 week	0.8	0.9	545.05	23.35	13.06	4463.69	66.81	49.15	6432.03	80.20	51.54
1 week	0.9	0.1	92.95	9.64	4.71	687.54	26.22	18.71	1052.61	32.44	17.37
1 week	0.9	0.2	99.19	9.96	4.73	741.10	27.22	19.58	1114.61	33.39	19.24

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
1 week	0.9	0.3	105.57	10.27	4.92	800.17	28.29	20.23	1187.65	34.46	20.80
1 week	0.9	0.4	111.63	10.57	5.24	864.77	29.41	20.83	1265.53	35.57	22.01
1 week	0.9	0.5	117.58	10.84	5.55	934.99	30.58	21.66	1348.05	36.72	22.97
1 week	0.9	0.6	123.70	11.12	5.90	1011.79	31.81	22.84	1436.88	37.91	23.79
1 week	0.9	0.7	130.24	11.41	6.22	1096.62	33.12	24.14	1533.85	39.16	24.57
1 week	0.9	0.8	137.41	11.72	6.52	1191.16	34.51	25.60	1640.56	40.50	25.52
1 week	0.9	0.9	145.42	12.06	6.80	1297.40	36.02	27.05	1758.69	41.94	26.45
2 weeks	0.1	0.1	13183.18	114.82	87.61	62480.22	249.96	195.52	8073.27	89.85	73.29
2 weeks	0.1	0.2	10991.16	104.84	80.70	59913.58	244.77	195.89	8958.86	94.65	76.01
2 weeks	0.1	0.3	9510.68	97.52	75.58	65024.35	255.00	206.52	10536.36	102.65	81.91
2 weeks	0.1	0.4	10093.48	100.47	78.96	79795.49	282.48	233.13	12500.70	111.81	90.83
2 weeks	0.1	0.5	11099.43	105.35	83.27	92472.86	304.09	256.74	13667.67	116.91	95.38
2 weeks	0.1	0.6	11146.03	105.57	83.13	93166.51	305.23	253.88	13361.67	115.59	92.47
2 weeks	0.1	0.7	10143.41	100.71	79.88	83163.42	288.38	241.46	12224.47	110.56	87.95
2 weeks	0.1	0.8	8736.18	93.47	73.58	69322.01	263.29	217.45	11055.58	105.15	84.19
2 weeks	0.1	0.9	7556.75	86.93	68.06	57455.68	239.70	196.05	10266.84	101.33	82.26
2 weeks	0.2	0.1	4553.70	67.48	45.48	28031.56	167.43	130.32	5883.51	76.70	61.66
2 weeks	0.2	0.2	4227.13	65.02	45.47	29937.28	173.02	139.66	6465.57	80.41	65.04
2 weeks	0.2	0.3	4157.86	64.48	45.20	30544.17	174.77	140.40	6715.13	81.95	66.58
2 weeks	0.2	0.4	3961.88	62.94	43.13	28179.82	167.87	135.85	6698.99	81.85	66.42
2 weeks	0.2	0.5	3798.86	61.63	42.70	25352.00	159.22	129.02	6745.25	82.13	67.26
2 weeks	0.2	0.6	3806.66	61.70	43.08	23622.64	153.70	125.10	6969.95	83.49	69.01
2 weeks	0.2	0.7	3964.00	62.96	44.11	23062.41	151.86	123.89	7369.75	85.85	71.08
2 weeks	0.2	0.8	4197.03	64.78	45.28	23237.73	152.44	126.10	7937.73	89.09	74.12

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
2 weeks	0.2	0.9	4429.49	66.55	46.32	23563.48	153.50	127.25	8670.38	93.11	77.51
2 weeks	0.3	0.1	2294.39	47.90	30.14	15367.46	123.97	97.03	4241.77	65.13	52.49
2 weeks	0.3	0.2	2212.62	47.04	30.17	15494.80	124.48	96.21	4494.49	67.04	54.21
2 weeks	0.3	0.3	2184.25	46.74	29.39	14817.73	121.73	96.31	4673.44	68.36	55.85
2 weeks	0.3	0.4	2198.41	46.89	29.64	14037.80	118.48	95.11	4905.86	70.04	57.15
2 weeks	0.3	0.5	2262.08	47.56	30.06	13582.03	116.54	94.88	5232.53	72.34	59.01
2 weeks	0.3	0.6	2341.33	48.39	30.53	13310.05	115.37	93.37	5632.50	75.05	61.09
2 weeks	0.3	0.7	2407.62	49.07	31.12	13015.53	114.09	91.22	6058.73	77.84	63.41
2 weeks	0.3	0.8	2453.96	49.54	31.79	12654.57	112.49	88.25	6446.10	80.29	65.61
2 weeks	0.3	0.9	2481.50	49.81	32.00	12311.30	110.96	86.87	6725.12	82.01	67.74
2 weeks	0.4	0.1	1282.85	35.82	21.07	9104.38	95.42	73.50	3054.28	55.27	43.79
2 weeks	0.4	0.2	1264.23	35.56	21.62	9015.87	94.95	73.44	3227.91	56.81	45.34
2 weeks	0.4	0.3	1273.81	35.69	21.12	8730.97	93.44	73.92	3400.98	58.32	46.24
2 weeks	0.4	0.4	1299.02	36.04	21.44	8497.53	92.18	73.51	3600.53	60.00	47.12
2 weeks	0.4	0.5	1326.57	36.42	21.68	8343.70	91.34	72.30	3805.90	61.69	48.80
2 weeks	0.4	0.6	1346.17	36.69	22.00	8236.63	90.76	71.45	3980.03	63.09	50.47
2 weeks	0.4	0.7	1354.88	36.81	21.86	8185.57	90.47	71.45	4093.09	63.98	51.77
2 weeks	0.4	0.8	1353.58	36.79	22.02	8212.88	90.62	71.29	4137.77	64.33	52.64
2 weeks	0.4	0.9	1345.85	36.69	22.05	8324.00	91.24	71.47	4130.85	64.27	53.04
2 weeks	0.5	0.1	727.03	26.96	15.13	5590.31	74.77	57.63	2145.88	46.32	36.08
2 weeks	0.5	0.2	724.77	26.92	15.48	5557.82	74.55	57.80	2266.21	47.60	36.96
2 weeks	0.5	0.3	733.20	27.08	15.36	5479.93	74.03	57.63	2384.99	48.84	37.91
2 weeks	0.5	0.4	743.90	27.27	15.48	5440.29	73.76	57.66	2500.09	50.00	39.18
2 weeks	0.5	0.5	751.81	27.42	15.35	5449.56	73.82	57.80	2596.27	50.95	40.52

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
2 weeks	0.5	0.6	755.44	27.49	15.32	5507.29	74.21	58.01	2664.46	51.62	41.86
2 weeks	0.5	0.7	755.85	27.49	15.57	5612.54	74.92	59.02	2708.11	52.04	42.69
2 weeks	0.5	0.8	755.00	27.48	15.63	5757.47	75.88	60.08	2738.45	52.33	43.10
2 weeks	0.5	0.9	754.49	27.47	15.71	5929.08	77.00	61.24	2766.64	52.60	43.31
2 weeks	0.6	0.1	395.83	19.90	10.69	3384.93	58.18	45.00	1429.28	37.81	28.95
2 weeks	0.6	0.2	396.55	19.91	10.99	3405.91	58.36	45.05	1509.19	38.85	29.93
2 weeks	0.6	0.3	400.79	20.02	10.95	3423.15	58.51	45.35	1585.51	39.82	30.96
2 weeks	0.6	0.4	404.90	20.12	10.88	3468.59	58.89	45.58	1655.51	40.69	32.29
2 weeks	0.6	0.5	407.60	20.19	10.92	3546.28	59.55	46.28	1715.84	41.42	33.42
2 weeks	0.6	0.6	409.04	20.22	11.23	3653.48	60.44	47.44	1768.32	42.05	34.11
2 weeks	0.6	0.7	409.90	20.25	11.43	3785.35	61.53	48.56	1818.15	42.64	34.57
2 weeks	0.6	0.8	410.67	20.27	11.46	3936.75	62.74	49.95	1870.54	43.25	34.92
2 weeks	0.6	0.9	411.53	20.29	11.42	4104.66	64.07	51.19	1929.08	43.92	35.23
2 weeks	0.7	0.1	195.16	13.97	7.21	1894.43	43.53	33.70	860.75	29.34	22.34
2 weeks	0.7	0.2	196.05	14.00	7.48	1935.72	44.00	33.72	912.17	30.20	23.29
2 weeks	0.7	0.3	198.05	14.07	7.58	1983.36	44.53	34.13	962.36	31.02	24.35
2 weeks	0.7	0.4	199.88	14.14	7.73	2049.05	45.27	35.05	1011.04	31.80	25.30
2 weeks	0.7	0.5	201.28	14.19	7.92	2133.66	46.19	36.04	1058.85	32.54	25.99
2 weeks	0.7	0.6	202.41	14.23	8.02	2235.55	47.28	37.17	1108.09	33.29	26.53
2 weeks	0.7	0.7	203.48	14.26	8.04	2353.16	48.51	38.37	1161.52	34.08	27.09
2 weeks	0.7	0.8	204.62	14.30	8.06	2485.99	49.86	39.42	1221.48	34.95	27.62
2 weeks	0.7	0.9	205.93	14.35	8.16	2634.67	51.33	40.60	1289.81	35.91	28.14
2 weeks	0.8	0.1	77.91	8.83	4.40	871.15	29.52	22.78	421.28	20.53	15.69
2 weeks	0.8	0.2	78.49	8.86	4.74	905.42	30.09	23.07	450.29	21.22	16.43

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
2 weeks	0.8	0.3	79.40	8.91	4.93	945.66	30.75	23.77	480.16	21.91	17.16
2 weeks	0.8	0.4	80.29	8.96	5.06	995.62	31.55	24.53	511.45	22.62	17.82
2 weeks	0.8	0.5	81.12	9.01	5.13	1055.72	32.49	25.46	545.11	23.35	18.40
2 weeks	0.8	0.6	81.96	9.05	5.16	1125.86	33.55	26.44	582.51	24.14	18.95
2 weeks	0.8	0.7	82.88	9.10	5.18	1206.33	34.73	27.46	625.09	25.00	19.49
2 weeks	0.8	0.8	83.96	9.16	5.24	1297.91	36.03	28.61	674.21	25.97	20.07
2 weeks	0.8	0.9	85.24	9.23	5.28	1401.83	37.44	29.86	731.29	27.04	20.74
2 weeks	0.9	0.1	17.88	4.23	2.12	232.72	15.26	11.83	119.22	10.92	8.37
2 weeks	0.9	0.2	18.09	4.25	2.34	246.35	15.70	12.16	129.15	11.36	8.80
2 weeks	0.9	0.3	18.37	4.29	2.43	262.44	16.20	12.60	139.97	11.83	9.22
2 weeks	0.9	0.4	18.67	4.32	2.49	281.82	16.79	13.17	152.02	12.33	9.62
2 weeks	0.9	0.5	18.98	4.36	2.52	304.81	17.46	13.80	165.74	12.87	10.01
2 weeks	0.9	0.6	19.34	4.40	2.54	331.71	18.21	14.49	181.68	13.48	10.41
2 weeks	0.9	0.7	19.75	4.44	2.56	363.05	19.05	15.24	200.46	14.16	10.85
2 weeks	0.9	0.8	20.25	4.50	2.58	399.56	19.99	16.06	222.82	14.93	11.34
2 weeks	0.9	0.9	20.84	4.57	2.59	442.27	21.03	16.95	249.64	15.80	11.92
4 weeks	0.1	0.1	14430.35	120.13	94.55	65010.46	254.97	196.71	6363.91	79.77	63.83
4 weeks	0.1	0.2	17004.44	130.40	104.33	75725.68	275.18	220.57	6971.37	83.49	67.96
4 weeks	0.1	0.3	21732.85	147.42	115.95	96518.23	310.67	246.11	7926.77	89.03	71.80
4 weeks	0.1	0.4	26131.02	161.65	128.79	119308.74	345.41	282.40	9038.85	95.07	76.98
4 weeks	0.1	0.5	27598.81	166.13	136.06	130207.60	360.84	301.52	9691.93	98.45	80.38
4 weeks	0.1	0.6	25362.95	159.26	131.84	122985.69	350.69	293.28	9759.95	98.79	79.79
4 weeks	0.1	0.7	21127.85	145.35	120.59	106285.86	326.02	274.45	9736.29	98.67	79.16
4 weeks	0.1	0.8	16935.83	130.14	106.81	90239.62	300.40	252.26	9852.67	99.26	79.82

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
4 weeks	0.1	0.9	13867.56	117.76	96.07	79282.91	281.57	234.11	10012.73	100.06	80.43
4 weeks	0.2	0.1	7336.04	85.65	60.03	36792.19	191.81	144.94	4764.49	69.03	56.44
4 weeks	0.2	0.2	8280.11	91.00	66.14	42351.98	205.80	159.13	5240.88	72.39	58.97
4 weeks	0.2	0.3	8144.32	90.25	68.29	43608.32	208.83	166.21	5593.76	74.79	60.91
4 weeks	0.2	0.4	7267.59	85.25	64.08	41434.59	203.55	164.15	5879.19	76.68	62.05
4 weeks	0.2	0.5	6556.68	80.97	61.33	39894.22	199.74	162.49	6218.83	78.86	63.92
4 weeks	0.2	0.6	6310.32	79.44	60.45	40451.28	201.13	165.55	6662.06	81.62	66.63
4 weeks	0.2	0.7	6405.93	80.04	60.53	42646.60	206.51	169.44	7220.04	84.97	69.54
4 weeks	0.2	0.8	6624.36	81.39	61.56	45339.25	212.93	173.58	7857.86	88.64	73.22
4 weeks	0.2	0.9	6796.49	82.44	62.24	47452.18	217.84	173.29	8516.29	92.28	77.21
4 weeks	0.3	0.1	3838.71	61.96	39.86	20914.95	144.62	107.51	3473.11	58.93	48.02
4 weeks	0.3	0.2	3920.12	62.61	43.28	22211.67	149.04	112.69	3752.30	61.26	49.08
4 weeks	0.3	0.3	3712.60	60.93	42.76	22217.72	149.06	117.27	4015.60	63.37	51.04
4 weeks	0.3	0.4	3556.14	59.63	41.89	22311.78	149.37	118.69	4307.57	65.63	53.17
4 weeks	0.3	0.5	3520.48	59.33	41.61	22762.09	150.87	120.00	4625.51	68.01	55.53
4 weeks	0.3	0.6	3526.92	59.39	41.67	23112.12	152.03	121.28	4932.06	70.23	57.59
4 weeks	0.3	0.7	3518.64	59.32	41.16	23009.88	151.69	120.88	5181.07	71.98	59.07
4 weeks	0.3	0.8	3486.39	59.05	41.09	22413.74	149.71	118.75	5335.89	73.05	59.85
4 weeks	0.3	0.9	3442.69	58.67	40.78	21487.52	146.59	116.75	5381.85	73.36	60.16
4 weeks	0.4	0.1	2113.37	45.97	27.72	12223.54	110.56	80.22	2479.52	49.79	40.20
4 weeks	0.4	0.2	2108.70	45.92	30.06	12671.28	112.57	84.89	2663.95	51.61	41.82
4 weeks	0.4	0.3	2044.40	45.21	29.91	12749.15	112.91	86.38	2844.36	53.33	43.29
4 weeks	0.4	0.4	2010.36	44.84	29.21	12789.11	113.09	88.37	3018.25	54.94	44.70
4 weeks	0.4	0.5	1996.38	44.68	29.00	12720.39	112.78	89.09	3164.35	56.25	46.06

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
4 weeks	0.4	0.6	1985.46	44.56	29.03	12484.02	111.73	88.38	3265.13	57.14	46.67
4 weeks	0.4	0.7	1976.49	44.46	28.70	12144.41	110.20	86.84	3319.90	57.62	47.16
4 weeks	0.4	0.8	1973.97	44.43	28.96	11809.69	108.67	84.88	3344.66	57.83	47.75
4 weeks	0.4	0.9	1980.99	44.51	29.15	11559.99	107.52	82.78	3362.61	57.99	48.07
4 weeks	0.5	0.1	1183.66	34.40	19.63	7183.53	84.76	62.03	1709.04	41.34	33.49
4 weeks	0.5	0.2	1178.97	34.34	21.34	7371.27	85.86	64.25	1826.42	42.74	34.67
4 weeks	0.5	0.3	1158.48	34.04	21.29	7396.80	86.00	65.81	1935.27	43.99	35.92
4 weeks	0.5	0.4	1148.60	33.89	20.96	7364.61	85.82	67.12	2029.76	45.05	36.92
4 weeks	0.5	0.5	1145.99	33.85	20.85	7283.25	85.34	67.08	2104.82	45.88	37.76
4 weeks	0.5	0.6	1148.31	33.89	20.63	7185.39	84.77	66.25	2164.29	46.52	38.46
4 weeks	0.5	0.7	1155.71	34.00	20.74	7113.94	84.34	65.91	2218.65	47.10	39.10
4 weeks	0.5	0.8	1168.03	34.18	20.97	7093.19	84.22	65.92	2278.50	47.73	39.62
4 weeks	0.5	0.9	1184.19	34.41	21.06	7125.20	84.41	66.61	2350.43	48.48	40.12
4 weeks	0.6	0.1	642.63	25.35	13.85	4092.19	63.97	46.85	1109.73	33.31	26.89
4 weeks	0.6	0.2	641.98	25.34	15.04	4183.83	64.68	48.65	1182.75	34.39	27.93
4 weeks	0.6	0.3	636.67	25.23	15.18	4204.79	64.84	49.74	1249.82	35.35	28.91
4 weeks	0.6	0.4	636.12	25.22	14.88	4204.97	64.85	50.22	1309.91	36.19	29.84
4 weeks	0.6	0.5	639.53	25.29	14.68	4203.02	64.83	50.71	1364.81	36.94	30.65
4 weeks	0.6	0.6	646.01	25.42	14.75	4216.42	64.93	51.37	1419.20	37.67	31.21
4 weeks	0.6	0.7	654.92	25.59	14.90	4255.20	65.23	51.74	1477.68	38.44	31.65
4 weeks	0.6	0.8	665.54	25.80	14.98	4320.01	65.73	52.43	1542.71	39.28	32.43
4 weeks	0.6	0.9	677.15	26.02	15.19	4406.66	66.38	52.97	1614.62	40.18	33.26
4 weeks	0.7	0.1	317.92	17.83	9.26	2139.22	46.25	33.88	647.80	25.45	20.44
4 weeks	0.7	0.2	319.03	17.86	10.25	2192.25	46.82	34.78	691.39	26.29	21.31

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
4 weeks	0.7	0.3	318.94	17.86	10.32	2220.48	47.12	36.00	732.94	27.07	22.15
4 weeks	0.7	0.4	321.00	17.92	10.15	2247.01	47.40	37.09	773.15	27.81	22.84
4 weeks	0.7	0.5	324.88	18.02	10.21	2281.30	47.76	37.81	813.75	28.53	23.33
4 weeks	0.7	0.6	330.07	18.17	10.34	2328.85	48.26	38.34	856.73	29.27	23.88
4 weeks	0.7	0.7	336.13	18.33	10.39	2391.48	48.90	38.91	903.31	30.06	24.60
4 weeks	0.7	0.8	342.70	18.51	10.48	2468.91	49.69	39.43	953.81	30.88	25.23
4 weeks	0.7	0.9	349.53	18.70	10.58	2560.73	50.60	40.16	1008.04	31.75	25.78
4 weeks	0.8	0.1	127.73	11.30	5.69	917.66	30.29	22.08	305.71	17.48	13.98
4 weeks	0.8	0.2	128.86	11.35	6.31	947.38	30.78	23.04	328.05	18.11	14.59
4 weeks	0.8	0.3	129.79	11.39	6.42	971.57	31.17	24.08	350.45	18.72	15.15
4 weeks	0.8	0.4	131.51	11.47	6.45	998.64	31.60	24.87	373.52	19.33	15.63
4 weeks	0.8	0.5	133.89	11.57	6.49	1031.94	32.12	25.40	398.05	19.95	16.09
4 weeks	0.8	0.6	136.73	11.69	6.50	1073.13	32.76	25.89	424.73	20.61	16.59
4 weeks	0.8	0.7	139.88	11.83	6.57	1122.97	33.51	26.48	454.03	21.31	17.06
4 weeks	0.8	0.8	143.25	11.97	6.62	1181.99	34.38	27.17	486.30	22.05	17.56
4 weeks	0.8	0.9	146.80	12.12	6.77	1250.91	35.37	27.89	521.97	22.85	18.10
4 weeks	0.9	0.1	29.53	5.43	2.68	229.07	15.13	11.18	83.10	9.12	7.20
4 weeks	0.9	0.2	29.97	5.47	3.00	239.23	15.47	11.70	90.02	9.49	7.53
4 weeks	0.9	0.3	30.41	5.51	3.10	249.27	15.79	12.24	97.31	9.86	7.84
4 weeks	0.9	0.4	31.04	5.57	3.10	260.96	16.15	12.68	105.21	10.26	8.12
4 weeks	0.9	0.5	31.81	5.64	3.12	275.07	16.59	13.04	113.97	10.68	8.42
4 weeks	0.9	0.6	32.71	5.72	3.14	292.07	17.09	13.46	123.84	11.13	8.72
4 weeks	0.9	0.7	33.71	5.81	3.19	312.35	17.67	13.92	135.10	11.62	9.03
4 weeks	0.9	0.8	34.82	5.90	3.24	336.34	18.34	14.41	148.08	12.17	9.34

Time interval			CPU			Memory			I/O		
	α	β	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
4 weeks	0.9	0.9	36.05	6.00	3.27	364.62	19.09	14.98	163.23	12.78	9.69

Appendix F – Prediction Results Using ARIMA Method

Table F1: ARIMA method for different time periods

Time interval				CPU test			Memory test			Disk test		
	p	d	q	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
5 minutes	0	1	0	0.0100	0.0900	0.0400	62.2100	7.8900	0.8100	0.2300	0.4800	0.1600
	0	1	1	5370.0400	73.2800	9.7400	10333890.4900	3214.6400	906.4700	25862.7900	160.8200	59.9200
	1	1	1	5572.3800	74.6500	9.4400	9245027.4600	3040.5600	861.6800	19541.2700	139.7900	55.2100
	1	0	0	5613.8600	74.9300	9.6200	8985012.1100	2997.5000	671.9700	18292.7300	135.2500	67.7500
	0	0	0	5522.2416	74.3118	10.7035	11847222.0683	3441.9794	763.8845	42122.7690	205.2383	99.4288
	0	0	1	5607.2026	74.8813	9.8501	9106816.9165	3017.7503	687.9435	22783.2652	150.9413	77.9168
	1	1	0	5158.1580	71.8203	5.8434	6638977.5550	2576.6213	319.6355	12295.6266	110.8856	31.9978
	1	0	1	5914.8569	76.9081	9.8149	9084603.1499	3014.0675	651.9676	23932.7113	154.7020	57.0296
10 minutes	0	1	0	0.0000	0.0700	0.0300	142.4100	11.9300	1.5900	0.4600	0.6700	0.2600
	0	1	1	29.5700	5.4400	3.6700	5864081.5000	2421.5900	908.2400	13333.9700	115.4700	52.1700
	1	1	1	22.8900	4.7800	3.2700	5088754.5900	2255.8300	853.1000	10728.2300	103.5800	47.8000
	1	0	0	37.4100	6.1200	4.7600	5217467.7600	2284.1800	675.4800	9802.5900	99.0100	57.2700
	0	0	0	86.7450	9.3137	7.1046	6717479.6611	2591.8101	763.6912	28367.4402	168.4264	96.5679
	0	0	1	52.7754	7.2647	5.6114	5052548.2002	2247.7874	672.6463	14515.7596	120.4814	71.7941
	1	1	0	14.3889	3.7933	2.2676	3381301.7161	1838.8316	327.1896	6601.6521	81.2506	27.2436

Time interval				CPU test			Memory test			Disk test		
	p	d	q	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
	1	0	1	28.0249	5.2939	3.8809	5145496.2656	2268.3686	659.7672	12604.3650	112.2692	49.9596
20 minutes	0	1	0	0.0100	0.0700	0.0400	294.3600	17.1600	3.1200	1.2300	1.1100	0.4700
	0	1	1	16.9900	4.1200	2.7500	3858264.6800	1964.2500	919.2500	8040.0900	89.6700	47.1700
	1	1	1	12.5600	3.5400	2.4400	3142284.7800	1772.6500	834.9400	8342.5000	91.3400	48.1300
	1	0	0	20.6600	4.5500	3.4200	3114341.5200	1764.7500	654.9800	7260.7400	85.2100	54.9700
	0	0	0	69.5921	8.3422	6.0503	4144011.5128	2035.6845	765.5782	20714.6051	143.9257	91.2628
	0	0	1	36.4512	6.0375	4.4180	3326162.8451	1823.7771	691.2077	11985.5369	109.4785	71.5792
	1	1	0	6.4598	2.5416	1.5650	2101154.9748	1449.5361	369.7714	7036.9501	83.8865	33.8887
	1	0	1	18.0084	4.2436	2.8614	3187064.5579	1785.2352	664.2569	7596.3236	87.1569	46.4696
1 hour	0	1	0	0.0110	0.1049	0.0729	588.8413	24.2661	6.9864	4.6156	2.1484	1.1271
	0	1	1	10.8500	3.2900	2.2600	1669981.6300	1292.2800	785.7100	2032.9000	45.0900	28.8800
	1	1	1	8.4976	2.9151	1.9666	1500114.2559	1224.7915	745.5265	2641.9366	51.3998	37.9756
	1	0	0	12.9000	3.5900	2.6100	1615564.6500	1271.0500	656.5300	2568.3000	50.6800	35.7300
	0	0	0	54.1088	7.3559	4.8077	1735836.7268	1317.5116	681.7991	12696.7911	112.6800	80.9877
	0	0	1	28.0758	5.2987	3.6323	1580083.5226	1257.0137	646.9297	5908.0146	76.8636	55.2598
	1	1	0	3.1198	1.7663	1.2682	815949.9846	903.2995	353.6103	1316.0611	36.2776	17.8696
	1	0	1	9.6365	3.1043	2.1290	1635976.2723	1279.0529	634.1526	2481.7944	49.8176	34.1583
4 hours	0	1	0	0.2000	0.4500	0.2000	138.5900	11.7700	5.1400	16.9900	4.1200	1.4000
	0	1	1	134.7000	11.6100	4.0500	387194.5000	622.2500	445.1700	34808.7900	186.5700	80.0400
	1	1	1	133.2300	11.5400	4.2700	570936.5200	755.6000	521.1500	71032.1600	266.5200	91.0700
	1	0	0	115.8000	10.7600	8.0700	546040.4500	738.9500	490.3300	29406.8700	171.4800	71.6800

Time interval				CPU test			Memory test			Disk test		
	p	d	q	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
	0	0	0	483.8516	21.9966	12.8557	403819.9095	635.4683	434.0459	80069.8510	282.9662	103.3000
	0	0	1	154.2760	12.4208	9.9428	593148.4135	770.1613	510.0577	46660.3654	216.0101	82.8588
	1	1	0	381.3970	19.5294	2.6225	234431.0299	484.1808	302.9909	75212.2657	274.2485	61.4646
	1	0	1	62.6126	7.9128	5.1238	569572.6881	754.7004	503.4899	34314.9528	185.2430	74.8901
12 hours	0	1	0	0.3600	0.6000	0.2900	273.7500	16.5500	10.2200	46.2800	6.8000	2.7500
	0	1	1	77.4300	8.8000	3.3800	72162.5500	268.6300	186.6500	5653.6000	75.1900	49.2100
	1	1	1	188.9800	13.7500	5.8300	128927.6300	359.0600	219.3600	33797.9500	183.8400	66.9100
	1	0	0	56.3300	7.5100	5.7000	167937.2700	409.8000	229.5100	2739.5600	52.3400	39.7400
	0	0	0	243.1750	15.5941	11.2604	90577.4049	300.9608	184.4093	31047.5610	176.2032	82.9365
	0	0	1	128.6734	11.3434	8.3495	152880.3870	390.9992	229.6821	15515.0283	124.5593	61.9489
	1	1	0	161.1244	12.6935	3.2071	142122.0922	376.9908	226.5475	38107.4861	195.2114	66.5921
	1	0	1	74.4076	8.6260	5.4238	147198.1170	383.6641	217.1519	7891.9691	88.8368	51.0317
1 day	0	1	0	348.9400	18.6800	7.0900	135.5800	11.6400	3.7600	0.8100	0.9000	0.7000
	0	1	1	52205.7100	228.4900	118.8700	10996.4700	104.8600	69.4600	1.3400	1.1600	0.8800
	1	1	1	60554.1500	246.0800	131.1700	41125.2100	202.7900	85.6500	1.3500	1.1600	0.8900
	1	0	0	167099.2900	408.7800	148.0100	3155.9400	56.1800	37.8000	1.5200	1.2300	0.9200
	0	0	0	64022.4993	253.0267	112.6197	30942.8139	175.9057	71.9606	1.1121	1.0545	0.8124
	0	0	1	182898.8028	427.6667	159.7091	8328.2905	91.2595	50.8834	1.1134	1.0552	0.8128
	1	1	0	44431.8950	210.7887	91.4869	54959.9937	234.4355	75.1266	1.6844	1.2979	0.7871
	1	0	1	241180.6241	491.1014	185.1832	8164.5580	90.3579	52.0501	0.7968	0.8926	0.6953
	0	1	0	69.3900	8.3300	5.8700	170.7100	13.0700	9.2600	7.9300	2.8200	1.8400

Time interval				CPU test			Memory test			Disk test		
	p	d	q	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
1 week	0	1	1	1023.4200	31.9900	11.6300	10655.7700	103.2300	52.7200	3956.6200	62.9000	41.4000
	1	1	1	3524.8700	59.3700	19.9700	8984.4800	94.7900	48.2500	3646.7600	60.3900	39.4900
	1	0	0	5015.1100	70.8200	17.4200	20531.8100	143.2900	90.1300	5394.2900	73.4500	50.0400
	0	0	0	15875.3434	125.9974	88.8476	78211.4318	279.6631	195.8262	4810.1579	69.3553	47.4945
	0	0	1	8633.1360	92.9147	56.7080	50200.6124	224.0549	162.4786	9461.8327	97.2720	60.9683
	1	1	0	399.1362	19.9784	9.1098	3216.5157	56.7143	34.6068	1223.6674	34.9810	20.0883
	1	0	1	5479.8654	74.0261	22.3406	21298.2875	145.9393	78.3289	12140.7820	110.1852	68.8334
2 weeks	0	1	0	34.8902	5.9068	4.0727	161.2819	12.6997	8.4943	19.0322	4.3626	2.7524
	0	1	1	250.8700	15.8400	8.8800	2081.8800	45.6300	32.7900	2524.6000	50.2500	36.3900
	1	1	1	347.8900	18.6500	9.3000	5872.1000	76.6300	57.3100	7038.7900	83.9000	60.4200
	1	0	0	32.2800	5.6800	4.3000	1230.4000	35.0800	26.2100	3488.2200	59.0600	42.2300
	0	0	0	13345.0271	115.5207	100.3746	62630.9655	250.2618	202.6924	7614.4048	87.2606	65.6482
	0	0	1	4489.1789	67.0013	56.8750	27123.5078	164.6922	132.3298	5493.6312	74.1190	55.7388
	1	1	0	311.3160	17.6441	7.3750	5082.7335	71.2933	50.9296	6425.8478	80.1614	56.6815
	1	0	1	263.3140	16.2270	9.9322	2893.0875	53.7874	38.3445	3248.0505	56.9917	42.8170
4 weeks	0	1	0	51.1600	7.1500	4.8600	205.6400	14.3400	9.9800	15.7800	3.9700	2.9300
	0	1	1	606.2700	24.6200	10.7000	1007.9800	31.7500	18.9800	917.0000	30.2800	19.5400
	1	1	1	602.5400	24.5500	13.1500	1577.5000	39.7200	22.5000	755.3300	27.4800	16.2800
	1	0	0	97.4100	9.8700	6.9700	721.7300	26.8600	18.3500	483.3100	21.9800	16.3600
	0	0	0	11130.5250	105.5013	84.4430	47225.1413	217.3135	166.7793	3381.6851	58.1523	43.1146
	0	0	1	3701.1875	60.8374	46.6885	17908.3271	133.8220	98.8997	1210.1003	34.7865	25.1138

Time interval				CPU test			Memory test			Disk test		
	p	d	q	MSE	RMSE	MAD	MSE	RMSE	MAD	MSE	RMSE	MAD
	1	1	0	518.8884	22.7791	9.4728	1477.5704	38.4392	20.9260	157.3327	12.5432	8.4443
	1	0	1	1021.8676	31.9667	18.6503	2449.6439	49.4939	33.4677	701.4740	26.4854	19.4477