10-2022

QUALITY-OF-SERVICE PROVISIONING FOR SMART CITY APPLICATIONS USING SOFTWARE-DEFINED NETWORKING

Basima Kurungadan

UAEU جامعة الإمارات العربية المتحدة United Arab Emirates University

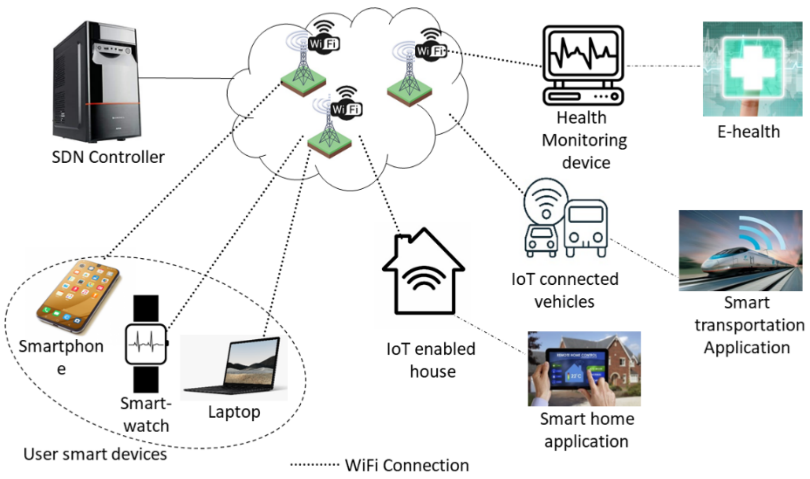
MASTER THESIS NO. 2022: 121

College of Engineering

Department of Electrical and Communication Engineering

QUALITY-OF-SERVICE PROVISIONING FOR SMART CITY APPLICATIONS USING SOFTWARE-DEFINED NETWORKING

Basima Kurungadan



November 2022

United Arab Emirates University

College of Engineering

Department of Electrical and Communication Engineering

QUALITY-OF-SERVICE PROVISIONING FOR SMART CITY APPLICATIONS USING SOFTWARE-DEFINED NETWORKING

Basima Kurungadan

This thesis is submitted in partial fulfilment of the requirements for the degree of Master of Science in Electrical Engineering

November 2022

United Arab Emirates University Master Thesis
2022: 121

Cover: System model of software defined networking enabled WiFi connected smart city services

(Photo: By Basima Kurungadan)

Declaration of Original Work

I, Basima Kurungadan, the undersigned, a graduate student at the United Arab Emirates University (UAEU), and the author of this thesis entitled "*Quality-of-Service Provisioning for Smart City Applications using Software-defined Networking*", hereby, solemnly declare that this is the original research work done by me under the supervision of Dr. Atef Abdrabou, in the College of Engineering at UAEU. This work has not previously formed the basis for the award of any academic degree, diploma or a similar title at this or any other university. Any materials borrowed from other sources (whether published or unpublished) and relied upon or included in my thesis have been properly cited and acknowledged in accordance with appropriate academic conventions. I further declare that there is no potential conflict of interest with respect to the research, data collection, authorship, presentation and/or publication of this thesis.

Student's Signature: [illegible]

Date: 17/11/2022

Approval of the Master Thesis

This Master Thesis is approved by the following Examining Committee Members:

1) Advisor (Committee Chair): Atef Abdrabou
   Title: Associate Professor
   Department of Electrical Engineering
   College of Engineering

   Signature Atef   Date 15/04/2023

2) Member: Qurban Ali M.
   Title: Associate Professor
   Department of Electrical Engineering
   College of Engineering

   Signature Qurban Ali   Date 17/04/2023

3) Member (External Examiner): Chow Chee Onn
   Title: Associate Professor
   Department of Electrical Engineering
   Institution: University of Malaya, Malaysia

   Signature [illegible]   Date 17/04/2023

This Master Thesis is accepted by:

Dean of the College of Engineering: Professor Mohamed Al-Marzouqi

Signature Mohamed AlMarzouqi

Date June 14, 2023

Dean of the College of Graduate Studies: Professor Ali Al-Marzouqi

Signature Ali Hassan

Date 14/06/2023

Abstract

In the current world, most cities have WiFi Access Points (AP) in every nook and corner. Hence upraising these cities to the status of a smart city is a more easily achievable task than before. Internet-of-Things (IoT) connections primarily use WiFi standards to form the veins of a smart city. Unfortunately, this vast potential of WiFi technology in the genesis of smart cities is somehow compromised due to its failure in meeting unique Quality-of-Service (QoS) demands of smart city applications. Out of the following QoS factors; transmission link bandwidth, packet transmission delay, jitter, and packet loss rate, not all applications call for the all of the factors at the same time. Since smart city is a pool of drastically unrelated services, this variable demand can actually be advantageous to optimize the network performance. This thesis work is an attempt to achieve one of those QoS demands, namely packet delivery latency. Three algorithms are developed to alleviate traffic load imbalance at APs so as to reduce packet forwarding delay. Software-Defined Networking (SDN) is making its way in the network world to be of great use and practicality. The algorithms make use of SDN features to control the connections to APs in order to achieve the delay requirements of smart city services. Real hardware devices are used to imitate a real-life scenario of city-wide coverage consisting of WiFi devices and APs that are currently available in the market with neither of those having any additional requirements such as support for specific roaming protocol, running a software agent or sending probe packets. Extensive hardware experimentation proves the efficacy of the proposed algorithms.

Keywords: Software-defined Networking, WiFi, traffic, load, control, delay, IoT, smart city, e-Health.

Title and Abstract (in Arabic)

توفير جودة الخدمة لتطبيقات المدن الذكية باستخدام الشبكات المحددة بالبرامج

الملخص

في العالم الحالي، تحتوي معظم المدن على نقاط وصول WiFi (AP) في كل زاوية وزاوية. ومن ثم فإن الارتقاء بهذه المدن إلى مكانة المدينة الذكية هو مهمة يمكن تحقيقها بسهولة أكبر من ذي قبل. تستخدم اتصالات إنترنت الأشياء (IoT) بشكل أساسي معايير WiFi لتشكيل عروق مدينة ذكية. لسوء الحظ، فإن هذه الإمكانات الهائلة لتقنية WiFi في نشأة المدن الذكية تتعرض للخطر إلى حد ما بسبب فشلها في تلبية متطلبات جودة الخدمة (QoS) الفريدة لتطبيقات المدن الذكية. من بين عوامل جودة الخدمة التالية؛ عرض النطاق الترددي لوصلة الإرسال وتأخير إرسال الرزم والارتعاش ومعدل فقدان الحزمة، لا تستدعي جميع التطبيقات جميع العوامل في نفس الوقت. نظرًا لأن المدينة الذكية عبارة عن مجموعة من الخدمات غير المرتبطة بشكل كبير، يمكن أن يكون هذا الطلب المتغير مفيدًا في الواقع لتحسين أداء الشبكة. يعد هذا العمل الخاص بالأطروحة محاولة لتحقيق أحد متطلبات جودة الخدمة هذه، ألا وهو زمن انتقال الحزم. تم تطوير ثلاث خوارزميات للتخفيف من اختلال حمل حركة المرور في نقاط الوصول وذلك لتقليل تأخير إعادة توجيه الحزمة. تشق الشبكات المعرفة بالبرمجيات (SDN) طريقها في عالم الشبكات لتكون ذات فائدة وعملية كبيرة. تستفيد الخوارزميات من ميزات SDN للتحكم في الاتصالات بنقاط الوصول من أجل تحقيق متطلبات التأخير لخدمات المدينة الذكية. تُستخدم الأجهزة الحقيقية لتقليد سيناريو حقيقي للتغطية الواسعة للمدينة التي تتكون من أجهزة WiFi ونقاط الوصول المتوفرة حاليًا في السوق مع عدم وجود أي من تلك الأجهزة التي لديها أي متطلبات إضافية مثل دعم بروتوكول تجوال معين أو تشغيل وكيل برمجيات أو الإرسال حزم التحقيق. أثبتت التجارب الواسعة للأجهزة فاعلية الخوارزميات المقترحة.

مفاهيم البحث الرئيسية: الشبكات المعرفة بالبرمجيات، WiFi، حركة المرور، التحميل، التحكم، التأخير، إنترنت الأشياء، المدينة الذكية، الصحة الإلكترونية.

Acknowledgements

I would like to thank my advisor Dr. Atef Abdrabou for the guidance, support, and assistance throughout my preparation of this thesis.

Dedication

To my beloved parents and family

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AP	Access Point
E2E	End-to-End
IoT	Internet-of-Things
LoRa	Long Range Radio
MAC	Medium Access Control
PHY	Physical Layer
QoS	Quality-of-Service
RSSI	Received Signal Strength Indicator
SDN	Software-Defined Networking
SDWLAN	Software-Defined Wireless Local Area Network
WiFi	Wireless Fidelity
WLAN	Wireless Local Area Network

Chapter 1: Introduction

1.1 Overview

With WiFi seeping its roots deep into human life day by day, it has even influenced the author of Pace Technical (n. d.) to add it to Maslow's Hierarchy of Needs (Maslow's Hierarchy of Needs, n. d.) on a basic level before human physiological needs. A new study expects more than two-thirds of the IP traffic volume to be from portable and WiFi-associated devices, outnumbering the wired traffic by more than double the amount (Cisco, 2019). With the leap in the usage of wearable smart gadgets, which are mostly connected through smartphones instead of having embedded cellular connectivity (Seneviratne et al., 2017), WiFi connection outdoors is becoming inevitable. Moreover, WiFi offloading is a current topic of interest due to its efficiency in saving the battery life of mobile devices while using real-time services like video streaming (Burger et al., 2015; Li et al., 2022). In addition, the presence of the COVID-19 pandemic broadly expanded the interest in involving the Internet through WiFi associations to achieve a large throughput in both communication directions (transfer and download), like in web-based video conferences/ meetings and online teaching.

Smart city is no more a topic of tomorrow, it is already present and it has been realized as the ultimate solution to mitigate challenges of the rising urban population such as urban management, resource shortage, environmental pollution, traffic congestion, et cetera (He, 2021). WiFi is commonly used as a candidate technology for the communication network design of smart cities to support the functional requirements of many IoT applications (Zanella et al., 2014). However, the capabilities of a smart city and its importance in the future are still unappealing to ordinary people due to numerous concerns over the latency, privacy, reliability, and availability (Tragos et al., 2014) it can offer. Information delivery latency is one issue that needs to be addressed seriously. Besides numerous tasks, transmitting a large real-time data volume from IoT nodes and processing this data represent important tasks of various smart city services. Among those, some are time-critical, for which any communication lag above a particular threshold means total worthlessness of such services. For instance, information transfer latency is vital for e-health services to realize their objectives of continuously monitoring patients and reporting emergencies (Islam et al., 2017). Similar delay-sensitive data might become important in enabling several other smart city services. For instance, while enabling smart transportation, timely communication of traffic status needs to be conveyed to passengers/drivers; such as the suggestion to avoid traffic congestion by choosing alternate routes, emergency alert regarding the occurrence of a road accident, et cetera. Similarly, the safety of citizens is one of the primary concerns in recent city development. Proper installation of surveillance systems along with live monitoring or alarm systems can achieve this purpose to a large extent in public as well as in industrial sites. Smart education systems are the norm after the Covid 19 pandemic. It has proved that classes can be continued online even if there are unfavorable conditions to continue daily classes such as the flu season, flood, extreme heat, et cetera. However, uninterrupted live communication is necessary to carry out the smooth functioning of academics online for the participation of both students and tutors. Several smart monitor, control, and automation systems are already available to conserve energy in buildings, offices, and industrial sites. There is even an anticipated drift of users toward electric-powered vehicles (del Cacho Estil-les et al., 2022). Energy management becomes much easier within a smart city due to connectivity, such as spotting a car charging station with the availability of vacant spots. Ensuring QoS to services is considered one of the success factors of smart city implementation (Wenge et al., 2014). In fact, authors Zakaria and Jawwad (2015) list efficiency as one of the many other challenges while proposing a smart city architecture with specific mention to instantaneous access to emergency services, be it in any field such as civil, construction, industrial or medical. Hence there is an ardent need to study and fulfill the unique QoS demands of different smart city services to ensure availability, reliability, and efficiency of these services.

1.2 Statement of the Problem

1.2.1 Example Scenario in Real Life

Figure 1 shows a sample scenario in a WiFi network. In case 1, there are two APs connected to the internet. AP1 is crowded with heavy data traffic; one of the users is performing data backup which consumes huge bandwidth, and other user with high-speed gaming devices also occupies high bandwidth. Hence both of these devices already fully occupy the bandwidth of AP1. Somehow a smartphone user is also trying to send some crucial health-related data using an e-health application through the same access point. Obviously, the user sending health data will face significant delay. Here, the gaming device and e-health application are both QoS demanding while the data backup function could have been delayed for a later time. Unless the network is able to differentiate connections based on QoS, this is not possible. However, at the same time, AP2 carries very little data traffic load leaving its resources mostly unused. Now consider case 2, where the data backup traffic and gaming service fully occupy the bandwidth of AP1, and the mobile user sending health data moved to AP2 compared to case 1, so that the user is offered a reliable connection. In case 1, load imbalance occurred between AP1 and AP2 because the AP selection was only based on the RSS factor. In this research, the main focus is to avoid case 1 scenarios in a WiFi network, especially in a smart city setting where QoS-demanding applications co-exist with non QoS-ones.

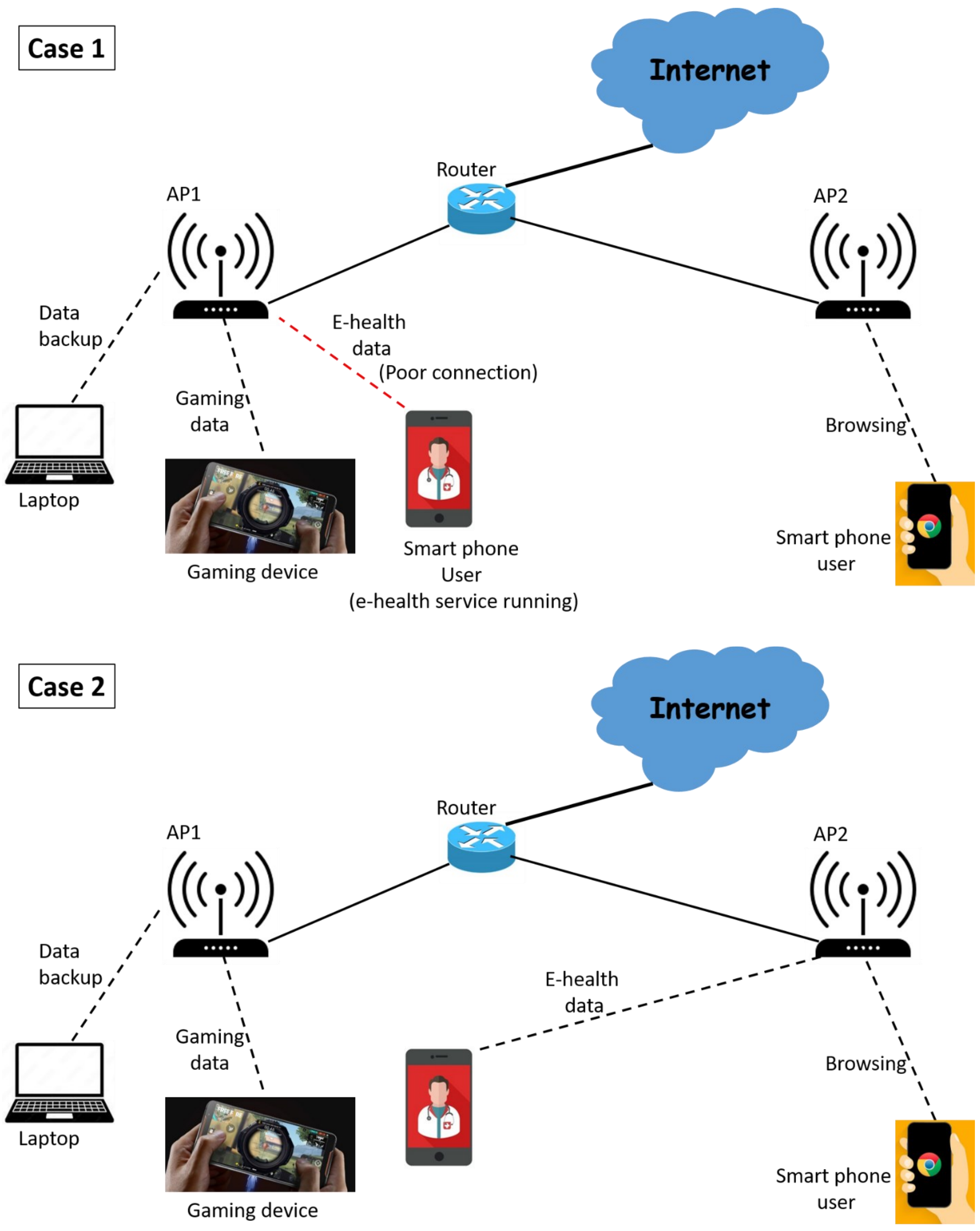


Figure 1: Example scenario for statement of problem

1.2.2 Problem Statement

Not that the benefits of smart cities are unknown to people, but there is still some hesitance in trusting most of its capabilities due to various reasons such as security, privacy, reliability, and availability issues (Tragos et al., 2014). Indeed, transferring a huge amount of real-time data from IoT devices, the analysis of this data, and taking proper actions are the primary functions involved in most smart city applications (Pflanzner et al., 2018). Some of the actions are time-critical, and hence any delay in data transfer would result in severe consequences. For example, smart city e-health applications are essential for both the elderly and younger citizens, but data transfer delay is critical for these applications to perform their intended functions, especially in emergencies (Islam et al., 2017). In many smart city initiatives, the city is covered by many WiFi networks with overlapped coverage. This may result in a scenario where some WiFi networks are loaded with a routine data collection, while other delay-sensitive traffic (e.g., reporting an emergency alert of a person's health) needs to be sent within a short time. Thus, the careful radio resource management of smart city WiFi networks is inevitable for smart city services to achieve their targets. This requires a dynamic configuration of network resources, which cannot be achieved by conventional methods.

Indeed, the number of wireless sensor nodes required to realize a full-fledged smart city is massive, while these nodes are of different types and should be installed in different locations. Some of them are not easy to frequently be replaced, like in the case of structure monitoring, and hence they need to maintain low power consumption. Others require long-range communication due to obstacles or being installed in remote city areas. This mandates the usage of different wireless technologies such as Zigbee and LoRa, which cannot support a high data rate. Thus, using WiFi-connected IoT gateways becomes inevitable (Mehmood et al., 2017). These gateways can receive sensed data over different wireless technologies and forward it to its management entities or servers using WiFi. Moreover, they can function as edge or fog computing nodes (Chiang & Zhang, 2016). However, these gateways need to maintain acceptable packet latency requirements of the forwarded data, given that these gateways often lie in spots with overlapped WiFi coverage (Kurungadan & Abdrabou, 2022). Admittedly, this is triggered in a situation where some APs receive periodic routine data while others are loaded with delay-sensitive traffic, such as alarm notifications or critical health status reporting. Consequently, this demands a dynamic network configuration to connect an IoT gateway to the AP with sufficient resources, which cannot be achieved with classic methods. While there are very few studies regarding the uplifting of smart cities using SDN in their functioning, the potential of such a system is boundless. With its agility and programmability (Liu et al., 2018), SDN can dynamically configure the WiFi network resources efficiently (Kreutz et al., 2015) since it can untie the complexity of the current network infrastructure.

1.3 Research Objectives

In this thesis, the objective is to present a scheme for reducing the delivery latency of data packets forwarded by WiFi-connected IoT gateways covered by multiple smart city APs working on different radio channels. This is an effort to fill the gap in research addressing issues of QoS smart city services specifically with delay-sensitive services. Because even though there are several studies on network security, speed, IoT networks, et cetera, few studies discuss QoS in a smart city scenario where different QoS applications co-exist. This research also tests all the proposed algorithms in a real laboratory setting, this is to observe the practicality of SDN controllers in acquiring network data and making decisions for dynamic configuration changes. Although SDN is a much-raved topic in the literature, very rarely it is tested practically in a real testbed, most of them rely on simulation tools. Moreover, instead of choosing AP based on the RSSI factor alone which is the case in a traditional network, this work also proves how packet latency can be significantly improved if the proposed algorithm is used to make handover decision. Tests are also conducted using Pollaczek–Khinchine formula to predict end-to-end delay for Poisson arrival rates. Lastly, an algorithm is used to find the least crowded AP. The algorithm selects the least crowded AP for the client to handover from crowded AP. It shows how selecting the least loaded AP for handover notably reduces the packet delay compared to the case of random handover. This thesis becomes useful in the implementation of delay-sensitive smart city services with the use of SDN given that SDN is already been realized as the future of wireless/wired network renovation.

1.4 Potential Contributions and Limitations of the Study

In this research, the SDN controller is programmed to select which IoT gateway should be handed over to another AP and which AP can accommodate the gateway traffic with a lower End-to-End (E2E) packet delivery latency. This results in redistributing the traffic load at the WiFi network APs leading to a considerable decrease in overall network delay. The contributions of this thesis are three-fold. First, an algorithm is proposed to find the IoT gateway with the highest end-to-end packet delivery delay transparently without altering the IoT gateway or the receiving node by adding any software agent or measuring probes. The algorithm is non-invasive as it solely depends on packet interarrival time measurements at the SDN controller. Second, another algorithm is devised to find the IoT gateway with the highest end-to-end latency by estimating this delay for each gateway connected to a certain AP. This is done by performing the necessary measurements at the SDN controller to carry out an M/G/1 analysis using the arrival rate information received from the IoT gateway. For both algorithms, the SDN controller reassociates the tagged IoT gateway seamlessly with another AP without changing the gateway configuration or exchanging handover-related messages. Third, an algorithm is developed on the SDN controller to continuously find the least loaded AP to handover the IoT gateway with the highest end-to-end packet delivery delay. The proposed algorithms were tested on a real laboratory setup of SDN based network, and it has shown significant improvement in delay performance. Hence, it opens a way to provision latency QoS to delay-sensitive applications, especially in a smart city scenario where varying QoS demand co-exist.

One of the main limitations of this thesis is in the implementation of an SDN controller for a large network. Obviously, a single controller cannot monitor all of the network devices in a real-life scenario. In that case, multiple SDN controllers need to be implemented for total coverage, each controller taking care of only a particular portion of a network (Isong et al., 2020). However, to make network decisions a global view of the whole network is inevitable, this would mean exchanging network data among controllers. This might result in additional data traffic overhead in the network. The placement of controllers is also a topic of research as the network capacity grows bigger. Researchers Chen et al. (2018) and Mohanty et al. (2021) look for the optimum placement of multiple controllers in large networks. Bouzghiba et al. (2017) discuss controller placement as well as other network performances that may be affected due to the separation of the control plane in SDN.

Chapter 2: Literature Review

2.1 General Background

2.1.1 WiFi AP Selection

Wireless local area network, also known as WLAN consists of many devices including computers, mobile phones, et cetera. which communicate with each other via radio waves within limited coverage areas such as campuses, shopping malls, corporate offices, and so on. Wireless connections enable mobility for users without connection loss. WLANs are often connected to the internet through a gateway. IEEE 802.11 protocol is the major protocol used in such networks.

WLAN can be either Infrastructure based or Ad-hoc network. A home or office Wi-Fi network is an example of a WLAN setup in infrastructure mode. The endpoints are all connected and communicate with each other through a base station, which may also provide internet access (Cisco, n. d.). It usually consists of one or more wireless routers or access points, which act as the base stations, and endpoints, which can be computers, mobile devices, printers, and other devices. In an ad-hoc configuration, a WLAN connects endpoints such as computer workstations and mobile devices without the use of a base station. In this research, only infrastructure-based networks are considered.

In an infrastructure-based WLAN, a wireless client station (STA) can find multiple APs in its vicinity. As per IEEE 802.11 standard, each STA has to associate with one and only one AP at a time. The proper selection of AP increases the user throughput and system performance. As per IEEE 802.11 standard, the STA selects the AP with the highest Received Signal Strength Indicator (RSSI). This approach will cause an uneven number of STAs to be connected at APs as more and more clients are added to the network which eventually results in overload at particular APs. As a consequence, the fairness in STA throughput is degraded and the network resources are not utilized effectively. Moreover, this method of AP selection does not consider many important parameters such as channel conditions, AP load and contending stations as illustrated by Kumar and Bindu (2014). In fact, an interesting experiment by Judd and Steenkiste

(2002) shows that even when a new AP was added to a cell to alleviate an overloaded AP, there was no change in the network. This issue rises from the fact that 802.11 only considers RSS criteria for AP selection whereas it completely ignores the load factor on each APs.

Since AP selection is a crucial factor in determining network performance, there are numerous kinds of research and proposals in this field. These are broadly classified as centralized and non-centralized approaches. Both of these approaches have their advantages and disadvantages in achieving optimum network performance.

In the non-centralized method, each client makes decision regarding choosing the AP based on some measurements such as link quality and received signal strength. The research work by Nicholson et al. (2006) proposes a system named Virgil. This system chooses an AP as a result of associating the client with the rest of the nearby APs to evaluate the connection quality. In a research by Kim et al. (2017), the Wifi client estimates the interference level of APs based on metrics derived by signal strength variance and beacon collision rate, unlike in the traditional method where the RSSI metric is usually used. However, the proposed scheme demands several modifications at the WiFi client. Sun et al. (2016) propose a modification to the current 802.11 protocol probing phase. Whenever the bandwidth of a connected AP goes below the required bandwidth, the client starts to probe neighboring APs. Instead of only considering RSS, a new bandwidth estimation scheme is proposed in the paper. Hence the paper ensures WiFi client association with APs of sufficient bandwidth, RSS, and load. But, this approach, also known as client-driven user-AP association, has many shortcomings. First, a client has local knowledge of the network state and its association scheme considers only signal quality from the APs. Second, user fairness and load balancing among APs cannot be ensured. Third, even if a client is optimally associated, it may not remain optimal if the client moves or the network state changes as described by Bayhan et al. (2020).

In the centralized approach, usually, a network controller is involved in the AP selection process. Until very recently, centrally controlled networks were quite unpopular. Hence studies on AP selection in this category are very rare. As SDN came to be known as a viable technology, it has become more and more popular nowadays. In a research by Bejerano et al. (2007), a Network Operation Center (NOC) collects the effective bit rate of APs in the network from the users and it runs the AP association algorithm to ensure network-wide max-min fair bandwidth allocation to users. Li et al. (2008) suggest that instead of evaluating network performance in terms of aggregate throughput alone, fairness among users should also be considered especially in the case of multi-rate WLAN networks. Based on simulations, their proposed algorithms for AP association control based on proportional fairness gave much better fairness and aggregate throughput than the max-min fairness-based algorithm. Pang et al. (2010) propose a collaborative service named WiFi-Reports. Clients using WiFi-Reports automatically submit information including metrics such as estimated backhaul capacity, ports blocked, and connectivity failures of the APs they connected in the WiFi network. Later on, new clients get historical data of individual APs in a network, thus enabling them to choose apt AP for their applications. As evident, this scheme needs a lot of data collection from users, which everyone may not be willing to share.

2.1.2 AP Load Balancing

According to Yen et al. (2009) load balancing approaches must address two primary issues: proper definition and measurement of load-related metrics, and method to distribute overall traffic load among all available APs. In load balancing, an AP can reject a connection from a user when its resources are exploited in contrast to the case of AP selection where any number of devices can connect to an AP as long as the RSSI condition is satisfied. An optimal load balancing system must be adaptive in nature; it should be able to change its behavior according to changing load in the network. For instance, an AP should be able to accept a connection from a client from which the connection was previously rejected when the load was much lesser or greater depending on the situation. This will necessitate a central view of the network topology with information regarding the amount of load on each AP. Even though various studies address AP selection mechanisms, few papers discuss WiFi AP load balancing. Out of these considerable number of researches demand the availability of specific functionality in WiFi clients, such as transmitting test packets (Gong et al., 2008; Xu et al., 2010) and running certain protocols (Adame et al., 2021). Xu et al. (2010) present an online algorithm in which a WiFi node chooses the AP that minimizes the norm of the traffic loads of other APs in its transmission range. Only throughput is addressed in Xu et al. (2010), which requires a modification of the client WiFi adapter to send custom test packets. Range extender selection for home APs is addressed in the study by Adame et al. (2021). The paper considers not only the traditional method of RSSI values but also the channel load, but it demands the existence of WiFi hardware supporting IEEE 802.11k/v (Adame et al., 2021). Industrial applications employ many IoT services. These applications also constitute an integral part of a smart city. Thus, Cheng et al. (2018) propose a Deterministic Load Balancing algorithm (Det- LB) based on a game-theoretic auction model. The research work by Hava et al. (2019) presents a Video Load Balancing Solution (ViLBaS) to increase the performance of video applications in a multi-hop wireless mesh network to enhance the user quality of experience. However, this method is more like a routing mechanism to avoid congested nodes in the route. By utilizing multiple transmission power levels in APs, authors Bejerano and Han (2009) try to achieve load among APs through dynamically changing power levels of transmission power of the AP beacons.

2.1.3 Software Defined Networking

The networking intelligence is distributed in the networking devices themselves currently. This has caused a lot of restraints on network designing to meet current data requirements. Today's networks are highly complex because each problem or protocol in the network are addressed individually which in turn resulted in the addition of a lot of middle devices such as multiple switches, firewall, web authentication portals, et cetera. Hence addition or removal of any device from the network becomes a hectic task making the network unable to scale. Vendors provide new features specific to their devices which limits the ability of network operators to tailor the network to their individual environments (Sloane, 2013).

The key idea of SDN is to decouple the control plane from the data plane and allow flexible and efficient management and operation of the network via software programs. Specifically, devices (e.g., switches and routers) in the data plane perform packet forwarding, based on rules installed by controllers. Controllers in the control plane oversee the underlying network and provide a flexible and efficient platform to implement various network applications and services. Under this new paradigm, innovative solutions for specific purposes (e.g., network security, network virtualization, and green networking) can be rapidly implemented in form of software and deployed in networks with real traffic (Xia et al., 2015). Figure 2 shows different layers of SDN.

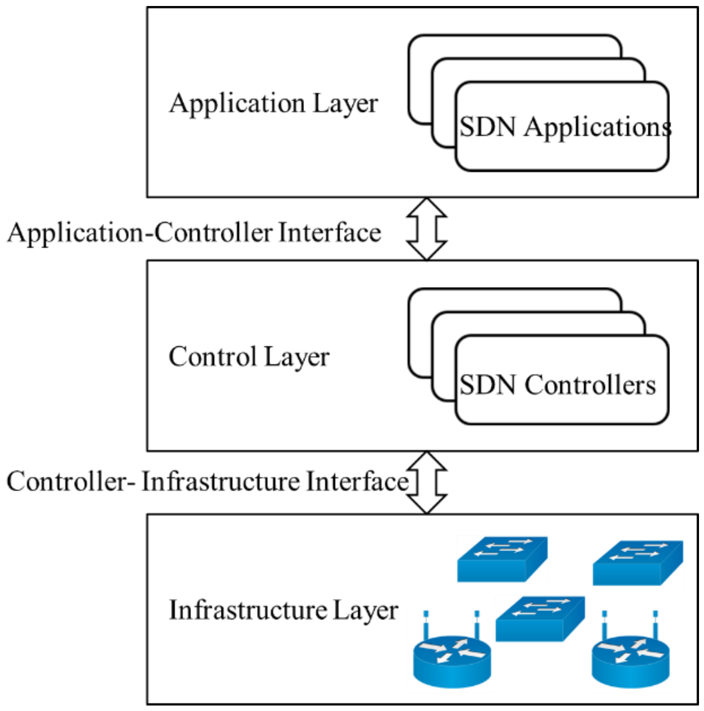


Figure 2: SDN reference model

2.1.4 SDN v/s Traditional Network

A traditional network is mainly built upon static dedicated hardware entities such as routers and switches (sometimes many more middleware will be added) to control network traffic. Due to its complex fixed structure, traditional network is difficult to scale. Moreover, network security and performance issues are another major concern in the current global network. SDN on the other hand promotes control and management of the network using software applications. SDN controls the network from a central controller using open APIs. This controller is able to see the network topology. Software Defined Network enhances performance by network virtualization. Using software applications written over SDN controller and open APIs, versatile network management and communication with end-to-end devices becomes possible. Being open-standards based, SDN pulls out a lot of network operation difficulties that otherwise had to be done with vendor-specific instructions and commands. SDN can automate most of the hectic repetitive configuration tasks in the IT sector. It also opens a lot of opportunities in the telecom sector. Table 1 shows the difference between SDN and traditional networks.

Table 1: Difference between SDN and traditional network

Software Defined Network	**Traditional Network**
It has centralized control over network topology.	It has distributed control.
SDN is programmable.	Traditional network Not programmable.
In Software Defined Network data plane and control plane are decoupled by software.	In traditional network data plane and control plane are mounted on same plane.
Easy to scale and program as per need.	Difficult to scale.
Low structural complexity.	Highly complex network structure.

Figure 3 shows difference in data plane and control plane coupling in traditional network and SDN.

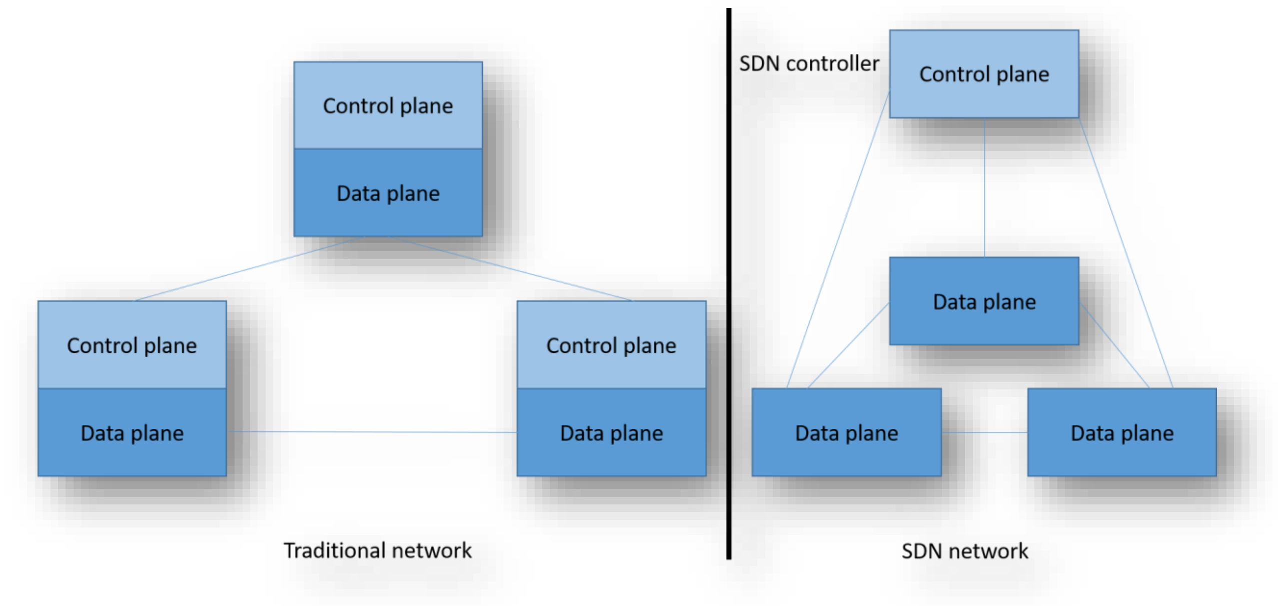


Figure 3: Data plane and control plane in traditional network and SDN

2.1.5 The OpenFlow Specification

OpenFlow protocol enables communication of SDN controller with the data plane of the network constituting routers and switches (Göransson et al., 2017). In SDN, the routing decisions (control plane) are decoupled from the data plane where forwarding decisions are made. An external controller gives forwarding instructions and decisions to the switches. Network applications communicate to the SDN controller regarding forwarding decisions and the controller converts these data to flow entries which are fed to the switch via OpenFlow. Each flow entry has packet match fields, flow priority, various counters, packet processing instructions, flow timeouts, and a cookie. These flow entries make up different flow tables inside the OpenFlow switch (Lara et al., 2014). An incoming data packet will be processed according to the entries in multiple "pipelined" tables before exiting on an egress port. Figure 4 illustrates the protocol in pictorial form.

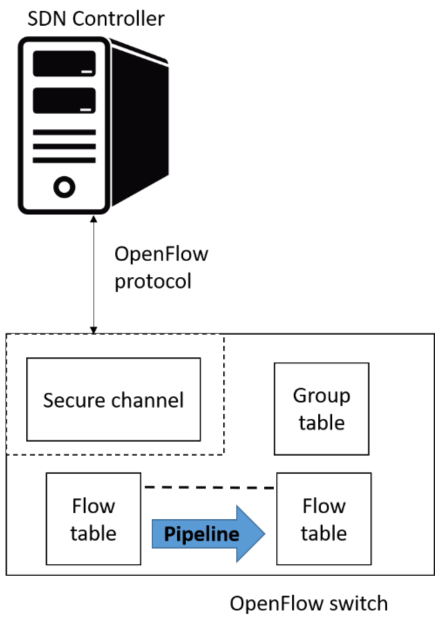


Figure 4: Openflow protocol

The controller can add, delete or change entries in the flow tables so s to:

- Based on various header fields of the packets or based on the input port, identify and group different packets.
- Process the packets in desired ways, including modifying the header; and,
- Drop or forward the packets to a particular output port or to the OpenFlow Controller.

Thus, a completely flexible and dynamic network is created. Owing to these properties, OpenFlow is chosen as a de-facto standard communications protocol in SDN architecture, which is similar to the TCP/IP protocol of the Internet as a communication standard (Haiyan et al., 2016). Some of the real-life examples of OpenFlow-enabled systems are (Sloane, 2013):

- Google's 100% OpenFlow global inter-data center WAN with centralized routing, traffic engineering, and bandwidth allocation. It had already realized 95% network utilization with simpler, faster configuration, management and provisioning.

- NTT Communications’ Enterprise Cloud IaaS service, offering network virtualization within and between data centers plus precise control of computing, network; virtual firewall, and load balancer functions.
- ATandT and IBM’s secure SDN cloud services, enabling fast and highly secure shared cloud storage and cloud services.

2.1.6 Smart City

People are migrating to cities searching for better lifestyles and resources. This urge has resulted in more than half of the world’s population living in cities currently. Consequently, urban areas have become congested and resources are being overexploited. According to statistics in Worlddata.Info (n. d.), 87% of the total population lives in urban. Hence careful planning and management of cities have become vital for environmental sustainability and socio-economic well-being (Mori & Christodoulou, 2012). Smart city has the capability to add this notion to existing cities. Although there are several ways to look at smart city, here we will mainly consider the Information Communication Technology (ICT) aspect. According to Washburn and Sindhu (2010), smart city can be defined as “the use of Smart computing technologies to make the critical infrastructure components and services of a city – which include city administration, education, healthcare, public safety, real estate, transportation and utilities- more intelligent, interconnected and efficient”. Smart cities can address a lot of challenges faced by cities in the current world. The well-structured transport system is one of the crucial parts in enabling a smooth work-life balance for residents. Using location-based services, wireless communication, cloud computing, et cetera can resolve travel headaches in cities. Proper waste management is unavoidable in cities. IoT technology can keep track of bin status, sort recyclable and degradable wastes, apply charges, et cetera. Cities are suffering from major air pollution side effects nowadays. New Delhi, the capital city of India is the most suffocating city due to air pollution globally. Careful installation of sensors and data processing can be used to assess air quality and take remedial measures to resolve such issues. Smart home is a fancy word currently but soon to become a common talk among households due to its benefits in energy efficiency, remote management of devices, enhancement of home security, and so on. Smart parking promotes efficient use of urban spaces and relieves a lot of stress for people living especially in crowded res. E-health systems may aid in saving lives if implemented suitably. It can notify emergency units regarding patient health status or other medical emergencies, set threshold values for determining the severity of situations, perform online consultation or even online surgery, et cetera. The list of smart city applications can go on; smart traffic management, e-governance, public safety and security, water management, smart infrastructure, and many more.

All of these applications ultimately share the same network for communication and data transfer. However, if observed carefully, it can be seen that each of these applications takes care of different areas of human life. Some of them are time-saving while others may be energy-conserving. E-health systems can be lifesaving whereas e-governance may include crucial data conveyance which demands utmost data security. There can be services that are just for fun too such as gaming facilities and aesthetic smart services installations. WiFi, along with an SDN controller can provide different QoS provisioning to different applications.

2.1.7 Internet of Things (IoT)

IoT is a collection of 'things' including living beings connected to or attached to smart devices, software applications, daily life gadgets, sensors, and many more smart devices and computational systems that communicate with each other. IoT is so relevant to smart city that most of its applications eventually contribute to the formation of a smart city. IoT has got countless applications currently implemented, and so much more to be found yet. Hence it is a growing technology with ambient possibilities to explore further. There are mainly five models to IoT design as in Figure 5, namely, IoT Domain Model, IoT Information Model, IoT Functional Model, IoT communication model, and IoT trust security and privacy model (Bauer et al., 2013).

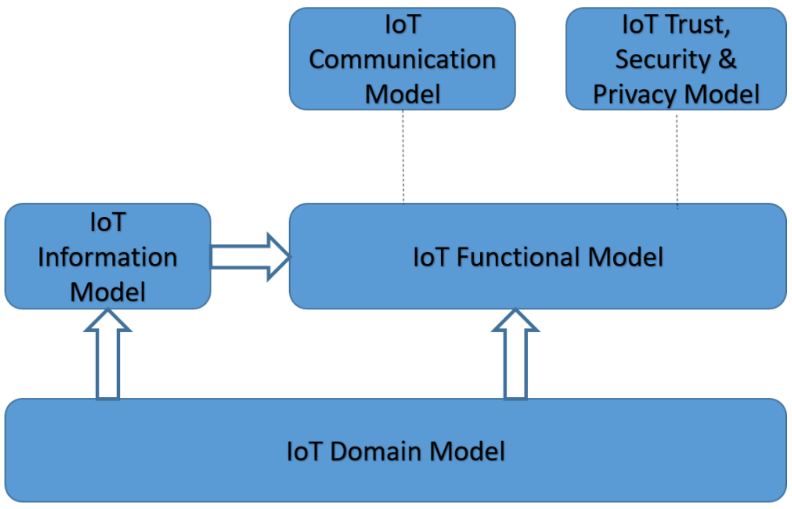


Figure 5: IoT models

2.2 Related Works

In the current situation there is no way to control the access points and end devices from a central controller. But this is essential in enabling dynamic changes to the network based on an updated global view of the network topology. A concept closest to this is the Control And Provisioning of Wireless Access Points (CAPWAP) protocol which defines a protocol to ease the implementation of large WLAN deployments that utilize the Controller-AP (Access Point) architecture (Stanley et al., 2009). It is used for managing Access Points by the Wireless LAN Controller in a Firmware upgrades, Configuration, and Authentication. It has no role in performing adaptive reconfiguration of network topology. SDN, however, can perform such adaptive changes to the network by making decisions regarding connection to APs. Several such studies based on SDN are found in the literature, the majority of which are simulation-based proposals. Mininet WiFi emulator is the most commonly used simulation unit for this purpose. Masaki et al. (2021), used this simulator to guarantee latency and bandwidth QoS in wireless networks using SDN for which customers can pay using IOTA cryptocurrency. A simulation-based study is done by Tadros et al. (2020) to show that out of several possible architectures for the SDN control plane, the Logically Centralized-Physically Distributed (LC-PD) architecture gives better communication efficiency and the Quality of Services (QoS) of running internet services in the 5G mobile network. Similarly, authors of Tivig et al. (2021) simulated a layer three logic for packet forwarding using the Ryu API and

Python language. Since the practicality of SDN is still a questionable topic simulation-based studies cannot help in its actual implementation in real life. More and more physical experiments on SDN are required to solve existing challenges and discover new ones, as well as to prove its much-acclaimed benefits over the traditional network.

2.2.1 SDN in AP Selection and Load Balancing

In the literature, a multitude of research works investigates the usage of SDN in AP selection. For instance, Raschellà et al. (2016), used a centralized SDN controller for AP selection according to a date rate-based fittingness factor without taking packet latency into account. Manzoor et al. (2019) present a software defined-WiFi network that addresses packet delay and loss for e-Health applications. However, both Raschellà et al. (2016) and Manzoor et al. (2019) use computer simulations to verify the proposed schemes. Lin et al. (2019) propose a new learning-based algorithm to monitor load imbalances in APs and perform handover accordingly to balance the load in all APs in a Software-Defined Wireless Network (SDWN). Their work mainly focuses on the throughput of TCP connections. Research by Lei et al. (2017) applies graph theory by modeling the station to AP association problem as a weighted bipartite graph matching problem and finding the optimal semi-matching using the Kuhn-Munkres (K-M) algorithm. The experiment is run on a simulated testbed with an algorithm running on an SDN controller. Using SDN to optimize the client association process with the help of a centralized controller while ensuring QoS for Software-Defined Wireless Local Area Network (SDWLAN) is studied by Chen et al. (2017). However, the authors Chen et al. (2017) assumed that all the APs and clients are tuned to the same radio channel and conducted only computer simulations to measure the performance of the proposed scheme. Han et al. (2016) propose an adaptive load balancing algorithm for Software-Defined Enterprise WLANs that takes into account AP load condition in addition to RSSI value as load metrics. An SDN controller makes AP selection decisions based on the global view of the network and continuous monitoring of load at APs. But this work demands periodic updates of load value from each AP to the controller.

2.2.2 SDN in Smart City

SDN is considered promising technology to promote smart city development for several reasons. Apart from its uses in basic network functions such as routing and end-to-end performance optimization, the ability to have a global view of the network at a central controller, programmability which helps dynamic network configuration, and virtualization to provide resource sharing and isolation, are factors which make it the ideal technology for the smart city (Rahouti et al., 2020). The introduction of Software-Defined Wireless Sensor Networks (SDWSN) to alleviate management complexity in WSN is also an added advantage (Jurado-Lasso et al., 2022). A study by Cedillo-Elias et al. (2018) is an example of how secure and reliable e-government services are possible with the use of SDN and cloud. The authors experimented with the state of Jalisco as part of transforming the Gudalajar metropolitan area into a smart city, to resolve the issue of long queues and energy consumption at tax collection offices during peak hours. They installed IoT devices in the network device and ensured uninterrupted connection to online financial services with the use of SDN. An SDN-based framework for satellite communication in smart cities is proposed by Priyadarsini et al. (2020). In a research by Alzubi et al. (2022), sensitive data from IoT devices in a smart city is sent from IoT to the SDN controller and then to the cloud, where the application resides. If the IoT device doesn't trust its neighbor, the controller gives another trustworthy route for the packet thereby ensuring the privacy of sensitive data. However, since the whole packet is transferred to the controller, the controller will be overloaded with a huge amount of packets considering the enormous amount of data that will be generated from IoT devices in real smart city scenarios. A Smart City Resilient System (SCRS) is presented by AlZoman and Alenazi (2020) to provide throughput QoS by detecting link failures using SDN. To strengthen the confidence of IoT-based smart city services by protecting data from eavesdropping, a multi route transmission scheme based on a k-n secret-sharing mechanism and SDN structure is proposed by Yuan et al. (2020). Under this scheme, the data is encoded into several parts and delivered to different parts through multiple routes calculated by an SDN controller, therefore protecting the data from leakage even if some of the routes may be compromised. A traffic-light scheduling framework using the deep reinforcement learning technique is proposed by Kumar et al.

(2022). It helps in balancing the traffic flow and preventing congestion in the dense regions of the city via a software-defined control interface. The authors propose an SDN-based architecture to monitor the traffic conditions and it generates the traffic light control signal (Red/Yellow/Green) accordingly. Ibrar et al. (2022) put forward a novel Reliability-Aware Flow Distribution Algorithm (RAFDA) in an SDN-based IoT network. The algorithm assigns more traffic to the more reliable links and vice versa, at the same time considering factors such as traffic load, bandwidth allocation, link utilization, and end-to-end delay. All of the above studies are simulation-based researches.

2.2.3 SDN in Delay QoS

SDN is currently realized as the primary technology to simplify network control and promote QoS provisioning (Dezfouli et al., 2019). By putting throughput, latency, jitter, and packet loss between the AP and STA metrics into the Weighted Sum Model Multi Decision Criteria Making (MCDM) algorithm along with the use of a Recurrent Neural network to predict future RSSI, handover is triggered from an SDN framework developed by themselves named 'HuMOR', in order to optimize AP load in the network (Zeljković et al., 2019). By using an SDN controller to get network parameters such as queue buffer size, queue bandwidth, flow number, and propagation delay, queueing delay and end-to-end delay estimation are done by Haiyan et al. (2016). Even though the authors aim to enhance latency QoS in delay-sensitive applications such as real-time video and online surgery, no specific algorithm is proposed to reduce the delay. These metrics data are retrieved from monitoring data from the APs monitoring block by an SDN controller. Abbou et al. (2021) check if a deterministic end-to-end delay is possible with the use of SDN. With the use of SDN along with fog computing and blockchain security, a low-latency, secure and reliable decision-making algorithm is proposed by Ren et al. (2022) to ensure important QoS in healthcare IoT. The simulation results of this research show the suitability of SDN based network in emergency alarm service in e-health. Different SDN-based research for enhancing security in mobile wireless networks are studied by Ding et al. (2014). In a research by Babu and Duttagupta (2020), multiple services demanding different QoS such as a disaster management service demanding stringent delay but less bandwidth while other applications that tolerate higher delay is considered. SDN controller is used to identify the Type of Service of data packets and accordingly assign different flow paths to minimize the delay. Chin et al. (2018) propose a framework to measure and minimize end-to-end delay using SDN. An optimal path selection scheme is used to prioritize network traffic based on the priority of data transmitted. However, their study demands gathering much information from all over the network. A similar flow rerouting mechanism is proposed by Takia et al. (2020). But the scheme mainly addresses queueing delay monitoring and also the controller periodically enquires queueing status from all switches in the network, which will be an overwhelming amount of data to be processed at the controller. Research conducted by Aswanth et al. (2021) tries to ensure minimum end-to-end delay and bandwidth in telemedicine by flow modifications in flow tables of switches using SDN. However, probe packets are sent from the controller to estimate link delays. Danielis et al. (2017) propose theoretical algorithms for dynamic flow migration that migrate paths at runtime to accommodate the new flow without interrupting ongoing flows. SDN is used for dynamic flow updates for reducing end-to-end delay in industrial applications. But the study doesn’t give any numerical verifications or simulation results and did not consider QoS also. Kurungadan and Abdrabou (2021) present an SDN-based algorithm for selecting WiFi nodes based on packet latency to be handed over to another AP.

Using SDN, flexibility in network configuration and resource control can be achieved to a large extent (Kreutz et al., 2015). The separation of the network data plane and control plane paves the way to control a network by software functions from a central controller without the involvement of many middleboxes. This thesis uses the SDN concept to control data traffic delay in a smart city wireless network consisting of several WiFi networks with overlapped coverage. The SDN controller decides which Access Point (AP) the wireless node will be connected to, as opposed to the case where the node itself makes this decision in traditional networks (IEEE Standard for Information Technology, 2012). Moreover, the decision is not based on Received Signal Strength Indicator (RSSI) values, which may result in uneven load distribution among different access points (Yen et al., 2009). Furthermore, most of the SDN-based previous studies on this subject either probe the network for link delay information (Aswanth et al., 2021) or periodically retrieve network data from switches or/and from the network to store at the controller (Chin et al., 2018; Haiyan et al., 2016; Ren et al., 2022; Takia et al., 2020) or forward the entire data packet to the controller (Alzubi et al., 2022; Babu & Duttagupta, 2020). This could impact the network performance by increasing data load or huge storage demand at the controller. In our study, the controller continuously monitors the network's state and redistributes the clients over different access points based on their End-to-End (E2E) delay requirements. Only some packet-in event messages are received from the switches which won't cause any storage overhead or data overload in the network or controller. Although some recent amendments to the IEEE 802.11 standards (IEEE 802.11k/v) (IEEE Standard for Information Technology, 2011; IEEE Standard for Information Technology, 2008) provide signaling messages for the WiFi nodes to change their AP attachments, these new standards usually are not implemented over IoT wireless sensor nodes since they are designed to be simple with a low cost.

Chapter 3: Analysis and Proposed Algorithms

3.1 System Model

This research addresses a smart city outdoor setting where a city-wide WiFi network consisting of many APs with overlapped coverage supports different services via IoT communications, as shown in Figure 6. In outdoor (also indoor) scenarios, overlapped coverage is used to make sure that the clients are able to receive sufficient signal strength anywhere given the obstacles in the surroundings (Jahromi et al., 2015). The traffic is assumed to be generated by different IoT devices serving numerous applications supported by various wireless technologies such as Zigbee, Bluetooth, LoRa, and others. These devices are connected to low-cost IoT gateways equipped with WiFi transceivers supporting the legacy IEEE 802.11n standard without any other amendments such as IEEE 802.11 v/k. Some applications are assumed to be delay-sensitive (e-health, e-governance, industrial control system, real-time applications), whereas others are delay-tolerant (routine data backup). The gateways are assumed fixed in their locations or have limited mobility. The city-wide WiFi network consists of APs, where the adjacent ones are tuned to non-overlapping radio channels. An SDN controller is assumed to communicate with all APs in the network and obtain information about the packets transmitted over any WiFi radio channel. It can also communicate with a WiFi gateway via a software agent that reports the packet arrival rate at the gateway. Also, it is assumed that the controller can seamlessly perform handover to an already connected gateway to another AP without reconfiguration or disconnection.

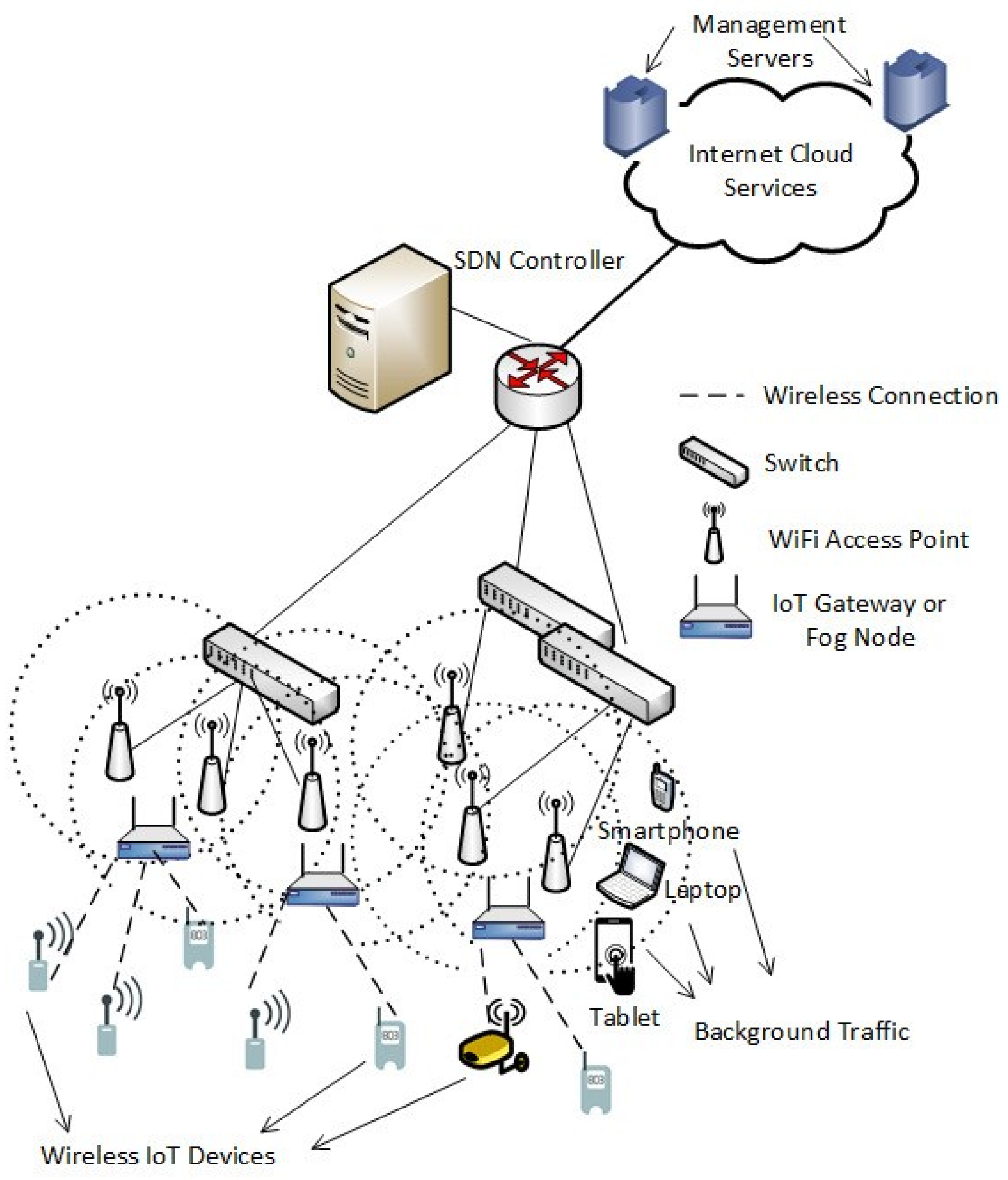
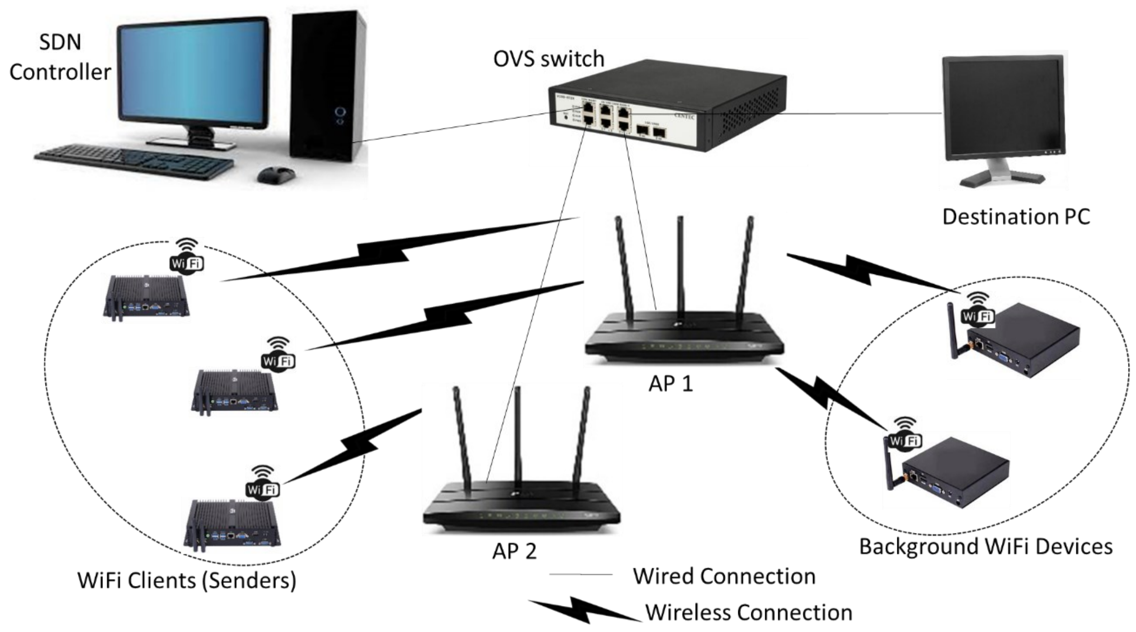


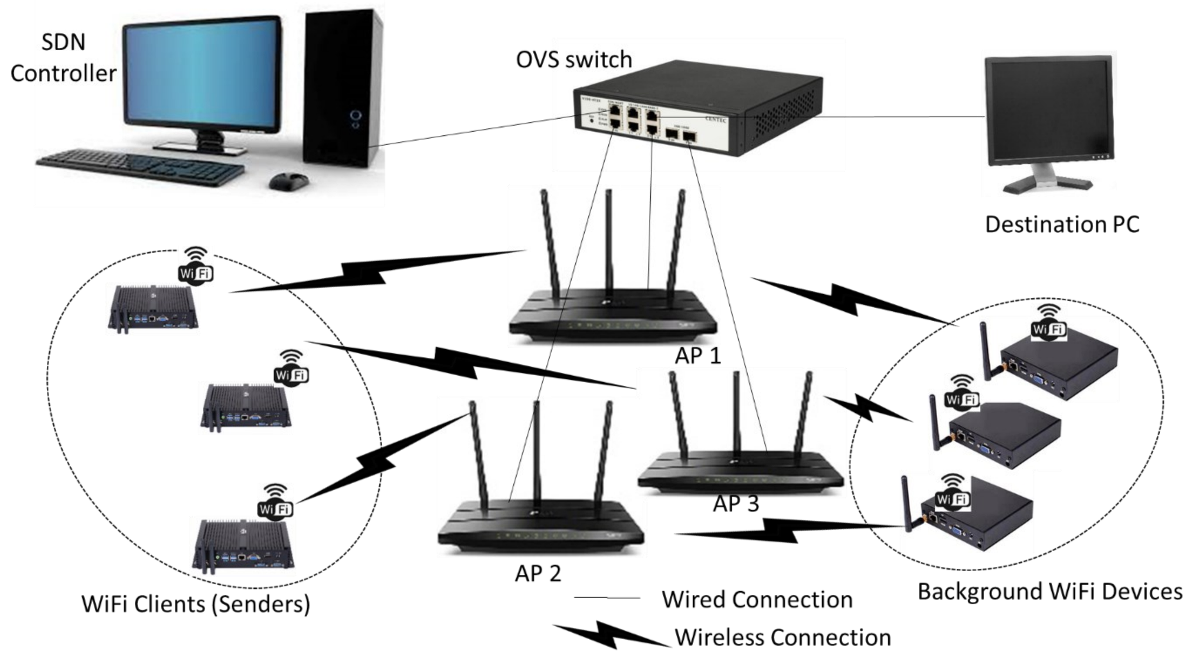
Figure 6: System model

3.2 Experimental Setup

Two experimental setups are used to mimic the system model and validate the proposed algorithms, as revealed in Figure 7. Two experiments were done with the first experimental setup shown in Figure 7(a), whereas the third experiment used the second experimental setup. Both setups use real hardware, not computer simulators. Different hardware equipment and software packages are employed in both setups, as described in the sequel.



(a)



(b)

Figure 7: Experimental setup (a) first experimental setup (b) second experimental setup

3.2.1 Hardware Equipment

APs are emulated using TP-LINK wireless dual-band routers in both experimental setups. For the first setup, two APs are used (marked as AP1 and AP2), whereas, for the second setup, a third AP is used (AP3). Three PCs emulate three IoT gateways in both setups. The WiFi adapters used in the three PCs (WiFi clients) support IEEE 802.11n. The data generated by the three gateways are sent to the destination PC, emulating a management entity (server) in the real world. Apart from the three gateway nodes, two other background WiFi clients (PCs) are used to emulate background data traffic that shares the city WiFi network but is not a part of the smart city sensed data traffic. All the used computers run Ubuntu 19.10 or 20.04 operating system. All Ethernet connections are made via Gigabit Ethernet ports.

3.2.2 Software Packages

In order to realize the SDN environment in the experiments, various software packages are used, namely, Ryu, Open vSwitch (OVS) and Empower-5G framework. The SDN controller runs on a separate computer (PC) and supports the OpenFlow protocol. The controller uses the Open Flow protocol to communicate with the OVS software, which runs on a multi-Ethernet port computer to emulate a soft SDN switch (referred to hereafter as the OVS switch) and also on the APs' firmware, as shown in Figure 7(a) and Figure 7(b). The SDN controller, the APs, and the destination PC are connected through the OVS switch.

3.2.3 Ryu

Ryu is a component-based software-defined networking framework (RYU SDN Framework—Ryubook 1.0 documentation, n. d.) used to create the SDN controller functionality. The Ryu Controller provides software components, with well-defined Application Program Interfaces (APIs), that make it easy for developers to create new network management and control applications. This component approach helps organizations customize deployments to meet their specific needs; developers can quickly and easily modify existing components or implement their own to ensure the underlying network can meet the changing demands of their applications.

3.2.4 Open vSwitch (OVS)

Open vSwitch is an open source, virtual multilayer software switch that can be run in virtual machine environments. It is licensed under the open source Apache 2 license. It provides access to the virtual networking layer with standard control and visibility interfaces, and enables distribution across multiple physical servers. Open vSwitch supports multiple Linux-based virtualisation technologies (Bouguerra, 2021).

The main components of OVS are the following (Open VSwitch 3.0.90 Documentation, n. d.):

- Ovs-vswitchd, a daemon that implements the switch, along with a companion Linux kernel module for flow-based switching.
- Ovsdb-server, a lightweight database server that ovs-vswitchd queries to obtain its configuration.
- Ovs-dpctl, a tool for configuring the switch kernel module.
- Scripts and specs for building RPMs for Red Hat Enterprise Linux and deb packages for Ubuntu/Debian.
- Ovs-vsctl, a utility for querying and updating the configuration of ovs-vswitchd.
- Ovs-appctl, a utility that sends commands to running Open vSwitch daemons.

Open vSwitch also provides some tools (Open VSwitch 3.0.90 Documentation, n. d.):

- Ovs-ofctl, a utility for querying and controlling OpenFlow switches and controllers.
- Ovs-pki, a utility for creating and managing the public-key infrastructure for OpenFlow switches.
- Ovs-testcontroller, a simple OpenFlow controller that may be useful for testing (though not for production).
- A patch to tcpdump that enables it to parse OpenFlow messages.

When building software-defined networks, an SDN controller is required to take care of the high level network policies to define things like how virtual networks should communicate with each other, and which flows should be permitted or blocked. OVS is the component that enforces these policies via OpenFlow.

3.2.5 Empower-5G Framework

Empower-5G framework is an open-source SDN platform for heterogeneous RANs consisting of: (i) a radio access agnostic Application Programming Interface (API) that clearly separates control plane from user plane; (ii) a software agent able to operate with several radio access nodes (Wi-Fi and LTE); and (iii) a proof-of-concept Software-Defined Radio Access Network (SD-RAN) Controller implementation. (Coronado et al., 2019).

The 5G-EmPOWER Operating System consists of the following components:

- Empower-core, the core library used to develop the 5G-EmPOWER controller.
- Empower-runtime, the Python-based 5G-EmPOWER Controller. This allows network apps to control Wi-Fi APs and LTE eNBs using either a REST API or a Python API.
- Empower-lvap-agent, the 5G-EmPOWER Wi-Fi agent. This agent allows controlling Wi-Fi access points using the empower-runtime.
- Empower-enb-agent, the 5G-EmPOWER LTE agent library. This agent allows controlling LTE eNBs using the empower-runtime.
- Empower-vbs-emulator, a basic dummy eNB implementing part of the 5G-EmPOWER southbound interface and meant to help in the development of the controller when a real eNB is not available.
- SrsLTE, a branch of srsLTE with the 5G-EmPOWER eNB agent.
- Empower-openwrt-packages, the 'empower-lvap-agent' package for OpenWRT 19.07.
- Empower-openwrt, a branch of OpenWRT 19.07 including some Kernel patches necessary for the correct operation of the 'empower-lvap-agent'.
- Empower-config, the configuration files for the Wi-Fi WTPs.

- Docker, some docker files of the main 5G-EmPOWER components.

3.2.6 Light Virtual Access Point (LVAP)

In wireless network, mobility of users is a matter of concern. Ideal case calls for uninterrupted service to users moving from the vicinity of one AP to another. Although TCP/IP is the primary network communication protocol, it doesn’t account for mobility of devices. Several other protocols are introduced to address mobility of users in wireless networks. IPv6 by IETF is the most common protocol to enable mobility in wireless network, hence it takes care of handover of clients from one AP to other . However, during this handover the client cannot receive or send packets resulting in high latency and packet loss (Proceedings of the 11th Annual International Conference on Mobile Computing and Networking, n. d.).

In SDN, the concept of Virtual Access Point (VAP) abstraction in SDN enables seamless handover while connected users maneuver voluntarily. With this abstraction, multiple clients connected to a single physical AP are treated as a set of logically isolated clients connected to different ports of a switch (Suresh et al., 2012). Light Virtual Access Point (LVAP) is one of the most popular abstraction simplifying handover task in wireless networks. The LVAP acts as a state management entity for the wireless devices connecting to APs and it absorbs the complexities of the IEEE 802.11 protocol stack such as handling of client associations, authentication, and handovers (Riggio et al., 2015). The APs send every new probe requests from clients to the controller, which will trigger probe response frame along with the creation of an LVAP at the requesting AP (GitHub, n. d.). Hence a new LVAP is created at the AP when a new client connects. Consequently, an AP will have as many LVAPs as the number of clients connected to it at the moment. Removing an LVAP from a WTP and instantiating it on another WTP effectively results in a handover, thereby simplifying network management and introducing seamless mobility support (GitHub, n. d.).

3.3 Proposed Algorithms

This section introduces the three proposed algorithms. Two algorithms aim at selecting the gateway with the highest end-to-end packet delivery latency to undergo a handover to another AP. The first algorithm is mainly based on packet service time estimation, whereas the second relies on queuing analysis. The third algorithm selects the least loaded AP to which the handover of the gateway, selected by any of the first two algorithms, will be performed.

3.3.1 Based on Service Time Estimation

The end-to-end delay, W_j, for a packet of a gateway node j, is given by the service time, V_{t_j}, and the queueing delay, U_{t_j}, as

$$W_j = V_{t_j} + U_{t_j} \qquad \text{Equation (1)}$$

Generally, U_{t_j} varies with the arrival rate and the service time. It is assumed either the arrival processes of all gateways are similar and have almost the same arrival rate _ or the arrival processes have different arrival rates, but the queuing delay is negligible compared with the service time. This happens when the network works in a region sufficiently far from the saturation point (Abdrabou & Zhuang, 2008). Hence, the end-to-end delay variation is captured by the service time variation (i.e., the longer the service time, the longer the end-to-end packet delay). A generalization of the average service time $E[V_{t_j}]$ of IEEE 802.11 mentioned in (Abdrabou & Zhuang, 2008) can be obtained as

$$E\left[V_{t_j}\right] = \sum_{i=1}^{N-1} \gamma_i \left[D_{s_i} + \frac{D_{c_i}}{2}\frac{p_c}{1-p_c}\right] + B(p_c) + D_{s_j} + \frac{D_{c_j}}{2}\frac{p_c}{1-p_c} \qquad \text{Equation (2)}$$

where $\gamma_i = V_{t_i}\lambda$ is the node i queue utilization, p_c is the probability of collision (constant for background nodes and gateways), λ is the rate of packet arrivals, and $B(p_c)$ is the average back off interval given by (Abdrabou & Zhuang, 2008).

$$B(p_c) = \frac{1 - p_c - p_c(2p_c)^{bs}}{1-p_c}\frac{CW_{min}}{2} \qquad \text{Equation (3)}$$

where CW_{min} is the minimum contention window size and bs is the number of backoff stages . The transmission and collision times of a packet of node i are represented by D_{s_i} and D_{c_i}, respectively. They can be expressed as $D_{s_i} = \frac{S}{R_i} + D_{M_1}$ and

$D_{c_i} = \frac{S}{R_i} + D_{M_2}$ respectively, where S is the packet size, R_i is the channel transmission rate for node i, and D_{M_1} and D_{M_2} are constant times related to the IEEE 802.11 protocol operation (Abdrabou & Zhuang, 2008). The proposed algorithm can be outlined as follows in Table 2.

Table 2: Steps for Service time estimation

Step 1	Once a packet arrives at the OVS switch, a Packet-In event is triggered and sent to the SDN controller. The controller obtains the average and standard deviation of the interarrival times of Packet-In events using the recorded arrival time of these events. The average and standard deviation of the interarrival time of Packet-In events correspond to their packet service time counterparts.
Step 2	The gateway that has the maximum average interarrival time will be the candidate for handover to the lowest loaded AP. In the case of the same maximum value is recorded for more than one gateway, the gateway that records the maximum standard deviation among them will be selected for handover. The controller is so programmed that if more than one client with the same maximum average delay and highest standard deviation occur, randomly choose one among them for handover.
Step 3	The interarrival time of Packet-In events will be continuously monitored for the gateway that underwent the handover. Step 2 shall be redone with another AP if the interarrival time is not decreased.
Step 4	The interarrival time of Packet-In events is rechecked by the SDN controller whenever a new gateway is attached to an AP or the traffic load of one of the APs increases because this influences the service time of all the attached gateways to this AP based on Equation (2). Step 2 shall be repeated when the average values of the packet interarrival time of the connected gateways increase.

Figure 8 shows flowchart of the algorithm.

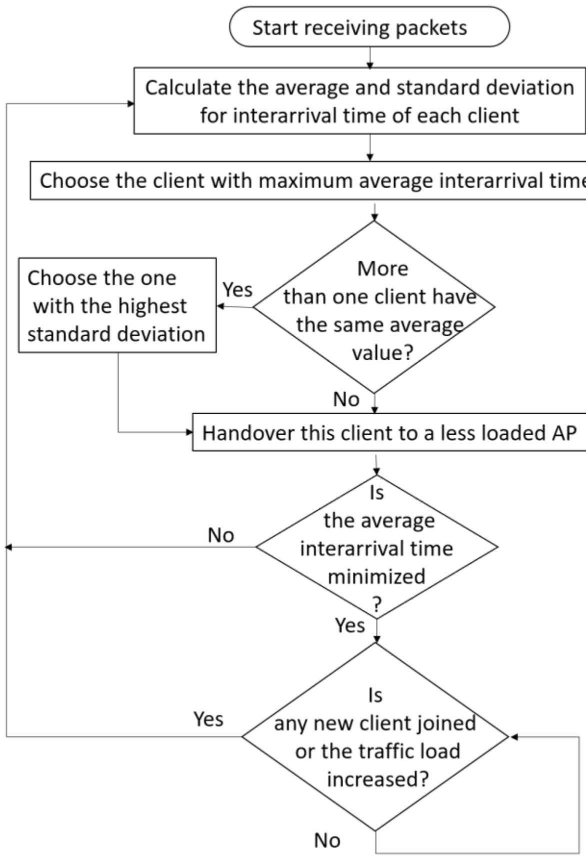

Start receiving packets
Calculate the average and standard deviation for interarrival time of each client
Choose the client with maximum average interarrival time
More than one client have the same average value?
Yes
Choose the one with the highest standard deviation
No
Handover this client to a less loaded AP
Is the average interarrival time minimized ?
No
Yes
Is any new client joined or the traffic load increased?
Yes
No

Figure 8: Service time estimation algorithm flow chart.

3.3.2 Using M/G/1 Analysis

Since the traffic received by a large number of IoT sources (mainly characterized as on-off in nature) is aggregated at the IoT gateway, modeling this aggregated traffic can be approximated to a Poisson arrival process (Cao & Ramanan, 2002). Thus, M/G/1 analysis (Shortle et al., 2018) is used to estimate the end-to-end delay for each IoT gateway using the Pollaczek-Khinchine (PK) formula. The whole process is shown as flowchart in Figure 9.

The mean waiting time for a packet in the queue at a sender gateway *j* according to the PK formula is

$$T_{q_j} = \lambda * \frac{\left(\sigma_{s_j}^2 + E\left(V_{t_j}\right)^2\right)}{2 * \left(1 - \lambda * E\left(V_{t_j}\right)\right)} \qquad \text{Equation (4)}$$

where λ is the packet arrival rate, σ_{s_j} is the standard deviation of the service time at sender j, $E\left(V_{t_j}\right)$ is the mean service time at sender j. From Equation (1) the average end-to-end packet delay, $E\left[W_j\right]$, for a packet sent by node j can be formulated as

$$E\left[W_j\right] = T_{q_j} + E\left[V_{t_j}\right] \qquad \text{Equation (5)}$$

The steps of the algorithm are tabulated in Table 3.

Table 3: Steps for M/G/1 analysis-based algorithm

Step 1	By recording the arrival time of packet-In events with the aid of the OVS, the SDN controller calculates the average and standard deviation of these events' interarrival time. This translates directly to the average and standard deviation of each gateway's packet service time.
Step 2	Each gateway informs the SDN controller about its packet arrival rate through a software agent as a client/server communication. The controller applies the PK formula (4) to estimate the end-to-end delay for each gateway (WiFi client).
Step 3	The client which has the maximum estimated end-to-end delay is selected for handover to an AP loaded with less traffic.
Step 4	The end-to-end packet delay of all the gateways, which recently performed a handover, is continued to be estimated by the SDN controller to check if the delay is lowered. Otherwise, Step 2 shall be redone, but with another AP.
Step 5	The end-to-end packet delay for all IoT gateways (clients) is estimated by the SDN controller when a new gateway joins an AP or an AP traffic load changes. Once the estimated end-to-end delay becomes higher, Step 2 shall be redone.

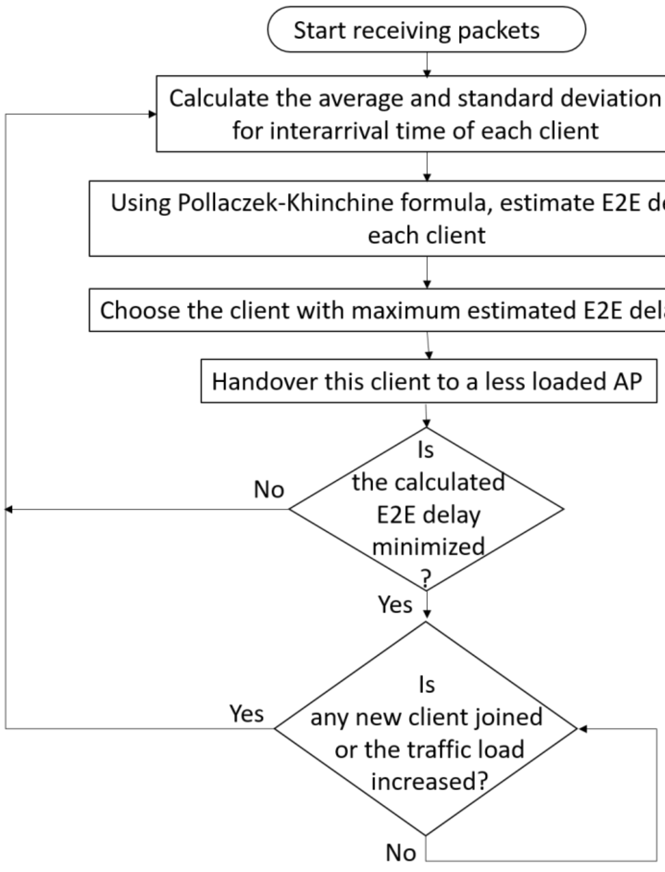

Start receiving packets
Calculate the average and standard deviation for interarrival time of each client
Using Pollaczek-Khinchine formula, estimate E2E delay for each client
Choose the client with maximum estimated E2E delay
Handover this client to a less loaded AP
Is the calculated E2E delay minimized ?
No
Yes
Is any new client joined or the traffic load increased?
Yes
No

Figure 9: The flowchart for the M/G/1 analysis-based algorithm

3.3.3 Finding the Least-Loaded AP

This algorithm complements the aforementioned algorithms since it finds the least loaded AP to which the SDN controller shall carry out a handover. This is performed after finding the gateway with the maximum average interarrival time (as in Subsection 3.3.1) for Packet-In events or the maximum estimated end-to-end packet delay (as in Subsection 3.3.2). The steps are described in Table 4. The integration between the algorithm mentioned in Subsection 3.3.1 and this algorithm is depicted as a flowchart in Figure 10 for illustration purposes.

Table 4: Steps for choosing least loaded AP

Step 1	The SDN controller receives Packet-In events from all the network APs after the packets are delivered to the OVS switch.
Step 2	Using the available information from the packet- In events, the SDN controller extracts the number of packets received from each AP separately over a certain time interval.
Step 3	The AP with the least packet count is identified by the SDN controller as the least loaded AP.
Step 4	The Packet-In events' data is continuously monitored by the SDN controller, which applies Step 2 to update the AP with the least load status.

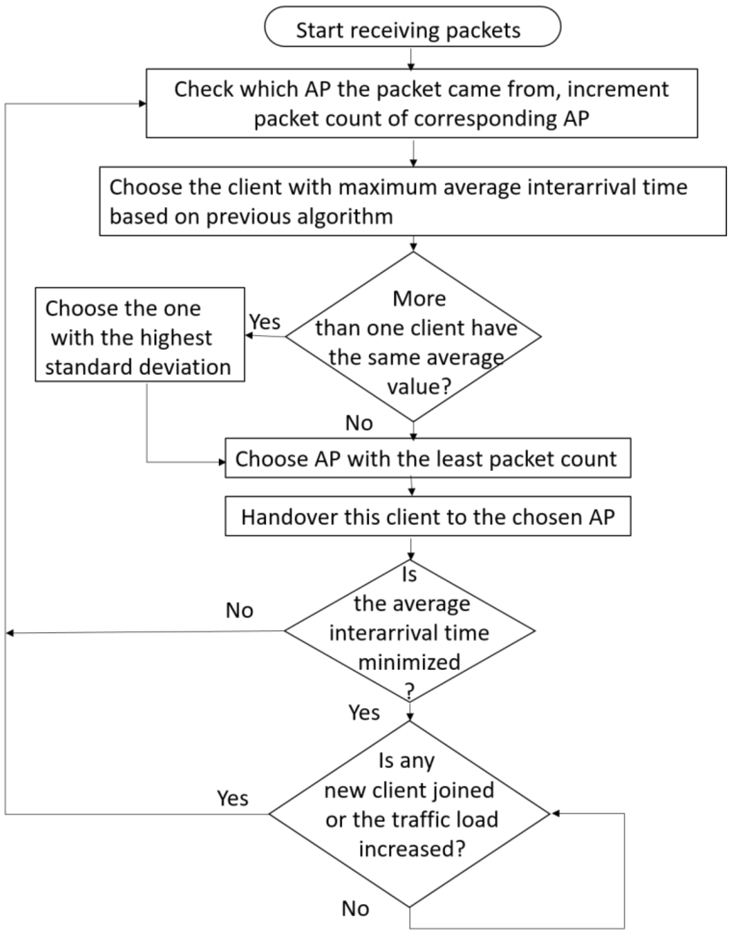

Start receiving packets
Check which AP the packet came from, increment packet count of corresponding AP
Choose the client with maximum average interarrival time based on previous algorithm
More than one client have the same average value?
Yes
Choose the one with the highest standard deviation
No
Choose AP with the least packet count
Handover this client to the chosen AP
Is the average interarrival time minimized ?
No
Yes
Is any new client joined or the traffic load increased?
Yes
No

Figure 10: The scheme for choosing least loaded AP and highest E2E client

In order to track changes in the network, all of the three algorithms check if there is any increase in traffic load at the access points apart from checking if any new clients joined the network in the last step.

3.4 Experimental Procedure

The seamless handover of WiFi gateways from one AP to another is performed using the Empower-5G framework (Coronado et al., 2019), which is an open SDN platform for radio access networks. This framework is installed and run on the SDN controller PC. When APs are registered with Empower-5G, it establishes a WiFi network of a certain Service Set IDentifier (SSID) for each of the registered APs. When a WiFi gateway connects to any of these APs, an agent corresponding to this gateway is installed at the AP by Empower-5G, known as a Light Virtual Access Point (LVAP) agent. Thus, handover is performed by transferring these virtual agents between two APs (i.e., removing the LVAP agent from the current AP and creating a new LVAP agent in another AP) without changing the WiFi configuration of the gateway (Riggio et al., 2013). The data traffic received by the gateway from IoT devices is emulated using the RUDE/CRUDE software tool, a UDP traffic generator running in a client/server fashion. The RUDE/CRUDE tool is also used to generate the background traffic with different data rates. The synchronization of senders and receivers is crucial to measure the end-to-end delay precisely. Hence, the Precision Time Protocol (PTP) runs on all the senders and destination computers.

Chapter 4: Results and Discussions

4.1 Results

Extensive laboratory experimentation is used to validate the performance of the algorithms explained earlier. The results of around 30 samples are recorded for each experiment.

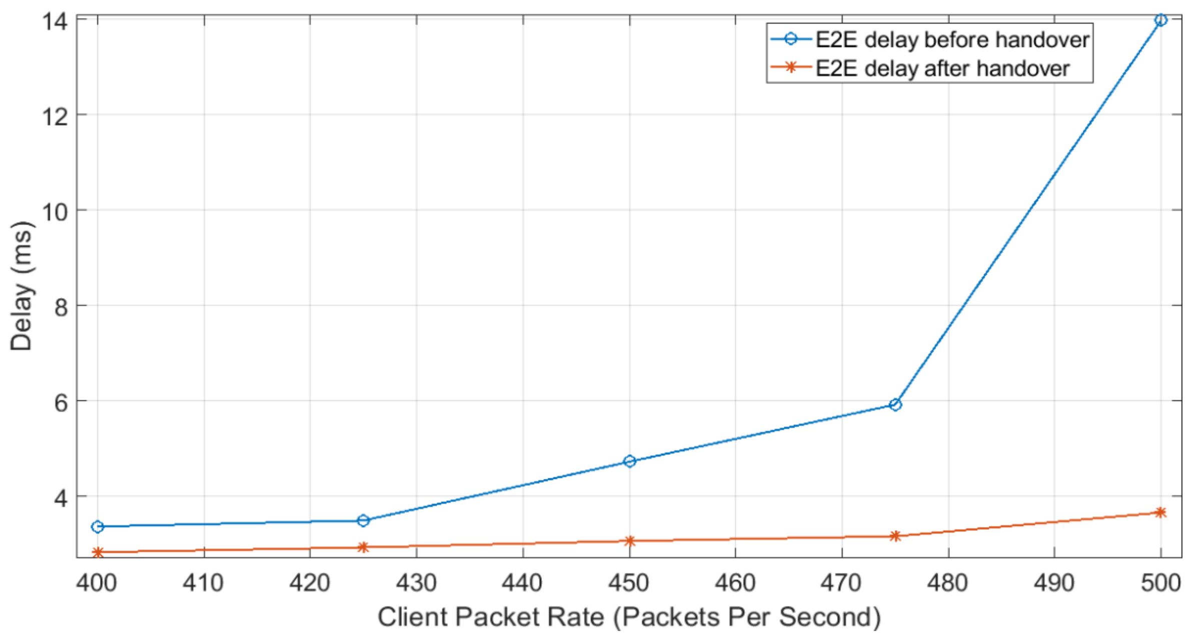
4.1.1 Handover Based on Service Time Estimation

Three performance indicators are used to validate the algorithm presented in Section 3.3.1. In the first indicator, the average end-to-end packet delay for the emulated gateways before handover is compared with its value after handover by varying the gateway traffic rate and a constant background traffic rate of 120 packets/s (each packet is 1450 Bytes). Figure 11(a) shows a considerable decrease in the end-to-end packet delay after handover, particularly when the gateway traffic rate is high.

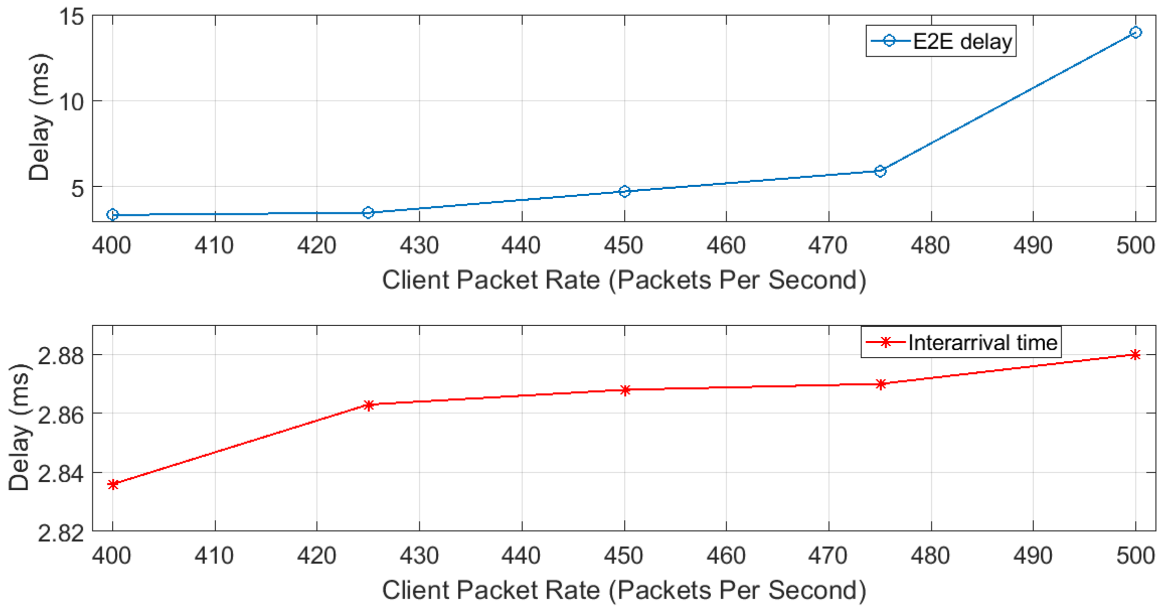
In the second indicator, the relationship between varying interarrival time of Packet-In events at the SDN controller and the end-to-end delay is shown in Figure 11(b) for an extensive range of gateway traffic rates.

The third indicator addresses the applicability and correctness of performing the handover based on the service time estimation algorithm by comparing it with the decision of an observer who can record the end-to-end delay at the destination computer (ideal decision). As depicted in Figure 11(c), the number of times the algorithm decision was different from the ideal one decreases when the gateway traffic rate increases.

Furthermore, more experiments are conducted to validate the algorithm's performance against the previously mentioned performance metrics when the gateway traffic rate is kept constant at 475 packets per second while the background traffic load of the network is varied.

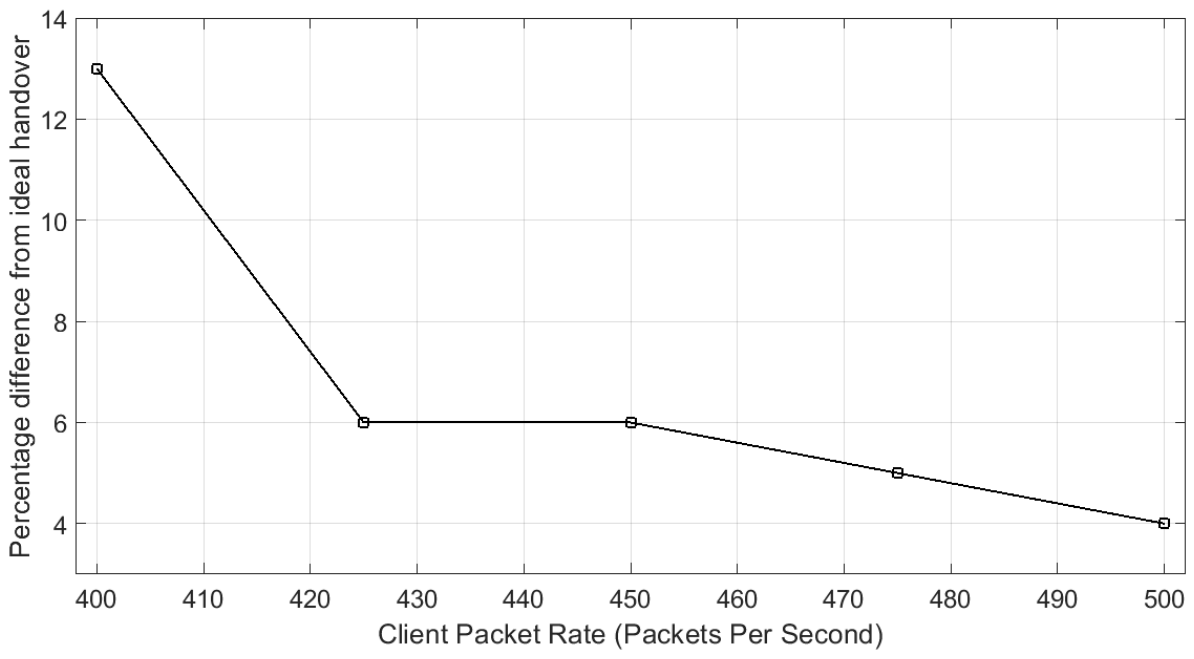


(a)



(b)

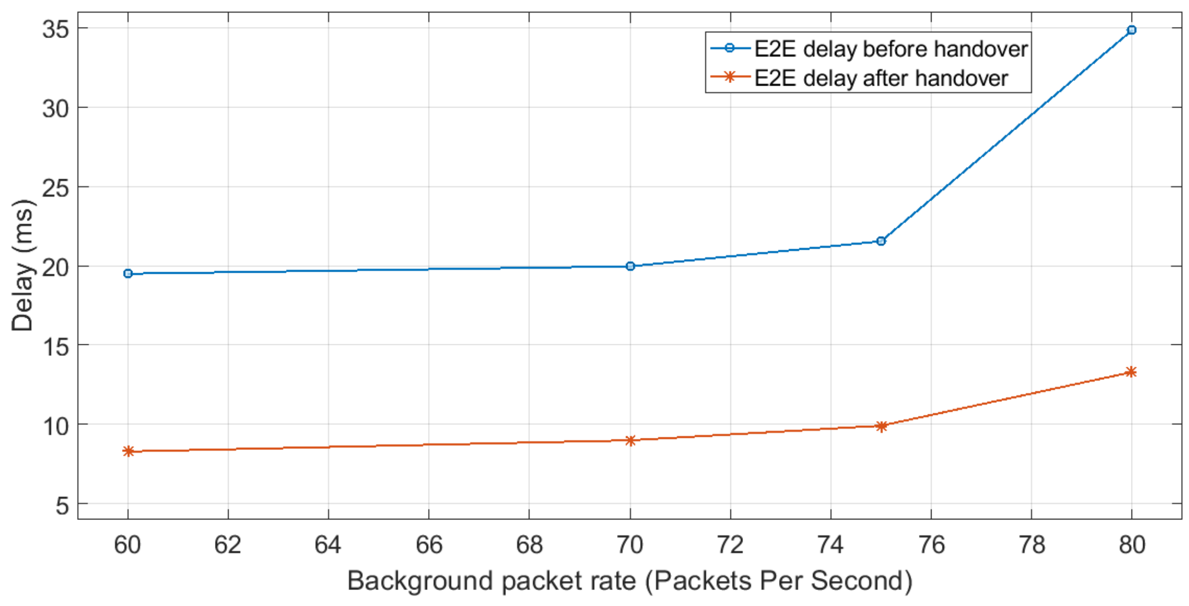
Figure 11 : Service time estimation-based algorithm performance with different client rates (a) End-to-end delay before and after handover (b) Variation of end-to-end delay with interarrival time at the SDN controller (c) Ideal handover and algorithm-based handover percentage difference



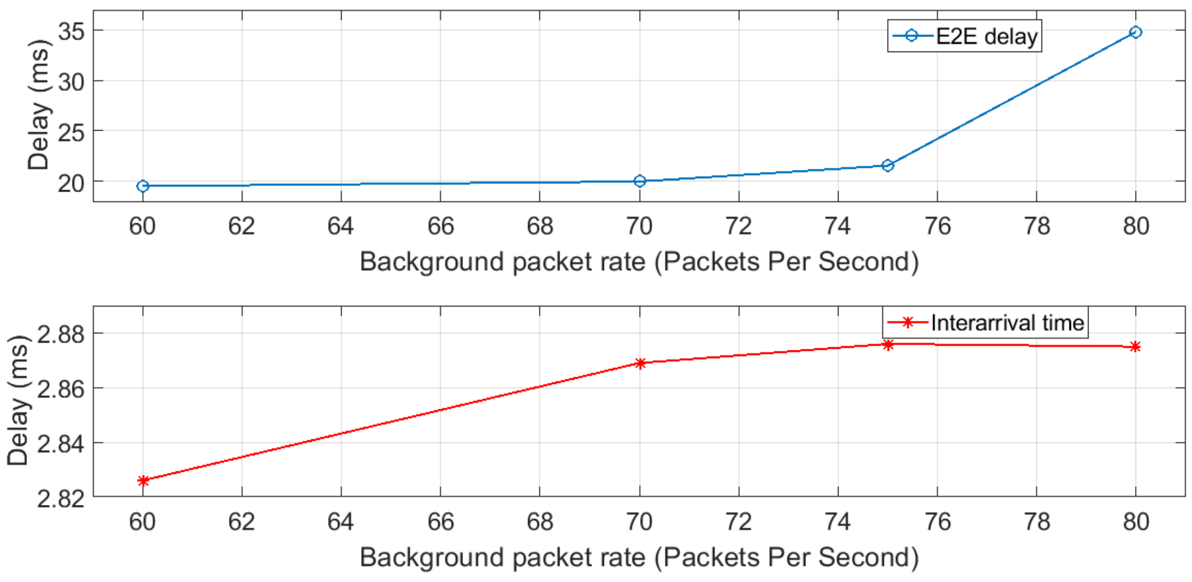
(c)

Figure 11 : Service time estimation-based algorithm performance with different client rates (a) End-to-end delay before and after handover (b) Variation of end-to-end delay with interarrival time at the SDN controller (c) Ideal handover and algorithm-based handover percentage difference (continued)

As shown in Figure 12(a), a notable end-to-end packet delay difference between the cases before and after handover is observed. Figure 12(b) reveals that the variation of the end-to-end delay is affected by changing the packet interarrival time at the controller in general. However, with a large background traffic volume, a small variation in the packet interarrival times leads to a considerable change in the end-to-end delay. Figure 12(c) shows that the algorithm handover decision approaches the ideal handover decision when raising the load of the network background traffic.

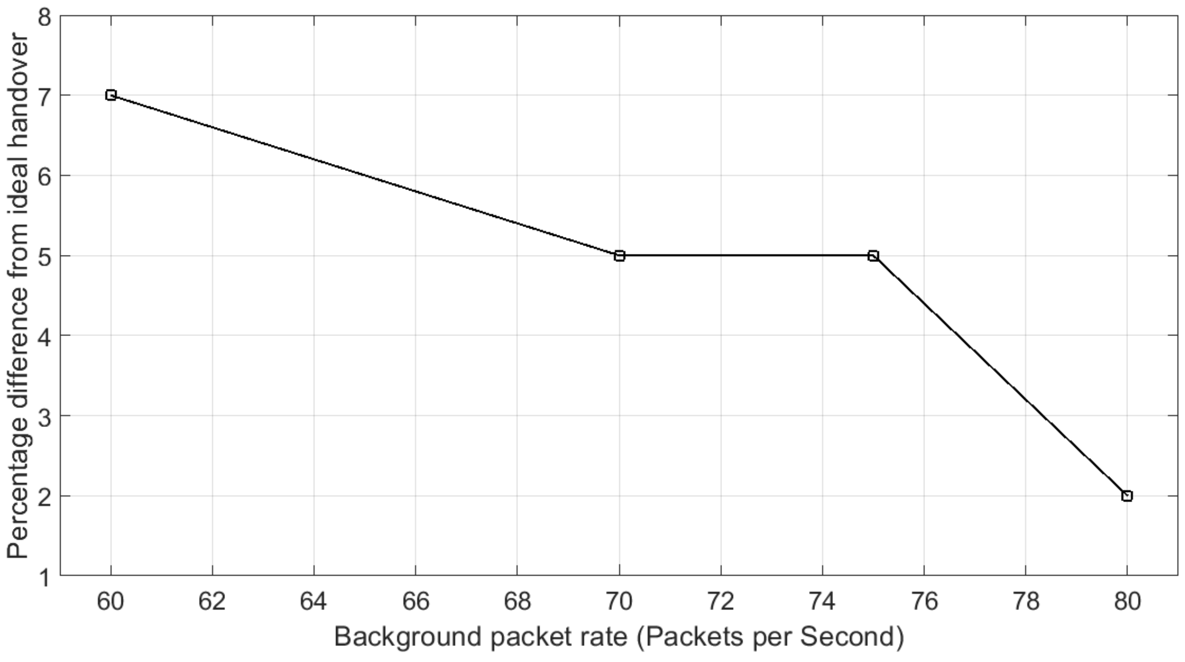


(a)



(b)

Figure 12 : Service time estimation-based algorithm performance with different traffic loads (a) End-to-end delay vs. background traffic load before and after handover (b) End-to-end delay and interarrival time at the SDN controller with background traffic load (c) Ideal handover and algorithm-based handover percentage difference

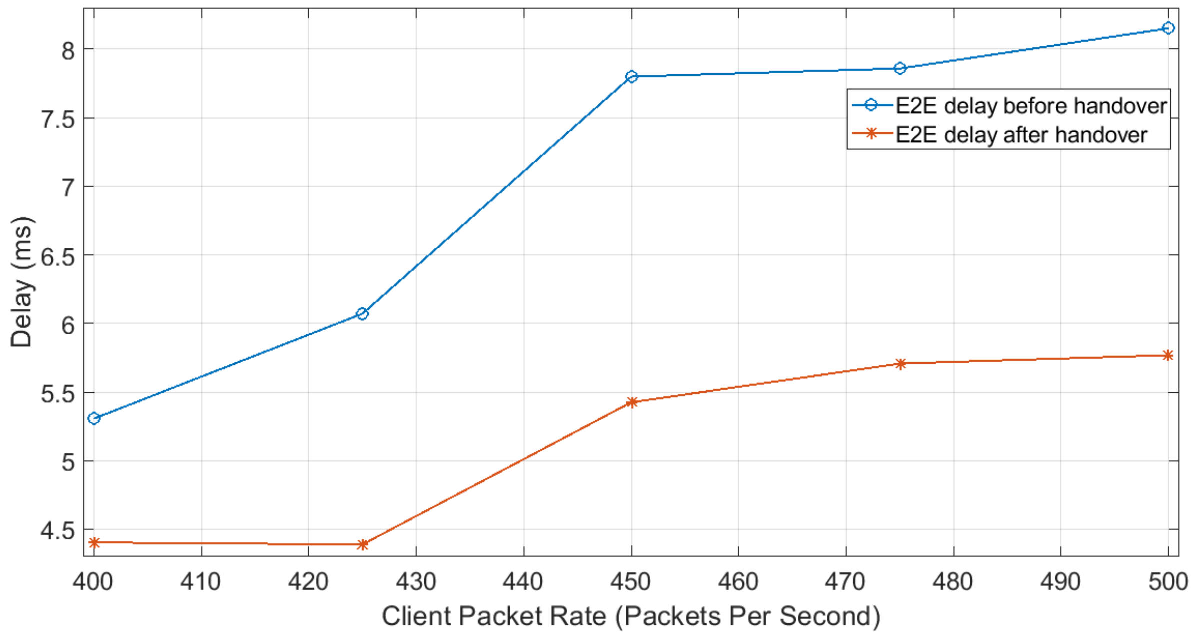


(c)

Figure 12 : Service time estimation-based algorithm performance with different traffic loads (a) End-to-end delay vs. background traffic load before and after handover (b) End-to-end delay and interarrival time at the SDN controller with background traffic load (c) Ideal handover and algorithm-based handover percentage difference (continued)

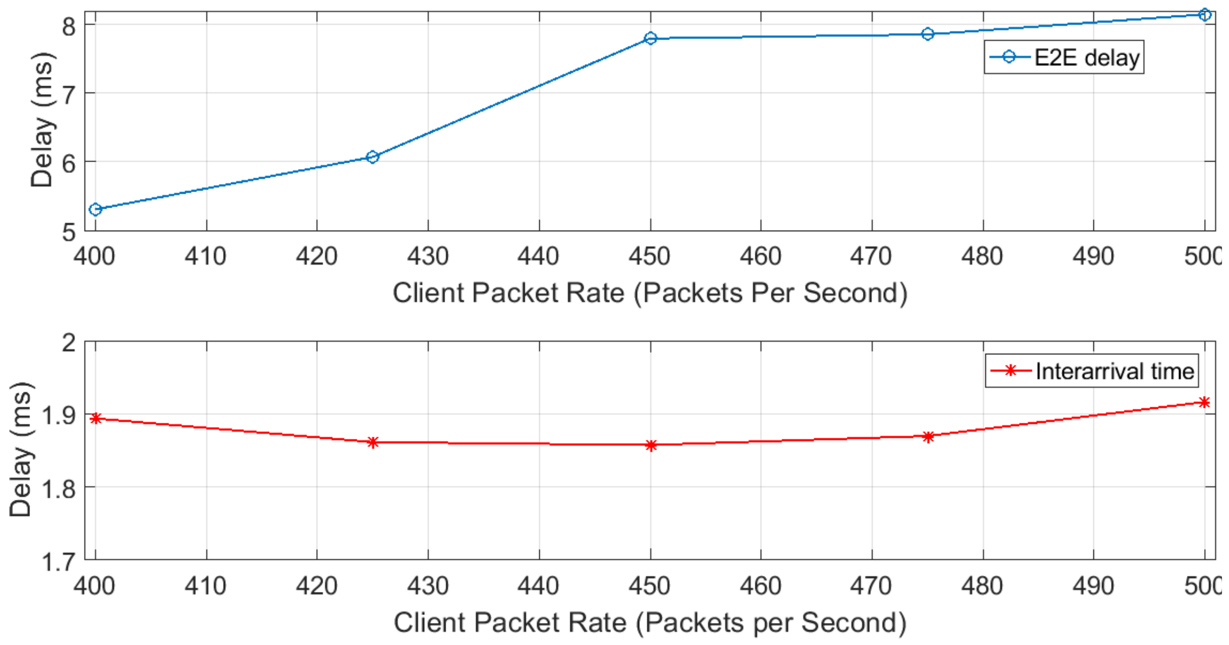
4.1.2 Handover Based on M/G/1 Analysis

This section presents the results of performing the handover using the algorithm introduced in Section 3.3.2. Figure 13(a) shows the variation of the end-to-end delay as the gateway (WiFi client) packet rate changes. As Figure 13(a) reveals, the algorithm is successful in significantly decreasing the end-to-end packet delay after handover. However, it is depicted in Figure 13(b) that the change of the packet interarrival time at the SDN controller with Poisson-distributed traffic arrivals does not change significantly with the arrival rate. Thus, it does not reflect the change of the end-to-delay, as shown previously with a fixed arrival rate (Figure 11(a)). Similar to the first algorithm, the performance of this algorithm in terms of end-to-end packet delay is investigated when the background traffic load is varied, as depicted in Figure 14. Figure 14(a) shows a significant reduction in end-to-end packet delay after handover is performed. However, the difference between the end-to-end delay before and after handover becomes smaller as the background traffic volume increases. The results revealed in Figure 14(b) affirm that the variation of the end-to-end packet delay is not reflected by the packet interarrival time at the SDN controller, as noticed in Figure 13(b). In addition, a comparison between the measured end-to-end delay before handover and the estimated end-to-end delay calculated by the SDN controller is depicted in Figure 13(c). The difference in the delay values is attributed to the measurement error due to the time taken by the operating system kernel to set a timestamp of the incoming and outgoing packets (Hernandez & Magana, 2007).

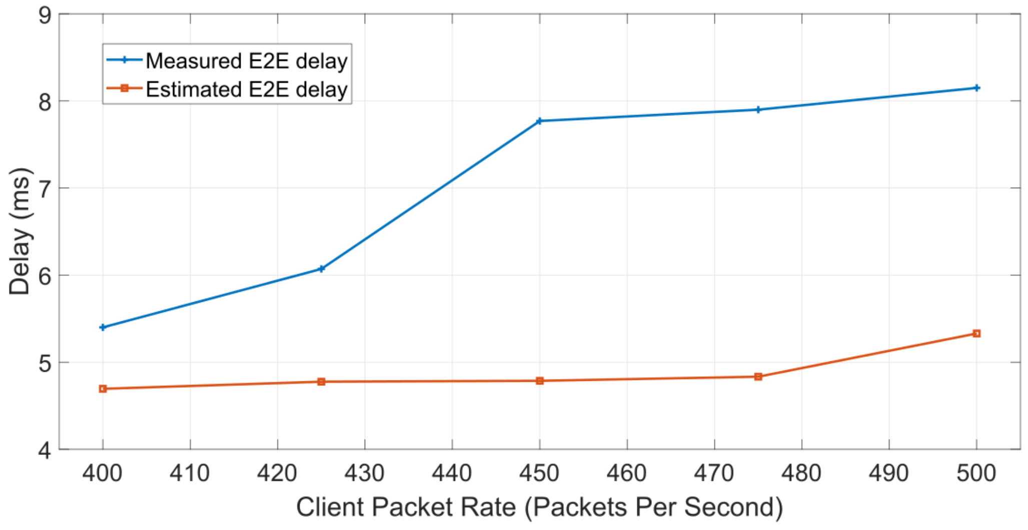


(a)

Figure 13 : Handover based on M/G/1 analysis with different client rates (a) End-to-end delay before and after handover (b) Variation of end-to-end delay with interarrival time at the SDN controller (c) Measured and estimated end-to-end delay

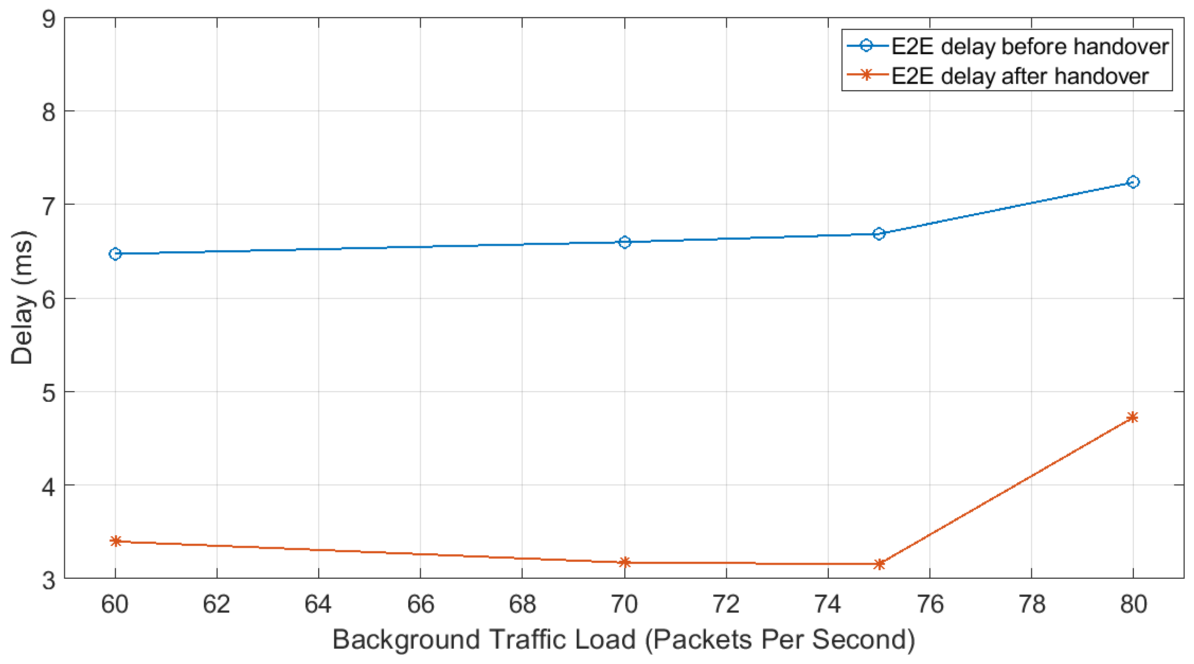


(b)

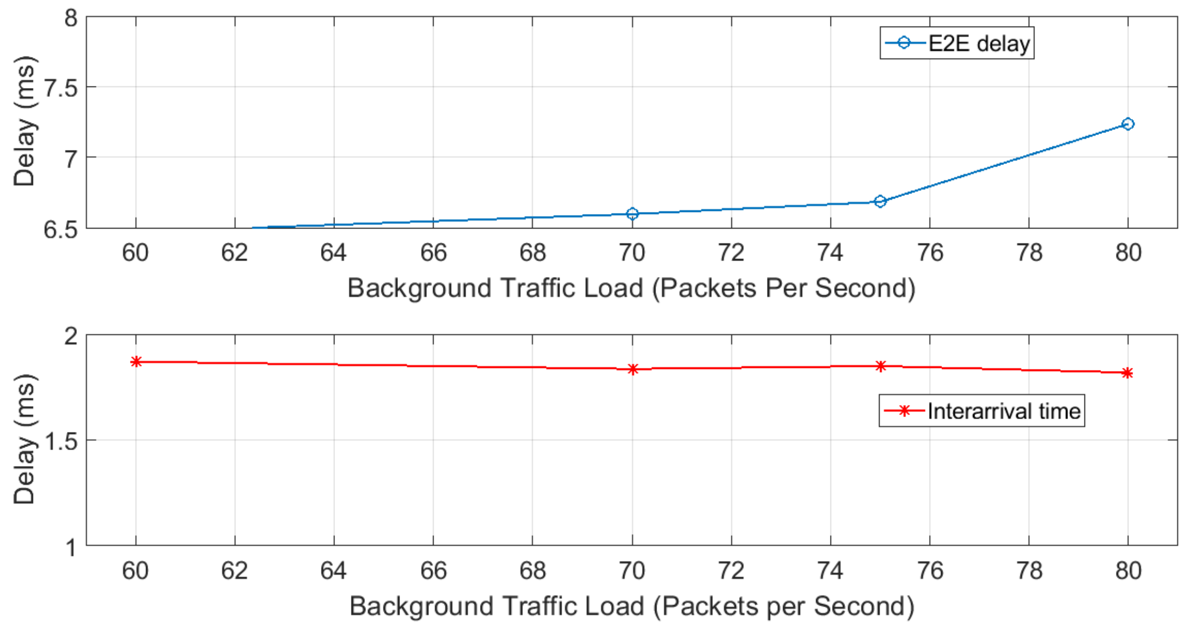


(c)

Figure 13 : Handover based on M/G/1 analysis with different client rates (a) End-to-end delay before and after handover (b) Variation of end-to-end delay with interarrival time at the SDN controller (c) Measured and estimated end-to-end delay (continued)



(a)



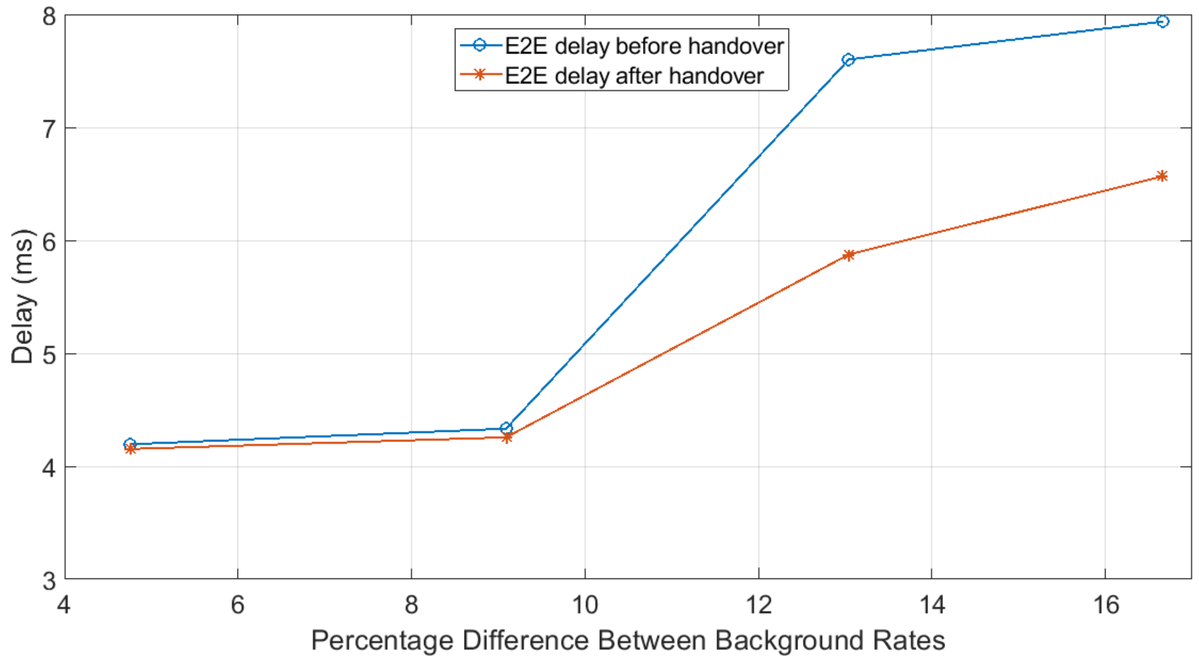
(b)

Figure 14 : Handover based on M/G/1 analysis with different traffic loads (a) End-to-end delay vs. background traffic load before and after handover (b) End-to-end delay and interarrival time at the SDN controller with background traffic load

4.1.3 Handover to AP with Least Load

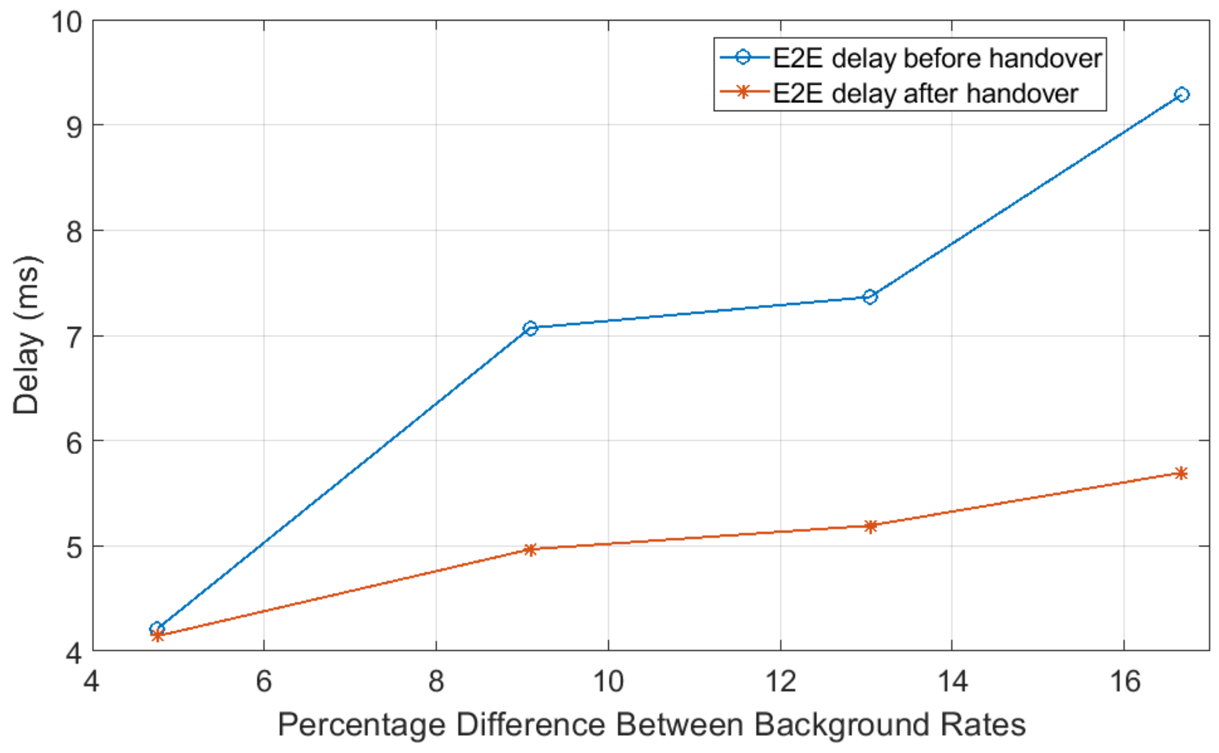
Figure 15 depicts the performance of the AP selection algorithm. The performance is evaluated by comparing the end-to-end packet delay of the gateways (clients) before and after handover for an algorithm randomly choosing the AP, as in

Figure 15(a), and the algorithm proposed in Section 3.3.3, as in Figure 15(b). The percentage difference in the traffic load between the available APs is varied in both figures, and the corresponding end-to-end packet delay is recorded. It is evident from Figure 15(a) and Figure 15(b) that the reduction in the end-to-end delay packet delay after handover is significantly larger when the AP is chosen using the proposed algorithm. An exception to this is when the percentage difference in traffic load between the available APs is very low.



(a)

Figure 15 : Comparison of algorithm-based and random AP selection (a) End-to-end delay before and after handover of with random AP selection (b) End-to-end delay before and after handover of with AP selection algorithm



(b)

Figure 15 : Comparison of algorithm-based and random AP selection (a) End-to-end delay before and after handover of with random AP selection (b) End-to-end delay before and after handover of with AP selection algorithm (continued)

4.2 Discussion

The presented results indicate that the three proposed algorithms can efficiently perform their intended functions. As evident from Figure 11(a) and Figure 12(a), the end-to-end packet delay of the IoT gateways (WiFi clients) can be significantly reduced by performing handover based on an algorithm running on the SDN controller that estimates the service time by measuring the packet interarrival time of Packet-In events. For a constant rate of gateway traffic, which often happens with synchronized periodic IoT sources (Navarro-Ortiz et al., 2020), Figure 11(b) and Figure 12(b)reveal that the interarrival time of Packet-In events is indicative of the end-to-end packet delay at the receiving node. Moreover, Figure 11(c) and Figure 12(c) show that the algorithm handover decision is less erroneous as the network traffic load increases either by increasing the gateways' traffic rate or the background traffic (increasing γ_i in Equation (2). This pushes the network towards saturation (a higher violation probability and backoff time as in Equation (3)), making the end-to-end delay more sensitive to service time variation.

While using M/G/1 analysis for handover decisions, a similar trend to the one with the service time estimation can be observed. Gateways' traffic exhibited a lower end-to-end packet delay after handover based on M/G/1 analysis, as shown in Figure 13(a) and Figure 14(a). However, the packet interarrival time at the SDN controller does not significantly vary with increasing either the gateway traffic rate (Figure 13(b)) or the background data rate (Figure 14(b)). This is due to the randomness of Poisson traffic, which makes the queuing delay impact the end-to-end delay more than the service time of the shared IEEE 802.11 channel. As evident from Figure 15, choosing the AP based on load gives much better performance in terms of end-to-end delay. It also avoids further unnecessary handover to compensate for uneven load on APs.

It is worth noting that the mobility of IoT devices does not affect the performance of the proposed algorithms. If an IoT device moves, it may connect to a different IoT gateway. The IoT gateways are assumed stationary and deployed to cover the connectivity of the IoT devices over the whole city. On the other hand, if any IoT gateway changes its location or is deployed elsewhere, this is considered limited mobility as it will still be under overlapped coverage of the other city APs. This also does not affect the operation of the proposed algorithms as they run mainly on the SDN controller based on the data received from the APs.

Although the results are from a limited laboratory experimental setup, for a real-life scenario where the number of APs, end nodes, and amount of data will be much higher. Nevertheless, during running the experiments, maximum effort is taken to imitate a realistic scenario. In addition to WiFi clients used for data transmission, background traffic was generated to maximize load at the APs. Apart from that, the main concern will be the overburden on the SDN controller while computing delay calculations to make handover decisions. As the network size grows, multiple controllers can be installed to alleviate the task load of a single controller (Li et al., 2022). Several studies are being done in this field to address controller issues such as the number of controllers and optimal placement of controllers (Dhar et al., 2022; Killi & Rao, 2019; Shirmarz & Ghaffari, 2021).

Chapter 5: Conclusion and Future Scope

5.1 Conclusion

Smart city is a very common term in the current technology world. Developers have already realized the scopes of smart city and developed a lot of services including waste management, parking lot status, air pollution indicator, e-health services, and so on. Even though the smart city services are ready to be deployed right away there lacks a reliable network that satisfies the different QoS demands of these services that are necessary by their nature. As a growing technology, SDN has immense potential in transforming the network world. It is expected to untie current network complexities and provide agility and easier management of network devices. By separating the control plane and data plane and adding programmability to the control plane, SDN 'opens' the traditionally 'closed' network. Consequently, the network can be centrally controlled using software applications via open APIs. This makes SDN the apt solution to the earlier mentioned problem of smart city. In this research, SDN capabilities are put into use to achieve packet delivery latency QoS demand of smart city applications.

This research leans on WiFi networks in city-wide coverage where multiple APs typically cover each WiFi client. The experiments were done in a laboratory setting that mimics such a scheme. An SDN controller oversights all the APs in the city WiFi network, which supports smart city applications with different delay requirements. Three algorithms are introduced to control the traffic load of each AP and reduce the end-to-end delay of connected WiFi devices by performing a seamless handover to the least-loaded AP. Experiments were performed in two scenarios; by keeping constant client-forwarded traffic when the traffic load was varied and vice versa.

The first algorithm selects the client to be handed over by estimating the packet service time at the SDN controller in a non-invasive fashion that does not require any extra configuration, protocols, or software agent to run on the clients. Extensive experimentation has shown a notable decrease in the end-to-end delay after the handover in both experimental scenarios. The handoff decision based on service time estimation was compared with the ideal handover decision based on end-to-end delay calculations. An average error of 4.75% in constant client forwarded traffic and an average error of

6.8% in constant background traffic load was observed. Moreover, the handover decision approached the ideal handoff decision as the network load was increased in both cases. Graphs were drawn to compare the variation of end-to-end packet delay with the interarrival time of data packets. It was found that the interarrival time of data packets indeed followed the end-to-end delay. In smart cities, IoT devices are mostly synchronized to periodically update data. Hence handoff based on interarrival time is reliable in reducing end-to-end delay.

The second algorithm addresses client traffic of Poisson arrivals, which typically happens when many independent on-off IoT sources are aggregated at the client. This algorithm uses M/G/1 analysis to estimate the end-to-end packet delay by estimating the service time statistics at the SDN controller. Substantial hardware experiments using different packet rates of clients and background traffic show that the algorithm efficiently decreases the end-to-end packet delay of the WiFi clients. Nevertheless, no relation between service time and end-to-end delay was to be found in both the cases of constant client rate as well as constant background load. The measured end-to-end delay was compared against the calculated values using M/G/1 analysis, there was significant difference in their values. This error occurred because of the delay caused by the operating system kernel in timestamping the arrival and departure of packets. Also, the randomness of Poisson traffic affected the end-to-end delay to be more dependent on the queueing delay, hence a plot comparing interarrival time and end-to-end delay has shown no particular relationship between the two. This indicates that for the poisson arrival process, the handoff decision cannot be taken based on interarrival time; it has to be based on the Pollaczek-Khinchine (PK) formula.

In the third algorithm, the SDN controller selects the least-loaded AP for handover. SDN along with Open vSwitch is a powerful combination because it is with the help of data from open vSwitch that the controller can make handover decisions. And in fact, it has given remarkably lower packet delay compared to the case when AP for handover was chosen randomly.

All in all, this study has shown that SDN controllers can effectively monitor the networks and make suitable decisions to change the network topology for better end-to-end delay performance. This enables smart city applications that demand low packet delay such as e-health systems to be well-founded and trustworthy. The big advantage here is that there are no additional changes to be done to the WiFi clients or APs, thus making this technology to be backward compatible with devices. This study leaves scope for many more future studies. Several other QoS demands for smart city services can be explored and implemented in the same way. Network security, system responsiveness, reliable data traffic, throughput, et cetera are some of several more QoS demands of smart city applications. Sound functioning of services is extremely important for proper smart city establishment, otherwise public loses trust in it. In the end, the success of a smart city lies in social involvement.

5.2 Future Scope

SDN is expected to change the global network towards more controllable and agile model. Yet it is still under study for its practicality. This research is an actual physical experimentation based on SDN in a laboratory experimental setup. The results have emphasized that SDN does help in improving network performance if implemented wisely. However, only latency QoS is given importance in this work. Hence this system applies to services that demand only delay QoS, such as smart city health alert alarm service. In reality, there are numerous applications that demand other QoS or combination of QoS factors. This work can be further improved to incorporate more complicated QoS applications. This will enhance its applications in smart city. Furthermore, this study is done with a single SDN controller. But in reality as the network gets bigger, more controllers will be deployed to cover the whole network. The algorithms proposed in this study can be ran on a multiple controller network to compare performance differences.

References

Abbou, A. N., Taleb, T., & Song, J. (2021). Towards SDN-based Deterministic Networking: Deterministic E2E Delay Case. *2021 IEEE Global Communications Conference (GLOBECOM)*, 1–6. https://doi.org/10.1109/GLOBECOM46510.2021.9685656

Abdrabou, A., & Zhuang, W. (2008). Stochastic delay guarantees and statistical call admission control for IEEE 802.11 single-hop ad hoc networks. *IEEE Transactions on Wireless Communications*, *7*(10), 3972–3981. https://doi.org/10.1109/T-WC.2008.070564

Adame, T., Carrascosa, M., Bellalta, B., Pretel, I., & Etxebarria, I. (2021). Channel Load Aware AP / Extender Selection in Home WiFi Networks Using IEEE 802.11k/v. *IEEE Access*, *9*, 30095–30112. https://doi.org/10.1109/ACCESS.2021.3059473

AlZoman, R., & Alenazi, M. J. F. (2020). Exploiting SDN to Improve QoS of Smart City Networks Against Link Failures. *2020 Seventh International Conference on Software Defined Systems (SDS)*, 100–106. https://doi.org/10.1109/SDS49854.2020.9143878

Alzubi, J. A., Movassagh, A., Gheisari, M., Najafabadi, H. E., Abbasi, A. A., Liu, Y., Pingmei, Z., Izadpanahkakhk, M., & Najafabadi, A. P. (2022). A Dynamic SDN-based Privacy-Preserving Approach for Smart City Using Trust Technique. *2022 9th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS)*, 1–5. https://doi.org/10.1109/CFIS54774.2022.9756458

Aswanth, A., Manoj, E., Rajendran, K., EM, S. K., & Duttagupta, S. (2021). Meeting Delay guarantee in Telemedicine service using SDN framework. *2021 IEEE 9th Region 10 Humanitarian Technology Conference (R10-HTC*), 1–5. https://doi.org/10.1109/R10-HTC53172.2021.9641695

Babu, A. A., & Duttagupta, S. (2020). Delay Constrained Communications in a Disaster Site using SDN Framework. *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 16–20. https://doi.org/10.1109/ICSSIT48917.2020.9214234

Bauer, M., Bui, N., De Loof, J., Magerkurth, C., Nettsträter, A., Stefa, J., & Walewski, J. W. (2013). IoT Reference Model. In A. Bassi, M. Bauer, M. Fiedler, T. Kramp, R. van Kranenburg, S. Lange, and S. Meissner (Eds.), *Enabling Things to Talk: Designing IoT solutions with the IoT Architectural Reference Model* (pp. 113–162). Springer. https://doi.org/10.1007/978-3-642-40403-0_7

Bayhan, S., Coronado, E., Riggio, R., & Zubow, A. (2020). User-AP Association Management in Software-Defined WLANs. *IEEE Transactions on Network and Service Management*, *17*(3), 1838–1852. https://doi.org/10.1109/TNSM.2020.3001869

Bejerano, Y., & Han, S.-J. (2009). Cell Breathing Techniques for Load Balancing in Wireless LANs. *IEEE Transactions on Mobile Computing*, *8*(6), 735–749. https://doi.org/10.1109/TMC.2009.50

Bejerano, Y., Han, S.-J., & Li, L. (2007). Fairness and Load Balancing in Wireless LANs Using Association Control. *IEEE/ACM Transactions on Networking*, *15*(3), 560–573. https://doi.org/10.1109/TNET.2007.893680

Bouguerra, F. (2021). *Data centre networking: What is OVS?. Ubuntu*. Retrieved September 27, 2022, from https://ubuntu.com/blog/data-centre-networking-what-is-ovs

Bouzghiba, S., Dahmouni, H., Rachdi, A., & Garcia, J.-M. (2017). Towards an Autonomic Approach for Software Defined Networks: An Overview. In R. El-Azouzi, D. S. Menasche, E. Sabir, F. De Pellegrini, and M. Benjillali (Eds.), *Advances in Ubiquitous Networking 2* (pp. 149–161). Springer. https://doi.org/10.1007/978-981-10-1627-1_12

Burger, V., Seufert, M., Kaup, F., Wichtlhuber, M., Hausheer, D., & Tran-Gia, P. (2015). Impact of WiFi offloading on video streaming QoE in urban environments. *2015 IEEE International Conference on Communication Workshop (ICCW)*, 1717–1722. https://doi.org/10.1109/ICCW.2015.7247428

Cao, J., & Ramanan, K. (2002). A Poisson limit for buffer overflow probabilities. *Proceedings.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, *2*, 994–1003. https://doi.org/10.1109/INFCOM.2002.1019347

Cedillo-Elias, E. J., Orizaga-Trejo, J. A., Larios, V. M., & Maciel Arellano, L. A. (2018). Smart Government infrastructure based in SDN Networks: The case of Guadalajara Metropolitan Area. *2018 IEEE International Smart Cities Conference (ISC2)*, 1–4. https://doi.org/10.1109/ISC2.2018.8656801

Chen, J., Liu, B., Zhou, H., Yu, Q., Gui, L., & Shen, X. (2017). QoS-Driven Efficient Client Association in High-Density Software-Defined WLAN. *IEEE Transactions on Vehicular Technology*, *66*(8), 7372–7383. https://doi.org/10.1109/TVT.2017.2668066

Chen, W., Chen, C., Jiang, X., & Liu, L. (2018). Multi-Controller Placement Towards SDN Based on Louvain Heuristic Algorithm. *IEEE Access*, *6*, 49486–49497. https://doi.org/10.1109/ACCESS.2018.2867931

Cheng, Y., Yang, D., & Zhou, H. (2018). Det-LB: A Load Balancing Approach in 802.11 Wireless Networks for Industrial Soft Real-Time Applications. *IEEE Access*, *6*, 32054–32063. https://doi.org/10.1109/ACCESS.2018.2802541

Chiang, M., & Zhang, T. (2016). Fog and IoT: An Overview of Research Opportunities. *IEEE Internet of Things Journal*, *3*(6), 854–864. https://doi.org/10.1109/JIOT.2016.2584538

Chin, T., Rahouti, M., & Xiong, K. (2018). Applying software-defined networking to minimize the end-to-end delay of network services. *ACM SIGAPP Applied Computing Review*, *18*(1), 30–40. https://doi.org/10.1145/3212069.3212072

Cisco. (2019). *Cisco visual networking Index: Forecast and trends, 2017–2022* (White Paper Document ID 1551296909190103). San José, CA: Cisco. Retrieved June 7, 2022, from https://twiki.cern.ch/twiki/pub/HEPIX/TechwatchNetwork/HtwNetworkDocuments/white-paper-c11-741490.pdf

Cisco. (n. d.). *What Is a Wireless LAN (WLAN)?*. Retrieved August 15, 2022, from https://www.cisco.com/c/en/us/products/wireless/wireless-lan.html

Coronado, E., Khan, S. N., & Riggio, R. (2019). 5G-EmPOWER: A Software-Defined Networking Platform for 5G Radio Access Networks. *IEEE Transactions on Network and Service Management*, *16*(2), 715–728. https://doi.org/10.1109/TNSM.2019.2908675

Danielis, P., Dan, G., Gross, J., & Berger, A. (2017). Dynamic Flow Migration for Delay Constrained Traffic in Software-Defined Networks. *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, 1–7. https://doi.org/10.1109/GLOCOM.2017.8254753

del Cacho Estil-les, M. A., Pia Fanti, M., Mangini, A. M., & Roccotelli, M. (2022). Electric Vehicles Routing Including Smart-Charging Method and Energy Constraints. *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, 1735–1740. https://doi.org/10.1109/CASE49997.2022.9926492

Dezfouli, B., Esmaeelzadeh, V., Sheth, J., & Radi, M. (2019). A Review of Software-Defined WLANs: Architectures and Central Control Mechanisms. *IEEE Communications Surveys and Tutorials*, *21*(1), 431–463. https://doi.org/10.1109/COMST.2018.2868692

Dhar, M., Debnath, A., Bhattacharyya, B. K., Debbarma, M. K., & Debbarma, S. (2022). A comprehensive study of different objectives and solutions of controller placement problem in software-defined networks. *Transactions on Emerging Telecommunications Technologies*, *33*(5), e4440. https://doi.org/10.1002/ett.4440

Ding, A. Y., Crowcroft, J., Tarkoma, S., & Flinck, H. (2014). Software defined networking for security enhancement in wireless mobile networks. *Computer Networks*, *66*, 94–101. https://doi.org/10.1016/j.comnet.2014.03.009

GitHub. (n. d.). *Python WiFi API · 5g-empower/5g-empower.github.io Wiki.* Retrieved September 27, 2022, from https://github.com/5g-empower/5g-empower.github.io/wiki/Python-WiFi-API#lvap

Gong, H., Nahm, K., & Kim, J. (2008). Distributed Fair Access Point Selection for Multi-Rate IEEE 802.11 WLANs. *2008 5th IEEE Consumer Communications and Networking Conference*, 528–532. https://doi.org/10.1109/ccnc08.2007.123

Göransson, P., Black, C., & Culver, T. (2017). Chapter 5—The OpenFlow Specification. In P. Göransson, C. Black, and T. Culver (Eds.), *Software Defined Networks (Second Edition)* (pp. 89–136). Morgan Kaufmann. https://doi.org/10.1016/B978-0-12-804555-8.00005-3

Haiyan, M., Jinyao, Y., Georgopoulos, P., & Plattner, B. (2016). Towards SDN based queuing delay estimation. *China Communications*, *13*(3), 27–36. https://doi.org/10.1109/CC.2016.7445500

Han, Y., Yang, K., Lu, X., & Zhou, D. (2016). An adaptive load balancing application for software-defined enterprise WLANs. *2016 International Conference on Information and Communication Technology Convergence (ICTC)*, 281–286. https://doi.org/10.1109/ICTC.2016.7763486

Hava, A., Ghamri-Doudane, Y., Murphy, J., & Muntean, G.-M. (2019). A Load Balancing Solution for Improving Video Quality in Loaded Wireless Network Conditions. *IEEE Transactions on Broadcasting*, *65*(4), 742–754. https://doi.org/10.1109/TBC.2019.2906783

He, Q. (2021). Smart City Network Security Evaluation System. *2021 International Conference on Intelligent Transportation, Big Data and Smart City (ICITBS)*, 249–252. https://doi.org/10.1109/ICITBS53129.2021.00070

Hernandez, A., & Magana, E. (2007). One-way Delay Measurement and Characterization. *International Conference on Networking and Services (ICNS ’07)*, 114–114. https://doi.org/10.1109/ICNS.2007.87

Ibrar, M., Wang, L., Shah, N., Rottenstreich, O., Muntean, G.-M., & Akbar, A. (2022). Reliability-Aware Flow Distribution Algorithm in SDN-Enabled Fog Computing for Smart Cities. *IEEE Transactions on Vehicular Technology*, 1–17. https://doi.org/10.1109/TVT.2022.3202195

IEEE Standard for Information technology. (2008). Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 1: Radio Resource Measurement of Wireless LANs. *IEEE Std 802.11k-2008 (Amendment to IEEE Std 802.11-2007)*, 1–244. https://doi.org/10.1109/IEEESTD.2008.4544755

IEEE Standard for Information technology. (2011). Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 8: IEEE 802.11 Wireless Network Management. *IEEE Std 802.11v-2011 (Amendment to IEEE Std 802.11-2007 as Amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, IEEE Std 802.11w-2009, IEEE Std 802.11n-2009, IEEE Std 802.11p-2010, and IEEE Std 802.11z-2010)*, 1–433. https://doi.org/10.1109/IEEESTD.2011.5716530

IEEE Standard for Information technology. (2012). Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, 1–2793. https://doi.org/10.1109/IEEESTD.2012.6178212

Islam, M., Razzaque, A., Hassan, M. M., Ismail, W. N., & Song, B. (2017). Mobile Cloud-Based Big Healthcare Data Processing in Smart Cities. *IEEE Access*, *5*, 11887–11899. https://doi.org/10.1109/ACCESS.2017.2707439

Isong, B., Molose, R. R. S., Abu-Mahfouz, A. M., & Dladlu, N. (2020). Comprehensive Review of SDN Controller Placement Strategies. *IEEE Access*, *8*, 170070–170092. https://doi.org/10.1109/ACCESS.2020.3023974

Jahromi, K. K., Meneses, F., & Moreira, A. (2015). On the impact of overlapping access points in detecting node encounters. *2015 14th Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET)*, 1–7. https://doi.org/10.1109/MedHocNet.2015.7173169

Judd, G., & Steenkiste, P. (2002). Fixing 802.11 access point selection. *ACM SIGCOMM Computer Communication Review*, *32*(3), 31–31. https://doi.org/10.1145/571697.571720

Jurado-Lasso, F. F., Marchegiani, L., Jurado, J. F., Abu-Mahfouz, A. M., & Fafoutis, X. (2022). A Survey on Machine Learning Software-Defined Wireless Sensor Networks (ML-SDWSNs): Current Status and Major Challenges. *IEEE Access*, *10*, 23560–23592. https://doi.org/10.1109/ACCESS.2022.3153521

Killi, B. P. R., & Rao, S. V. (2019). Controller placement in software defined networks: A Comprehensive survey. *Computer Networks*, *163*, 106883. https://doi.org/10.1016/j.comnet.2019.106883

Kim, H., Lee, W., Bae, M., & Kim, H. (2017). Wi-Fi Seeker: A link and Load Aware AP Selection Algorithm. *IEEE Transactions on Mobile Computing*, *16*(8), 2366–2378. https://doi.org/10.1109/TMC.2016.2618790

Kreutz, D., Ramos, F. M. V., Veríssimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2015). Software-Defined Networking: A Comprehensive Survey. *Proceedings of the IEEE*, *103*(1), 14–76. https://doi.org/10.1109/JPROC.2014.2371999

Kumar, G. V., & Bindu, C. S. (2014). *A Study on Access Point Selection Algorithms in Wireless Mesh Networks*. *6*(1). Retrieved June 6, 2022, from https://www.proquest.com/openview/18fef54fcae3cb16953db690f7a85fd1/1?pq-origsite=gscholar&cbl=886380

Kumar, N., Mittal, S., Garg, V., & Kumar, N. (2022). Deep Reinforcement Learning-Based Traffic Light Scheduling Framework for SDN-Enabled Smart Transportation System. *IEEE Transactions on Intelligent Transportation Systems*, *23*(3), 2411–2421. https://doi.org/10.1109/TITS.2021.3095161

Kurungadan, B., & Abdrabou, A. (2021). A Software-defined Delay-aware Traffic Load Control for WiFi-based Smart City Services. *2021 International Conference on Computer, Information and Telecommunication Systems (CITS)*, 1–5. https://doi.org/10.1109/CITS52676.2021.9617928

Kurungadan, B., & Abdrabou, A. (2022). Using Software-Defined Networking for Data Traffic Control in Smart Cities with WiFi Coverage. *Symmetry*, *14*(10), Article 10. https://doi.org/10.3390/sym14102053

Lara, A., Kolasani, A., & Ramamurthy, B. (2014). Network Innovation using OpenFlow: A Survey. *IEEE Communications Surveys and Tutorials*, *16*(1), 493–512. https://doi.org/10.1109/SURV.2013.081313.00105

Lei, T., Wen, X., Lu, Z., & Li, Y. (2017). A Semi-Matching Based Load Balancing Scheme for Dense IEEE 802.11 WLANs. *IEEE Access*, *5*, 15332–15339. https://doi.org/10.1109/ACCESS.2017.2733083

Li, L., Pal, M., & Yang, Y. R. (2008). Proportional Fairness in Multi-Rate Wireless LANs. *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, 1004–1012. https://doi.org/10.1109/INFOCOM.2008.154

Li, S., Li, Z., & Zhang, W. (2022). A Dynamic Association Strategy for Controller Load Balancing in Software-Defined Networks. *2022 23rd Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 01–06. https://doi.org/10.23919/APNOMS56106.2022.9919948

Li, Y., Xia, S., Zheng, M., Cao, B., & Liu, Q. (2022). Lyapunov Optimization-Based Trade-Off Policy for Mobile Cloud Offloading in Heterogeneous Wireless Networks. *IEEE Transactions on Cloud Computing*, *10*(1), 491–505. https://doi.org/10.1109/TCC.2019.2938504

Lin, S., Che, N., Yu, F., & Jiang, S. (2019). Fairness and Load Balancing in SDWN Using Handoff-Delay-Based Association Control and Load Monitoring. *IEEE Access*, *7*, 136934–136950. https://doi.org/10.1109/ACCESS.2019.2942717

Liu, C.-F., Samarakoon, S., Bennis, M., & Poor, H. V. (2018). Fronthaul-Aware Software-Defined Wireless Networks: Resource Allocation and User Scheduling. *IEEE Transactions on Wireless Communications*, *17*(1), 533–547. https://doi.org/10.1109/TWC.2017.2768358

Manzoor, S., Zhang, C., Hei, X., & Cheng, W. (2019). Understanding Traffic Load in Software Defined WiFi Networks for Healthcare. *2019 IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW)*, 1–2. https://doi.org/10.1109/ICCE-TW46550.2019.8991777

Masaki, H., Nguyen, K., & Sekiya, H. (2021). A QoS-guaranteed System with Software Defined Networking and Micropayment. *2021 26th IEEE Asia-Pacific Conference on Communications (APCC)*, 94–98. https://doi.org/10.1109/APCC49754.2021.9609938

Maslow’s Hierarchy of Needs. (n. d.). *The Interaction Design Foundation*. Retrieved August 27, 2022, from https://www.interaction-design.org/literature/article/maslow-s-hierarchy-of-needs

Mehmood, Y., Haider, N., Imran, M., Timm-Giel, A., & Guizani, M. (2017). M2M Communications in 5G: State-of-the-Art Architecture, Recent Advances, and Research Challenges. *IEEE Communications Magazine*, *55*(9), 194–201. https://doi.org/10.1109/MCOM.2017.1600559

Mohanty, S., Shekhawat, A. S., Sahoo, B., Apat, H. K., & Khare, P. (2021). Minimizing Latency for Controller Placement Problem in SDN. *2021 19th OITS International Conference on Information Technology (OCIT)*, 393–398. https://doi.org/10.1109/OCIT53463.2021.00083

Mori, K., & Christodoulou, A. (2012). Review of sustainability indices and indicators: Towards a new City Sustainability Index (CSI). *Environmental Impact Assessment Review*, *32*(1), 94–106. https://doi.org/10.1016/j.eiar.2011.06.001

Navarro-Ortiz, J., Romero-Diaz, P., Sendra, S., Ameigeiras, P., Ramos-Munoz, J. J., & Lopez-Soler, J. M. (2020). A Survey on 5G Usage Scenarios and Traffic Models. *IEEE Communications Surveys and Tutorials*, *22*(2), 905–929. https://doi.org/10.1109/COMST.2020.2971781

Nicholson, A. J., Chawathe, Y., Chen, M. Y., Noble, B. D., & Wetherall, D. (2006). Improved access point selection. *Proceedings of the 4th International Conference on Mobile Systems, Applications and Services*, 233–245. https://doi.org/10.1145/1134680.1134705

Open vSwitch 3.0.90 documentation. (n. d.). *What Is Open vSwitch?*. Retrieved September 27, 2022, from https://docs.openvswitch.org/en/latest/intro/what-is-ovs/

Pace Technical. (n. d.). *Is Fast WiFi the Most Basic of Human Needs?* Retrieved August 27, 2022, from https://pacetechnical.com/fast-wifi-basic-human-needs/

Pang, J., Greenstein, B., Kaminsky, M., McCoy, D., & Seshan, S. (2010). Wifi-Reports: Improving Wireless Network Selection with Collaboration. *IEEE Transactions on Mobile Computing*, *9*(12), 1713–1731. https://doi.org/10.1109/TMC.2010.151

Pflanzner, T., Leszko, K. Zs., & Kertesz, A. (2018). SUMMON: Gathering smart city data to support IoT-Fog-Cloud simulations. *2018 Third International Conference on Fog and Mobile Edge Computing (FMEC)*, 71–78. https://doi.org/10.1109/FMEC.2018.8364047

Priyadarsini, M., Mittal, P., & Bera, P. (2020). Smart City Renovation using SDN Framework. *2020 International Conference on COMmunication Systems and NETworkS (COMSNETS)*, 794–799. https://doi.org/10.1109/COMSNETS48256.2020.9027417

Proceedings of the 11th annual international conference on Mobile computing and networking. (n. d.). *Self-management in chaotic wireless deployments*. Retrieved October 13, 2022, from https://dl.acm.org/doi/abs/10.1145/1080829.1080849

Rahouti, M., Xiong, K., & Xin, Y. (2020). Secure Software-Defined Networking Communication Systems for Smart Cities: Current Status, Challenges, and Trends. *IEEE Access*, *PP*, 1–1. https://doi.org/10.1109/ACCESS.2020.3047996

Raschellà, A., Bouhafs, F., Seyedebrahimi, M., Mackay, M., & Shi, Q. (2016). A centralized framework for smart access point selection based on the Fittingness Factor. *2016 23rd International Conference on Telecommunications (ICT)*, 1–5. https://doi.org/10.1109/ICT.2016.7500386

Ren, J., Li, J., Liu, H., & Qin, T. (2022). Task offloading strategy with emergency handling and blockchain security in SDN-empowered and fog-assisted healthcare IoT. *Tsinghua Science and Technology*, *27*(4), 760–776. https://doi.org/10.26599/TST.2021.9010046

Riggio, R., Marina, M. K., Schulz-Zander, J., Kuklinski, S., & Rasheed, T. (2015). Programming Abstractions for Software-Defined Wireless Networks. *IEEE Transactions on Network and Service Management*, *12*(2), 146–162. https://doi.org/10.1109/TNSM.2015.2417772

Riggio, R., Rasheed, T., & Granelli, F. (2013). EmPOWER: A Testbed for Network Function Virtualization Research and Experimentation. *2013 IEEE SDN for Future Networks and Services (SDN4FNS)*, 1–5. https://doi.org/10.1109/SDN4FNS.2013.6702538

RYU SDN Framework—Ryubook 1.0 documentation. (n. d.). Retrieved August 27, 2022, from https://book.ryu-sdn.org/en/html/

Seneviratne, S., Hu, Y., Nguyen, T., Lan, G., Khalifa, S., Thilakarathna, K., Hassan, M., & Seneviratne, A. (2017). A Survey of Wearable Devices and Challenges. *IEEE Communications Surveys and Tutorials*, *19*(4), 2573–2620. https://doi.org/10.1109/COMST.2017.2731979

Shirmarz, A., & Ghaffari, A. (2021). Taxonomy of controller placement problem (CPP) optimization in Software Defined Network (SDN): A survey. *Journal of Ambient Intelligence and Humanized Computing*, *12*(12), 10473–10498. https://doi.org/10.1007/s12652-020-02754-w

Shortle, J. F., Thompson, J. M., Gross, D., & Harris, C. M. (2018). *Fundamentals of Queueing Theory*. John Wiley and Sons. https://doi.org/10.1002/9781119453765

Sloane, T. (2013). OpenFlow. *Open Networking Foundation*. Retrieved February 22, 2021, from https://opennetworking.org/sdn-resources/customer-case-studies/openflow/

Stanley, D., Montemurro, M., & Calhoun, P. R. (2009). Control And Provisioning of Wireless Access Points (CAPWAP) Protocol Specification (Request for Comments RFC 5415). *Internet Engineering Task Force*. https://doi.org/10.17487/RFC5415

Sun, T., Zhang, Y., & Trappe, W. (2016). Improving Access Point Association Protocols Through Channel Utilization and Adaptive Probing. *IEEE Transactions on Mobile Computing*, *15*(5), 1157–1167. https://doi.org/10.1109/TMC.2015.2442254

Suresh, L., Schulz-Zander, J., Merz, R., Feldmann, A., & Vazao, T. (2012). Towards programmable enterprise WLANS with Odin. *Proceedings of the First Workshop on Hot Topics in Software Defined Networks - HotSDN '12*, 115. https://doi.org/10.1145/2342441.2342465

Tadros, C. N., Rizk, M. R. M., & Mokhtar, B. M. (2020). Software Defined Network-Based Management for Enhanced 5G Network Services. *IEEE Access*, *8*, 53997–54008. https://doi.org/10.1109/ACCESS.2020.2980392

Takia, S. S., Omee, R. R., Hossain, R. A., & Islam, M. (2020). Delay-Optimized Traffic Forwarding Mechanism in Software Defined Networking. *2020 IEEE Region 10 Symposium (TENSYMP)*, 1424–1427. https://doi.org/10.1109/TENSYMP50017.2020.9230704

Tivig, P.-T., Borcoci, E., Brumaru, A., & Ciobanu, A.-I.-E. (2021). Layer 3 Forwarder Application—Implementation Experiments Based on Ryu SDN Controller. *2021 International Symposium on Networks, Computers and Communications (ISNCC)*, 1–6. https://doi.org/10.1109/ISNCC52172.2021.9615685

Tragos, E. Z., Angelakis, V., Fragkiadakis, A., Gundlegard, D., Nechifor, C.-S., Oikonomou, G., Pöhls, H. C., & Gavras, A. (2014). Enabling reliable and secure IoT-based smart city applications. *2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS)*, 111–116. https://doi.org/10.1109/PerComW.2014.6815175

Washburn, D., & Sindhu, U. (2010). Helping CIOs Understand "Smart City" Initiatives. *Smart City*. Retrieved September 8, 2021, from https://s3-us-west-2.amazonaws.com/itworldcanada/archive/Themes/Hubs/Brainstorm/forrester_help_cios_smart_city.pdf

Wenge, R., Zhang, X., Dave, C., Chao, L., & Hao, S. (2014). Smart city architecture: A technology guide for implementation and design challenges. *China Communications*, *11*(3), 56–69. https://doi.org/10.1109/CC.2014.6825259

Worlddata.Info. (n. d.). *United Arab Emirates: Country data and statistics*. Retrieved October 14, 2022, from https://www.worlddata.info/asia/arab-emirates/index.php

Xia, W., Wen, Y., Foh, C. H., Niyato, D., & Xie, H. (2015). A Survey on Software-Defined Networking. *IEEE Communications Surveys and Tutorials*, *17*(1), 27–51. https://doi.org/10.1109/COMST.2014.2330903

Xu, F., Tan, C. C., Li, Q., Yan, G., & Wu, J. (2010). Designing a Practical Access Point Association Protocol. *2010 Proceedings IEEE INFOCOM*, 1–9. https://doi.org/10.1109/INFCOM.2010.5461909

Yen, L.-H., Yeh, T.-T., & Chi, K.-H. (2009). Load Balancing in IEEE 802.11 Networks. *IEEE Internet Computing*, *13*(1), 56–64. https://doi.org/10.1109/MIC.2009.11

Yuan, B., Lin, C., Zhao, H., Zou, D., Yang, L. T., Jin, H., & Rong, C. (2020). Secure Data Transportation With Software-Defined Networking and k-n Secret Sharing for High-Confidence IoT Services. *IEEE Internet of Things Journal*, *7*(9), 7967–7981. https://doi.org/10.1109/JIOT.2020.2993587

Zakaria, N., & A., Jawwad. (2015). Smart City Architecture: Vision and Challenges. *International Journal of Advanced Computer Science and Applications*, *6*(11). https://doi.org/10.14569/IJACSA.2015.061132

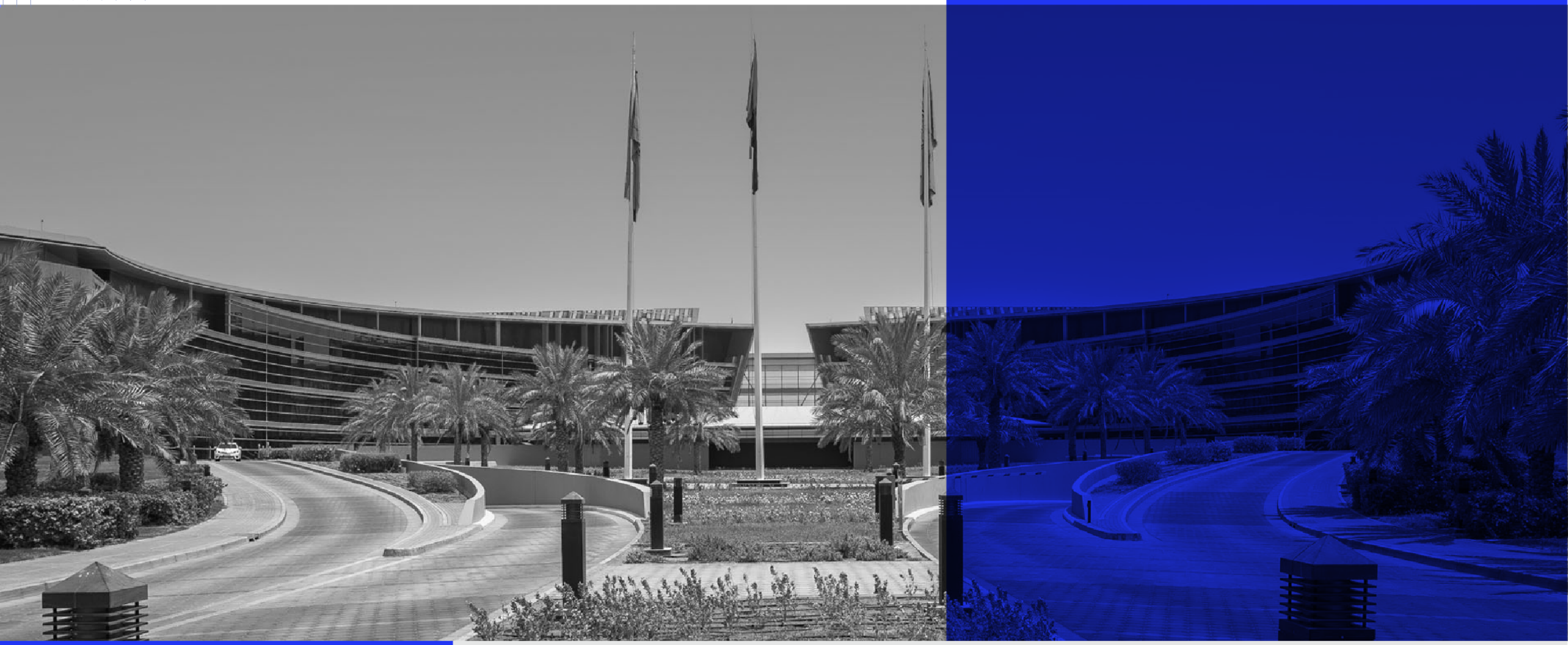
Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014). Internet of Things for Smart Cities. *IEEE Internet of Things Journal*, *1*(1), 22–32. https://doi.org/10.1109/JIOT.2014.2306328

Zeljković, E., Slamnik-Kriještorac, N., Latré, S., & Marquez-Barja, J. M. (2019). ABRAHAM: Machine Learning Backed Proactive Handover Algorithm Using SDN. *IEEE Transactions on Network and Service Management*, *16*(4), 1522–1536. https://doi.org/10.1109/TNSM.2019.2948883

List of Publications

Kurungadan, B., & Abdrabou, A. (2021). A Software-defined Delay-aware Traffic Load Control for WiFi-based Smart City Services. *2021 International Conference on Computer, Information and Telecommunication Systems* (CITS), *1–5*. https://doi.org/10.1109/CITS52676.2021.9617928

Kurungadan B., & Abdrabou, A. (2022). Using Software-Defined Networking for Data Traffic Control in Smart Cities with WiFi Coverage. *Symmetry*, 14(10):2053. https://doi.org/10.3390/sym14102053



UAE UNIVERSITY MASTER THESIS NO. 2022: 121

This thesis tries to implement a novel approach for Quality of Service provisioning to delay sensitive smart city services. It makes use of Software defined Networking technology to control the wireless network topology for optimum results.

Basima Kurungadan received her Master of Science in Electrical Engineering from the Department of Electrical and Communication Engineering, College of Engineering at UAE University, UAE. She received her Bachelor of Technology in Electronics and Communication Engineering from the University of Calicut, India.