Admission Control in Sliced Networks, with Predictive Analytics

Presented by

PEROSE MUTOMBO NGUFOR

This dissertation is submitted to the Department of Electrical Engineering at the University of Cape Town in partial fulfilment of the requirements for the degree

Master of Engineering.

Faculty of Engineering and the Built Environment, University of Cape Town,

Supervisor:

Dr. Joyce B. Mwangama

Dept. of Electrical Engineering, University of Cape Town

Co-Supervisor:

Prof. Albert A. Lysko

Council for Scientific and Industrial Research (CSIR)

October 2022

Dedication

I dedicate this work to my parents Peter and Rosemary, who supported me in every possible way through out this post-graduate journey.

To my siblings and to my friends, for always cheering me on especially when I did not believe in myself.

To my supervisors, Dr. Mwangama and Dr. Lysko, for their guidance, encouragement, patience, and availability throughout this journey.

To God the Almighty for giving me the opportunity and the ability to complete this work.

Declaration

I know the meaning of plagiarism and declare that all the work in this document, save for that which is properly acknowledged, is my own.

The IEEE referencing style has been used in this document.

Signed by candidate

Perose Mutombo Ngufor

Abstract

Over the years, the telecommunications industry has constantly adapted to accommodate the rising demand for more specialised network connectivity. Network slicing was introduced as a solution for providing specialised networks to customers. However, network slicing has a set of objectives which require that legacy network functions be revisited and updated to support network slicing. One such function is admission control. This work proposes two admission control algorithms and investigates how the admission control function can be improved by incorporating traffic forecasting into the admission control process.

In this work, we present the state-of-the-art in admission control in sliced networks and the state-of-the-art of the application of predictive analytics to admission control. We design and evaluate two intra-slice admission control algorithms namely, *the Decision Matrix algorithm* and *the Utility Index algorithm*. A real IP network dataset, containing network flows collected from the University of Cauca, Popayán network is used for the simulation and evaluation of these admission control algorithms. The proposed admission control algorithms presented various strengths, with the Utility Index algorithm being highly profitable to the operator and the Decision Matrix algorithm being suitable for traffic with a large proportion of high priority traffic.

A traffic forecasting model was implemented based on the Holt-Winters Exponential Smoothing predictive model. This forecasting model was trained using the network data from the real IP network dataset and then incorporated into the admission control process. For prediction-based admission control, the traffic forecasting model was used to forecast resource requirements of future network traffic in each slice and pre-emptively make provisions for the upcoming traffic.

The performance of the intra-slice admission control algorithms with and without the influence of traffic forecasting was analysed and it was found that the use of predictive analytics to predict future slice traffic allows for dynamic allocation of slice resources. Prediction-based admission control, when compared to the admission control without predictions, showed better performance in terms of probability of blocking, system utilisation, and profitability to the operator.

Table Of Contents

List Of Figures

List Of Tables

List Of Acronyms

Acronym	Definition
3GPP	3rd Generation Partnership Project
AC	Admission Control
AI	Artificial Intelligence
ANN	Artificial Neural Networks
ARIMA	Autoregressive Integrated Moving Average
ARMA	Autoregressive Moving Average
CDF	Cumulative Distribution Function
CPU	Central Processing Unit
CSV	Comma-Separated Values
DES	Double Exponential Sequencing
FIFO	First-In-First-Out
GBR	Guaranteed Bit Rate
GRG	Generalised Reduced Gradient
HWES	Holt-Winters Exponential Smoothing
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
MCPTT	Mission-critical Push-to-Talk
ML	Machine Learning
NFV	Network Function Virtualisation
NGMN	Next Generation Mobile Network
NRMSE	Normalised Root Mean Square Error
PDF	Probability Density Function
QCI	QoS Class Identifier
QoS	Quality of Service
RAM	Random Access Memories
RB	Resource Blocks
RMSE	Root Mean Square Error
RRM	Radio Resource Management
SDN	Software-Defined Networking
SLA	Service-Level Agreement
SlaaS	Slice as a Service
TCP	Transmission Control Protocol
V2X	Vehicle-to-Everything

Chapter 1: Introduction

1.1. Background

Since the introduction of mobile communication to the commercial world in the 1970s, the technologies behind mobile communications have evolved, and every major development has incorporated a response to an increase in the demand for connectivity. Cisco [1] predicts that the number of devices connected to IP networks (fixed broadband, Wi-Fi, and mobile networks included) will be up to 29.3 billion by 2023, with mobile devices accounting for almost half of that number (13.1 billion). Not only is the volume of traffic transmitted over networks increasing exponentially with respect to time, but an ever-increasing variety of services is to be accommodated on the network as per the demands of the industry verticals to be served by next generation networks such as 5G. This is outlined in the use cases and business models proposed by the Next Generation Mobile Network Alliance (NGMN) for the operation of 5G networks [2]. The concept of network slicing has been introduced as a solution which allows network operators to accommodate this diversity of demands on their networks [2]–[4].

Network slicing is the implementation of end-to-end logical sub-networks called slices which are tailored to support a particular connection type with specific network requirements [2]. These network slices are created and exist simultaneously with each other on the same physical network. Each slice is customised to serve a specific type of customer whose demands are different from those in the other slices. A customer or specific service which occupies a network slice is called a *tenant*. Since these network slices share the physical resources of the network, resource management strategies must be implemented to ensure that the network slices are served in the most optimal way possible.

Even though the number of connections to be accommodated by networks is increasing exponentially with respect to time, the network resources that serve them remain limited, and the industry must remain innovative in finding ways to accommodate an increasing number of users while maintaining a desirable quality of service. This leads to the introduction of the general concept of radio resource management (RRM) and specifically the function of admission control which will be further explored in this work. Admission control for sliced networks must go a step further to not only ensure that as many users are accommodated on the network as possible, but also to ensure that the objectives of network slicing such as slice isolation are respected.

The evolution of telecommunication networks from 1G right up to 5G networks has seen a shift in the network architecture from fully circuit-switched systems (1G, 2G) to a dual-

domain system with both circuit-switching and packet-switching (2.5G, 3G), and now to fully packet-switched networks with a unified packet core (4G, 5G) [5][6]. Being fully packet-switched networks, 4G and 5G networks are based on Internet Protocol (IP) communications. Voice and data calls on these IP-based networks are widely referred to as *flows* and this is evidenced by the use of the term 'flow' in the technical specification documents for 4G and 5G [2][3], as well as in the majority of peer-reviewed research published on the topic. A flow on an IP network is generally defined as a stream of packets moving from a given source IP address and port to a given destination IP address and port within a specific time interval, and the packets in this stream all have specific Quality of Service requirements and the same transport protocol [7][8]. All the packets in a flow carry the same header information. The 3GPP simply defines a packet flow as a collection of packets that flows between the user equipment and the network within a given time interval for a certain application. In a lot of the literature presented in Chapter 2, the terms 'call' and 'flow' are used interchangeably.

The term *Quality of Service* (QoS) refers to the quantification of the performance of a system as perceived by the users of that system. The QoS required of a telecommunications system is often defined by parameters such as network availability, accessibility, service integrity and more [9].

Radio resource management (RRM) is a control functionality in the network. RRM ensures efficient use of the available network resources. It does so while fulfilling Quality of Service (QoS) requirements. RRM employs functions such as admission control, resource allocation (packet scheduling), radio bearer control, mobility control, load balancing and inter-cell interference coordination [10].

Admission control (AC) is the function that admits or rejects users on the network. The admission decision is generally based on a number of factors, such as the QoS requirements of the incoming user, the QoS requirements of currently connected users, the availability of radio resources on the network for the requested connection and other constraints which may have been set by the network administrator [10].

In sliced networks, the concept of admission control is extended to include the process of determining whether requests from tenants for the creation of new slices should be accepted or denied [11] and this decision is also based on the availability of resources and whether or not the objectives of the incoming slice can be met while fulfilling those of ongoing slices. The fulfilment of Service-Level Agreements (SLA), or QoS guarantee is fundamental to 5G networks.

In sliced networks, admission control is implemented on two main levels: at the network level (inter-slice) where requests for the creation of a slice could be admitted or rejected, and at the slice level (intra-slice) where individual incoming calls (users) could be admitted or

rejected. These concepts and the supporting literature are explored more in Chapter 2. All the systems and approaches for admission control explored in Chapter 2 focused on admission control at either the intra-slice level or the inter-slice level. A system model that considers the need for network resource management at both the inter-slice and intra-slice levels is proposed in Chapter 4.

Admission control strategies and algorithms are not standardised and even though there are existing strategies [12], [13], the choice of strategy to implement is made at the discretion of the network and infrastructure providers. This has served as a cue for researchers to propose various admission control strategies for various network objectives. This dissertation focuses on admission control techniques as they pertain to sliced networks.

Network function virtualisation (NFV) and software-defined networking (SDN) are two key concepts that have been leveraged to achieve network slicing [14], [15]. NFV is the software implementation of network functions, ensuring that the network functions are programmable and scalable. SDN decouples the control plane from the data plane for a more centralised control point, allowing for easier network management, scalability, and a global network view.

The admission control schemes implemented on sliced networks must support the main principles of network slicing. The authors in [16] summarise the main principles on which the concept of network slicing is built as follows: automation, isolation, customisation, elasticity, programmability, end-to-end service, and hierarchical abstraction. For an admission control mechanism to support these principles, it must:

i. Have *automated admission request management*: since network slices can be requested by a third party, the admission control scheme must be able to immediately compute whether or not incoming requests can be accommodated by the current network and either admit or reject accordingly.
ii. Ensure *slice isolation*: radio resources must be allocated so that the activities in one slice do not affect services being delivered on another slice on the same network.
iii. Support *dynamic resource allocation*: the admission control scheme should support the scaling up and down of the radio resources allocated to slices. This is to fulfil the principles of elasticity and hierarchical abstraction mentioned in [16].

In addition to the characteristics outlined above for admission control schemes in sliced networks, Khodapanah et al. show in [17] that an admission control scheme for network slicing must also be:

iv. Slice-aware: The admission control scheme must be aware of the Service-Level Agreement (SLA) requirements of each slice, and the control parameters that are

needed to guarantee that the slice SLA is being fulfilled in all the slices. This addresses the issue of guaranteeing QoS.

The survey carried out by Ojijo and Falowo in [11] identifies four main objectives of slice admission control: optimise revenue, provide and maintain the agreed QoS, inter-slice congestion control, and ensure fairness in the admission of slices. These objectives cannot all be achieved equally and always. Trade-offs must be made, depending on which objectives are a priority to the network operator. Based on these objectives, the characteristics below can be added to the list of characteristics to be incorporated into admission control (AC) mechanisms for sliced networks. An AC mechanism for sliced networks should also:

v. *Incorporate congestion control*: a congestion control mechanism should be established to manage the requests which cannot be fulfilled in the current network state.
vi. *Assure slice fairness*: biases in the types of slices admitted to the network should be avoided by the admission control scheme, for the network to serve a variety of tenants, as is the point of network slicing.
vii. *Efficiently utilise the available resources*: the admission control mechanism should maximise resource efficiency and system utilisation, and this in turn can drive up revenue for the operator or infrastructure provider.

The goal of this body of work is to review the existing admission control mechanisms with respect to how they support the principles of network slicing as outlined above and to develop and implement a system that incorporates the characteristics of effective radio resource management in sliced mobile networks.

1.2. Problem Statement

Several resource management and admission control schemes have been proposed and implemented on cellular networks. These schemes have served the previous generations of networks well. Advancing on this, next generation networks (5G, 6G networks) employ network slicing to cater to the various business models and requirements of applications. Network slicing largely changes the way resources are allocated within a cellular network. Slice isolation must be ensured, and legacy admission control schemes were not designed for sliced networks.

Several admission control schemes have recently been proposed for network slicing as covered in Chapter 2. However, they have a few shortcomings such as enforcing the objectives of network slicing at the expense of individual user objectives. The works reviewed in Chapter 2 address admission control at either the intra-slice level or the inter-slice level and the intersection between inter-slice admission control and intra-slice admission control

was not explored by the related works reviewed in Chapter 2. To ensure that both slice-level and user-level objectives are met in the most efficient way possible, a dynamic hierarchical admission control method is needed. The hierarchical admission control method should allow for the implementation of admission control and resource allocation both amongst network slices within the network (inter-slice) and amongst users within the slice (intra-slice).

1.3. Research Questions

The research questions which were considered in this study are outlined as follows:

i. How can admission control methods be designed and evaluated for sliced networks and evaluated?
ii. How can a real network dataset be used for the simulation of admission control?
iii. How can a prediction model be developed and trained for the forecasting of network traffic?
iv. How does the integration of predictive analytics into the admission control and resource allocation process impact the performance of admission control algorithms?

1.4. Research Objectives

The objectives of this study are outlined as follows:

i. To design, implement, and compare the performances of admission control methods that decide whether to admit incoming flows within a given slice (intra-slice admission control).

ii. To identify a relevant dataset, preform data mining and data pre-processing on that dataset, such that it can be used for admission control simulations.

iii. To test the impact of using forecasted traffic information for each slice to dynamically adjust the resources allocated to future slices. To do this, the following sub-objectives are first addressed.

    - Develop and train a traffic forecasting model based on the Holt-Winters Exponential Smoothing method, using real network data from the dataset mentioned in ii. above.

    - Dynamically allocate resources to each slice, based on the traffic load forecasts produced by the traffic forecasting model.

- Evaluate the impact of this dynamic slice resource allocation on the performance of the admission control methods in i. above.

1.5. Research Methodology Overview

A literature review is first conducted so as to identify and critically review the related work that has already been done in the field. A system model is then designed and modelled using MATLAB, which is a powerful computational software tool. Two admission control algorithms (the Decision Matrix algorithm and the Utility Index algorithm) are designed and implemented on this system model. In addition, a First-In First-Out admission control algorithm is implemented and a performance comparison of all three admission control algorithms is presented. This evaluation identifies the strengths and weaknesses of all three admission control algorithms.

A traffic prediction model is developed based on the Holt-Winter Exponential Smoothing mathematical model. This traffic prediction model is trained using 4 possible approaches, outlined in Chapter 4, and the best-performing approach is used to train the final prediction model. This trained traffic prediction model is used to incorporate traffic forecasting into the admission control process, and then the performance of the admission control algorithms with the traffic forecasting is compared with their performance without the addition of traffic forecasting. This provides an assessment of the impact of the incorporation of traffic forecasting into the admission control process.

All performance evaluation in this work is based on quantitative analysis. A real network dataset is used in all the implementations and experiments in this work.

1.6. Main Contributions

This work investigates the impact of incorporating predictive techniques into the admission control process in sliced networks. The effect of using network traffic forecasting to make resource management decisions is observed both at the intra-slice and inter-slice levels. The main contributions of this work are outlined as follows:

i. The design and implementation of two admission control algorithms for admitting or rejecting users within a given slice (intra-slice admission). The admission decision in the proposed algorithms is based on policies designed for each algorithm and this is explored in Chapter 4. These proposed algorithms are tailored to and tested on a real network dataset.

ii. Implementation and training of a prediction model for network traffic forecasting.

iii. Design, implementation, and evaluation of a dynamic Slice Resource Allocation module which adjusts the resources allocated to ongoing slices, based on the forecasted traffic load of each of the ongoing slices.

iv. Benchmarking the performance of the proposed algorithms against the performance of a legacy admission control algorithm. In this work, the performance of the admission control algorithms with the addition of forecasting is also evaluated.

1.7. Scope and Limitations

The focus of this work is the implementation of admission control algorithms within a network slice and investigating the impact of predictive analytics on the performance of these admission control algorithms. Prediction-based admission control is implemented and compared with legacy admission control methods. Additional aspects such as user mobility were not considered in this work, as these aspects go beyond the scope of this work and can be considered in further studies or extensions of this work.

The resources assigned to a user, or to a flow is assumed to be used for both the uplink and downlink (bidirectional).

One limitation is that there is a lack of publicly available datasets for sliced networks so for this study, a campus IP network was used to simulate real life incoming traffic. The proposed algorithms in this work can however be reproduced on other traffic datasets.

These admission control methods were simulated and tested in MATLAB and not on network test beds, so even though real network data was used, the real-time performance of these methods is not presented.

1.8. Dissertation Outline

The remaining text is structured as follows:

Chapter 2 covers background on admission control approaches, the state-of-the-art of the domain of admission control for cellular networks, background on predictive techniques, state-of-the-art of the specific application of predictive techniques to the forecasting of network traffic and lastly, the state-of-the-art of the application of network traffic forecasting to admission control.

Chapter 3 presents and justifies the research methodology employed in this work. The real-world dataset that is used for the practical simulations is introduced here as well.

Chapter 4 takes the reader through the design of the system model, to the construction and training of the prediction model. The proposed algorithms are also presented here.

In Chapter 5, the simulation parameters are detailed, and the simulation results presented. The performance of the algorithms described in Chapter 4 is presented, and finally, the results of the simulation are discussed, and recommendations made.

Finally, Chapter 6 presents a summary of the work done, the limitations of the work and recommendations for a future extension of this work.

Chapter 2: Literature Review

In this chapter, we introduce the concept of admission control and the state-of-the-art of its applications in sliced networks. We also introduce the concepts and processes surrounding predictive analytics and its application in the case of network traffic forecasting.

2.1. Admission Control in Sliced Networks - Related Work

In legacy networks, admission control (AC) is a radio resource management function that determines whether or not a new user can be admitted to the network. However, in the context of sliced networks, the functionalities of admission control mechanisms have been extended to manage the admission of incoming slices.

In sliced networks, admission control is the function that not only determines which new users are to be admitted to the network [17], [18] but also decides which new slice requests are to be accepted or rejected [19], [20]. Rejection of slices may invoke some congestion control mechanism. The concept of congestion control shall be further explored in this section. AC decisions are mainly based on the available resources, the SLA of the current slices and users, and the resources required by the incoming slice or user to fulfil its SLA.

AC mechanisms are triggered by the arrival of requests for network access by new users or by the arrival of new slice requests. Before the bearer for the user request is set up, the admission control mechanism starts running. AC ensures that each slice can guarantee the SLA of its current users for the duration of their connection.

A variety of approaches to admission control in sliced networks, catering to different objectives has been proposed in the literature. The following sections explore the various existing categories.

2.1.1. Implementation Level

The work in [11] identifies two main implementation categories for admission control mechanisms based on the hierarchical level at which the admission control is implemented in the network, namely inter-slice admission control and intra-slice admission control.

Inter-slice admission control refers to admission control schemes that are invoked to manage incoming requests for new slices from tenants (customers). Inter-slice admission control also manages the admission control for the transfer of users between slices.

The works of Han et al. in [19], [20] propose an admission control mechanism that considers a network providing Slice as a Service (SlaaS), and makes decisions regarding the creation of

new slices. Here, the admission control module is considered a slice broker or mediator between the tenants (buyers of the slices) and the infrastructure provider selling the slices. Such an admission control model assumes that the tenant to whom an active slice is assigned will implement an admission control mechanism within the slice to manage the individual user requests for access to that slice.

A solely inter-slice admission control scheme would address only half of the problem if implemented in a network where the infrastructure provider is also a mobile network provider, creating slices to cater to its individual consumers. The slice-level objectives may be met but the individual user objectives have not been considered.

The second implementation level is the intra-slice admission control, where individual users send requests for network access to a tenant occupying an already existing slice. A decision is made within that slice as to whether a user is accepted or rejected from that slice. The decision is based on whether or not that user can be accommodated by the currently available resources in that slice and while maintaining the minimum required QoS of the active users.

Caballero et al. [18], Khodapanah et al. [17], Ali and Zarai [21] each propose admission schemes that are considered intra-slice approaches because these schemes are invoked for each incoming user request and a decision is made for each user. The approach in [18] considers a network model where admission control is implemented within the slice, hence the AC can be managed by the tenant and [17] considers a network where admission control is implemented in an orchestrator which manages a multitude of cells, hence the AC must be managed by the infrastructure provider. Since in [17], AC is managed by a network-level orchestrating entity, it is slice-aware and this scheme has more control over inter-slice influences, hence better traffic isolation between slices.

Like the inter-slice approach, a solely intra-slice admission control scheme is limited in that it does not consider the decisions to be made upon a new request for the creation of a slice. Infrastructure providers also operating as mobile virtual network operators need an approach that implements admission control at both the intra-slice and the inter-slice levels.

The strength of an admission control approach is best demonstrated during network congestion. There are a number of approaches for managing congestion control and the following section addresses some of them.

2.1.2. Admission Control with Congestion Control

Generally, the decision made by the slice admission control mechanism is binary; AC decides whether or not an incoming slice request is to be accepted or rejected. By default, if the necessary resources are available and the maximum possible QoS is being attained by all active users and slices, all incoming user and slice requests are granted on their respective

allowed networks. If the currently available resources are insufficient to accommodate the incoming slices, the network is faced with an issue of congestion and can invoke a congestion control mechanism to manage the surplus user or slice requests. The following congestion control approaches have been explored in the literature:

a. No Congestion Control (Blocking):

In the admission schemes in [8] and [14], no congestion control mechanism is implemented. The incoming user requests which cannot be accommodated are blocked. In these cases, users who were not initially granted network access would have to resubmit a request for network access later. This results in a higher probability of blocking in these schemes than it would in similar schemes implementing a congestion control mechanism such as queueing. The admission control mechanisms presented in [8] and [14] are both intra-slice admission control mechanisms.

b. Slice Degradation:

Han et al. [19] propose a slice degradation approach to congestion control where if an incoming slice request is met with insufficient resources, some resources are pulled from other slices with flexible (elastic) QoS requirements. This degradation limits the amount of isolation possible between slices, as congestion in one slice has an effect on other slices on the same network. A trade-off should be found between congestion control by slice degradation and the enforcement of slice isolation.

c. Call Degradation:

The scheme proposed by Ali and Zarai in [21], employs a call degradation congestion control mechanism. Here, some resources are pulled from tolerant calls (calls without strict QoS requirements) and allocated to incoming calls that would otherwise have been blocked due to insufficient resources. This is possible because when calls are first allocated their resources, the calls are given the maximum possible amount of resources for the best possible QoS. When need be, this QoS is degraded to the minimum allowed QoS levels for that call by taking some of the resources which were allocated to the call. This method of congestion control must be limited to intra-slice admission schemes in order to ensure isolation between slices. Even though it is not specified in the scheme in [21], for traffic isolation, resources should not be pulled from calls in other slices to solve congestion in one slice.

d. Queueing:

Queueing schemes can be used in conjunction with other congestion control techniques, such as in [19] where slice degradation is implemented and heterogeneous slice queueing is also used to manage congestion. The strategy in [21] also uses queueing alongside call

degradation. Here different queues represent different service classes, associated with different priorities on the network.

In [20], Han et al. propose a block-and-queue approach where the focus of the scheme is on efficient queue management and fairness in a bid to manage incoming slice requests without affecting currently active slices. This approach is tailored to networks with short-term slices and not long-standing tenants, as there is a limited amount of time a tenant can wait in a queue.

A summary of the admission schemes which have been reviewed in this section is shown in Table 1 below.

Table 1 Summary of Admission Control Schemes Considered in the Literature

Authors	Admission control goals	Trade-offs	Performance Metrics	Inter-slice Resource Allocation	Network Hierarchy level	Congestion Control	Queueing Approach	Shortcomings
Caballero, et al. [18], 2018	Guarantee SLA of current slices, considering tenants with strict QoS requirements	Maximising isolation vs maximising efficiency	Network utilisation, blocking probability, throughput gains	Predefined network shares (cap/threshold) per slice with dynamic distribution.	Intra-slice	Reject and block new users	No queueing	Queuing was not considered; traffic isolation is not ensured as overloading on one slice can cause users in other slices to be dropped.
Han et al. [19], 2018	Guarantee SLA of current slices, Account for inconsistencies in client slice requests	Performance objectives of each slice are capped so as to maintain balance throughout the network.	Slice servicing rate (arrival vs departure), slice dropping probability, network utilisation	Resource sharing between slice instants	Inter-slice	Slice degradation (resources pulled from elastic slices)	Multi-queue, heterogeneous queues (slice type)	No admission control for incoming end-users
Khodapanah et al. [17], 2019	Guarantee slice SLA, ensure isolation	Isolation issues: Some inter-slice activity affects the lower priority slices, but they benefit from multiplexing gains	Average throughput, number of admitted users (admission rate).	Predefined network shares (thresholds) per slice.	Intra-slice	Reject and block new users	No queuing	Queueing is not considered.
Ali, Zarai [21], 2019	Reduced call blocking probability	Isolation vs resource utilisation	Packet loss ratio (due to dropped connections), call blocking probability, resource utilisation	Resource reservation for each QoS class with dynamic thresholds.	Intra-slice	Call degradation (resources pulled from tolerant calls)	Multi-queue, heterogeneous queues (service class)	Inter-slice effects are not minimised (isolation is not guaranteed). Resource reservation is for QoS classes, not slices.
Han et al. [20], 2019	Guarantee fairness, account for impatient tenants	Focuses on queue management and incoming slice requests from queues.	Slice inter-acceptance time	Complete sharing amongst slices (resources are assigned for informing slices from a resource pool)	Inter-slice	Block and queue	Multi-queue, heterogeneous queues (slice type), first-come-first-served, random delays	No specific measures in place to ensure traffic isolation between slices. Tailored to slices with short-term duration.

The admission control schemes we have explored thus far, in conjunction with their respective congestion control measures, are reactive. This means that they attempt to resolve the issues as they are occurring. In contrast, preventive AC should anticipate the problems and make provisions to address the problems beforehand. As suggested in [22], with the incorporation of predictive analytics and traffic forecasting into network functions such as admission control and congestion control schemes, higher QoS performance would be possible because congestion would be avoided, rather than resolved.

2.2. Introduction to Predictive Analytics

Predictive analytics refers to a range of statistical and analytical techniques used to develop models that can predict future events or behaviours [23]. In predictive analytics, historical data is analysed, and relational information such as trends and patterns are identified and extracted from the data. This relational information is used to develop predictive models which can then use statistical techniques to make predictions about future events, based on historical data.

In a broad sense, the process of predictive analytics starts with analysing the current and historical data for trends, patterns and relationships [23][24]. This is also known as *data mining* or *feature extraction.* This is considered the first step of predictive analytics, under the assumption that the data has already been collected, cleansed, and stored in some sort of data warehouse. There are various techniques for the collection and cleaning of data. Data cleaning (also called data cleansing, or data scrubbing) is key to reducing errors, removing redundancies like repeated entries, and improving overall data quality [25]. Data cleaning also prepares data into a format that can be read by an analytical tool [24]. However, the issue of data collection and data cleansing is beyond the scope of this dissertation.

After the data is mined, the data is then divided in to 2 portions: a *training dataset* and a *testing dataset*. The training dataset is used to develop a predictive model based on statistical tools, which will be discussed in the subsequent sections of this chapter. Depending on the intended application of the predictive model, a statistical tool is chosen for the predictive model and an algorithm which emulates the chosen statistical tool is chosen. The statistical tool on which a predictive model is based can be a regression technique, a machine learning technique, a combination of both in a hybrid model as can be seen in [26].

The model is then trained and tested in an iterative manner. The model is trained and retrained such that with each iteration, the model is more robust than it was in the previous iteration [23]. Training the model involves running the chosen algorithm against the training dataset, which as mentioned above, is part of the relational information that was mined from the data. For example, if data had been collected over the course of 1 year, the training data could comprise of the data that was collected within the first, say 9 months. An optimal

number of iterations or performance level must be determined so that the training is stopped once the model attains that performance [27]. This is important to avoid 'overtraining' the model. An overtrained model would yield near perfect results on its training and testing datasets but would not be able to make accurate predictions on other more general datasets.

The testing dataset is the portion of data that was not included in the training data. In the example cited above, the testing dataset would be the data collected in the last 3 months of the 1 year of data collection. After training, the model is then tested by redacting the variable to be predicted (the *dependent variable*) from the testing dataset and then running predictions on the incomplete testing dataset to see how close the predictions come to the redacted information. Model testing, also known as *model validation*, is key to verifying the accuracy of a model. The accuracy of the prediction obtained from the predictive model is measured by the error, which is the difference between the predicted and actual values. Since the actual values in the testing dataset are already known and were redacted by the model developer, the accuracy of the predictive model can easily be verified. The validity of a predictive model depends on the size and quality of the dataset available for its development.

In the context of machine learning, this method of training and testing a model where the training data contains the inputs as well as the expected output is called *supervised learning*. All the predictive techniques covered in subsequent sections are trained via supervised learning.

When the predictive model has been validated and is sufficiently accurate, it goes into the deployment phase where it is applied to the real processes for which it was built. Predictive models, like most systems, must be maintained after deployment to ensure and improve performance and efficiency. An example of the type of maintenance that goes into predictive models is the updating of significant prediction factors which may have changed since deployment, such as geographical locations, population numbers, among other examples.

Most predictive models generate a probability variable (also known as a *score*) which indicates the likelihood of the occurrence of a given event. In a nutshell, predictive analytics uses past events to anticipate future events. A summary of the processes explained above with respect to the development of a predictive model is shown in Figure 1.

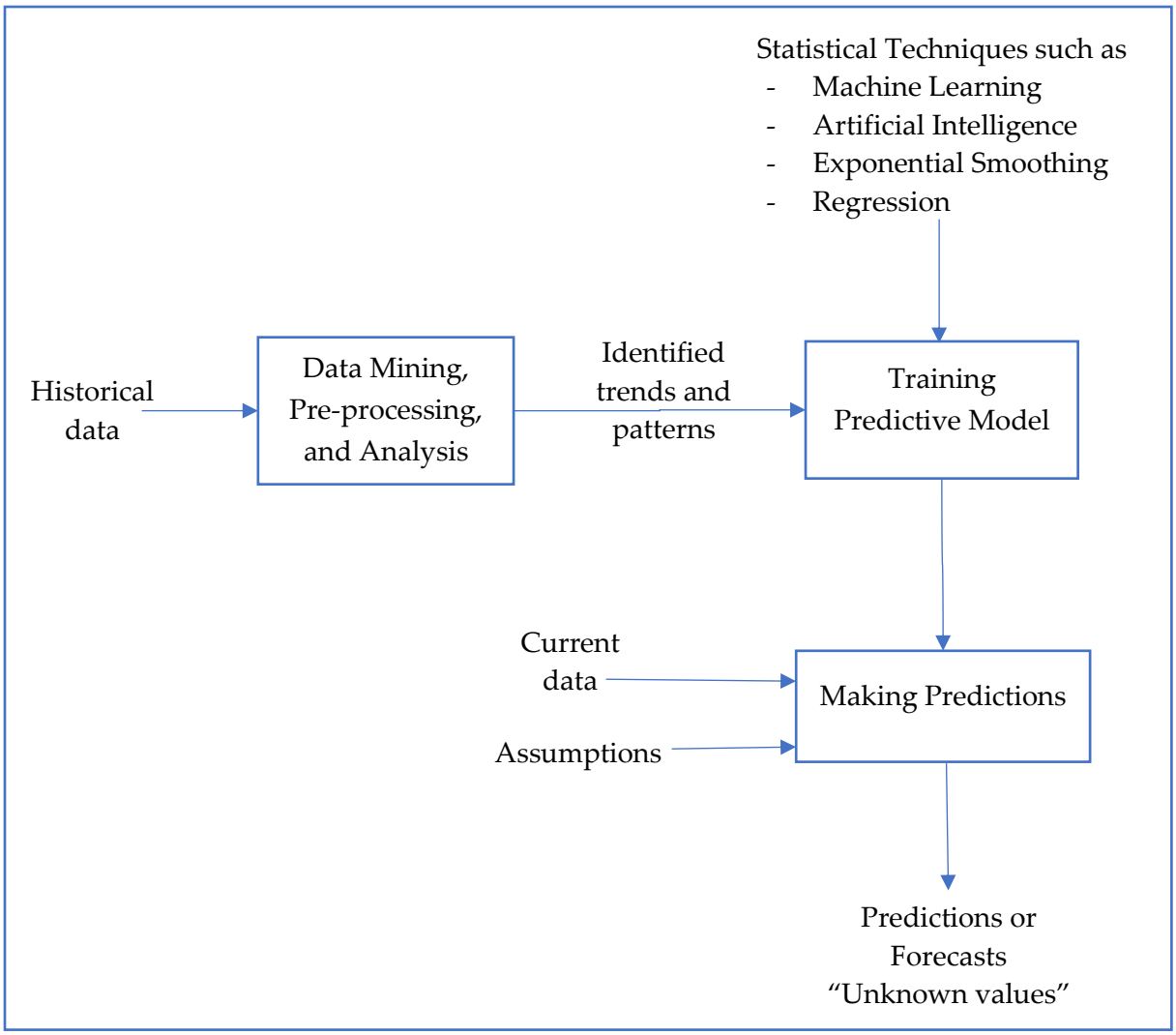

Figure 1 Summary of Flow of Processes in Predictive Analytics

There are some predictive models called non-training models that have been proposed for cases where there is not enough data available to constitute a training data set. In these non-training models, the model makes a prediction based on the last few observed values [28]. An example of such a model is the Linear Minimum Mean Square Error. Non-training models generally do not achieve as high a performance as the trained models and will not be considered in this dissertation.

While discussing the topic of Predictive Analytics, it is important to highlight the difference between the terms, 'prediction' and 'forecasting'. Prediction, in predictive analytics, refers to the estimation of unknown values of a variable, based on historical data, possibly as a response to changes in other factors that might affect the unknown variable, while forecasting is the branch of predictive analytics that focuses on estimating the value of a variable at a

given time in the future, based on mathematical models created from historical data [29]. Since the focus of forecasting is the temporal feature of data, the mathematical models used are usually time-series models. Simply put, forecasting answers the question, "Based on how A has behaved in the past, how would A behave at this specific time in the future?" while prediction answers the broader question, "If B and C change, how will that affect A?"

2.2.1. Statistical Tools for Predictive Modelling

The tools used to develop predictive models are founded on statistical methods and these methods or tools have been broadly categorised in the literature into regression techniques and machine learning techniques [23] .

i) Regression Techniques

Regression modelling describes the relationship between the variable to be predicted (the dependent variable) and the known variables (also known as independent variables, predictors, or features) [23]. Regression models generally represent the dependent variable as a function of the independent variables and their coefficients with an added error value. For regression modelling, the relationship (i.e., the afore mentioned function) to be modelled must be specified during the model development such that it fits the training data as closely as possible. As outlined in Chapter 6 of [30], linear regression models generally have the following mathematical representation:

$$Y_i = f(X_i, \beta) + e_i \qquad (2.1)$$

Where Y_i is the dependent variable, X_i represents the independent variables, β represents the coefficients of the independent variables, also called *regression coefficients*, and e_i represent the error terms (also called the *residuals*).

During modelling, based on the training data, the function f is chosen such that it best approximates the relationship between Y_i and X_i. β is calculated using estimation methods to ensure that the e_i error term is as small as possible.

Some relevant examples of regression techniques are explained below.

a. Linear Regression

In the simplest linear regression model, the function f in equation 2.1 is a straight line and the error component has a distribution with zero mean and unknown variance. Here, the dependent variable, Y, has just one predictor, X and their relationship is represented by a straight line shown in equation 2.2 [31]:

$$Y = \beta_0 + \beta_1 X + e \qquad (2.2)$$

The regression coefficients are unknown and are usually estimated using least squares estimation.

b. Time Series Regression

In some cases, relevant data can only be observed over time, and not as a function of another variable, as in the example of linear regression above. When the data is recorded as observations of a given variable (e.g., the performance of a unit, revenue, or the demand for an item in a market) over fixed, equally-spaced time intervals (e.g., hourly, daily, monthly, quarterly observations), this data is referred to as a time series [32] and time series regression analysis is an appropriate approach to analysing such data in conjunction with data from other relevant predictors so as to produce models for predictions.

Basically, time series regression is a method of predicting future values of a variable based on the history of that variable as well as the impact of relevant predictors. Time series regressions account for dynamic factors in the data such as seasonal variations and because of the temporal nature of the data, time series regression can be used to predict the behaviour (response) of the given parameter at some specific point in the future (forecasting).

Usually in time series regression, only one variable is considered, and the predictions are based on past observations of that variable as expressed in equation 2.4.

$$Y_{T+1} = f(Y_1, Y_2, \dots, Y_T) \qquad (2.4)$$

Where Y_{T+1} is the prediction of the variable Y at a future time T+1 and $(Y_1, Y_2, \dots, Y_T)$ are observations of the variable Y in the time series (1, 2, …, T).

In time series regression, a 'design' matrix containing the past (and current) observations of the predictors is typically built and the entries of this matrix are organised in chronological order of occurrence. An estimation method is then applied to the design matrix in order to calculate the regressor coefficients.

Various linear methods have been developed for time series regression and the most popular examples include the Autoregressive Moving Average (ARMA), Autoregressive Integrated Moving Average (ARIMA) used for non-stationary time series, Double Exponential Sequencing (DES), and the Holt-Winters Exponential Smoothing (HWES) algorithm.

We present the case study of the Holt-Winters Exponential Smoothing algorithm, and this algorithm is a fundamental feature of the prediction model implemented in this work.

Holt-Winters Exponential Smoothing (HWES)

The Holt-Winters Exponential Smoothing model is chosen as a case study because it is a favourable model for the prediction of network traffic. Network traffic tends to be periodic

and seasonal and the HWES model is known for its suitability for data that has components with level, trend, and seasonality. Level refers to the magnitude of a measurement in a set of measurements. Trend refers to the general slope of the overall graph, irrespective of peaks and troughs. Seasonality refers to how often the patterns in the data are repeated. HWES forecasting is also known as *Triple Exponential Smoothing* due to the fact that it is based on these 3 parameters. Like the regression techniques introduced above, level, trend and seasonality each have a smoothing equation shown by equations 2.5, 2.6 and 2.7 respectively.

The Holt-Winters ES model has two main modes of application: additive seasonality and multiplicative seasonality. The choice of the mode to use depends on the nature of the data to be modelled. The multiplicative variety is used when the previous value within a season is a factor of the future value. The additive variety is used when the behaviour of the seasons is linear.

Because the smoothing factors are basically a ratio of how much weight should be placed on current values versus previous values, the smaller the smoothing factor, the smaller the weight or impact of the current value on the forecast and the larger the weight of the previous values.

The forecasted values are given by equation 2.8.

$$L_t = \alpha * (Y_t - S_{t-M}) + (1 - \alpha) * (L_{t-1} + b_{t-1}) \quad (2.5)$$

$$b_t = \beta * (L_t - L_{t-1}) + (1 - \beta) * (b_{t-1}) \quad (2.6)$$

$$S_t = \gamma * (Y_t - L_t) + (1 - \gamma) * S_{t-M} \quad (2.7)$$

$$F_{t+k} = L_t + k * b_t + S_{t+k-M} \quad (2.8)$$

Where Y_t = actual value at time t

L_t = current level

L_{t-1} = previous level

b_t = current trend

b_{t-1} = previous trend

S_t = current seasonal value

S_{t-M} = corresponding seasonal value in previous season.

M = duration of a season (number of observation points in a season) / order of seasonality

k = number of datapoints into the future to forecast

F_{t+k} = forecast of k steps ahead

Values must be chosen for the smoothing constants, and initial values for L_t, b_t and S_t must be calculated [33] and then the model will be extended from these initial values.

The initial value for L_t is denoted as L_0, and one way to calculate L_0 is to average the values for Y_t over the first season as done in [34] and as shown in equation 2.9. Equation 2.10 is used to obtain the initial value for the trend (b_0).

$$L_0 = \frac{1}{M}\sum_{t=1}^{M} Y_t \qquad (2.9) \qquad\qquad b_0 = \frac{1}{M}\sum_{t=1}^{M} \frac{Y_{t+M} - Y_t}{M} \qquad (2.10)$$

ii) Machine Learning Techniques

Machine learning (ML) is an application of artificial intelligence (AI) which allows algorithms to improve by themselves 'through experience' and without explicit programming [35], and it is an advanced approach compared to the regression techniques presented in section i) above. Machine learning approaches are said to be an appropriate solution for situations where no mathematical model sufficiently represents the system [36].

Machine learning can be regarded as a tool used to implement the regression techniques explored in section i) above and the implementation of time series forecasting is commonly done using machine learning.

The application of machine learning in predictive analytics follows the same predictive analytics steps outlined in the introduction to section 2.2.; machine learning facilitates the creation and training of a model using training data such that it can process new data to make predictions.

In machine learning, there is no one-size-fits-all algorithm, and each application needs an algorithm which has been tailored to its data structure and domain.

The most popular machine learning techniques are artificial neural networks and decision trees, and they are described as follows:

a. Artificial Neural Networks

Artificial neural networks (ANN) were inspired by the naturally occurring neural networks naturally in animal brains. In ANN, the neural network is a collection of interconnected nodes (representative of the neurons in the brain) where each node is a function (*transfer function*) which receives input from the node(s) of the preceding layer and passes its output to node(s) of the next layer [37]. The inputs and outputs are passed between the nodes via the connections, each of which bears a learned weight (learned during the training phase) and

the cumulation of the nodes, the connections and their respective weights determines the overall behaviour of the ANN.

When designing an ANN, important issues to consider include the number of *layers* of neurons (nodes) to use, the transfer function of the neuron and the connections between the neurons. The first layer of neurons is referred to as the *input layer*, the middle layers are called the *hidden layers* and the last layer is called the *output layer*. The layers of artificial neural networks are depicted in Figure 2.

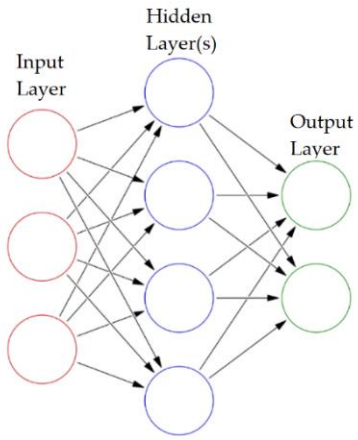

Figure 2 Neural Network Layers

b. Decision Trees

In decision trees, observations are made about a variable and from these observations, conclusions are drawn about target values of that variable. Decision trees are ideal for classification and decision-making problems.

2.3. Traffic Forecasting - Related Work

2.3.1. Traffic Forecasting Using Predictive Analytics

Over the years, the field of predictive analytics has evolved, and one of the several areas of application of predictive techniques is the forecasting of network traffic. Network traffic has been found to be fairly predictable [37], even though the authors in [28] pointed out that based on the literature, prediction models for network traffic should be adaptive so as to accommodate the fact that network traffic presents temporal and spatial variations. In addition to the variations across time and space in network traffic, in recent years, there has been an explosion in communication network traffic, in terms of the number of users accessing the networks and in terms of the number of new services being offered by the

network. This has resulted in a significant change in the traffic patterns which were first observed on communication networks when network traffic forecasting was first being proposed. The literature shows that there is no one-size-fits-all solution in the application of predictive techniques to network forecasting.

The choice of predictive model to use in various communication systems has been widely governed by the prediction accuracy, as shown by the authors in [28], [38], [37], [39], [40]. The prediction accuracy is generally a function of the mean square error of the predictions. An example of such a function is the root mean square error (RMSE) as employed by the authors in [28], [38], and the normalised mean square error (NRMSE) as used by the authors in [37]. The terms RMSE and NRMSE are further defined in Chapter 4, section 4.3.2. A desirable prediction error would be <10% as per Wittig [27] and several researchers, but [27] points out that prediction accuracy should not be the sole determinant of how effective a prediction model is, but ease of use and interpretability should be considered as well.

Other performance metrics have been used in addition to the prediction accuracy to further inform the choice of predictor to use. Iqbal et al. in [41] focused on identifying an accurate predictor with minimal computational complexity and minimal power requirements. Their work showed that artificial neural network (ANN) predictors are highly accurate but also very computation-intensive. This meant that an optimal trade-off point was to be found between the cost (in terms of complexity and power requirements) and the accuracy of the predictor.

Predictors based on time-series models are a popular choice amongst researchers. In some of the earlier works considered in this dissertation (2007 – 2012), network traffic was broadly categorised into voice (circuit-switched calls) and data calls (packet-switched calls) as seen in [37], [38]. Tikunov and Nishimura in [38] applied Holt-Winters Exponential Smoothing to the prediction of circuit-switched cellular network traffic. The network traffic under consideration in [38] is seasonal over spans of 24 hours, as well as over spans of 7 days. However, 4G networks and beyond favour a unified packet system (Evolved Packet System [5]) where there is no longer a circuit-switched route and models must adapt to these new traffic types.

Despite the popularity of the time-series based predictors, some researchers found machine-learning based predictors to be a better fit for certain network applications than their time-series counterparts. Zhani, Elbiaze and Kamoun in [28] investigated and compared two time-series models, ARMA (Autoregressive Moving Average) and ARIMA (Autoregressive Integrated Moving Average) with the hybrid neurofuzzy model [42]. The comparison was done with respect to their application in the prediction of Internet traffic. The hybrid neurofuzzy model, which is based on neural networks, performed best in terms of prediction

accuracy but the traffic structure used in their study does not necessarily reflect today's sliced networks and the fact that today's traffic model has two hierarchies: the arrival of slice requests and the arrival of individual users trying to access these slices.

Existing network traffic prediction or traffic forecasting models have been analysed in the literature and proposed for various types of networks and with various objectives such as improved power efficiency, improved dynamic scalability, lower computation complexity, and improved prediction accuracy.

When dealing with time-series data, it is important to consider the *dependence range* of the data. The dependence range of a process describes how dependent the occurrence of point A is on the occurrence of point B, based on the time or spatial distance between the two points. This dependence between 2 points in a process is statistical dependence and is usually measured with the auto-covariance function. Most network traffic is presented and analysed as time series data, and for a time series model, if a process is 'short-range dependent', this simply means that the statistical dependence decreases rapidly as the time delay between them increases [43]. This is also referred to as short-memory processes. Nie et al. in [40] identified that because of the aspect of random access in networks, modelling short-range dependence is an important stage in network traffic prediction for the purpose of capacity planning and admission control. This is reflected in the work done in the field of network traffic forecasting by the authors in [28], [38], [22], and [23].

A network's ability to forecast the traffic behaviour (trends and patterns) is a contributing factor to how efficiently the available network resources are allocated and used, and how the quality of service can be maximised.

Traffic forecasting has found application in several areas of network management [28] such as network security, network resource allocation, congestion control and capacity planning. The subsequent sections and chapters of this dissertation will focus on the application of network traffic forecasting to admission control in the context of sliced networks.

2.3.2. Predictive Analytics and Admission Control

Several studies [37] have shown that predictive analytics and network traffic prediction provide proactive approaches to resource management in communication networks with application in areas like admission control, congestion control and resource allocation. In adopting such proactive approaches, issues on the network can be anticipated and prepared for beforehand (preventive measures), rather than managing issues during or after their occurrence (curative measures). Resources can be utilised more efficiently if the future traffic can be predicted accurately. The authors in [36] outline the current and potential applications of machine learning in communication networks, citing resource management and interference mitigation as examples of these applications.

Predictive analytics, when applied to admission control, can help admission control algorithms to reduce blocked calls and improve overall QoS. Predictive analytics can use historical data to predict how long a user or tenant is likely to stay on the network. Based on this prediction, measures can be taken to free up network resources for incoming users.

The capacity planning strategy proposed in [38] intuitively compares predicted traffic levels with a pre-determined threshold in the network which defines a particular blocking rate. If predicted traffic levels are above that threshold, this indicates a higher potential for blocking and the network is warned about potential future overload and the cells are dynamically allotted more resources to accommodate the potential influx of traffic.

The work of Buyakar et al. in [45] explored inter-slice admission control and resource allocation using traffic forecasting to ensure certain QoS. They opted to do their predictions using neural networks.

Owing to the introduction of the network slicing architecture [2], [4] in 5G networks and beyond, some works introduce a slice broker entity [46], [47] into the 3GPP network management architecture and the broad function of this slice broker is to allocate physical resources on the network to incoming slice requests based on their SLA requirements.

Sciancalepore et al. [44] present a slice management scheme for sliced networks where they applied predictive analytics to the prediction of slice traffic, improved admission control and scheduling, and optimal system utilisation. They proposed a modification of the above-mentioned network slice broker, first proposed in [46], into a three-module system. This three-module system would comprise of the following:

i. a *forecasting unit* [44] [45] which would predict the future slice traffic load based on the observations from that same network,

ii. an *admission control module* [44] [45] which would receive the predictions from the forecasting module and make a decision on the incoming slice requests, and lastly,

iii. a *slice scheduler* [44] which would immediately schedule these requests, as well as provide a feedback path to the forecasting module. This feedback to the forecasting module is the data such as the status of SLA, and the forecasting error. This information can then be used to adjust the forecasting model and improve future predictions.

Buyakar et al. [45] also opted for a similar modular system with the 3 main modules being the forecasting unit, the admission control unit and a resource allocation unit. The resource allocation unit in [45] feeds back the actual resources used by each request.

In both cases studies, the feedback path allows for the actual values to be compared to the predicted value and the difference is used to improve the forecasting model.

Also worthy of note is that Sciancalepore et al. [44] perform traffic predictions on the expected traffic per tenant while Buyakar et al. [45] perform traffic predictions per slice. Given the definition of a network slice as an end-to-end isolated logical network, and the replicability of network slices as opposed to the replicability of tenants, Buyakar's decision to predict traffic on a slice-by-slice basis appears to be more feasible on larger scale networks.

In their implementations, Sciancalepore [44] and Buyakar [45] both employed an observation period during which incoming requests were gathered to be processed at the end of the observation period by the admission control unit.

Sciancalepore et al. [44] opted for the additive Holt-Winters predictive model for the traffic forecasting. The main goal of this slice management scheme is to maximise system utilisation and to find the best forecasting configuration for this application. The work done by Sciancalepore et al. [44] and Buyakar et al. [45] are the main case studies and their system architecture will be extended and tested at the intra-slice level in the subsequent chapters of this dissertation.

2.4. Chapter Summary

The literature review done in this chapter shows that a trade-off must be found between the implementation of slice congestion control and the enforcement of true slice isolation. Slice isolation is a fundamental feature of network slicing, and this trade-off is necessary because slice congestion control solutions tend to limit slice isolation and on the other hand, the enforcement of true slice isolation prevents the network from effectively responding to congestion. With the prediction tools explored in this chapter, one could design a system in which the incoming traffic for each slice would be predicted at set intervals and the slice resource allocations pre-emptively made to accommodate the predicted incoming traffic. This would eliminate the need for impromptu slice degradation because slice resources would only be reviewed at pre-determined intervals, maintaining true slice isolation for set periods of time. This approach is explored further in subsequent chapters.

Chapter 3: Methodology

In this chapter, we reiterate our research objectives and then we present the hardware and software tools that were used to achieve the results in this work. This chapter also specifies which prediction model is used in this work and we present the performance metrics by which the proposed algorithms will be evaluated. Lastly, the dataset used in this work is presented and described in this chapter.

3.1. Research Objective

As seen in Chapter 2, there is not a lot of literature exploring the application of predictive analytics to admission control, specifically in the context of sliced networks. The aim of our work is reiterated as follows:

i. To design, implement, and compare the performances of admission control methods that decide whether to admit incoming flows within a given slice (intra-slice admission control).

ii. To identify a relevant dataset, preform data mining and data pre-processing on that dataset, such that it can be used for admission control simulations.

iii. To test the impact of using forecasted traffic information for each slice to dynamically adjust the resources allocated to future slices. To do this, the following sub-objectives are first achieved.

   - Develop and train a traffic forecasting model based on the Holt-Winters Exponential Smoothing method, using real network data from the dataset mentioned in ii. above.
   - Dynamically allocate resources to each slice, based on the traffic load forecasts produced by the traffic forecasting model.
   - Evaluate the impact of this dynamic slice resource allocation on the performance of the admission control methods in i. above.

3.2. Tools, Models and Methodical Approach

To achieve the research objectives, a system model was designed, and algorithms for admission control were developed and implemented in MATLAB® running on an Intel® Core™ i7-8550U CPU @ 1.80 GHz, 4 Core(s), and 8 GB RAM. MATLAB is a powerful computational tool with which both simulations and quantitative evaluations can be performed. The ideal data structure for the representation of multi-faceted network traffic is the array or matrix because each row or column can be used to carry characteristic information about that traffic. MATLAB was built for array (or matrix) manipulations and so is a justified choice for this research work. The predictive model was also successfully implemented in the MATLAB computing environment.

The predictive model used here is the additive Holt-Winters exponential smoothing (HWES). As seen in section 2.2.1., this model is good for predicting data that has a seasonal component, such as network traffic. The performance of predictive models and estimators is commonly quantified using an error term, as discussed in section 2.2. In this dissertation, the performance of the traffic forecasting model was evaluated by calculating both the root mean square error (RMSE) and the normalised root mean square error (NRMSE). These terms RMSE and NRMSE have been mathematically defined in Chapter 4, section 4.3.2.

A real network traffic dataset containing traffic information from a campus network was used to test the admission control methods. A selection of network traffic data was extracted from the dataset and split into two portions: one for training the predictive model, and the other for testing the predictive model. The use of a real network traffic dataset allowed for a near representation of the behaviour of a real packet data network employing the proposed admission control methods.

3.3. Quantitative Analysis and Evaluation

Admission control mechanisms are generally assessed by calculating performance metrics such as the probability of blocking (PoB), the system utilisation, and in some cases, the profitability of the given admission control mechanisms. In this work, admission control methods were developed, implemented, and compared based on these same metrics. This choice allows for a common-to-industry understanding of the performance of the methods which were developed and implemented in this work. In addition, the performance of the admission control algorithms with the implementation of forecasting was benchmarked against the performance of the same admission control algorithms without forecasting, and all this is presented in Chapter 5.

3.4. About the Dataset

The dataset used in this study contains approximately 2,000,000 sample headers of real network flows, observed over the span of 3 months (April 2019 to June 2019) on the University of Cauca, Popayan, Columbia network [48]. This dataset has been used by researchers in the domain of network traffic management. One example is the work done by Rojas et al. [49], where the dataset was used for network traffic flow measurement and analysis. Another dataset that was considered for this work was the 5G Campus Networks: Measurement Traces [50] but that dataset was too large to be managed with the resources available to this work. After studying the University of Cauca dataset [48], it was found to contain flow information that would be useful for the application in this dissertation.

The dataset was originally available in CSV format, and as part of the data preparation process, a sample of the data was extracted and pre-processed using Microsoft Excel. The pre-processing involved data cleaning by removing flows which have a duration of zero seconds, extension of the dataset with relevant data fields and storing it in a spreadsheet format. The flow data were then visually represented, and the seasonality of the data was identified, and this is presented in Chapter 4, section 4.1.3.

The term 'the dataset' as used in this chapter and in the subsequent chapters refers to the specific dataset [48] used in this work.

3.5. Chapter Summary

Three main research objectives were presented in this chapter and to achieve these objectives, the MATLAB® software tool is used in this work to run simulations on a computer. The additive Holt-Winters exponential smoothing model is the chosen prediction model for this work. Variations of the root mean square error are used in this work to evaluate the performance of the chosen traffic forecasting model. The admission control algorithms are evaluated using three performance metrics namely, probability of blocking, system utilisation and profitability. A publicly-available network traffic dataset from the University of Cauca is used for the simulation of network traffic.

Chapter 4: Prediction Model Development, System Design and Admission Control Algorithms

This chapter covers the preparation of the dataset for exploitation, the development and training of the prediction model, the system model, and the design of the admission control algorithms.

4.1. Data Mining and Pre-processing

The dataset used in this work [48] contains a large sample of bidirectional flow headers with related information such as the flow identifier, the start and stop times of each flow, the packet sizes in each flow, the duration of each flow and the category of service of each flow. The category of service in turn tells us the QoS Class Identifier (QoS) and the Quality of Service (QoS) requirements of the flow. Within the dataset, each row represents an individual flow header, and each column represents the various data fields of the flow headers. Each data field contains specific information about the flow. A complete description of the relevant data fields used in this work from the dataset in [48] is included in Appendix I. The description of the field 'octetTotalCount' in the dataset [48] indicates that the flow information is that of the payload only, so we can conclude that no control data (signalling data) is included in this dataset.

The exercise of data mining and pre-processing for this work mainly involved cleaning, classifying, and extending the data. This was done in 4 main steps as detailed in sections 4.1.1. and 4.1.2. below.

4.1.1. Cleaning the Data

Given that the dataset [48] is a raw dataset, the data must be cleaned before processing through a forecasting model. This is to ensure that the information in the dataset is meaningful and can be exploited by the relevant applications.

Step 1: Irrelevant data was removed. For this specific application, irrelevant data includes the following:

- Flows with 0 (zero) seconds duration. This 0-second duration is an indication that no traffic is carried during the flow. Such flows might be an indication of dropped, blocked, or pre-maturely ended flows. The processing of flows with duration of 0 seconds does not add value to the admission control algorithms being implemented in this work. Since admission control decisions are made based on the characteristics

of the incoming payload, a more accurate analysis of our system and its own performance would require purely information on traffic transmitted on the network.

- Flow information which does not factor into the admission control decisions of our algorithm was removed, such as information on the packet interarrival times, transport protocol numbers, and port numbers. This was a necessary step to avoid overloading our processor with unnecessary data when processing the dataset.

Step 2: Some parameters (flow duration and bandwidth per flow) were brought down to a suitable scale for simulation purposes. This is achieved by normalising the data. The process of normalising data allows data to be rescaled on a scale from 0 to 1 without loss of its intrinsic and relative information.

Given a set of values, an entry X in that set would be normalised using the standard formula shown in equation 4.1. Normalisation as shown in equation 4.1 reduces the scale of the set of values from its original range [$Xmax\ \ Xmin$] to [0 1].

$$Normalized\ X\ = \frac{X - Xmin}{Xmax - Xmin} \tag{4.1}$$

Where X = any entry to be normalised,

$Xmax$ = maximum value for X in the range of values of X,

$Xmin$ = minimum value for X in the range of values of X,

and $0 \leq Normalized\ X \leq 1$.

Based on equation 4.1, the flow duration and bandwidth per flow were normalised as follows:

- The original flow duration which was provided in the dataset [48] was normalised to a 120 time units (t.u.) scale using equations 4.2 and 4.3.

$$normalised\ flow\ duration = \frac{original\ duration - mimimum\ duration}{maximum\ duration - minimum\ duration} \tag{4.2}$$

$$flow\ duration\ on\ 120\ scale\ =\ normalised\ duration\ \times\ 120 \tag{4.3}$$

- The average bandwidth required per flow was normalised to a scale of 50 bandwidth units (b.u.) using equations 4.4 and 4.5.

$$normalised\ bandwidth = \frac{original\ bandwidth - mimimum\ bandwidth}{maximum\ bandwidth - minimum\ bandwidth} \tag{4.4}$$

$$bandwidth\ on\ 50\ scale\ =\ normalised\ bandwidth\ \times\ 50 \tag{4.5}$$

4.1.2. Classification and Extension of the Dataset

According to the 3GPP technical specification [51], each flow is assigned one QoS Class Identifier (QCI). The QCI specifies some of the QoS requirements of certain services and applications and this enables the operator to make an informed decision on the resource management policies to be applied to each flow. Organisation of incoming traffic into traffic classes is also done based on the QCI of each traffic flow. The dataset [48] did not contain QCI values but it did contain the flow categories which enabled us to determine the appropriate QCI per flow in Step 3.

Step 3: Classification and extension of the dataset was done by inferring and adding relevant parameters to the dataset.

- QoS Class Identifiers (QCI) were added. The list of QCI and their corresponding QoS profiles are detailed in the 3GPP Technical Specification [51]. Comparing the standardised QCI descriptions detailed in Table 2 to the category and application of each flow enabled the identification of 6 QCI groups in the data.
- The priority level of each flow was added to the data selection based on the corresponding QCI. Based on the 3GPP assigned priorities, the flows with 3GPP priority 2, 4, 3 and 0.7 were classified as 'high priority' and 3GPP priority 6 and 9 were classified as 'normal priority'.
- The bandwidth required by each flow was not provided in the dataset, but this parameter was estimated using the average packet size in each flow, which was provided in the dataset. Given that the packets in a flow are transmitted successively and not all at once, we conclude that the flow requires a bandwidth allocation which is directly proportional to the average packet size on that flow. As mentioned in section 4.2.2., the flows are assumed to have a constant packet size.
- Based on the flow start time, the flows were assigned hour numbers for better tracking and to enable the seasonality of the data to be more easily visualised. e.g., flows with start times that fall within the first 3600 seconds are assigned 'hour = 1', those that fall within the second 3600 seconds are assigned 'hour = 2' and so on.

Table 2 Standardised QCI Characteristics With Determined Priority Class [51]

QCI	Resource Type	Priority	Packet Delay Budget	Packet Error Loss Rate	Example Services	Priority Class
1	GBR	2	100ms	10^{-2}	Conversational Voice	High
2	GBR	4	150ms	10^{-3}	Conversational Video (Live Streaming)	High
3	GBR	3	50ms	10^{-3}	Real Time Gaming, V2X messages	High
65	GBR	0.7	75ms	10^{-2}	Mission Critical user plane Push To Talk voice (e.g., MCPTT)	High
6	non-GBR	6	300ms	10^{-6}	Video (Buffered Streaming) TCP-Based (for example, www, email, chat, ftp, p2p and the like)	Normal
9	non-GBR	9	300ms	10^{-6}	Video (Buffered Streaming) TCP-Based (for example, www, email, chat, ftp, p2p and the like). Typically used as default carrier	Normal

Step 4: Summarisation of the Dataset.

- For this admission control application, the dataset was truncated down to a selection of 20,861 successive flow entries. This truncated selection corresponds to network data collected over the span of 312 hours or 13 days on a real campus network. This truncated selection is sufficient for the scope and purposes of this dissertation.
- The selection of 13 days' data was then divided using a 10:3 ratio into 2 sets namely, the training data and the testing data respectively. The training data was comprised of the data from the first 240 hours (10 days) in the data selection. This corresponds to 17,906 flow entries. The testing data was comprised of the remaining 3 days' worth of data, corresponding to 72 hours and 2955 flow entries.

The training data was used to train the prediction model as well as to calibrate the admission control model. The testing data was used to test the prediction model, and the proposed admission control algorithms.

4.1.3. Analysis of Data Selection

This analysis provided some of the parameters needed for the construction and evaluation of the admission control algorithms and the prediction model.

The flow interarrival times for all the flows were calculated as $t_i - t_{i-1}$ where t_i is the flow start time of the current flow and t_{i-1} is the flow start time of the previous flow.

The probability density function of the flow interarrival time displayed in Figure 3 shows that the data points are not equally spaced in time. However, time-series forecasting is most effective for data with observations equally spaced in time. To achieve equally spaced data points, we implemented an observation interval of 1 hour by aggregating the flows within each hour into one entry. With this hourly data, the traffic forecasting could be done for the hourly arrival of future flows, rather than on an individual flow by flow basis.

The hourly arrival of flows follows an Inverse Gaussian distribution as shown in Figure 4 which was gotten from running the dataset through the distribution fitting tool in MATLAB. This suggests that the flows in this dataset arrive with a Poisson distribution.

A graph of the total bandwidth units requested hourly by incoming flows across the 240 hours in the training data is shown in Figure 5. A visual inspection of the graph in Figure 5 shows a sinusoidal pattern in the graph which repeats every 48 hours, showing seasonality in the data. This pattern is outlined in Figure 6 using the green seasonality graph.

A study of the plot of the autocorrelation function of the requested bandwidth in Figure 7 also shows a repeating pattern every 48 lags. This autocorrelation plot is displayed in Figure 7 and the seasonality is outlined in this plot. Based on Figures 6 and 7, we can conclude that the hourly data in this dataset is seasonal and can be forecasted using a time-series forecasting model. The shape of autocorrelation function plot also shows that statistical independence between the entries increases with the increase in lag.

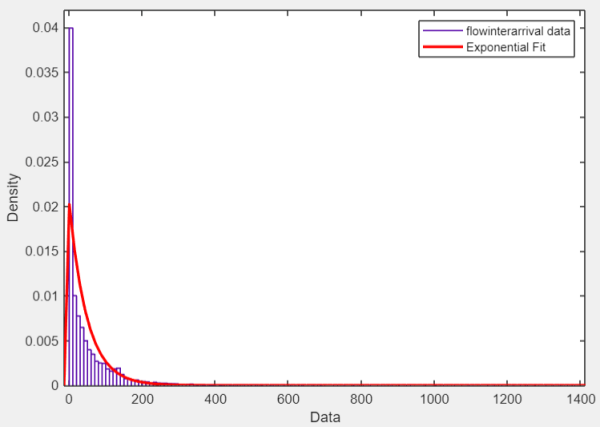

Figure 3 Probability Density Function of Flow Interarrival Time in Training Data, fitted with an exponential curve

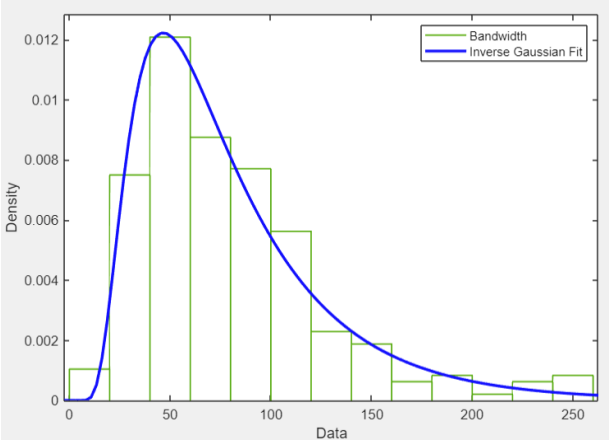

Figure 4 PDF of Total Bandwidth Requested Hourly in Training Data fitted with an Inverse Gaussian curve

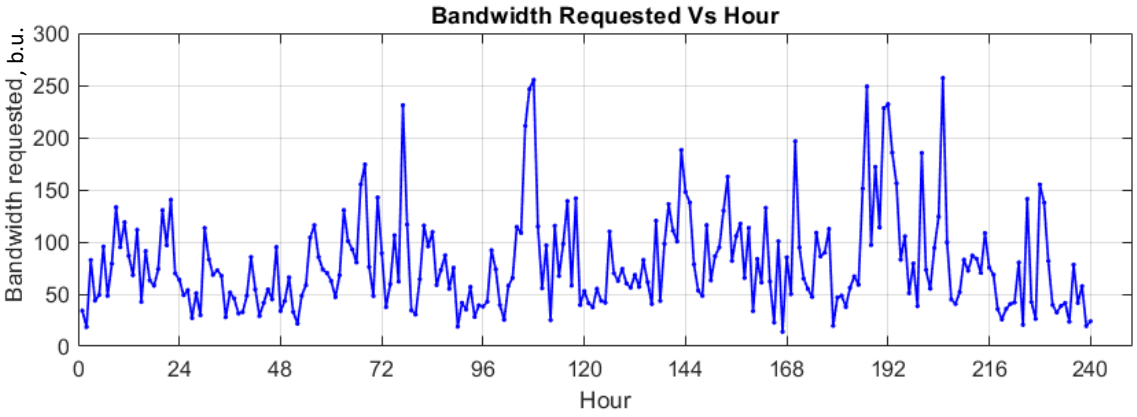

Figure 5 Graph of Total Requested Bandwidth each Hour, Depicted Across 240 hours (Training Data)

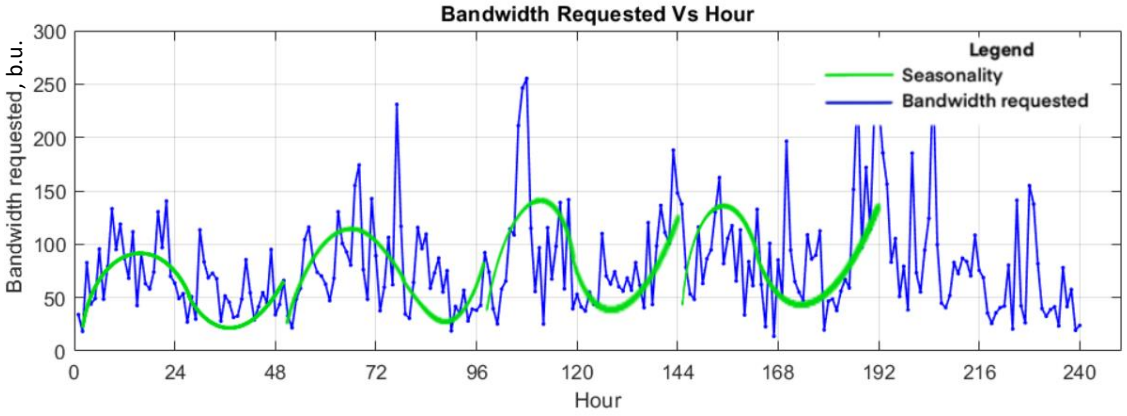

Figure 6 Graph of Total Bandwidth Requested Hourly with Seasonality

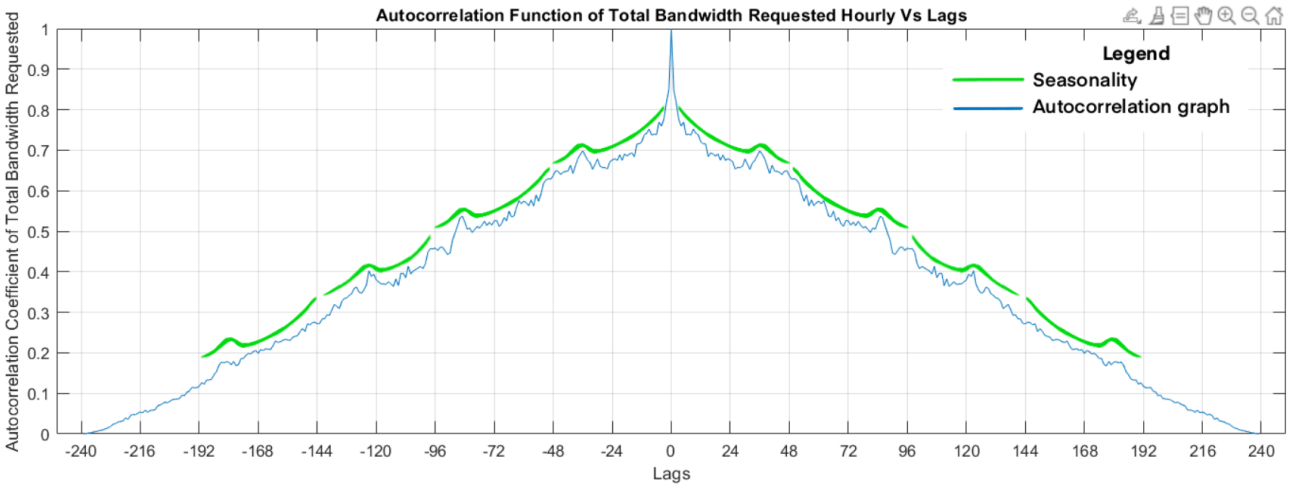

Figure 7 Plot of Autocorrelation Function of Total Bandwidth Requested Hourly with Seasonality

For this study, each flow requests system resources in the form of bandwidth and time. Each flow requests a certain number of bandwidth units, and each flow requests to be accommodated on the network for a number of time units. The bandwidth and time used by a flow can be jointly represented using a term called the 'resource block' and it is defined in equation 4.6.

$$1\ resource\ block, RB\ =\ 1\ bandwidth\ unit\ \times 1\ time\ unit \qquad (4.6)$$

The ratio of the high priority traffic, compared to the normal priority traffic in the training data was calculated and is presented in Table 3 below. It is clear that this particular dataset contains majority normal priority flows.

Table 3 Ratio of Normal Priority Traffic to High Priority Traffic

Priority	Number of Flows	Percentage of Total Traffic
Normal	17698	99.57%
High	76	0.43%

4.2. System Model and Design

4.2.1. Motivation for this system design

In Chapter 2, the state-of-the-art of admission control in sliced networks and admission control using predictive techniques was critically reviewed, and the three-module systems proposed by both Sciancalepore [44] and by Buyakar [45] was found to be one of the few systems which explicitly explored the intersection of admission control in sliced networks and predictive analytics. These systems however did not address the issue at the intra-slice level.

In this dissertation, our system model considers the need for admission control at the intra-slice level and investigates how admission control at the intra-slice level is impacted by the use of predictive analytics to dynamically adjust resources between slices. This intersection is an issue that was not explored in any of the related works reviewed in Chapter 2.

The literature reviewed in Chapter 2 shows that the admission control schemes applied to inter-slice admission control only provide pseudo-isolation as slices still undergo degradation when there is congestion. In cases where slice allocation is too rigid, there tends to be poor system utilisation. This work proposes a halfway point between these two approaches, where prediction techniques are used to predict the future traffic load and the predicted incoming traffic is used to pre-emptively calculate and allocate the necessary resources to slices at pre-determined time intervals. In this system model, once a slice allocation is done and the servicing time window has begun, the slice can no longer be

degraded, until the next resource allocation interval. This ensures better isolation between slices, while keeping resource allocation fairly dynamic. This system model also enforces congestion control by instating a queueing system for flows waiting to be admitted into the network.

4.2.2. Assumptions

Below are the assumptions made in the implementation of the described system:

- The mobile network operator oversees inter-slice admission control and resource allocation as well as intra-slice admission control and resource allocation. This assumption is made because this type of case where the mobile network operator is the infrastructure provider and also manages the slices on its network is one of the configurations in which network slicing is applied in mobile networks.
- Once the slices have been created and have been allocated resources, the slices achieve true isolation and are indivisible for a pre-determined time window, called the servicing window. This assumption is made because slice isolation and indivisibility during the servicing window is a feature of the system model in this work.
- The network speed is constant across the network and once a flow begins, the data transfer within that flow is consistent. This assumption is made because the dataset used in this work does not provide information on the variation patterns of the network speed.
- Network flows are statistically independent. This assumption is made because statistical independence is a common feature of network traffic.
- For the sake of simplicity, each flow is assumed to have a constant packet size, represented by the average packet size of the flow.

4.2.3. System Model

The system model is as illustrated in Figure 8 and it comprises 5 main components: the Flow Admission Control module, the Traffic Forecasting module, the Flow Servicing module, the Slice Resource Allocation module, and the Queue Management (Q.M.) module. This model extends the 3-module system proposed by the authors in [44] to comprise 5 main components.

Servicing Window, Observation Period and Batch Processing

In this system model, a slice is designed to run for a pre-determined length of time which we called the servicing window, W_t. At the start of a servicing window, resources are allocated to a slice and during the servicing window, the slice may not be degraded. This means that

network resources which have been allocated to that slice may not be removed from the slice until the servicing window is over.

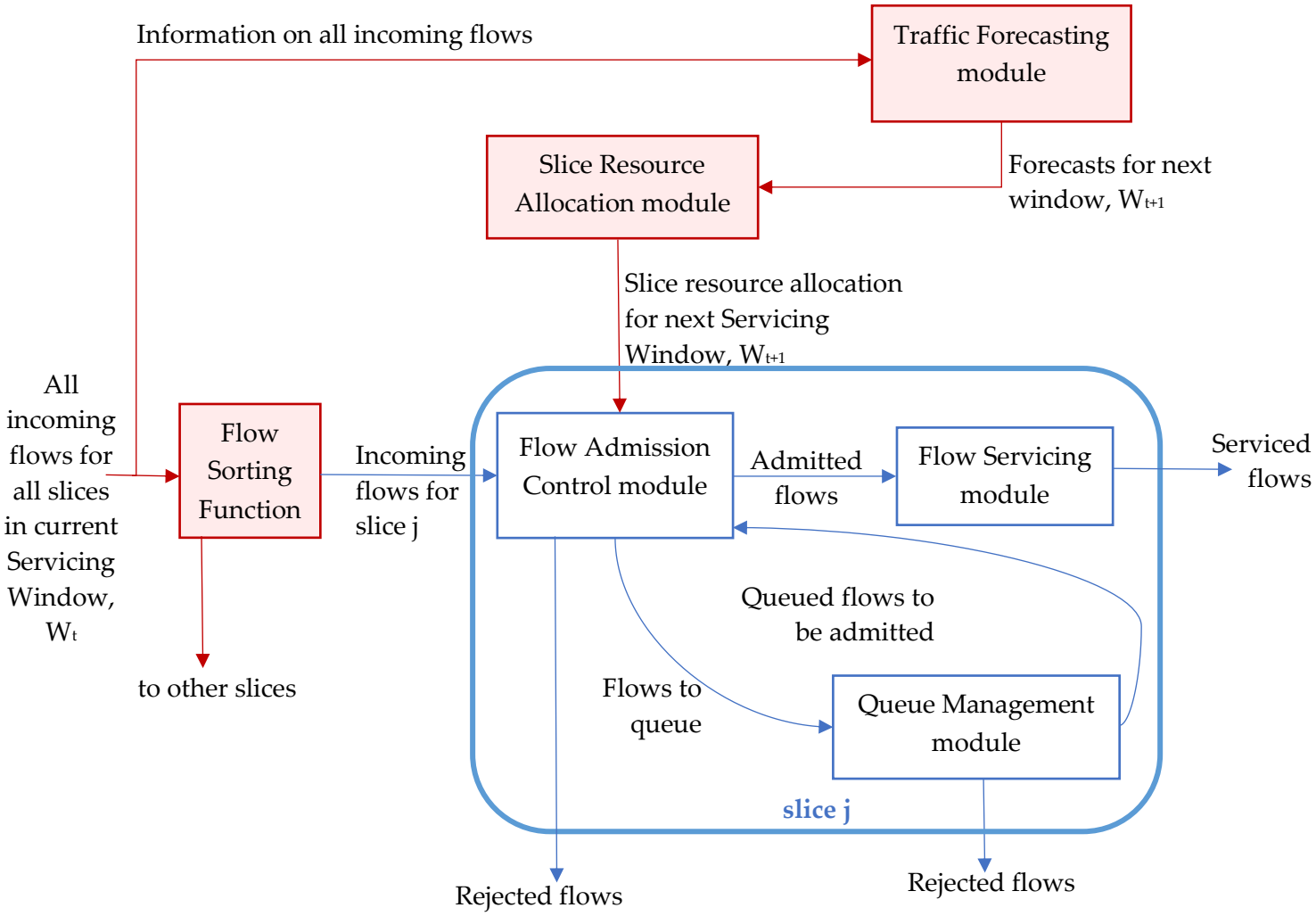

Figure 8 System Design Showing 5 Functional Modules and 1 External Module

For the purposes of feeding the real time flow data into the simulation, an observation period, T_{OBS} is defined. Flow requests are not fed into the simulation one after the other but rather, for a period of time (the observation period), the incoming flow requests are gathered and processed as a batch.

This means that if on a timeline of length, $2T_{OBS}$, 8 flow requests are recorded with the first 5 requests falling in the first half of the timeline ($\leq 1T_{OBS}$), and the remaining 3 requests occurring in the second half of the timeline ($1T_{OBS} < t \leq 2T_{OBS}$), then the requests will be divided into two batches to be processed through the system. 5 flows will be processed in the first batch during the first servicing window. The next 3 flows will be processed as the second

batch during the second servicing window. The Servicing Window, W_t is the length of time for which a slice runs uninterrupted. During W_t, the batch of incoming flows which were gathered during the observation period for that slice are processed by the Flow Admission Control. A similar approach of batch processing was used for the collection of slice requests in [44], [45] .

Description of Modules and System Model

The system shown in Figure 8 is focused on intra-slice admission control. Here, we consider that the flow requests which reach the Flow Admission Control module of a slice are all flows designated for that slice. To make this a feasible consideration, a Flow Sorting Function has been included one stage before the Flow Admission Control module and the role of this Flow Sorting Function is to determine the destination slice of each incoming flow request and route them to their designated slices.

The Flow Admission Control module receives the set of incoming flows and their corresponding flow profiles which contain flow information such as slice specification, flow duration, bandwidth requested, and priority class. The Flow Admission Control module examines the information contained in the incoming flow profiles, and based on the programmed admission control policies, a decision is made as to whether to admit, reject, or queue the incoming flow requests. The Flow Admission Control module makes admission control decisions based on the specific admission control algorithm it is programmed with. The admission control algorithms proposed for this work are covered in section 4.4.

The Flow Servicing module does all the processing needed to service each flow. Each flow has an associated flow duration and when the flow has been serviced up until the point where the flow duration expires, the Flow Servicing module terminates the flow and releases the system resources which were being used up by that flow. Flows which have been admitted, serviced and terminated are shown in Figure 8 as serviced flows.

The flows to be queued are sent to the Queue Management module where they are put in a first-in-first-out queue until network resources are available. The queue is modelled as a buffer of infinite length, however in practice, the queue would not hold more flows than are contained in the incoming batch of flows. When resources are available for queued flows, the admissible flows are sent to the Flow Servicing module. If no resources are made available for a queued flow within the Servicing Window, the queued flow is dropped.

The Flow Admission Control, Flow Servicing and Queue Management modules are slice-level modules, and they perform admission control and flow servicing functions within a given slice. These 3 modules are designed to serve a single slice, therefore, if there are N slices running on the network, they would each have their respective Flow Admission Control,

Flow Servicing and Queue Management modules to manage their incoming traffic. This ensures traffic isolation within slices. These modules can function without traffic forecasting, however this system includes 3 additional modules at the global network level: a Traffic Forecasting module, a Slice Resource Allocation module and a Flow Sorting Function.

The Traffic Forecasting module collects information on the traffic entering the network for each slice. This traffic information comprises of all incoming traffic requests including those admitted, queued, or rejected. This information is collected and used to train the Holt-Winters prediction model in the Traffic Forecasting module to provide a traffic forecast for each slice. The Traffic Forecasting module then sends all the forecasted traffic statistics and data to be used in the next Servicing Window, W_{t+1} to the Slice Resource Allocation module.

The slice resource allocation module may operate in one of two scenarios. The first scenario is that in which the information provided by the Traffic Forecasting module is not considered and the second scenario is that in which the information provided by the Traffic Forecasting module is considered and used to allocate resources to slices before the incoming traffic demands these resources.

In the case where forecasting is not applied, the Slice Resource Allocation module allocates a fixed number of resource units to each slice. Here, the number of resource units assigned to a given slice are not tailored to adequately accommodate the incoming traffic. The same number of resource units are used to accommodate incoming traffic as the demand increases and decreases over time.

In the scenario where the information provided by the Traffic Forecasting module is used by the Slice Resource Allocation module, the allocation of resources to the slice is dynamic and adaptive. The Slice Resource Allocation module receives the traffic forecast for the next servicing window and based on this data, decides how resources should be allocated between the existing slices in the next servicing window, W_{t+1}. This functionality is important because Figure 5 shows that that the incoming traffic load is not constant and fluctuates periodically. Since there is higher demand for resources in some periods over others, it is necessary to have a module which can pre-emptively reduce and increase the number of resource units allocated to each slice at the start of each servicing window. Even though in this case the slice resource allocation is dynamic, once resources have been allocated and a slice enters a servicing window, no slice degradation is done until the next servicing window, where the slice allocations are revised based on the forecasts. This is to ensure slice isolation.

The Flow Sorting Function as indicated in the diagram is in charge of mapping and redirecting the incoming flows to their designated slices. This system works on the assumption slices have already been admitted and are already running on the network.

The Traffic Forecasting module, the Slice Resource Allocation module and Flow Sorting Function are network-level modules, meaning that a single Traffic Forecasting module or a single Slice Resource Allocation module would serve all the slices admitted to the network.

This system focuses on the intra-slice admission control and on prediction-based slice resource allocation.

4.2.4. The Network Model

Each slice j receives a stream of flow requests at pre-defined intervals, the duration of which we have defined as the servicing window, W_t. Within a slice, each incoming flow f_i competes with the other flows within the same servicing window for a portion of the total slice bandwidth capacity, C and for a time slot within the servicing window W_t.

As verified in section 4.1.3, the flows arrive with a Poisson distribution.

Each flow f_i is associated with a flow profile represented by the vector $f_i = (b_i, d_i, v_i, a_i, p_i, s_i, t_i)'$ where:

b_i = bandwidth requested by flow i,

d_i = duration of flow i,

v_i, = traffic class (or profit per resource block) of flow i,

a_i = utility index of flow i,

p_i = revenue potential of entire flow i,

s_i = slice of flow i,

t_i = time at which flow i is admitted into network.

2 main traffic classes are considered in this system model, namely high priority and normal priority. It is assumed that each resource block assigned to flows has an associated profit. The profit from assigning 1 RB to a high priority flow is greater than the profit from assigning 1 RB to a normal priority flow.

If $High\ Priority\ profit\ per\ RB\ =\ Y$

and $Normal\ Priority\ profit\ per\ RB = Z$,

then, $Y > Z$

The profit that could be gotten from servicing flow i is then given by

$$revenue\ potential, p_i = b_i \times d_i \times v_i \qquad (4.7)$$

It should be noted that the profit term could represent monetary profit, or any other sort of perceived value or benefit to the operator for assigning a resource block to a particular traffic class.

In terms of resource blocks, the number of resources requested by each flow is given by

$$RB_i = b_i \times d_i \qquad (4.8)$$

The total number of resource blocks on offer in slice j is given by equation 4.9.

$$total\ RB_j = C \times W \qquad (4.9)$$

The slice utilisation of an admitted flow, i is the percentage of the total slice resources that are used up by that flow and is given by

$$Utilization\ of\ flow\ i = \frac{resource\ blocks\ consumed\ by\ i}{total\ slice\ resource\ blocks} = \frac{RB_i}{C \times W} \qquad (4.10)$$

Based on the system utilisation of flow i and on the profitability of flow i, we define a *Utility Index* term to measure the desirability of flow i to the network operator.

$$Utility\ Index\ of\ flow\ i, a_i = Utilization\ of\ flow\ i \times profit\ per\ RB_i = \frac{RB_i \times v_i}{C \times W} \qquad (4.11)$$

Each flow has a flow profile and the set of incoming flows in each Servicing Window can be represented by a traffic matrix where each column represents a flow profile e.g., the traffic profile of flow 1, f_1 is in the first column.

$$Traffic\ Matrix = \begin{pmatrix} b_1 & b_2 & b_3 & \\ d_1 & d_2 & d_3 & \\ v_1 & v_2 & v_3 & \\ a_1 & a_2 & a_3 & \cdots \\ p_1 & p_2 & p_3 & \\ s_1 & s_2 & s_3 & \\ t_1 & t_2 & t_3 & \end{pmatrix} \qquad (4.12)$$

The incoming traffic matrix is a 7×n matrix where n is the number of flows collected for that servicing window. The last row (row 7) is initially a set of zeros and each entry in row 7 is updated within the system with the time of admission of the corresponding flow. This last row is used by the Flow Servicing module to verify whether a flow has run its course and needs to be ended and its resources released.

Admission Control, which is at the centre of this study, is most relevant when the resources requested surpass the availability, causing congestion. To support this, when simulating this system model, C and W will be chosen such that the system mimics congestion.

For 5G networks and beyond, ensuring QoS is paramount. The high priority traffic class comprises of flows with stringent QoS requirements and normal priority traffic class has less

stringent QoS requirements. The Flow Admission Control module must apply different admission policies to the various traffic classes based on their priorities.

4.3. The Prediction Model

The Holt-Winters Exponential Smoothing (HWES) model is preferred for this traffic forecasting application due to its proven simplicity, intuitiveness, and versatility in the modelling of time series data. Also, HWES accounts for the seasonal component of data which is important for the prediction of seasonal data such as network traffic. A visual presentation of the network data shown in Figures 6 and 7 in section 4.1.3 shows that the seasonality is constant and in such a case, the additive Holt-Winters method is the appropriate choice as opposed to the multiplicative Holt-Winters method.

As covered in Chapter 2, the additive HWES model is based on the following 3 characteristic equations for level, trend, and seasonality respectively:

$$L_t = \alpha * (Y_t - S_{t-M}) + (1 - \alpha) * (L_{t-1} + b_{t-1}) \quad (2.5)$$

$$b_t = \beta * (L_t - L_{t-1}) + (1 - \beta) * (b_{t-1}) \quad (2.6)$$

$$S_t = \gamma * (Y_t - L_t) + (1 - \gamma) * S_{t-M} \quad (2.7)$$

A forecast of *k* steps into the future is given by the equation 2.8.

$$F_{t+k} = L_t + k * b_t + S_{t+k-M} \quad (2.8)$$

The initial values for the level (L_0) and trend (b_0) are calculated as shown in equations 2.9 and 2.10 respectively.

$$L_0 = \frac{1}{M} \sum_{t=1}^{M} Y_t \quad (2.9)$$

$$b_0 = \frac{1}{M} \sum_{t=1}^{M} \frac{Y_{t+M} - Y_t}{M} \quad (2.10)$$

Table 4 provides a definition of all the terms in the HWES equation.

The model represented by equations 2.5, 2.6, 2.7 and 2.8 is programmed into MATLAB and trained with the available network data from the dataset.

Table 4 Definition of Terms in the HWES Equations

Term	Description
L_0	initial level
L_t	current level
L_{t-1}	previous level
b_0	initial trend
b_t	current trend
b_{t-1}	previous trend
S_t	current seasonal value
S_{t-M}	corresponding seasonal value in previous season
Y_t	actual value at time t
M	duration of a season (number of observation points in a season) / order of seasonality
k	number of datapoints into the future to forecast
S_{t+k-M}	estimated seasonal value of the previous season at k steps ahead
F_{t+k}	forecast at k steps ahead based on additive seasonality

a. Training the Model

At this point the best values for these smoothing constants, α, β and γ, are not known yet. Although there are some suggestions in the literature as to what good values for these smoothing constants could be, the prediction model is not yet optimised for our use case and dataset until it is fitted to our data. Fitting the prediction model to the available data so as to improve the accuracy of future predictions is referred to as *training the model*.

In our use case of the additive Holt-Winters Exponential Smoothing method, the model is trained through an optimisation process which based on the data, informs us on the best 'optimum' values for the smoothing constants, α, β and γ. The smoothing constants are essentially weights thus, they must fall in the range [0 1]. The optimisation process aims at determining the values for α, β and γ at which the prediction error is acceptable. We do not aim for 0 error when fitting a prediction model so as not to overtrain the model, but we need an accurate enough prediction for the model to be of use. As is common practice [38], the prediction error is represented by the root mean square error (RMSE). The NRMSE is also used to represent the error as a percentage for easy comparison (normalised across the range). Equations 4.13 and 4.14 represent the RMSE and NRMSE respectively.

$$RMSE = \sqrt{\frac{\sum_i^N (y_i - \hat{y}_i)^2}{N}} \quad (4.13)$$

$$NRMSE = \frac{RMSE}{y_{max} - y_{min}} \quad (4.14)$$

Where i = count of predicted datapoints

y_i = i-th actual measured value

$\hat{y}_i$ = i-th forecasted value

y_{max} = maximum actual measured value y

y_{min} = maximum actual measured value y

N = number of datapoints

The data selection was divided into a training set and testing set in a 10:3 ratio. 10 days' worth of data (240 hours) was used for training the model and 3 days' worth of data (72 hours) was used to test the model. A link to the working data selection has been included at the end of the chapter, in section 4.5.

Several approaches were tried for training the prediction model and are detailed below with the most effective approach being the 4th approach which was a MATLAB-based Non-Linear Optimisation solution. The approaches tried for the training of the prediction model are:

1. Arbitrary values between 0 and 1 were assumed for the smoothing constants, α, β and γ and the training data was run through the model with these arbitrary values. The performance of the model, depicted by the error terms RMSE and NRMSE, was observed.

2. Using 0.1 as a starting value for all three smoothing constants (α, β and γ), the values of α, β and γ were varied manually and tested. This variation was done on a trial and error basis where values higher and lower than 0.1 were tested and the corresponding RMSE and NRMSE were observed. The ending values in this exercise yielded improved RMSE and NRMSE compared to the starting values and to the approach in 1. above. This exercise showed that the prediction model performed better with lower values of β and γ.

3. An attempt to solve for optimal values for the smoothing constants, α, β and γ was made using the basic Microsoft Excel solver function. The objective was set to minimise the value of the cell containing the RMSE using a GRG Non-linear solving method. The cells containing the values of the smoothing constants were set as the 'Variable Cells' to be changed and the constraints were defined such that the value of these Variable Cells must range from 0 to 1.

4. In MATLAB, using the Optimisation Tool, the smoothing constants, α, β and γ were defined as the solution to a non-linear optimisation problem with upper and lower bounds as constraints.

Constraints were $0 \leq \alpha \leq 1, 0 \leq \beta \leq 1, \ and \ 0 \leq \gamma \leq 1$.

The objective was to minimise the objective function, which was defined as below:

$Objective\ function = RMSE(\hat{y})$,

Where RMSE returns the RMSE of $\hat{y}$, and $\hat{y}$ is the set of HWES forecasted values based on the network data.

As shown in Figure 9, the optimisation function in Approach 4 iteratively tests for a lower value of the RMSE function and after 14 iterations, the best RMSE value was found to be 42.68, or NRMSE = 0.1718. The number of iterations done to get the optimal values of the smoothing constants is 14 because increasing the number of iterations beyond 14 does not yield any improvement in the RMSE or NRMSE. Thus, for our training data, 14 iterations are sufficient to produce the best values for the smoothing constants.

Table 5 presents a summary of the values for alpha, beta and gamma gotten from the implementation of the four approaches which were considered for the training of the prediction model as described above.

Table 5 Summary of HWES Model Training Approaches and Performances

Forecasted Variable: Total Requested Bandwidth each Hour					
Training Approach	**Approach 1: Arbitrary values**	**Approach 2: Trial and Error (Start values vs End values)**		**Approach 3: Excel Solver Function**	**Approach 4: Optimisation (14 iterations)**
		Start values	**End values**		
alpha, α	0.25	0.1	0.25	0.947	0.377
beta, β	0.025	0.1	0.00025	0.00018	4.991×10^{-9}
gamma, γ	0.5	0.1	0.0005	1	3.691×10^{-8}
RMSE	**62.6105**	**126.818**	**43.8296**	**48.579**	**43.4**
NRMSE	**0.2575**	**0.5216**	**0.1803**	**0.1998**	**0.1718**

The prediction model was duplicated, and the duplicate was trained to forecast the traffic load. Table 6 is an extract of Table 5 and Table 6 presents the optimised values for the smoothing constant after 14 iterations of Approach 4. These are the values that are used in the final prediction model. Figure 9 shows the output of the MATLAB Optimisation Tool after 14 iterations with the best value of the RMSE at 43.4.

Table 6 Optimised Smoothing Constant for Prediction of Hourly Bandwidth Demand

alpha, α	0.377
beta, β	4.991×10^{-9}
gamma, γ	3.691×10^{-8}
RMSE	43.4
NRMSE	0.1718

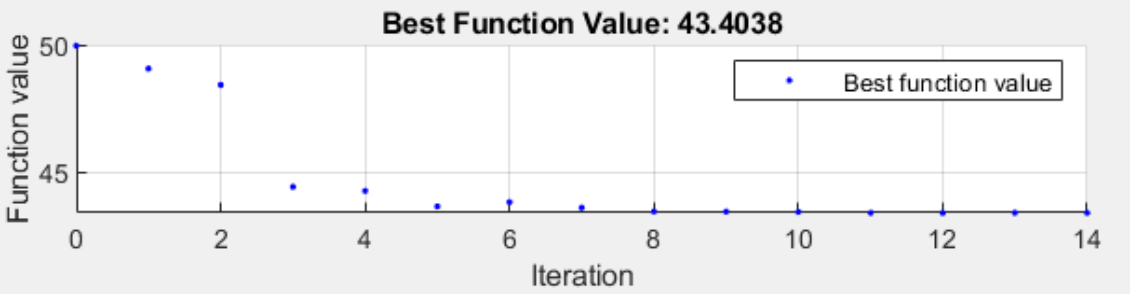

Figure 9 Variation of the Function Value (RMSE) with Number of Optimisation Iterations

A graph of the traffic forecasts gotten from the prediction model compared to the actual values is shown in Figure 10. The forecasts lag the actual values by one step, showing that the data is short-range dependent.

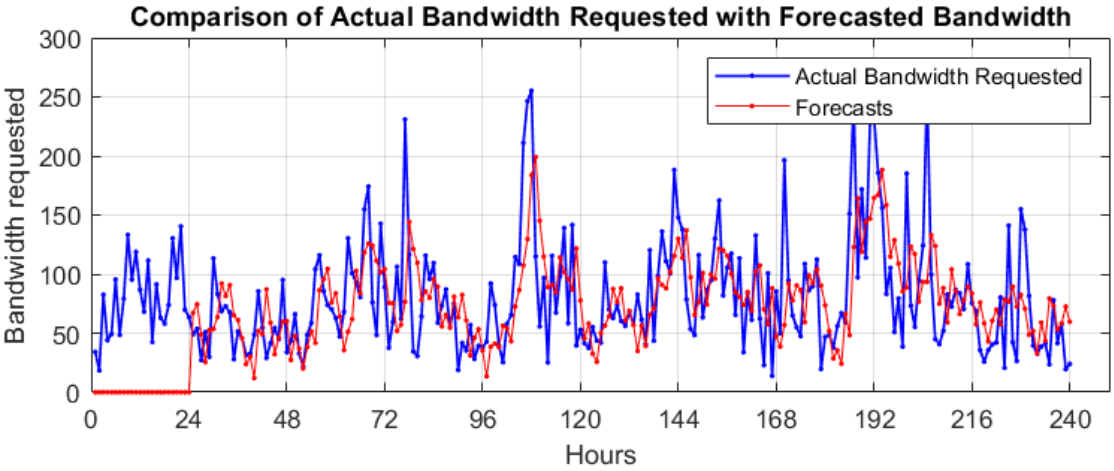

Figure 10 Comparison of Forecasted Values Using the Trained Model with Coefficients from Approach 4 (Optimisation) with Actual Data from the Dataset

The seasonal unit, M for this dataset is chosen to be 24 hours.

The actual error incurred in the forecasting of $\hat{Y}_t$ using the trained model is calculated as $E_t = Y_t - \hat{Y}_t$. When forecasting using HWES, no forecasts are done within the first season (M = 24). This means that for the training data, we effectively have 240 – 24 = 216 forecasts, shown by

the red graphs in Figure 10. The error values of all 216 forecasts are calculated and the cumulative distribution of the error is shown in Figure 11 and Figure 12.

53.24% of the error is less than or equal to 0, meaning that the predictor overestimated 53.24% of the expected bandwidth requests. For these cases, we hypothesise that resource allocation based on the forecasted bandwidth requests would ensure QoS, but system utilisation would be poor. In the remaining 46.76% of cases, the predictor underestimated the incoming bandwidth demand and we hypothesis that for these cases, resource allocation based on the forecasts would yield under-provision of resources but possibly high system utilisation.

The second CDF plot below shows that 90.28% of the error is less than or equal to 54. It follows that for a 90.28% guarantee of bandwidth availability for all incoming flows for a given servicing window, the system will have to be allocated 54 bandwidth units above the forecasted value.

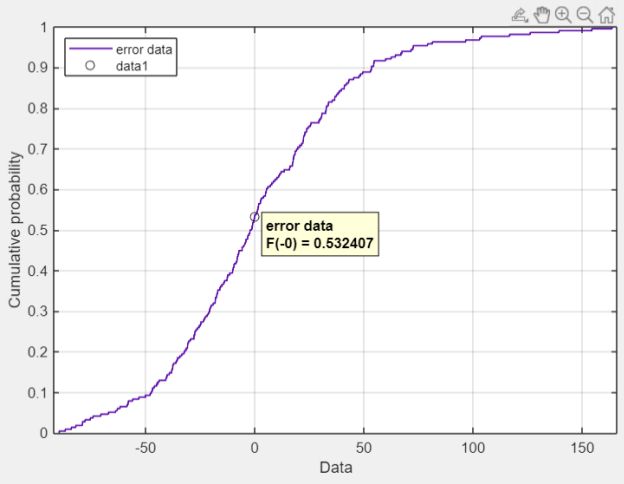

Figure 11 CDF plot of Error, Probability(Error≤0) = 0.5324

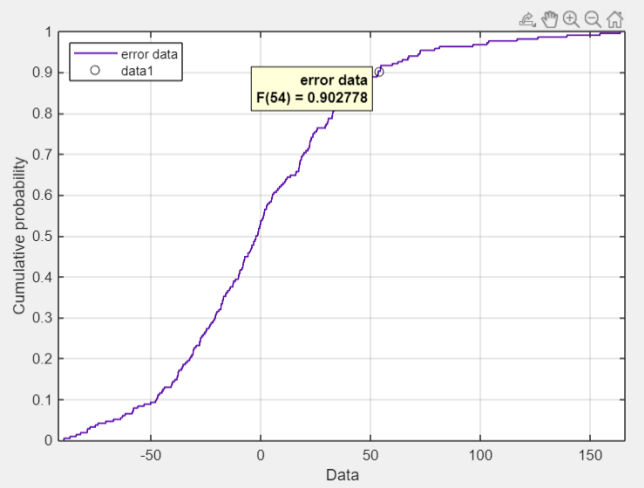

Figure 12 CDF plot of Error, Probability (Error≤54) = 0.9028

A summary of all the parameters of the forecasting model is shown in Table 7.

Table 7 Summary of Trained HWES Forecasting Parameters

Summary of Forecasting Model Parameters			
Model	Holt-Winters Additive	**alpha, α**	0.377
Prediction period	1 hour	**beta, β**	4.991×10^{-9}
Length of Season	24 hours	**gamma, γ**	3.691×10^{-8}
Training data	240 data points	**Performance Metrics**	RMSE, NRMSE
Testing data	72 data points	**Feature to forecast**	Requested bandwidth

4.4. Proposed Admission Control Algorithms

According to the system model presented in section 4.2.3., the Flow Admission Control module is programmed with an admission control algorithm which governs the admission control process.

To tackle the issue of intra-slice admission control, we propose 2 admission control algorithms which will be designed, simulated, and evaluated in subsequent sections. The proposed algorithms are the *Decision Matrix algorithm* and the *Utility Index algorithm*. We also compare these two algorithms with a basic legacy *First-in-First-out* system. These admission control algorithms are intended for implementation within a running slice.

4.4.1. Intra-slice Admission Control based on Decision Matrix Algorithm

For every flow request coming into the slice, the Flow Admission Control module makes one of the 3 decisions outlined in equation 4.15.

$$A.C\ decision\ set\ = \{admit, queue, reject\} \qquad (4.15)$$

The decision to either admit, queue or reject is guided by the Decision Matrix shown in Table 8. This Decision Matrix is inspired by the Eisenhower Matrix [52], [53] for time management and task scheduling.

The Decision Matrix algorithm considers the flow classes and ensures the prioritisation of high priority flows over normal priority flows. This is done by reserving some bandwidth for the high priority flows through the implementation of a bandwidth threshold, C_t. The bandwidth threshold is set at a percentage, say 70% of the slice capacity. This means that incoming normal priority and high priority flows can be admitted into the slice until 70% of the slice capacity has been used up. Beyond this 70% capacity, only high priority flows are admitted into the available resources and normal priority incoming flows are either queued up or rejected from the network based on further admission control policies.

Table 8 Decision Matrix for Admission Control Policies

High priority	**ADMIT**	**ADMIT**	**QUEUE**
Normal priority	**ADMIT**	**QUEUE**	**REJECT**
	Available bandwidth + requested bandwidth ≤ threshold	Threshold ≤ Available bandwidth + requested bandwidth ≤ Capacity	Available bandwidth + requested bandwidth > Capacity

The admission control policies for new incoming high priority flows, normal priority flows, and queued flows are outlined below.

- For a high priority flow request,

  $admit$ if $available\ bandwidth\ +\ requested\ bandwidth\ \leq\ capacity$
  Otherwise, $queue$.

- For a normal priority flow request,

  $admit$ if $available\ bandwidth\ +\ requested\ bandwidth\ \leq\ threshold$
  $queue$ if $threshold < available\ bandwidth\ +\ requested\ bandwidth\ \leq\ capacity$
  Otherwise, $reject$

- For a queued flow request,

  $admit$ if $available\ bandwidth\ +\ requested\ bandwidth\ \leq\ capacity$
  Otherwise, $reject$.

A flowchart of the Decision Matrix algorithm is shown in Figure 14. The simulation is set to run for one servicing period, during which the flow requests gathered in one observation period will be serviced, as explained in section 4.2.3. Once the system is initialised, the admission control policies are enforced by the algorithm as shown in Figure 14 to decide for each incoming flow whether to admit, queue, or reject. Once an admission control decision has been made, the algorithm calls the servicing function subroutine illustrated in Figure 13.

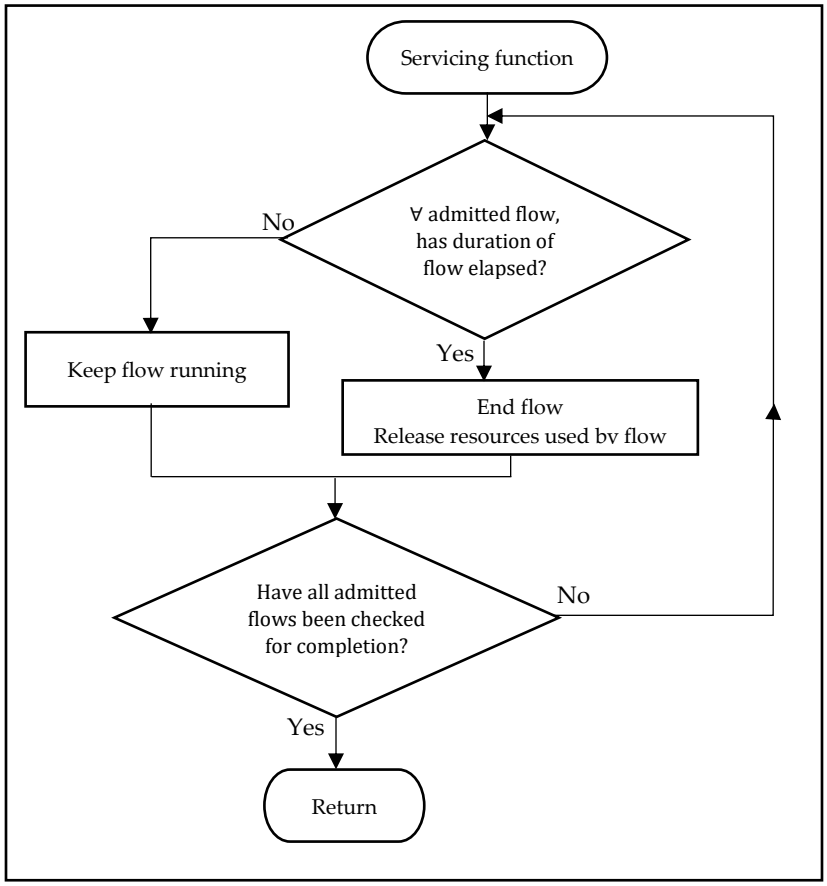

Figure 13 Servicing Function Subroutine

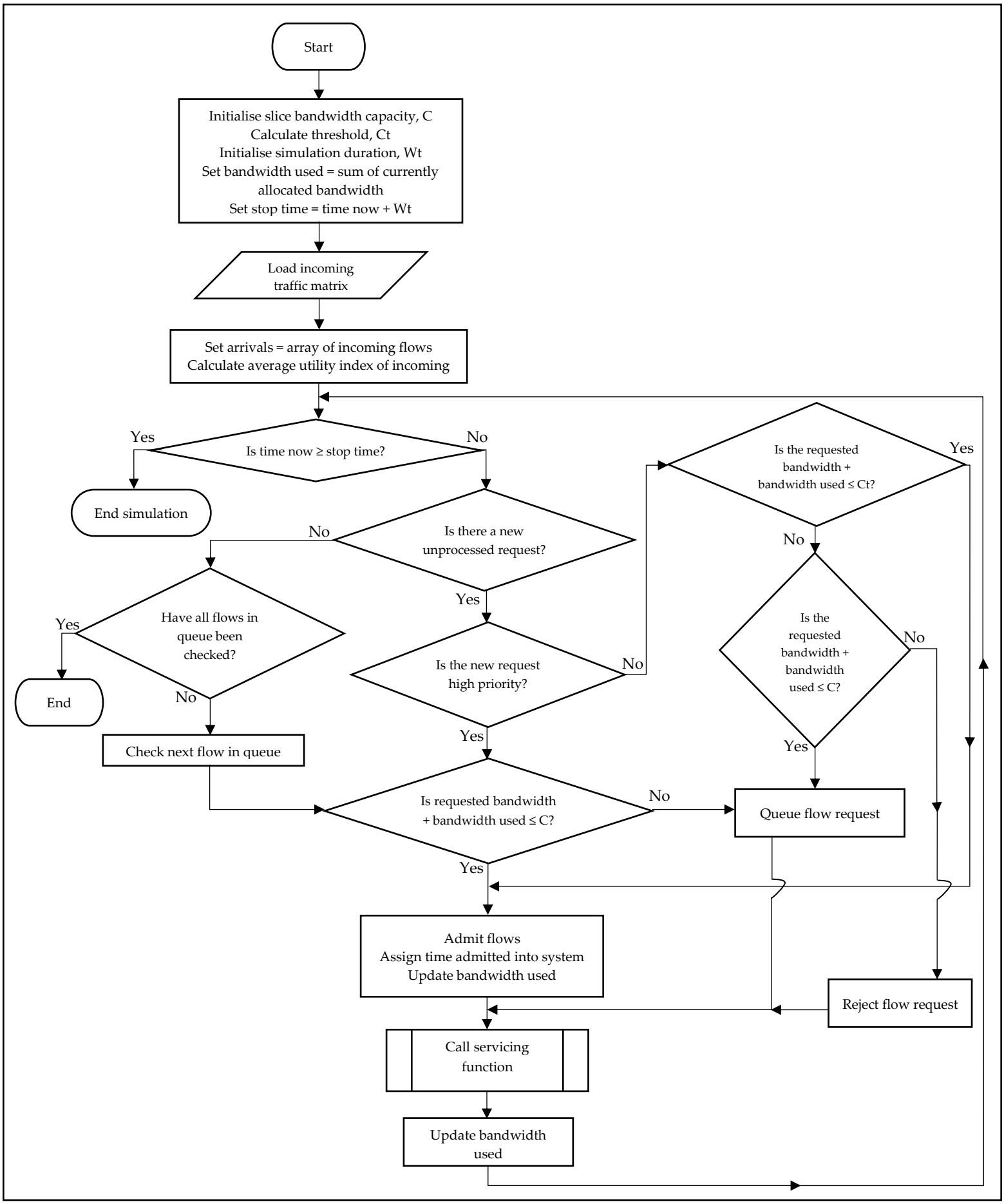

Figure 14 Flowchart for Decision Matrix Algorithm

4.4.2. Intra-slice Admission Control based on Utility Index Algorithm

The Utility Index algorithm prioritises flows based on their perceived utility to the operator. The concept of Utility Index, a_i was introduced in Section 4.2.4. and is calculated using equation 4.11. Flows with higher Utility Index are hypothesised to yield higher system utilisation and higher profits.

$$a_i = \frac{RB_i}{C \times W} \times V \qquad (4.11)$$

Where RB_i = number of resource blocks requested by flow i,

$C \times W$ = capacity of system in terms of resource blocks,

V = value per resource block of flow i.

V for high priority traffic > V for normal priority traffic

In this admission control method, the average utility index, a_{avg} of the batch of incoming flows is calculated and using a_{avg}, the incoming flows are classified into high and low utility index flows as shown in Table 9. Flows whose utility indices are above average are high utility flows and flows whose utility indices are below average are low utility flows.

Table 9 Distinction Between High and Low Utility Index

High Utility	$a_i \geq a_{avg}$
Low Utility	$a_i \leq a_{avg}$

The Utility Index algorithm services the flows with the highest utility index first, giving priority to the high utility flows, however, the flows with the highest utility indices are bandwidth-heavy applications which run for a long duration. If priority is given to such flows without provision for the servicing of low utility flows, the high utility flows will monopolise the system, leaving little to no system resources for other flows. To counter this issue, a bandwidth threshold is set which limits the amount of bandwidth that can be allotted to the high utility flows.

The bandwidth threshold ensures fairness and allows flows with low demand on resources to be serviced as well.

Fairness among the traffic classes is also ensured by the fact that even though the value per resource block of normal priority traffic is lower than that for high priority traffic, normal priority traffic can still bear a high utility index by virtue of their requested bandwidth and/or flow duration. This is shown in Table 10 which is an excerpt from our real network data

selection. The flows in the first two rows have high utility indices but are of different traffic classes. The flows in the last two rows have low utility indices but are of different traffic classes.

Table 10 Extract of Data from Working Selection

Flow Key (ID)	Hour	Normalised flow start time	Requested Bandwidth on 50 scale	Flow Duration on 120 scale	Value/Class (1 or 3)	Utility Index	Category
a1877b17596b8bf8 15ec9a59aa5fb801	288	1035064	2.25	6.661701	3 – High Priority	12.487	Network
9a642de21123835ba 4079f77c4b9feff	288	1035102	1.60	20.067678	1 – Normal priority	13.344	Web
81d47d0a10addb7 cb3187cada8b22ffb	288	1035024	0.43	0.000044	3 – High Priority	1.556E-05	VoIP
e27c4cf832f1bb0 feede24eadc8b0b6c	288	1035101	0.79	0.000041	1 – Normal priority	1.367E-05	Web

The admission control policies for the Utility Index algorithm are outlined below:

- For a high Utility Index requests,

  $admit$ if $available\ bandwidth + requested\ bandwidth \leq threshold$
  Otherwise, $queue$.

- For a low Utility Index request,

  $admit$ if $available\ bandwidth + requested\ bandwidth \leq capacity$
  Otherwise, $queue$

- For a queued flow request,

  $admit$ if $available\ bandwidth + requested\ bandwidth \leq capacity$
  Otherwise, $reject$.

The admission control policies of the Utility Index algorithm are further illustrated in the flowchart in Figure 15. Once an admission control decision is made, the Utility Index algorithm enters the servicing function subroutine whose flowchart is illustrated in Figure 13.

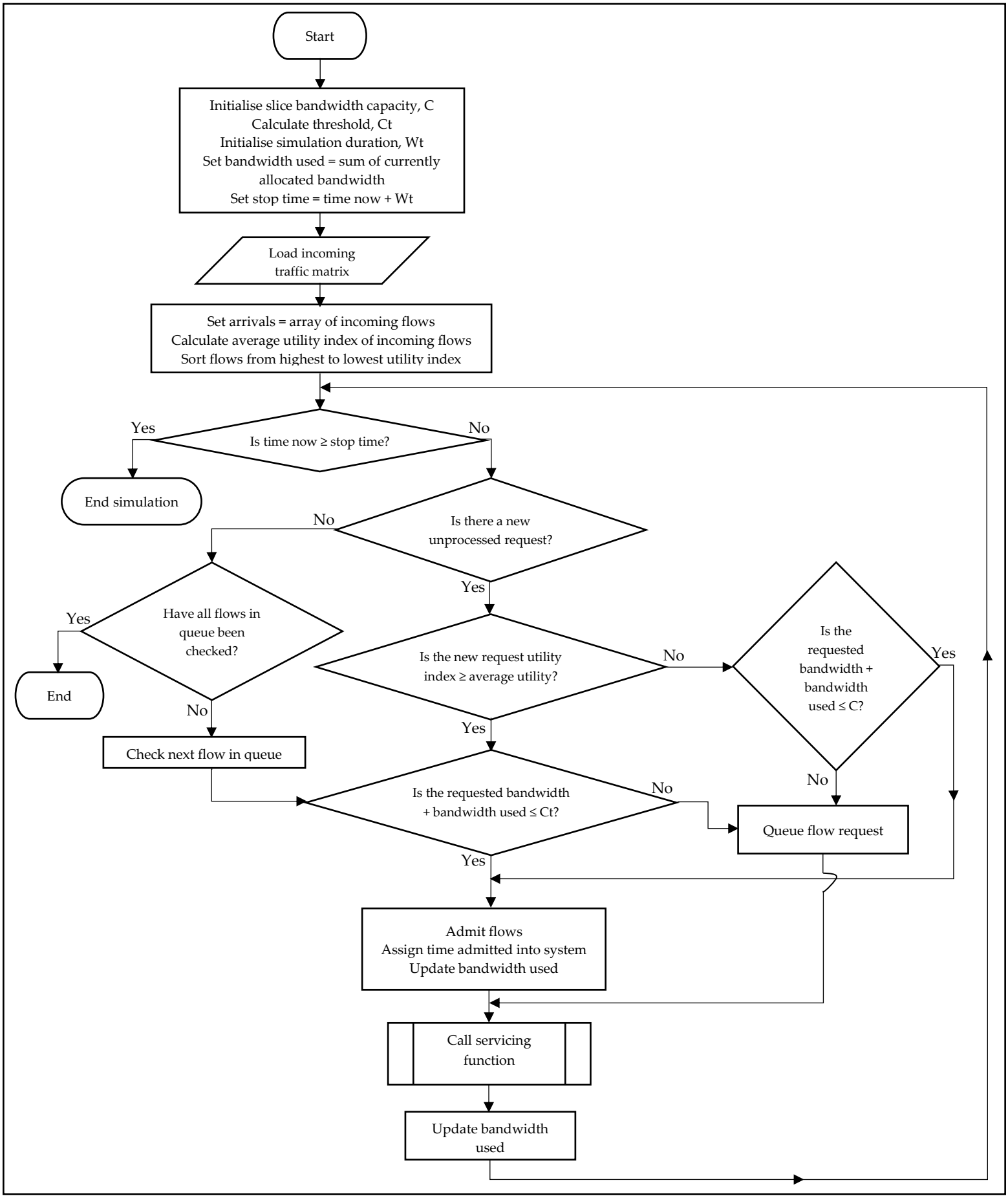

Figure 15 Flowchart for Utility Index Algorithm

4.4.3. Legacy First-In-First-Out (FIFO) Admission Control

The legacy FIFO admission control method was implemented as a control for evaluating the two methods designed in this work. This method is very simple and does not consider prioritisation or any flow characteristics in its decision-making. It admits users on a first-come-first-serve basis. However, since the two methods proposed above have implemented queueing as a congestion control mechanism, queueing is added to this FIFO algorithm to allow for a fair evaluation.

The admission control policies for the FIFO algorithm are as follows:

- For an incoming requests,

  $admit$ if $available\ bandwidth\ +\ requested\ bandwidth\ \leq\ capacity$
  Otherwise, $queue$

- For a queued flow request,

  $admit$ if $available\ bandwidth\ +\ requested\ bandwidth\ \leq\ capacity$
  Otherwise, $reject$.

The admission control policies for the FIFO algorithm are illustrated in the flowchart in Figure 16. As holds for the Decision Matrix and the Utility Index algorithms, once an admission control decision is made, the FIFO algorithm enters the servicing function subroutine whose flowchart is illustrated in Figure 13.

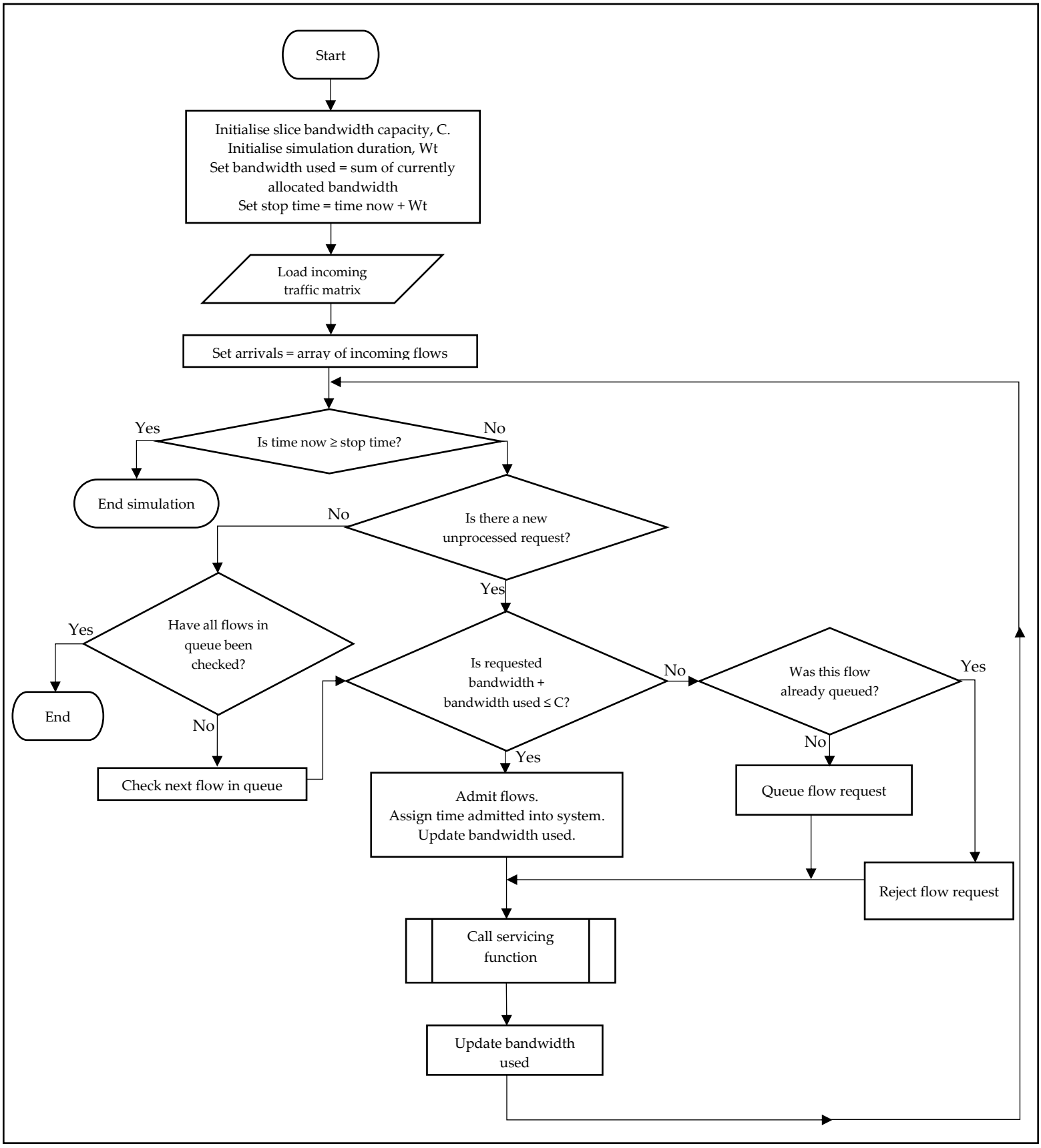

Figure 16 Flowchart for FIFO Algorithm

4.4.4. Prediction-Based Admission Control

To incorporate traffic forecasting into our admission control methods, we implement the unit which was defined in Section 4.2.3 as the Traffic Forecasting module. This module performs traffic forecasting using the HWES model. It feeds forecasts for the next batch of incoming flows to the Slice Resource Allocation module. The slice allocation function of the Slice Resource Allocation (S.R.A.) module is implemented manually by the experimenter by adjusting the capacity of the slice for the simulation of the next batch of flows, based on the forecasted data.

The slice allocation module computes an appropriate slice capacity for the next batch of flows based on equation 4.16.

$$Calculated\ slice\ capacity\ for\ W_{t+1} = Forecasted\ bandwidth\ /\ 18.52 \qquad (4.16)$$

Where 18.52 is the ratio of the incoming bandwidth demand for a model slice to the static bandwidth capacity of that model slice operating within the fixed servicing window, W_t.

The Slice Resource Allocation module then allocates the calculated slice capacity to the slice for the upcoming servicing window and the slice holds those resources until the bandwidth allocation is revised for the following window. No slice degradation takes place during the servicing window, ensuring slice isolation.

The forecasts will be generated for one period into the future ($\hat{Y}_{t+1}$). While it is possible to forecast several future points at a time, we forecast one point at a time so that the next forecast can be informed by the current slice traffic.

The NRMSE of the model implemented in the Traffic Forecasting module is known, based on the work done in Chapter 4.

4.5. Source Codes

The source codes and workspace variables for the admission control algorithms and the HWES traffic forecasting model described in this chapter can be downloaded from this Github link:

https://github.com/PeroseNgufor/AdmissionControlWithPredictions.git

The pre-processed data selection with normalised values for time and bandwidth related parameters can be downloaded from this link:

https://docs.google.com/spreadsheets/d/1SDUFSf3Zqw-hRZhVU7KrcZ8AgOHLCzH0/edit?usp=sharing&ouid=114415186482110144996&rtpof=true&sd=true

4.6. Chapter Summary

This chapter covers the process of developing our prediction model, starting with the preparation of the dataset using data mining and data pre-processing. Seasonality was identified in the data, and this is served as an indication that the Holt Winters Exponential Smoothing (HWES) model is a suitable prediction model for the given dataset. This chapter presents the development and training of the HWES prediction model using the given dataset. A modular system model where each function is implemented on a separate module was presented in this chapter. The network model was presented as well. New concepts such as the desirability of a flow, represented by the utility index were introduced alongside the network model.

The modular nature of the system model allows for admission control algorithms to be run with and without the input of predictive analytics. Two new admission control algorithms and their admission control policies (the *Decision Matrix* and the *Utility Index* algorithms) were introduced and explained in this chapter, and a legacy admission control algorithm (the *First-In-First-Out* algorithm) was also presented as a control.

The performance of these 3 admission control algorithms and the effects of incorporating predictive analytics into the admission control process are discussed in Chapter 5.

Chapter 5: Simulations and Results

In this chapter, we first define the relevant performance metrics used to evaluate the performance of the various admission control approaches. These performance metrics also provide a basis on which the various admission control approaches can be compared. Next, we present and analyse the results of the simulations which were carried out. We compare the performances of the proposed admission control algorithms without predictive analytics to the performance of the same admission control algorithms with predictive analytics. The pre-processed data selection can be found at [54] and the raw dataset is available at [48].

5.1. Performance Metrics

The performance metrics chosen for this study are the probability of blocking, system utilisation and the profitability to the operator. This is because as seen in the literature review presented in Chapter 2, the probability of blocking, system utilisation and profitability are commonly used by researchers and industry professionals to evaluate various admission control mechanisms. Using commonly-used performance metrics allows for easy comparison between the algorithms proposed in this work and other algorithms in the field.

1. **Probability of Blocking, PoB**

When a request is made to the network for the creation of a flow, the Flow Admission Control module decides to either admit or block (reject) the request. The probability of blocking is a measure of the likelihood for a request for network access to be rejected on a given network within a given period of time.

In the proposed admission control algorithms, blocking occurs when incoming normal priority requests arrive when there are no available resources to accommodate that request.

The higher the probability of blocking, the poorer the performance of the admission control mechanism. The probability of blocking is given by equation 5.1.

$$PoB = 1 - \frac{number\ of\ admitted\ flows}{number\ of\ incoming\ flows} \qquad (5.1)$$

It is desirable for an admission control algorithm to maintain very low PoB.

2. System Utilisation, S.U.

The system utilisation is the ratio of how much of the system's resources are used up within a given time frame compared to the total usable system resources. In this work, the main system resources being considered are bandwidth and time, collectively represented in resource blocks.

The system utilisation is given by equation 5.2.

$$s.u. = \frac{\sum_{i=1}^{N} RB_i}{C \times W} = \frac{\sum_{i=1}^{N} (b_i \times d_i)}{C \times W} \quad (5.2)$$

Where N = number of serviced flows,

RB_i = number of resource blocks requested by flow i,

$C \times W$ = capacity of system in terms of resource blocks,

b_i = bandwidth allocated to flow i during servicing,

d_i = duration for which flow i was serviced.

The system utilisation is expressed as a percentage, and it is desirable for an admission control algorithm to maintain a high system utilisation.

3. Profit or Profitability, P

The profitability is a measure of how beneficial it is for the operator to run a given admission control algorithm. Profit may represent revenue or any other value that an operator may choose to attach to the different traffic classes it serves.

Profitability is calculated on the premise that the gains from servicing a high priority call requesting 1 RB is higher than the gains from serving a normal priority call requesting 1 RB. The formula used in this work for profitability is given by equation 5.3.

$$P = \sum_{i=1}^{N} b_i \times d_i \times v_i \quad (5.3)$$

Where N = number of serviced flows,

b_i = bandwidth allocated to flow i during servicing

d_i = duration for which flow i was serviced

v_i = value/gain/profit per resource block of flow i.

It is desirable for an admission control algorithm to maintain high profitability.

5.2. Simulations

The three admission control algorithms namely, Decision Matrix, First-In-First-Out with Queue, and Utility Index algorithms, were implemented using MATLAB. These three admission control algorithms were covered in Chapter 4. The Holt-Winters Exponential Smoothing prediction model, also covered in Chapter 4, was also implemented, trained and tested using MATLAB.

In this implementation, because of the nature of the available dataset, our resource parameters are bandwidth and time, but the resources in this system could as well be substituted with other types of communication network resources such as computational power or energy.

5.2.1. Simulation Parameters

Table 11 outlines the simulation parameters which were used in this study. The values of these simulation parameters such as Bandwidth Capacity and Observation Period were chosen so as to simulate a congested system. Congestion is created by giving these parameters values which are lower than the average demands from the dataset used. This is important because the usefulness of an admission control system is only perceived when there is congestion and during congestion, the functioning and performance of the various admission control algorithms can be observed and assessed.

Table 11 System Simulation Parameters

ADMISSION CONTROL SIMULATION PARAMETERS	
Parameter	**Value**
Bandwidth Capacity, C	varied
Threshold	70% of Bandwidth capacity
Servicing Window, W_t	10 time units
Number of Traffic Classes	2: high priority, normal priority
High priority v_i	3
Normal priority v_i	1
Observation Period, T_{OBS}	3600 time units
Number of T_{OBS} used	10
Queue (buffer) length	infinite
HWES FORECASTING MODEL PARAMETERS	
Dependent variable (to be forecasted)	Slice load (RBs)
alpha, α	0.347
beta, β	2.178×10^{-10}
gamma, γ	3.769×10^{-10}
k (number of points to forecast head)	1

Flows in the data selection are grouped into various observation periods based on their flow start time.

5.3. Presentation and Discussion of Results

The simulation was run using the testing dataset which comprised of 72 observation periods or 72 batches of flows, totalling 2955 flows. For the sake of brevity and a concise presentation, the results presented section 5.3.2. are the simulation results from processing the data from the last 24 hours (hour 49 to hour 72) in the testing data. The complete set of results can be found in [55].

5.3.1. Evaluating the Holt-Winters Prediction Model

As covered in section 4.3.2., the performance of a prediction model is evaluated using the prediction accuracy, which is indicated by an error term such as the root mean square error (RMSE), or the normalised root mean square error (RMSE). Since the prediction model was constructed with a seasonality of 24 hours, during testing, we chose to retrain the model every 24 hours, making it adaptive to the new data received in the last 24 hours. As seen in Figures 17 and 18, the accuracy of the prediction model improves with every new training iteration, and this is indicated by the downward trend in the error terms RMSE and NRMSE.

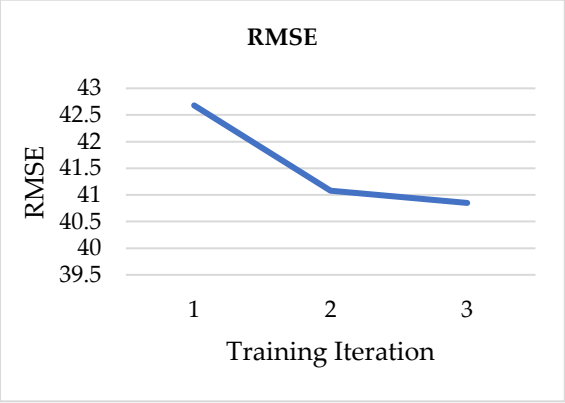

Figure 17 RMSE for each Training Iteration

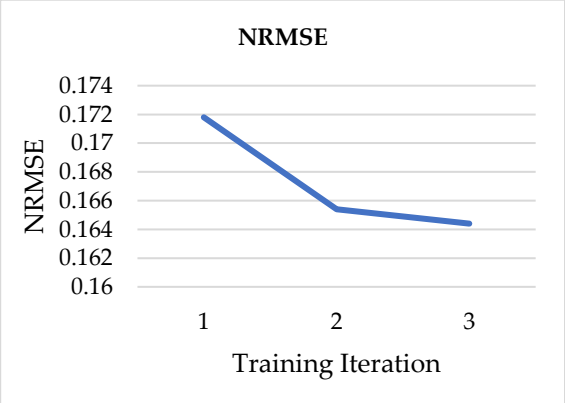

Figure 18 NRMSE for each Training Iteration

The 1st training iteration is the training which was done on the model using the training data set and this 1st training iteration was used to forecast the incoming traffic for the first 24 hours of the testing data. After forecasting was done for the first 24 hours of the testing data, the actual traffic values from the first 24 hours of the testing data were added to the training data and used to retrain the prediction model in a 2nd training iteration. The newly retrained model was in turn used to forecast the traffic from the second 24 hours of the testing data and the model was retrained again in a 3rd training iteration. This 3rd training iteration yielded the lowest NRSME term of 0.164 on a scale of 0 to 1 and this was used to forecast the traffic for the last 24 hours (49 to 72) of the testing data. The forecasted traffic is shown plotted against the actual incoming traffic for 49 to 72 in Figure 19.

Figure 19 shows that the trend of the actual traffic load has been captured by the forecasts. Within this sample, 50% of cases will face over-provisioning of slice resources due to the negative forecast error. 25% of forecasted traffic values are exact matches with the actual received traffic, and 25% of the forecasted traffic values are lower than the actual values and will lead to under-provisioning of slice resources.

The results from running the data from observation period 49 to 72 are presented in detail in Appendix II.

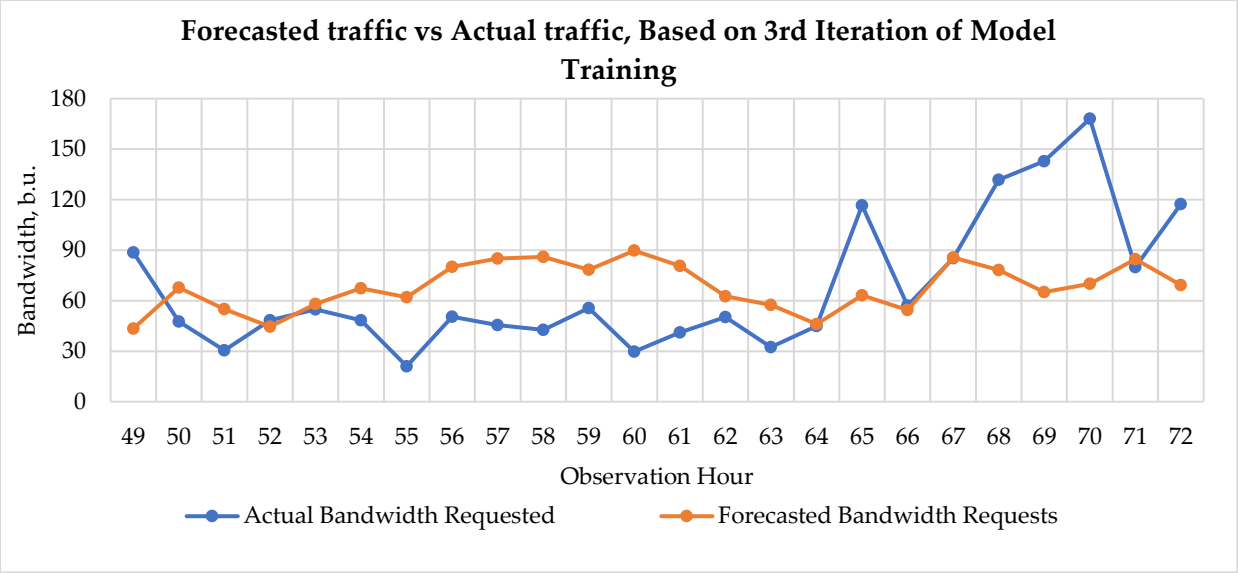

Figure 19 Plot of Forecasted Traffic Bandwidth Against Plot of Actual Traffic Bandwidth per Hour

The blue graph in Figure 19 shows the plot of the actual bandwidth requested and upon observing the levels in the graph, it can be seen that the bandwidth requested in hours 65, 68, 69 and 70 shoot much higher than the bandwidth requested in the rest of the hours. These are high congestion hours and because of this uncharacteristic deviation from the average,

the prediction model does not forecast the expected bandwidth in these hours as accurately as in hours where the bandwidth requests are closer to the average, such as hours 52, 64 and 66.

5.3.2. Evaluating the Performance of Intra-Slice Admission Control Methods With and Without Predictive Analytics

To simulate the admission control algorithms without the application of predictive analytics, we implemented a network slice which was allotted a capacity of 2 bandwidth units (b.u.) and a Servicing Window, W_t of 10 time units (t.u.). This means that the simulated network slice ran for 10 time units and the capacity of the network slice was 2 bandwidth units. This gives the slice a total of 20 RBs to accommodate incoming flow requests. Each batch of flows was processed through this slice using the 3 different admission control algorithms covered in Chapter 4.

The traffic load arriving hourly at the Flow Admission Control module during this simulation was calculated using the formula in equation 5.4 below and the hourly variation in traffic load is shown in Figure 20 below.

$$Load = total\ bandwidth\ requested \times average\ flow\ duration \quad (5.4)$$

Where $total\ bandwidth\ requested$ = sum of bandwidth requested by all flows within the observation period,

$average\ flow\ duration$ = sum of all flow durations divided by the number of flow requests within the observation period.

Based on the graph in Figure 20, given that the bandwidth and servicing windows are kept constant, the 6 most congested hours are 62, 65, 68, 69, 70 and 72, with incoming load requests above 300 RBs.

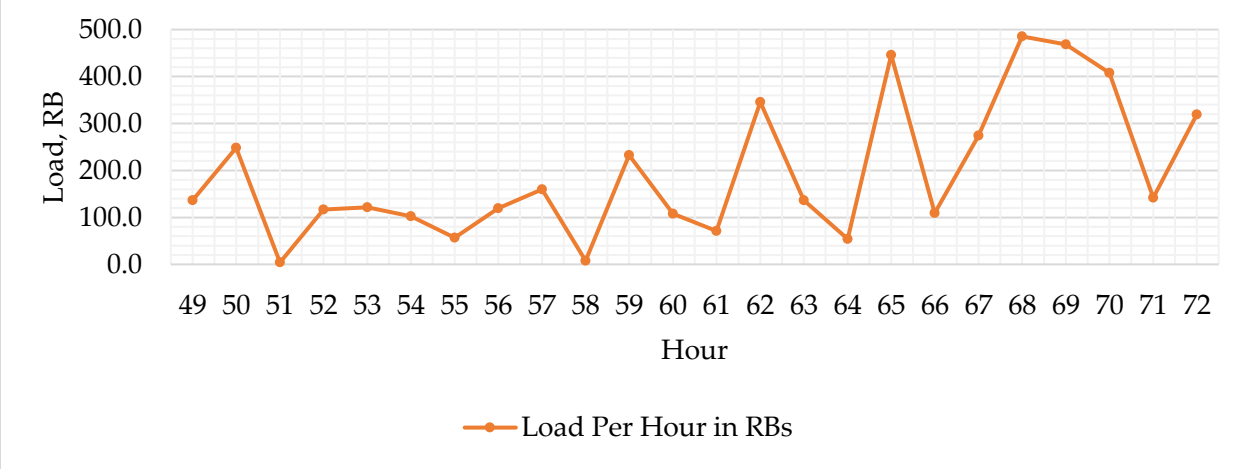

Figure 20 Plot of Hourly Traffic Load for Hours 49 to 72

To simulate the application of predictive analytics to the admission control process, the slice capacity for each servicing window is varied during simulation based on the forecasted traffic values. The length of the servicing window is kept constant at 10 t.u.

The results of the simulations with and without traffic forecasting are compared and presented.

1. **Probability of Blocking, PoB**

The probability of blocking of the 3 admission control methods are shown in Figure 21.

Without the application of predictive analytics, the First-In-First-Out with Queuing algorithm performs better than the other algorithms in terms of probability of blocking. The FIFO algorithm maintains a PoB of less than 0.5, except in the most congested observation periods (65, 68, 70, 72). The worst performance in terms of PoB is presented by the Utility Index algorithm, and this is shown clearly in Figure 21.

The PoB of the Decision Matrix algorithm is better than that of the Utility Index algorithm but is not as good as that of the more rudimentary FIFO algorithm because our dataset has a very small number of high priority flows, and the implementation of a threshold prevents the more frequent normal priority data from using the entire slice bandwidth capacity. This makes admission control using Decision Matrix unsuitable for slices with a low percentage of high priority traffic.

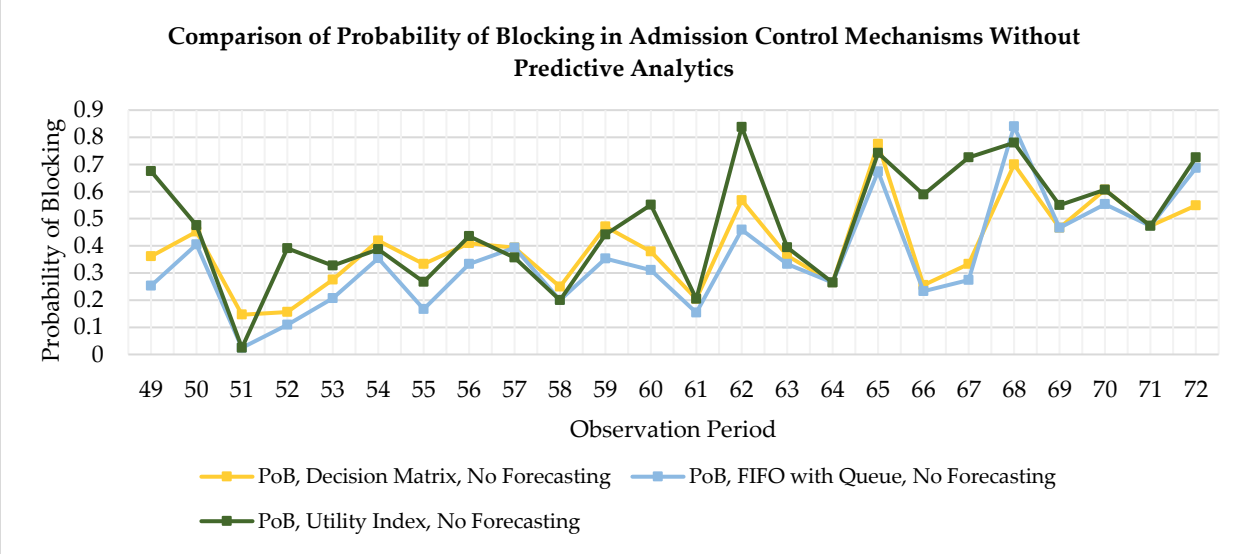

Figure 21 Comparing the Probability of Blocking of Decision Matrix, Utility Index and FIFO Algorithms Without Forecasting

When predictive analytics are used to pre-allocate the slice capacity using forecasted traffic values, there is a marked decrease in the probability of blocking across all three admission

control algorithms. This is shown in Figure 22. With forecasting, the PoB in the more congested hours improves significantly for the FIFO and Decision Matrix algorithms. The PoB performance of the Decision Matrix algorithm with forecasting in congested hours like 62 and 69 is comparable with the PoB performance of the same Decision Matrix algorithm in less congested hours like 63 and 64.

The performance of the Utility Index algorithm, however, does not improve much with the application of traffic forecasting, especially in the congested hours.

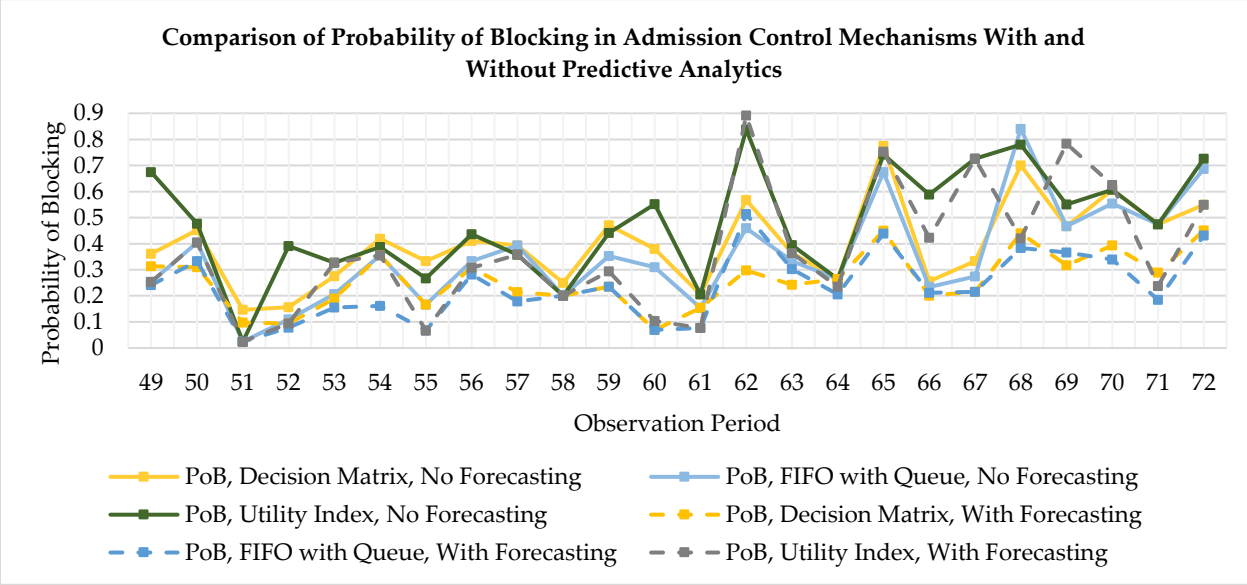

Figure 22 Comparing the Probability of Blocking of Decision Matrix, Utility Index and FIFO Algorithms With and Without Forecasting

2. System Utilisation

Figure 23 shows that the of the three admission control algorithms considered, the Utility Index algorithm is the best-performing algorithm when it comes to system utilisation. The worst-performing algorithm in this case is the Decision Matrix algorithm and again, this is due to the fact that a portion of the total slice capacity is reserved in a threshold for high priority flows, which do not occur frequently in this dataset. The Utility Index algorithm performs best in terms of system utilisation because even though there is a threshold, the first flows to be serviced are those with high utility index (a_i). Based on the formula for a_i, flows with high utility index will demand a higher number of resource blocks (RBs), hence will occupy a large portion of the available slice resources and increase system utilisation.

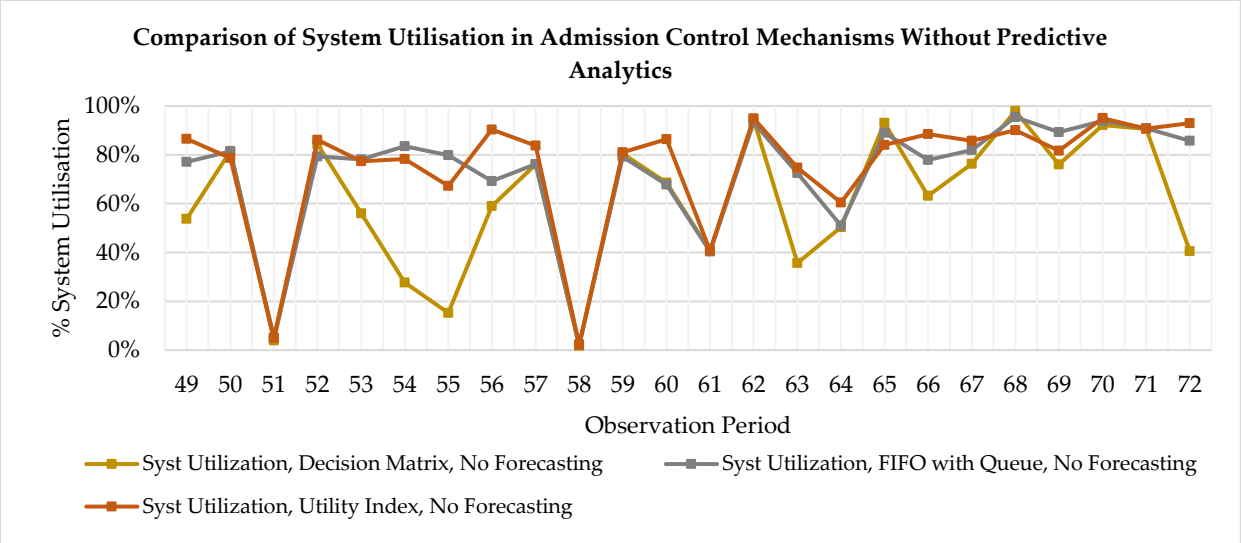

Figure 23 Comparing the System Utilisation of Decision Matrix, Utility Index and FIFO Algorithms Without Forecasting

In the more congested hours such as 62, 65, 68, 70, all three admission control algorithms yield high system utilisation of more than 85% system utilisation. High system utilisation during high congestion is expected because system resources are expected to be used up during congestion. In the congested hour 72 the Decision Matrix algorithm however performed poorly. This is due to the lack of high priority traffic during hour 72 whereas the Decision Matrix algorithm is designed to reserve some bandwidth for high priority traffic. This lack of high priority traffic means that the reserved bandwidth is not being used up.

Another note-worthy result is that all 3 admission control algorithms had poor system utilisation in hours 51 and 58. Referring back to Figure 20, it is clear that these hours do not have heavy traffic load and especially considering the fact that the PoB of hour 51 is less than 0.15 for all three algorithms, it is clear that most of the traffic for that hour could have been sufficiently serviced using fewer slice resources. The unused resources in these low congestion hours were wasted in that slice.

Predictive analytics is used to pre-provision the slice resources based on the forecasted incoming traffic improves the system utilisation of all 3 admission control techniques as shown in Figure 24. However, even with predictive analytics, the Decision Matric algorithm does not perform very well overall in terms of system utilisation. The poor system utilisation of the Decision Matrix algorithm despite the incorporation of predictions is due to the bandwidth reserved in this algorithm for only high priority traffic.

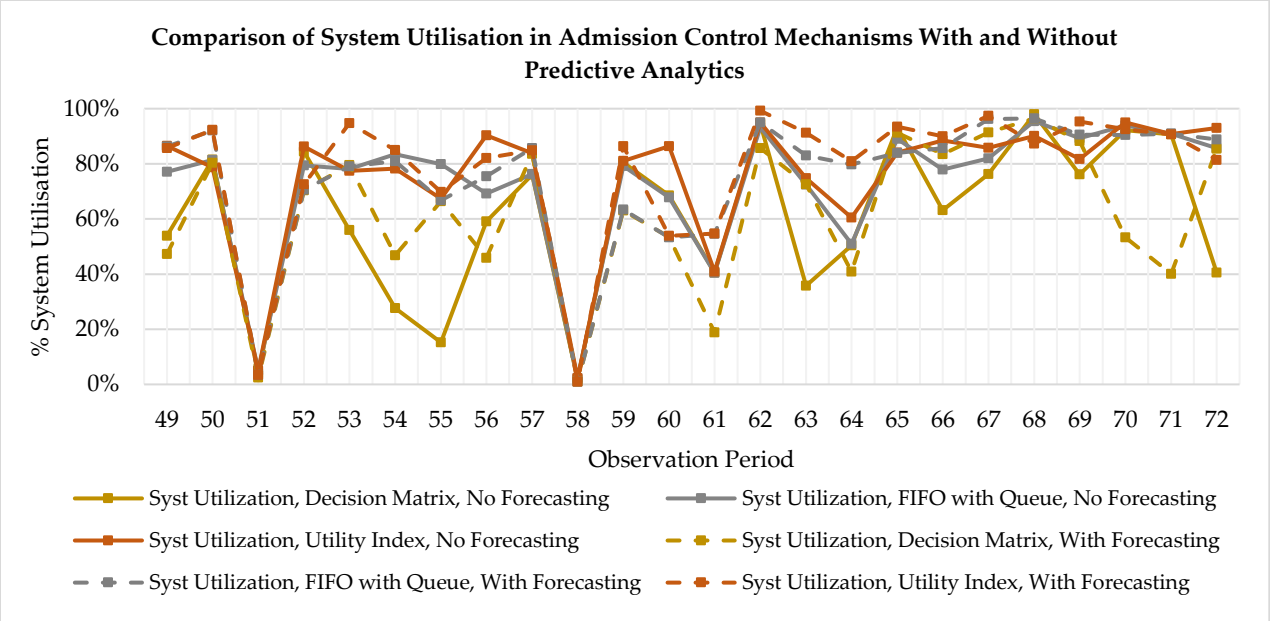

Figure 24 Comparing the System Utilisation of Decision Matrix, Utility Index and FIFO Algorithms With and Without Forecasting

3. Profitability, P

Figure 25 shows the admission control based on the Utility Index to be the most profitable algorithm, closely followed by the FIFO algorithm. This is expected because based on equations 4.11 and 5.3, Utility Index, a_i, is a function of profitability, P and the Utility Index algorithm services the highest utility indices first.

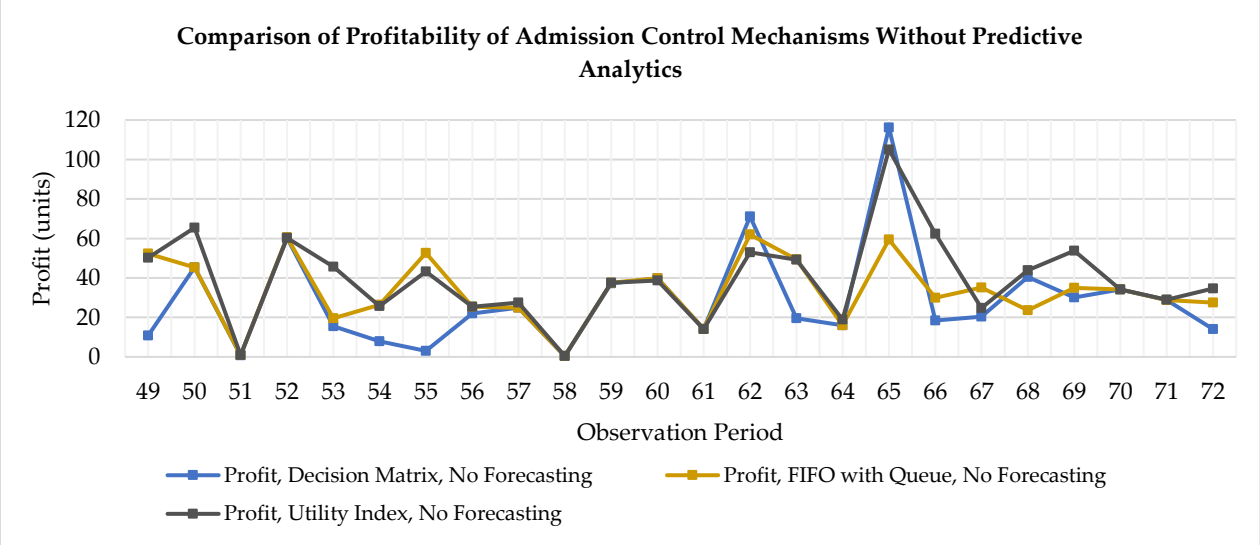

Figure 25 Comparing the Profitability of Decision Matrix, Utility Index and FIFO Algorithms Without Forecasting

In the less-congested observation periods, the profitability of the Decision Matrix algorithm is less than that of the FIFO and Utility Index algorithms but outperforms the others in the high congestion hours like 62 and 65. This is because in low congestion hours, some of the resources in the Decision Matrix go unused due to the fact that bandwidth has been reserved for high priority traffic and there is little high priority traffic to use up these reserves and generate profits. However, high congestion hours present more high priority traffic to use up the bandwidth reserves in the Decision Matrix algorithm, thus generating profits.

The Decision Matrix algorithm also performs on par with the other algorithms in terms of profits in some other congested hours such as 68 and 70.

The incorporation of traffic forecasting for pre-emptive slice resource results in a marked increase in the profitability of all three algorithms as show in Figure 26. This is because traffic forecasting allows for more tailored provisioning of slice resources where resources provided are closer to the resources demanded.

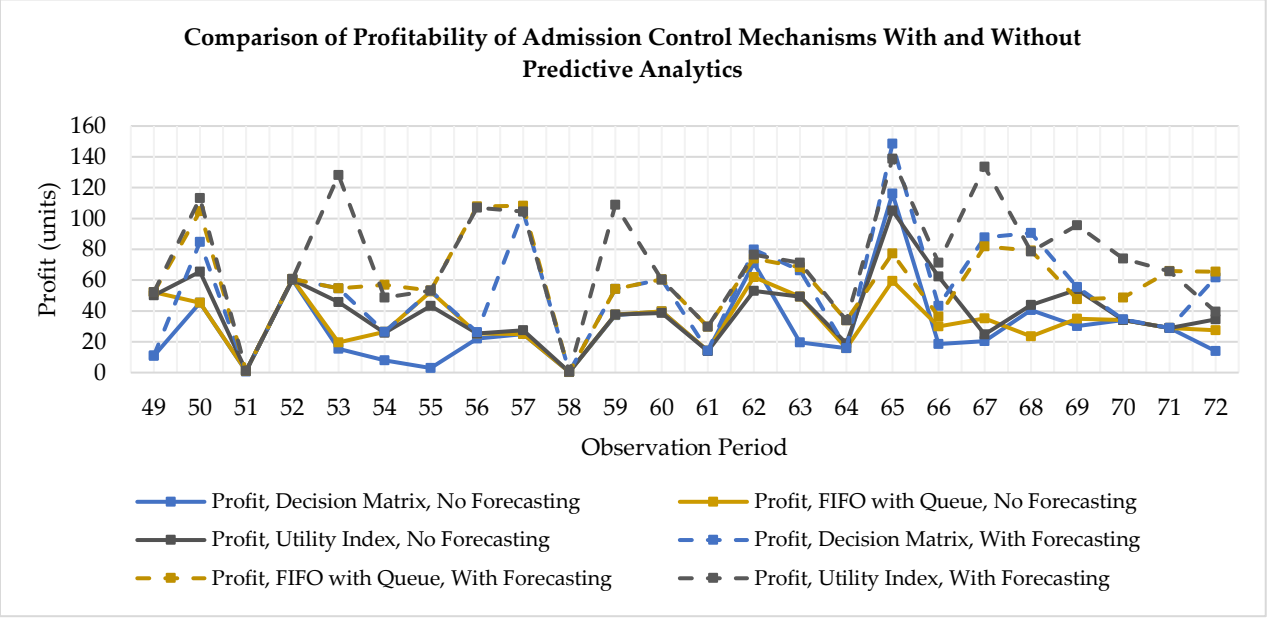

Figure 26 Comparing the Profitability of Decision Matrix, Utility Index and FIFO Algorithms With and Without Forecasting

The raw data (results) from these simulations is included in Appendix III and the graphs presented in this chapter are based on those results.

5.4. Discussion of Results

Based on the comparisons in section 5.3 between the Decision Matrix algorithm, the Utility Index algorithm and the FIFO with Queue algorithm, the Utility Index algorithm performed best overall in terms of system utilisation and profitability. This is explained by the fact that the Utility Index algorithm is designed to first fit the flows with the most utility and profit into the resources available in a slice. These high utility flows are both bandwidth-intensive and time-intensive and use up a good portion of the available slice resources. For these same reasons, the Utility Index algorithm under-performed in terms of the probability of blocking. Because few resource-intensive flows could be serviced by this algorithm, more flows were being blocked compared to the other two algorithms.

Since the Decision Matrix algorithm was designed to prioritise high priority traffic classes, implementing this algorithm on a campus network with a very small proportion of high priority traffic does not take advantage of this prioritisation feature. There are not enough high priority flows to use up the reserved bandwidth, thus that portion of the slice capacity goes unused, and this is seen in the low system utilisation presented by the Decision Matrix algorithm. It is wasteful of system resources in a network that does not have high priority traffic to use up the reserved bandwidth.

The legacy FIFO algorithm remains a robust choice as it performs well in terms of probability of blocking and system utilisation, but it is the least profitable algorithm especially in the more congested observation periods. This can be explained by the fact that the FIFO algorithm does not employ any form of discernment in the process of admission control. It is simply a first-come-first-serve system with no sophisticated policies to maximise profit, as opposed to the Decision Matrix algorithm which prioritises more expensive high priority traffic, or the Utility Index algorithm which prioritises high profit and high utility traffic.

This study also set out to investigate the effect of incorporating predictive analytics into the resource allocation and admission control process and the results show that the employment of traffic predictions for pre-emptive capacity adjustment yields an overall improvement in the performance of all admission control methods. The performance of the Decision Matrix algorithm, the Utility Index algorithm and the legacy FIFO algorithm improved in terms of probability of blocking, system utilisation and profitability. However, the accuracy of the prediction techniques must be increased to avoid wasting network resources by overprovisioning slice resources.

The results also show an improvement in the accuracy of the prediction model when the historical data is updated every 24 hours and the model retrained with the data update. As evidenced by the downward trend in the error graphs in Figure 17 and Figure 18, the prediction model should be adaptive (retrained periodically) in order to keep the prediction

model updated with the new trends in network data and yield better performance in the prediction model.

5.5. Chapter Summary

This chapter presented and defined the performance metrics used for the evaluation of the admission control algorithms. The probability of blocking, system utilisation and the profitability of the algorithms were the performance metrics of choice. The simulation parameters were also defined in this chapter. The results presented in this chapter also show the accuracy of the prediction model used and this accuracy is represented by the error term. The results go further to show a comparison between the performances of the three admission control algorithms discussed (Decision Matrix, Utility Index, and FIFO with Queue algorithms). The results also show a comparison of the performances of these three algorithms when the prediction model has been used to determine the future incoming slice requests. The results presented in this chapter are also discussed extensively.

Chapter 6: Conclusion and Recommendations

6.1. Conclusion

Admission control is one of the fundamental radio resource management functions and new approaches must be developed in tandem with the evolution of mobile communication networks. This work presented the state-of-the-art in admission control approaches and proposed two intra-slice admission algorithms namely the Decision Matrix algorithm and the Utility Index algorithm. These algorithms employ queueing as a method of congestion control. This work implemented and simulated the two proposed algorithms and benchmarked them against a legacy FIFO admission control algorithm.

This study was carried out to answer the research questions outlined in Chapter 1 section 1.3, which are reiterated below:

i. How can admission control methods be designed for sliced networks and evaluated?
ii. How can a real network dataset be used for the simulation of admission control?
iii. How can a prediction model be developed and trained for the forecasting of network traffic?
iv. How does the integration of predictive analytics into the admission control and resource allocation process impact the performance of admission control algorithms?

Through the completion of the tasks involved in this work, the research questions were all addressed and answered.

Admission control algorithms should be designed with specific goals in mind. These goals help determine the main features of each admission control algorithm. In this work, the main feature of the Decision Matrix algorithm is the prioritisation of high priority network traffic, and the main feature of the Utility Index algorithm is the maximisation of operator profits. The evaluation of admission control methods could be done using widely-used performance metrics such as the probability of blocking, system utilisation and the profitability of the methods.

By implementing and training a traffic forecasting model based on Holt-Winter's Exponential Smoothing, this work shows how a prediction model can be developed and trained for the forecasting of network traffic. The best method for training the forecasting model turned out to be the use of a non-linear optimisation tool where the objective was set to minimize the error term. This work found that the training of a traffic forecasting model should be done using training data from the network on which the forecasting model is to be used. This is because the behaviour of a prediction model is governed by the data with which it is trained.

The traffic forecasting model was used to predict the total demand for bandwidth every hour and this forecasted information was used to pre-emptively allot resources to a network slice in anticipation of incoming network traffic. This study also investigates and reports on the performance of the admission control algorithms with and without the use of traffic forecasting and it was found that the use of traffic forecasting improves the overall performance of the admission control algorithms.

To increase the accuracy of the traffic forecasting model, we made the prediction model adaptive by feeding back new historical data into the traffic model every 24 hours and retraining the model with the new data. With every iteration of incorporation of new data and training and the forecasting model was found to become more accurate.

We used a real network dataset containing data collected from a campus network to train and then test the HWES forecasting model, and the same real network data was used to test and evaluate the three admission control algorithms considered in this study. In order for the data to be exploited in this manner, data pre-processing and data mining had to be carried out on the dataset.

The three admission control algorithms (Decision Matrix algorithm, Utility Index algorithm, and FIFO with Queueing algorithm) were compared based on three main performance metrics namely probability of blocking, system utilisation and profitability. The respective strengths and weaknesses of the three algorithms considered were discussed. The Decision Matrix algorithm was found to be good for use in a network with a large proportion of high priority traffic, the Utility Index algorithm was found to be the best for prioritising operator profits and the FIFO algorithm was found to perform well in simple networks where prioritisation of certain traffic classes is not a prime feature.

The experiments carried out in this study prove that the use of traffic forecasting to pre-emptively allocate resources to slices in anticipation of incoming traffic greatly improves the overall performance of the admission control algorithms. The application of predictive analytics to admission control resulted in reduced probability of blocking and increased system utilisation and increased profitability of all the admission control algorithms considered.

This work designed and implemented a system model which operated in a midpoint between dynamic slice resource allocation and true slice isolation through the implementation of a servicing window during which slices are not degraded.

Overall, this work fills the gap identified in the literature review presented in Chapter 2 by proposing new admission control algorithms fitted with predictive analytics and implemented on a system which allows for network slice isolation as well as dynamic slice

resource allocation. This is a novel solution in the field of admission control in sliced networks and it considers both the objectives of network slicing and individual user objectives. This work also proves that the use of traffic forecasting for pre-emptive resource allocation in sliced networks elevates the general performance of admission control algorithms.

6.2. Recommendations

Based on the results gotten in Chapter 5, the following recommendations can be made to network infrastructure owners and operators:

- The Decision Matrix algorithm is ideal for networks with a large proportion of high priority traffic which needs prioritisation. This is because the Decision Matrix is designed such each admission decision prioritises incoming high priority traffic and a pool of resources is reserved for high priority traffic.
- The Utility Index algorithm is ideal for networks where operator profits are priority and efficient system utilisation. This is because this algorithm pushes traffic with high utility ahead of traffic with low utility.
- In systems where resource provisioning is done, the use of predictive analytics to forecast the resource needs of the system is advised as this leads to better overall performance of the system and efficient use of the available resources.
- When employing the use of a prediction model in real time applications, the prediction model should be made adaptive by periodic retraining using newly-acquired data. This keeps the prediction model updated with the new trends in network data and yields better performance in the prediction model.
- The system model implemented in this work should be considered for networks where a trade-off point is needed between slice isolation and dynamic resource allocation. This system model ensures slice isolation while allowing for dynamic slice resource allocation by enforcing a servicing window during which slice resources cannot be degraded but at the end of which slice resource allocation is revisited and modified.

6.3. Further Work

The time and resources available for this work only permitted the current scope of this work but further work can be done to build upon the work done in this dissertation.

Further work can be done to develop an algorithm which implements a trade-off between the Utility Index algorithm which maximises profitability and the FIFO algorithm which presents a low probability of blocking.

While it was found that predictions generally improved the performance of the admission control algorithms, improving the prediction accuracy is important to ensure that network resources are not wasted. More sophisticated predictive methods, such as machine learning methods as outlined in Chapter 2, section 2.2 can be used to fine-tune the forecasting process and improve prediction accuracy.

Other aspects of mobile communication traffic such as mobility can be forecasted and factored into the admission control decision for an overall improvement in network performance.

References

[1] CISCO, 'Cisco Annual Internet Report (2018–2023)', 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf.

[2] NGMN Alliance, '5G White Paper', *A Deliv. by NGMN Alliance*, p. 124, 2015, [Online]. Available: https://www.ngmn.org/uploads/media/NGMN_5G_White_Paper_V1_0_01.pdf.

[3] 3GPP, 'Feasibility Study on New Services and Markets Technology Enablers; Stage 1 (Release 14)', 2016.

[4] NGMN Alliance, 'Description of Network Slicing Concept', 2016. [Online]. Available: https://www.ngmn.org/fileadmin/user_upload/161010_NGMN_Network_Slicing_framework_v1.0.8.pdf.

[5] 3GPP, 'General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access (Release 17)', 2021. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=849.

[6] Frédéric Firmin, 'The Evolved Packet Core', *3GPP MCC*. https://www.3gpp.org/technologies/keywords-acronyms/100-the-evolved-packet-core (accessed May 22, 2022).

[7] C. Cárdenas, M. Gagnaire, V. López, and J. Aracil, 'Performance evaluation of the flow-aware networking (FAN) architecture under Grid environment', *NOMS 2008 - IEEE/IFIP Netw. Oper. Manag. Symp. Pervasive Manag. Ubiquitous Networks Serv.*, pp. 481–487, 2008, doi: 10.1109/NOMS.2008.4575171.

[8] I. Bisio, M. Marchese, and M. Mongelli, 'Resource Allocation over a GRID Military Network', *Mil. Commun.*, pp. 6.1-6.20, 2006, Accessed: May 22, 2022. [Online]. Available: https://moam.info/resource-allocation-over-a-grid-military-network_5c53bf57097c47d27f8b459e.html.

[9] GSM Association, 'Definition of Quality of Service parameters and their computation', Aug. 2017. Accessed: Sep. 09, 2022. [Online]. Available: https://www.gsma.com/newsroom/wp-content/uploads/IR.42-v8.0.pdf.

[10] E. Hossain, L. Bao Le, and D. Niyato, *Radio Resource Management in Multi-Tier Cellular Wireless Networks*, 1st ed. John Wiley & Sons, Inc., 2014.

[11] M. O. Ojijo and O. E. Falowo, 'A Survey on Slice admission Control Strategies and Optimization Schemes in 5G Network', *IEEE Access*, vol. 8, no. January, pp. 14977–

14990, 2020, doi: 10.1109/aCCESS.2020.2967626.

[12] M. M. Badawy and S. A. AlQahtani, 'Adaptive Terminal-Modality-Based Joint Call Admission Control for Heterogeneous Cellular Networks', *Int. J. Commun. Netw. Syst. Sci.*, vol. 06, no. 09, pp. 395–406, 2013, doi: 10.4236/ijcns.2013.69043.

[13] O. E. Falowo and H. A. Chan, 'Joint call admission control algorithms: Requirements, approaches, and design considerations', *Comput. Commun.*, vol. 31, no. 6, pp. 1200–1217, Apr. 2008, doi: 10.1016/j.comcom.2007.10.044.

[14] ETSI, 'Network Functions Virtualisation - White Paper on NFV priorities for 5G', Bilbao, Spain, Feb. 2017. Accessed: Nov. 30, 2020. [Online]. Available: http://portal.etsi.org/NFV/NFV_White_Paper_5G.pdf.

[15] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, '5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges', *Comput. Networks*, vol. 167, Feb. 2020, doi: 10.1016/j.comnet.2019.106984.

[16] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, 'Network slicing and softwarization: A survey on principles, enabling technologies, and solutions', *IEEE Commun. Surv. Tutorials*, vol. 20, no. 3, pp. 2429–2453, Mar. 2018, doi: 10.1109/COMST.2018.2815638.

[17] B. Khodapanah, A. Awada, I. Viering, J. Francis, M. Simsek, and G. P. Fettweis, 'Radio Resource Management in context of Network Slicing: What is Missing in Existing Mechanisms?', *IEEE Wirel. Commun. Netw. Conf. WCNC*, vol. 2019-April, 2019, doi: 10.1109/WCNC.2019.8885942.

[18] P. Caballero, A. Banchs, G. De Veciana, X. Costa-Perez, and A. Azcorra, 'Network slicing for guaranteed rate services: Admission control and resource allocation games', *IEEE Trans. Wirel. Commun.*, vol. 17, no. 10, pp. 6419–6432, 2018, doi: 10.1109/TWC.2018.2859918.

[19] B. Han *et al.*, 'Admission and congestion control for 5G network slicing', *arXiv*, 2018.

[20] B. Han, V. Sciancalepore, D. Feng, X. Costa-Perez, and H. D. Schotten, 'A utility-driven multi-queue admission control solution for network slicing', *arXiv*, 2019.

[21] K. Ben Ali and F. Zarai, 'Adaptive Radio Resource Management Scheme in 5G networks support for IoT Applications', in *2019 6th International Conference on Internet of Things: Systems, Management and Security, IOTSMS 2019*, Oct. 2019, pp. 270–276, doi: 10.1109/IOTSMS48152.2019.8939240.

[22] S. Kumar, M. K. Pandey, A. Nath, K. Subbiah, and M. K. Singh, 'Comparative study on machine learning techniques in predicting the QoS-values for web-services recommendations', *Int. Conf. Comput. Commun. Autom. ICCCA 2015*, pp. 161–167, Jul. 2015, doi: 10.1109/CCAA.2015.7148398.

[23] C. Nyce, 'Predictive Analytics White Paper', 2007. Accessed: May 07, 2021. [Online]. Available: www.aicpcu.org.

[24] W. W. Eckerson, 'TDWI Best practices Report: Predictive Analytics, Extending the Value of Your Data Warehousing Investment', 2007. [Online]. Available: http://download.101com.com/pub/tdwi/Files/PA_Report_Q107_F.pdf.

[25] M. Kamber and J. Han, *Data Mining : Concepts and Techniques*, 2nd ed. Morgan Kaufmann Publishers, 2006.

[26] S. Wu, W. Mao, T. Hong, C. Liu, and M. Kadoch, 'Compressed sensing based traffic prediction for 5G HetNet IoT Video streaming', in *2019 15th International Wireless Communications and Mobile Computing Conference, IWCMC 2019*, Jun. 2019, pp. 1901–1906, doi: 10.1109/IWCMC.2019.8766662.

[27] A. R. Gray and S. G. MacDonell, 'A comparison of techniques for developing predictive models of software metrics', *Inf. Softw. Technol.*, vol. 39, no. 6, pp. 425–437, Jan. 1997, doi: 10.1016/S0950-5849(96)00006-7.

[28] M. F. Zhani, H. Elbiaze, and F. Kamoun, 'Analysis and Prediction of Real Network Traffic', *Artic. J. Networks*, Nov. 2009, doi: 10.4304/jnw.4.9.855-865.

[29] L. Yi, X. Ke, and S. Junde, 'Research on forecasting and early-warning methods', in *Proceedings - IEEE 9th International Conference on Mobile Ad-Hoc and Sensor Networks, MSN 2013*, 2013, pp. 570–576, doi: 10.1109/MSN.2013.57.

[30] M. Kuhn and K. Johnson, *Applied Predictive Modeling*. 2013.

[31] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis*, 6th ed. John Wiley & Sons, Inc., 2021.

[32] C. W. Ostrom, *Time Series Analysis: Regression Techniques*, 2nd ed., vol. 9. Sage Publications, Inc, 1990.

[33] C. Chatfield and M. Yar, 'Holt-Winters Forecasting: Some Practical Issues', *Stat.*, vol. 37, no. 2, p. 129, 1988, doi: 10.2307/2348687.

[34] P. Newbold and C. W. J. Granger, 'Experience with Forecasting Univariate Time Series and the Combination of Forecasts', *J. R. Stat. Soc. Ser. A*, vol. 137, no. 2, p. 131, 1974, doi: 10.2307/2344546.

[35] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*, 2nd ed. Cambridge, MA: The MIT Press, 2018.

[36] S. J. Nawaz, S. K. Sharma, S. Wyne, M. N. Patwary, and M. Asaduzzaman, 'Quantum Machine Learning for 6G Communication Networks: State-of-the-Art and Vision for the Future', *IEEE Access*, vol. 7, no. Ml, pp. 46317–46350, 2019, doi: 10.1109/ACCESS.2019.2909490.

[37] M. F. Iqbal and L. K. John, 'Power and performance analysis of network traffic prediction techniques', in *ISPASS 2012 - IEEE International Symposium on Performance Analysis of Systems and Software*, 2012, pp. 112–113, doi: 10.1109/ISPASS.2012.6189212.

[38] D. Tikunov and T. Nishimura, 'Traffic prediction for mobile network using Holt-Winter's exponential smoothing', in *2007 15th International Conference on Software, Telecommunications and Computer Networks, SoftCOM 2007*, 2007, pp. 310–314, doi: 10.1109/SOFTCOM.2007.4446113.

[39] I. Alawe, A. Ksentini, Y. Hadjadj-Aoul, and P. Bertin, 'Improving Traffic Forecasting for 5G Core Network Scalability: A Machine Learning Approach', *IEEE Netw.*, vol. 32, no. 6, pp. 42–49, Nov. 2018, doi: 10.1109/MNET.2018.1800104.

[40] L. Nie *et al.*, 'A Reinforcement Learning-Based Network Traffic Prediction Mechanism in Intelligent Internet of Things', *IEEE Trans. Ind. Informatics*, vol. 17, no. 3, pp. 2169–2180, Mar. 2021, doi: 10.1109/TII.2020.3004232.

[41] M. Faisal Iqbal, M. Zahid, D. Habib, and L. Kurian John, 'Efficient Prediction of Network Traffic for Real-Time Applications', 2019, doi: 10.1155/2019/4067135.

[42] F. A. Rouai and M. B. Ahmed, 'A new approach for fuzzy neural network weight initialization', in *Proceedings of the International Joint Conference on Neural Networks*, 2001, vol. 2, pp. 1322–1327, doi: 10.1109/ijcnn.2001.939553.

[43] G. Rangarajan and M. Ding, *Processes with Long-Range Correlations: Theory and Applications*. Berlin: Springer, 2003.

[44] V. Sciancalepore, K. Samdanis, X. Costa-Perez, D. Bega, M. Gramaglia, and A. Banchs, 'Mobile traffic forecasting for maximizing 5G network slicing resource utilization', Oct. 2017, doi: 10.1109/INFOCOM.2017.8057230.

[45] T. V. K. Buyakar, H. Agarwal, B. R. Tamma, and A. A. Franklin, 'Resource Allocation with Admission Control for GBR and Delay QoS in 5G Network Slices', *2020 Int. Conf. Commun. Syst. NETworkS, COMSNETS 2020*, pp. 213–220, Jan. 2020, doi: 10.1109/COMSNETS48256.2020.9027310.

[46] K. Samdanis, X. Costa-Perez, and V. Sciancalepore, 'From network sharing to multi-tenancy: The 5G network slice broker', *IEEE Communications Magazine*, vol. 54, no. 7. Institute of Electrical and Electronics Engineers Inc., pp. 32–39, Jul. 01, 2016, doi: 10.1109/MCOM.2016.7514161.

[47] G. Tseliou, K. Samdanis, F. Adelantado, X. C. Perez, and C. Verikoukis, 'A capacity broker architecture and framework for multi-tenant support in LTE-A networks', Jul. 2016, doi: 10.1109/ICC.2016.7511042.

[48] R. J. Sebastián, 'Labeled Network Traffic flows - 141 Applications | Kaggle', *Kaggle*, 2019. https://www.kaggle.com/jsrojas/labeled-network-traffic-flows-114-applications

(accessed Nov. 22, 2021).

[49] J. S. Rojas, A. Pekar, A. Rendon, and J. C. Corrales, 'Smart User Consumption Profiling: Incremental Learning-Based OTT Service Degradation', *IEEE Access*, vol. 8, pp. 207426–207442, 2020, doi: 10.1109/ACCESS.2020.3037971.

[50] '5G Campus Networks: Measurement Traces | IEEE DataPort'. https://ieee-dataport.org/open-access/5g-campus-networks-measurement-traces#files (accessed Feb. 14, 2022).

[51] 3GPP, 'Technical Specification 23.203, Policy and charging control architecture', 2021. [Online]. Available: http://www.3gpp.org.

[52] Appfluence, 'The Eisenhower Method: What is it, and can it help you prioritize effectively?' https://appfluence.com/productivity/what-is-the-eisenhower-method/ (accessed Feb. 12, 2022).

[53] A. H. Mfoundoum, M. Tchindjang, V. Mrondoum, and I. Makouet, 'Eisenhower matrix * Saaty AHP = Strong actions prioritization? Theoretical literature and lessons drawn from empirical evidences', *IAETSD J. Adv. Res. Appl. Sci.*, 2019.

[54] P. M. Ngufor, 'Pre-processed Data Selection PMN.xlsx - Google Sheets'. https://docs.google.com/spreadsheets/d/1SDUFSf3Zqw-hRZhVU7KrcZ8AgOHLCzH0/edit#gid=715847007 (accessed Sep. 09, 2022).

[55] P. M. Ngufor, 'Final Simulation Results.xlsx - Google Sheets'. https://docs.google.com/spreadsheets/d/1myXZq3CNVY2af2xVbbU-WvCHSC4Msc48/edit#gid=1111362477 (accessed Sep. 09, 2022).

Appendix I

Description of relevant data fields used in this work from the dataset [48]

Data field as named in dataset [48]	**Description of data field**	**Application of data field in this work.**
flow_key	Flow identifier	Used to identify individual flows
pktTotalCount	Total number of packets in both directions	Calculate the average packet size per flow
octetTotalCount	Total number of bytes exchanged in both directions focusing on IP payload only	Calculate the average packet size per flow
avg_ps	Average packet size on the flow in both directions	Used to estimate the bandwidth required by each flow
flowStart	Flow start time in seconds	Used to batch the incoming flows into observation periods and determine the order in which flows should be fed into the simulation
flowDuration	Total flow duration in seconds	Used to determine how long a flow should be held in the simulation
category	Category of the communication	Used to determine the QCI of each flow

Appendix II

A. Results from 3 training iterations.

	Base Parameters	Adaptive Parameters (+24 hours)	Adaptive Parameters (+48 hours)
alpha, α	0.377	0.302	0.350097984
beta, β	4.991×10^{-9}	0.005	0.005251386
gamma, γ	3.691×10^{-8}	1.509 × 10-7	6.85296E-09
RMSE	42.68	41.08	40.85
NRMSE	0.1718	0.1654	0.1644

B. Simulation results from running the network data through the trained Holt Winters Exponential Smoothing Prediction Model trained in the 3rd iteration.

Observation Period	**49**	**50**	**51**	**52**	**53**	**54**
Actual Bandwidth Requested (b.u.)	88.62	47.55	30.53	48.44	54.76	48.35
Forecasted Bandwidth Requested (b.u.)	43.50	67.84	54.99	44.57	57.98	67.35
Observation Period	**55**	**56**	**57**	**58**	**59**	**60**
Actual Bandwidth Requested (b.u.)	21.04	50.53	45.46	42.75	55.66	29.79
Forecasted Bandwidth Requested (b.u.)	62.13	80.18	85.03	85.92	78.36	89.72
Observation Period	**61**	**62**	**63**	**64**	**65**	**66**
Actual Bandwidth Requested (b.u.)	41.08	50.20	32.37	44.99	116.55	57.19
Forecasted Bandwidth Requested (b.u.)	80.73	62.64	57.49	46.13	63.16	54.36
Observation Period	**67**	**68**	**69**	**70**	**71**	**72**
Actual Bandwidth Requested (b.u.)	85.31	131.7	142.87	168.06	79.92	117.3
Forecasted Bandwidth Requested (b.u.)	85.59	78.24	65.11	69.97	84.59	69.27

Appendix III

Simulation results from running 3 different Admission Control algorithms with and without the application of predictive analytics (traffic forecasting).

The unit for capacity, threshold and forecasted incoming total bandwidth is bandwidth units, b.u. The unit for window (servicing window) is time units, t.u.

A. Without Forecasting

WITHOUT FORECASTING													
	DECISION MATRIX WITHOUT PREDICTION												
Observation	49	50	51	52	53	54	55	56	57	58	59	60	61
Capacity	2	2	2	2	2	2	2	2	2	2	2	2	2
Threshold	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4
Window	10	10	10	10	10	10	10	10	10	10	10	10	10
Requests	83	42	41	64	58	31	30	39	28	20	34	29	39
Queued	48	20	22	45	36	9	13	15	11	5	13	15	23
Rejected	30	15	6	7	16	12	10	16	11	5	15	11	7
Q_Unprocessed	0	4	0	3	0	1	0	0	0	0	1	0	1
PoB, Decision Matrix	0.36	0.45	0.15	0.16	0.28	0.42	0.33	0.41	0.39	0.25	0.47	0.38	0.21
Syst Utilisation, Decision Matrix	53.82 %	81.52 %	3.99 %	84.40 %	56.00 %	27.71 %	15.25 %	59.06 %	76.05 %	1.79 %	80.49 %	68.55 %	40.55 %
Profit, Decision Matrix	10.76	45.26	0.80	60.33	15.45	7.97	3.05	21.98	24.91	0.36	37.31	39.80	14.06
	FIFO, QUEUE WITHOUT PREDICTION												
Observation	49	50	51	52	53	54	55	56	57	58	59	60	61
Capacity	2	2	2	2	2	2	2	2	2	2	2	2	2
Threshold	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Window	10	10	10	10	10	10	10	10	10	10	10	10	10
Requests	83	42	41	64	58	31	30	39	28	20	34	29	39
Queued	77	38	35	56	52	22	20	33	24	12	26	24	37
Rejected	0	0	0	0	0	0	0	0	0	0	0	0	0
Q_Unprocessed	21	17	1	7	12	11	5	13	11	4	12	9	6
PoB, FIFO with Queue	0.25	0.40	0.02	0.11	0.21	0.35	0.17	0.33	0.39	0.20	0.35	0.31	0.15
Syst Utilisation, FIFO with Queue	77.09 %	81.42 %	4.99 %	79.40 %	78.09 %	83.48 %	79.87 %	69.22 %	76.24 %	2.22 %	79.18 %	67.85 %	40.37 %

Profit, FIFO with Queue	52.25	45.35	1.00	60.64	19.51	26.40	52.66	25.36	24.91	0.44	37.63	39.81	14.19
	UTILITY INDEX WITH PREDICTION												
Observation	49	50	51	52	53	54	55	56	57	58	59	60	61
Capacity	2	2	2	2	2	2	2	2	2	2	2	2	2
Threshold	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4
Window	10	10	10	10	10	10	10	10	10	10	10	10	10
Requests	83	42	41	64	58	31	30	39	28	20	34	29	39
Queued	79	37	38	59	53	26	25	36	24	13	28	25	30
Rejected	0	0	0	0	0	0	0	0	0	0	0	0	0
Q_Unprocessed	56	20	1	25	19	12	8	17	10	4	15	16	8
PoB, Utility Index	0.67	0.48	0.02	0.39	0.33	0.39	0.27	0.44	0.36	0.20	0.44	0.55	0.21
Syst Utilisation, Utility Index	86.50 %	78.69 %	4.99 %	86.20 %	77.43 %	78.26 %	67.33 %	90.36 %	83.76 %	2.22 %	81.08 %	86.37 %	40.86 %
Profit, Utility Index	50.18	65.42	1.00	60.31	45.64	25.76	43.29	25.35	27.53	0.44	37.60	38.70	14.06

WITHOUT FORECASTING											
		DECISION MATRIX WITHOUT PREDICTION									
Observation	62	63	64	65	66	67	68	69	70	71	72
Capacity	2	2	2	2	2	2	2	2	2	2	2
Threshold	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4
Window	10	10	10	10	10	10	10	10	10	10	10
Requests	37	33	34	89	90	51	50	60	56	38	51
Queued	17	18	17	56	61	26	25	30	19	21	17
Rejected	13	12	9	27	23	17	21	27	27	13	28
Q_Unprocessed	8	0	0	42	0	0	14	1	7	5	0
PoB, Decision Matrix	0.57	0.36	0.26	0.78	0.26	0.33	0.70	0.47	0.61	0.47	0.55
Syst Utilisation, Decision Matrix	94.48%	35.74%	50.36%	93.07%	63.18%	76.26%	98.00%	76.11%	92.13%	90.62%	40.50%
Profit, Decision Matrix	71.06	19.58	15.93	116.17	18.45	20.40	40.55	30.07	34.16	28.85	14.06
		FIFO, QUEUE WITHOUT PREDICTION									
Observation	62	63	64	65	66	67	68	69	70	71	72
Capacity	2	2	2	2	2	2	2	2	2	2	2
Threshold	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Window	10	10	10	10	10	10	10	10	10	10	10
Requests	37	33	34	89	90	51	50	60	56	38	51
Queued	31	30	25	85	84	45	49	58	51	33	46

Rejected	0	0	0	0	0	0	0	0	0	0	0
Q_Unprocessed	17	11	9	60	21	14	42	28	31	18	35
PoB, FIFO with Queue	0.46	0.33	0.26	0.67	0.23	0.27	0.84	0.47	0.55	0.47	0.69
Syst Utilisation, FIFO with Queue	93.20%	72.50%	51.02%	89.08%	77.90%	82.01%	95.48%	89.24%	93.92%	90.92%	85.72%
Profit, FIFO with Queue	61.98	49.32	15.93	59.45	30.01	35.15	23.57	34.91	34.18	28.85	27.54
		UTILITY INDEX WITH PREDICTION									
Observation	62	63	64	65	66	67	68	69	70	71	72
Capacity	2	2	2	2	2	2	2	2	2	2	2
Threshold	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4
Window	10	10	10	10	10	10	10	10	10	10	10
Requests	37	33	34	89	90	51	50	60	56	38	51
Queued	33	29	29	84	87	47	45	57	51	34	47
Rejected	0	0	0	0	0	0	0	0	0	0	0
Q_Unprocessed	31	13	9	66	53	37	39	33	34	18	37
PoB, Utility Index	0.84	0.39	0.26	0.74	0.59	0.73	0.78	0.55	0.61	0.47	0.73
Syst Utilisation, Utility Index	94.92%	74.73%	60.46%	84.02%	88.45%	85.80%	90.03%	81.73%	95.00%	90.73%	92.96%
Profit, Utility Index	53.03	49.31	18.86	105.09	62.43	24.71	43.91	53.76	34.16	28.91	34.60

B. With Forecasting

WITH FORECASTING													
	DECISION MATRIX WITH PREDICTION												
Observation	49	50	51	52	53	54	55	56	57	58	59	60	61
Forecasted Total Incoming Bandwidth	43.50	67.84	54.99	44.57	57.98	67.35	62.13	80.18	85.03	85.92	78.36	89.72	80.73
Capacity	2.35	3.66	2.97	2.41	3.13	3.64	3.35	4.33	4.59	4.64	4.23	4.84	4.36
Threshold	1.64	2.56	2.08	1.68	2.19	2.55	2.35	3.03	3.21	3.25	2.96	3.39	3.05
Window	10	10	10	10	10	10	10	10	10	10	10	10	10
Requests	83	42	41	64	58	31	30	39	28	20	34	29	39
Queued	52	20	29	43	40	9	17	21	13	5	18	14	22
Rejected	26	13	4	6	11	10	5	12	5	4	7	2	6
Q_Unprocessed	0	0	0	0	0	1	0	0	1	0	1	0	0
PoB, Decision Matrix	0.31	0.31	0.10	0.09	0.19	0.35	0.17	0.31	0.21	0.20	0.24	0.07	0.15

Syst Utilisation, Decision Matrix	47.29 %	80.33 %	2.45 %	70.38 %	79.47 %	46.78 %	66.35 %	45.82 %	83.55 %	0.96 %	63.07 %	53.41 %	18.88 %
Profit, Decision Matrix	11.11	84.85	0.73	60.63	54.70	26.40	53.05	26.20	104.53	0.44	54.31	60.45	14.19
	FIFO, QUEUE WITH PREDICTION												
Observation	49	50	51	52	53	54	55	56	57	58	59	60	61
Forecasted Total Incoming Bandwidth	43.50	67.84	54.99	44.57	57.98	67.35	62.13	80.18	85.03	85.92	78.36	89.72	80.73
Capacity	2.35	3.66	2.97	2.41	3.13	3.64	3.35	4.33	4.59	4.64	4.23	4.84	4.36
Threshold	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Window	10	10	10	10	10	10	10	10	10	10	10	10	10
Requests	83	42	41	64	58	31	30	39	28	20	34	29	39
Queued	77	33	32	56	50	25	20	30	18	8	24	23	29
Rejected	0	0	0	0	0	0	0	0	0	0	0	0	0
Q_Unprocessed	20	14	1	5	9	5	2	11	5	4	8	2	3
PoB, FIFO with Queue	0.24	0.33	0.02	0.08	0.16	0.16	0.07	0.28	0.18	0.20	0.24	0.07	0.08
Syst Utilisation, FIFO with Queue	86.31 %	92.04 %	3.36 %	70.61 %	79.08 %	80.89 %	66.76 %	75.47 %	85.63 %	0.96 %	63.44 %	53.27 %	54.56 %
Profit, FIFO with Queue	52.28	104.67	1.00	60.64	54.72	56.93	53.21	107.92	108.25	0.44	54.31	60.45	29.73
	UTILITY INDEX WITH PREDICTION												
Observation	49	50	51	52	53	54	55	56	57	58	59	60	61
Forecasted Total Incoming Bandwidth	43.50	67.84	54.99	44.57	57.98	67.35	62.13	80.18	85.03	85.92	78.36	89.72	80.73
Capacity	2.35	3.66	2.97	2.41	3.13	3.64	3.35	4.33	4.59	4.64	4.23	4.84	4.36
Threshold	1.64	2.56	2.08	1.68	2.19	2.55	2.35	3.03	3.21	3.25	2.96	3.39	3.05
Window	10	10	10	10	10	10	10	10	10	10	10	10	10
Requests	83	42	41	64	58	31	30	39	28	20	34	29	39
Queued	78	37	36	59	55	25	26	33	21	12	29	21	34
Rejected	0	0	0	0	0	0	0	0	0	0	0	0	0
Q_Unprocessed	21	17	1	6	19	11	2	12	10	4	10	3	3
PoB, Utility Index	0.25	0.40	0.02	0.09	0.33	0.35	0.07	0.31	0.36	0.20	0.29	0.10	0.08

Syst Utilisation, Utility Index	85.72 %	92.30 %	3.36 %	72.55 %	94.70 %	85.00 %	69.75 %	82.11 %	84.92 %	0.96 %	86.34 %	53.90 %	54.74 %
Profit, Utility Index	52.26	113.15	1.00	60.63	128.15	48.71	53.21	107.08	104.49	0.44	108.84	60.35	29.73

WITH FORECASTING											
		DECISION MATRIX WITH PREDICTION									
Observation	62	63	64	65	66	67	68	69	70	71	72
Forecasted Total Incoming Bandwidth	62.64	57.49	46.13	63.16	54.36	85.59	78.24	65.11	69.97	84.59	69.27
Capacity	3.38	3.10	2.49	3.41	2.94	4.62	4.23	3.52	3.78	4.57	3.74
Threshold	2.37	2.17	1.74	2.39	2.05	3.24	2.96	2.46	2.64	3.20	2.62
Window	10	10	10	10	10	10	10	10	10	10	10
Requests	37	33	34	89	90	51	50	60	56	38	51
Queued	18	22	16	64	69	35	31	39	23	16	24
Rejected	10	8	9	23	12	11	13	19	22	11	21
Q_Unprocessed	1	0	0	17	6	0	9	0	0	0	2
PoB, Decision Matrix	0.30	0.24	0.26	0.45	0.20	0.22	0.44	0.32	0.39	0.29	0.45
Syst Utilisation, Decision Matrix	85.62%	72.56%	40.90%	91.23%	83.46%	91.39%	96.45%	88.25%	53.30%	40.03%	85.50%
Profit, Decision Matrix	79.72	66.27	15.93	148.43	43.28	87.67	90.55	55.47	34.45	28.91	61.81
		FIFO, QUEUE WITH PREDICTION									
Observation	62	63	64	65	66	67	68	69	70	71	72
Forecasted Total Incoming Bandwidth	62.64	57.49	46.13	63.16	54.36	85.59	78.24	65.11	69.97	84.59	69.27
Capacity	3.38	3.10	2.49	3.41	2.94	4.62	4.23	3.52	3.78	4.57	3.74
Threshold	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Window	10	10	10	10	10	10	10	10	10	10	10
Requests	37	33	34	89	90	51	60	60	56	38	51
Queued	32	28	26	81	83	44	57	57	47	27	46
Rejected	0	0	0	0	0	0	0	0	0	0	0

Q_Unprocessed	19	10	7	39	19	11	23	22	19	7	22
PoB, FIFO with Queue	0.51	0.30	0.21	0.44	0.21	0.22	0.38	0.37	0.34	0.18	0.43
Syst Utilisation, FIFO with Queue	95.00%	83.03%	79.73%	83.95%	85.69%	96.23%	96.44%	90.51%	90.41%	90.86%	88.77%
Profit, FIFO with Queue	73.93	68.37	33.98	77.37	36.32	81.82	79.11	47.64	48.68	65.89	65.48
		UTILITY INDEX WITH PREDICTION									
Observation	62	63	64	65	66	67	68	69	70	71	72
Forecasted Total Incoming Bandwidth	62.64	57.49	46.13	63.16	54.36	85.59	78.24	65.11	69.97	84.59	69.27
Capacity	3.38	3.10	2.49	3.41	2.94	4.62	4.23	3.52	3.78	4.57	3.74
Threshold	2.37	2.17	1.74	2.39	2.05	3.24	2.96	2.46	2.64	3.20	2.62
Window	10	10	10	10	10	10	10	10	10	10	10
Requests	37	33	34	89	90	51	50	60	56	38	51
Queued	33	29	27	83	85	47	42	56	50	31	46
Rejected	0	0	0	0	0	0	0	0	0	0	0
Q_Unprocessed	33	12	8	67	38	37	21	47	35	9	28
PoB, Utility Index	0.89	0.36	0.24	0.75	0.42	0.73	0.42	0.78	0.63	0.24	0.55
Syst Utilisation, Utility Index	99.23%	91.26%	80.86%	93.41%	89.98%	97.43%	87.26%	95.29%	92.55%	90.70%	81.43%
Profit, Utility Index	76.41	71.20	33.95	138.84	71.24	133.67	78.48	95.60	73.97	65.72	39.60

Q_Unprocessed is the number of unprocessed called in a queue.

Syst Util is the system utilisation.