Efficient energy-aware controller placement in software-defined wireless sensor networks

RRS MOLOSE

orcid.org/0000-0003-3669-1504

Dissertation accepted in fulfilment of the requirements for the degree Master of Computer Science at the North West University

Supervisor: Prof. Bassey Isong, Mrs Nosipho Dladlu-Mntuwaphi, Prof. Adnan Abu-Mahfouz (CSIR)

Graduation ceremony: October 2022

Student number: 25311999

Declaration

I, REORAPETSE RAMOLITI SAMUEL MOLOSE, declare that "**Efficient energy-aware controller placement in software-defined wireless sensors networks**" is my work which has never been presented for the award of any degree at any university. All sources of information used in this research have been deservedly acknowledged in text and references.

Signature: _____ Date: _____

APPROVAL

Supervisor: **Prof Bassey Isong**
Department of Computer Science
Faculty of Natural and Agricultural Science
North-West University
South Africa

Signature: _____ Date: _____

Co-Supervisor: **Mrs Nosipho Mntuwaphi-Dladlu**
Department of Computer Science
Faculty of Natural and Agricultural Science
North-West University
South Africa

Signature: _____ Date: _____

Co-Supervisor: **Prof. Adnan Abu-Mafhouz**
CSIR
Emerging Digital Technologies for 4IR (EDT4IR)
Pretoria
South Africa

Signature: _____ Date: _____

Dedication

I dedicate this work to my late parents and grandparents who, in their lifetime have gone through struggles to see me succeed. They have been my motivation to thrive both in academia and in my personal life. With this dissertation, I honour their lives as it was never in vain but a source of encouragement and a constant reminder that their endless love is and has always been warmly felt.

Acknowledgement

I give praise to the Lord for gracing me with blessings not only in completing my MSc degree but in positive relationships I have built-in and outside academia. Furthermore, I thank Him for being my pillar of strength throughout all obstacles I have encountered, I am and will always be grateful.

Acknowledgement to my family and friends, their patience and understanding have shown pure love throughout my academic years, therefore, I will always be grateful for their presence in my life.

My supervisors, Prof. B Isong and Mrs N Dladlu-Mntuwaphi have showed not only compassion towards my studies but in my personal growth, with a great sense of care and love. May the merciful God bring joy and happy endings in your lives for the work and commitment you have invested in me.

To all my colleagues and working staff in the Department of Computer Science, North-West University Mafikeng campus, respect, and appreciation to you for the guidance and exceptional working environment that we have shared during the years. The highest appreciation goes to the Council for Scientific and Industrial Research (CSIR) for their support in my academic finances and having the confidence to invest in my education.

Finally, I would like to give a special acknowledgement to my uncles, grandmother and grandfather for their support and patience in my academic journey, their constant motivation and wise words have been fuel for successfully achieving my academic and personal goals. God bless you all.

Abstract

A centralized controller in the Software-defined Wireless Sensor Networks (SDWSN) environment poses a single point of failure and is inapt for a large-scale network. As leverage, multiple controllers have been introduced but are confronted with controller placement problems (CPP) for a better quality of service and network requirements. CPP challenge in SDWSN lies in finding the numbers, location and allocation of controllers in given network topology as well as sensors assignments. This is important in positively impacting the network's performance in terms of latency and cost minimization and reliability, and energy efficiency maximization. Moreover, several Software-defined networking (SDN) based CPP approaches have been proposed and developed over the years but only a few proposed techniques addressed energy efficiency in the SDWSN. Therefore, an efficient and dynamic CPP approach that is generic and considers energy consumption is important in the SDWSN. In this research, a hybrid central CPP algorithm is designed and developed to reduce or get rid of the wireless sensor network (WSN) and SDN-based network performance objectives for improved SDWSN network performance. The proposed algorithm considered energy consumption, propagation latency and cost metrics to prolong the lifetime of wireless sensors and minimize the delay and cost spent for placements of controllers in networks. The algorithm is associated with real controllers and wireless sensor devices that use certain types of modules. Furthermore, the technique utilized the threshold-sensitive energy efficient sensor network (TEEN) routing protocol and particle swarm optimization (PSO)- K-means algorithm chosen after empirical evaluations were performed with other protocols and algorithms. The approach is evaluated through a series of simulations and the results indicate that the proposed efficient CPP energy-aware algorithm is effective on SDWSN in terms of number, location and allocation of controllers compared to the traditional SDN and WSN. The proposed algorithm also outperformed other algorithms and significantly increases propagation latency. The proposed algorithm minimizes delay and improves energy consumption however, it is short on reliability and load balancing which is part of the future work. We, therefore, recommend using real or emulated CC2530 devices to create an open-source architecture and framework that can be used on network simulation tools to test centrally designed algorithms in SDWSN.

Keywords: *Controllers, Energy efficiency, Latency, Wireless sensor network, Software defined networks, Software defined wireless sensor networks.*

Table of Contents

Table of Figures

List of Tables

Definition of Concepts

Software-defined Networks: A network architecture with a logically centralized controller which seeks to make the network more agile and adaptable. The goal of SDN is to improve network control by enabling the configuration of forwarding traffic by multiple devices to quickly adapt to changing business needs using OpenFlow as a communication medium example [1, 2].

Wireless Sensor Networks: A collection of sensors that are physically scattered and dedicated to monitoring and recording physical parameters such as temperature and humidity, of the environment through wireless communication mediums. These sensors are equipped with radio transceivers and microcontrollers to achieve their specific sensory task [3, 4].

Software defined wireless sensor networks: A networking paradigm that isolates the data plane from the control logic. It constitutes of different architectures but serves a common goal, to control the WSN with a strong controller that offers a comprehensive view of the entire network [5, 6].

Quality of Service: Ineffective attributes or features affecting the output condition or quality is defined as a quality of service. Examples may include the time taken for a response, audio/video quality and/or synchronization, etc [7].

Internet of things: The idea of common physical items talking with one another over the internet and being able to identify themselves to other devices or mediums. An example of a communication method is Radio Frequency Identification (RFID), although it includes other sensor technologies, wireless technologies or quick response (QR) codes [8, 9].

Wide Area Network: A telecommunication network composed of a collection of computers and network resources linked by a broad network that spans a geographic area. Wide area networks are frequently associated with national boundaries, metropolitan areas, and so on [10, 11].

Latency: Latency is a measurement of the response time of a particular entity. Thus, it is described as the time required for a computer on a network to respond to a request, meaning, the lower the number of requests, the faster the response [12, 13].

Load Balancing: A mechanism that considers multiple requests or processes with the aim of distribution across multiple devices on a network depending on how busy each device is. It serves to achieve even flow assignment distributed across the paths aiding network utilization improvement [14, 15].

5G: 5th Generation, is a cellular communication network standard. Thus, it performs packet switching meaning it can accommodate diverse coverage and mobility features, differentiating between an unlicensed and licensed spectrum. 5G can produce cellular data up to speeds of 20 Gbps, resulting in advancements to cater for an increasing number of IoT devices [16, 17].

List of Acronyms

Software defined Networks:	**SDN**
Binary Integer Program:	**BIP**
Wireless Sensor Networks:	**WSN**
Quality of Service:	**QoS**
Software Defined Wireless Sensor Networks:	**SDWSN**
Internet of things:	**IoT**
Controller Placement Problem:	**CPP**
Chaotic scalp swarm algorithm:	**CSSA**
Energy Profit Threshold:	**EPT**
Dynamic slave controller assignment:	**DSCA**
Bounded Maximum Latency Placement:	**BMLP**
Bounded Average Latency Placement:	**BALP**
Balanced Cost-Latency Placement:	**BCLP**
Clustering network partition algorithm:	**CNPA**
Cooperative Load Balancing Scheme:	**COLBAS**
Polynomial-Time Approximation Algorithm:	**PTAA**
Wide Area Network:	**WAN**
Pareto Integrated Tabu Search:	**PITS**
Multi-Start Hybrid Non-Dominated Sorting Genetic Algorithm:	**MHNSGA**
Multi-objective ant lion optimization:	**MO-ALO**
Pareto-based Optimal Controller placement:	**POCO**
Control-path Based-saving Routing Algorithm:	**CERA**
Energy consumption by Path:	**ECP**
Fifth-generation wireless:	**5G**

Chapter 1

Introduction and Background

1.1 Introduction

The logically centralized control or the programmability attribute of Software-Defined Networks (SDN) is leveraged in the traditional Wireless Sensor Networks (WSN). This is important to overcome the shortcomings that traditional WSNs possess which include resource constraints or limitations, reliability and the necessary Quality of Service (QoS) when there is an increase in sensor nodes deployment in small to large-scale networks. Software-Defined Wireless Sensor Networks (SDWSN) is a very important concept, especially in the day and age of "smart things" or the Internet of things (IoT) [3] where there is an increase in traffic flow in a network. The increase in traffic results in the growth of the network which brings about a lot of challenges particularly in energy conservation as sensor nodes operate at a limited power using batteries. Furthermore, scalability challenges are also a limitation that deserves to be handled with much attention because they involve multiple controllers' placement.

Thus, the Controller Placement Problem (CPP) is a scalability challenge faced in the realm of SDN and SDWSN. CPP arises due to the multi-controller placement architecture that forms part of the elimination of a centralized or single controller architecture which inherently poses a single point failure in the network. It is challenged by 1). the number of controllers in a network, 2). Where in the topology are the controllers to be placed and 3). the assignment of controllers and switches [18]. These are all the factors that affect the optimal placement of the controller in an SDWSN. Moreover, for better network performance, the optimal placement should consider the network performance metrics including latency, reliability, load balancing, energy efficiency, etc. All these metrics conflict when they are all considered simultaneously making CPP a non-deterministic hard problem (NP-hard) [18].

Performance optimality is the main goal to be achieved by an SDWSN, but the performance depends on the above-mentioned metrics. For instance, optimality in latency relies on the communication between controllers and switches, and among controllers themselves to handle the load in the network. The number and the location of controllers in the network are crucial

as the scale of a network becomes larger. This brings about the management of the load within the network to promote consistency between controllers by decreasing the delay hence the location of the controllers in the topology [18]. The volume of traffic or the load in the network between switches and controllers affect the energy of a network (sensor nodes), therefore the assignment of controllers to switches must be considered to provide a dynamic routing solution to increase energy efficiency [19]. With the load balancing being of core contribution towards the performance of a network and an independent variable in optimization of the performance metrics, reliability is also affected by the load or traffic. A controller can crash due to overloading and this can cause a disconnection between controller-to-controller and switch-to-controller. Therefore, this calls for a distributed controller placement and a fast recovery mechanism to increase reliability and resilience [20].

Recently, new technologies have tremendously increased the processing and management of a network [8, 21]. Thus, efficient and dynamic approaches are important to cope with such trends. Though several CPP approaches have been proposed and developed from the perspective of SDN, solving CPP results is time-consuming due to its' NP-hard trait. An efficient algorithm is one of the recent research focuses that this work tends to focus on [18]. The approach should optimally search or obtain possible controller placement, and promote or increase reliability by eliminating fault tolerance and increasing energy efficiency in a network.

Therefore, this research work is designed to address the CPP from the perspective of energy efficiency and reliability using multiple metrics to achieve performance boast in the network. These metrics include optimal placement of controllers, recovery mechanism for the increase in reliability, and load balancing for energy efficiency. As indicated above, multiple metrics ought to conflict with one another. The expected outcome is a better performance that is efficient and better QoS for SDWSN.

1.2 Problem statement

A centralized SDNWSN controller is effective for small to medium-sized networks [22]. However, it limits network scalability, reliability and availability as it poses a single point of failure and in the event of faults, it may lead to the entire network failing or being paralyzed [23]. Multiple controllers stand as leverage in terms of network scalability, efficiency, reliability and availability for large-scale networks. Despite the benefits, multiple controllers come with the challenge of controller placement problem (CPP) [18, 23, 24] for better QoS. CPP characterized as an NP-hard [18] is challenged by finding answers to critical questions

such as: how many controllers are to be placed given a network topology which can positively impact the network performance? what is the appropriate position of the controller in the topology? and, how are the controllers allocated or assigned to switches to minimize latency, maximise reliability and energy efficiency and so on? [18, 25]. Moreover, the optimal controller number, their placement, and assignment also affect the network performance in terms of energy consumption, latency, reliability, capacity etc. Therefore, the network performance depends upon controller placement optimality.

Furthermore, several approaches have been proposed and developed to solve CPP in large-scale SDNs as reported by Lu *et al*. [18]. However, an efficient solution is still far from being achieved. Also, the CPP solutions have been one-sided favouring only SDN with none or few developed for SDWSN, despite the inherent challenges faced by WSN [26]. Particularly, an efficient and dynamic CPP algorithm is still lacking [18, 24] and existing approaches only incorporate a few performance metrics in their objective function and not all metrics, though metrics conflict among themselves. Thus, a generic CPP solution is yet to be achieved. An efficient and dynamic algorithm is one of the key trends in solving or finding the optimal solution in a short period while providing good QoS and less resource utilization [18, 24]. Given the critical context, this research aims at developing an efficient and dynamic CPP algorithm that is generic from the SDWSN point of view taking into consideration, all the performance metrics with more emphasis on energy consumption. This is in line with [22, 27] which advocates efficient solutions for CPP in a large-scale network with dynamic attributes like dynamic load balancing.

1.3 Research Questions

This research seeks to provide answers to the following research questions (RQs):

RQ1: What is the state-of-the-art practice of CPP and the challenges that impact its efficiency?

RQ2: How can an efficient algorithm be designed and developed for SDWSN-based CPP which minimizes energy consumption and increase reliability?

To answer RQ2, the following sub-questions will be answered:

RQ2.1: How can a CPP algorithm be designed for optimal controller placement?

RQ2.2: How can a reliable energy-aware CPP be achieved from RQ2.1?

RQ3: How can we assess the effectiveness and performance of the proposed solution?

1.4 Research Goal

This research is aimed at developing an algorithm for optimal controller placement in the SDWSN that is efficient and energy-aware.

1.5 Research Objectives

To meet the above-stated aim and to address the RQs, the following research objectives (ROs) will be addressed:

RO1: Conducting a comprehensive literature review of existing CPP schemes in the realm of SDN and SDWSN to bring together their state-of-the-art practice.

RO2: Designing and implementing an efficient CPP algorithm that is energy-aware for SDWSN.

RO3: Evaluating the effectiveness and performance of the proposed CPP scheme using simulations and experiments.

1.6 Research Motivation

The drawbacks of WSN including energy efficiency are still adopted by emerging and promising technology, SDWSN. Nonetheless, our proposed energy-aware CPP algorithm makes use of existing protocols and algorithms in SDN and WSN to fulfil our defined set of ROs. Existing algorithms and protocols that have been used by other authors reduce the complexities of implementation and justification for algorithm/protocol runtime, the comparison is still performed on the choice of algorithm and protocol being used for our algorithm in Chapters 4 and 5. Data aggregation through the deployment of PSO clustering ensures good management for data transmission and a central and/or local control in the network through multiple controller placement. In addition, the TEEN protocol is developed to reduce sensor node energy consumption and thus provide a longer network lifetime. Furthermore, introducing SDN-enabled sink nodes to achieve multiple local controllers enables the use of TEEN as a distributed method. Therefore, the network is partitioned into sub-networks controlled by multiple local SDN-enabled sink nodes ensuring a prolonged network lifetime.

1.7 Research Contribution

Following proposed methods in existing literature, frameworks like TinySDN and SDN-WISE are commonly used creating a gap in the use of central standalone algorithms that achieve energy and latency efficient, reliable, and CAPEX/OPEX aware methods. Our ROs show exploration of energy efficiency (lifetime), propagation latency, node congestion (reliability) and CAPEX/OPEX (cost) as performance objectives for WSN and CPP drawbacks thus, incorporating them as a single unit (algorithm) to provide a solution to SDWSN challenge(s). Furthermore, this research used nature-based metaheuristics for addressing CPP and energy-efficient hierarchical routing protocol for combating energy consumption. However, these methods have been highly explored in both WSN and SDN including some of their respective enhancement or advancements in their classes. SDWSN being an advancing research area, the use of these two techniques serves as a contribution to existing approaches though they are used in a different type of network.

Additionally, the contributions in this research explain that the incorporation of existing energy-aware protocols together with existing nature-based metaheuristic algorithms to form a hierarchical clustering efficient energy-aware approach is a unique technique in solving controller placement and energy consumption issues in the SDWSN.

Our proposed method will administer a new algorithm that will be tested and evaluated to justify its minimization and efficient/effective support or attribute in SDWSN intricacy, further performing following the outlined objectives. The use of hybridizing or combining TEEN and PSO, and adopting the CC2530 module for SDN-enabled sink nodes has never been used before, as far as we can tell. In existing works, researchers have mostly implemented these techniques individually and/or adopted frameworks focusing only on one objective i.e., energy or latency-aware controller placement.

1.8 Methods of Investigation

In this section, the research methodology and various methods appropriate for this research to obtain the stated research objectives and answer the stated research question are discussed.

1.8.1 Research Methodology

The methodology most suitable for this research is a constructive methodology which is defined as a methodology that practically solves problems additionally contributing towards appreciating academic theory. The methodology aims at providing a “pragmatic goal-oriented

umbrella for tailor-made research designs" employing constructs [28]. In this research, we aim at developing an efficient energy-aware controller placement construct or algorithm as the methodology suggests i.e., employing constructs. This methodology is imperative as it strengthens the work with existing theories and further combines these theories and real-world problems, hence it is of relevance. Constructive research includes different phases within its respective process namely: selection of a relevant and practical research problem, obtaining pre-understanding and designing the construct [28].

1.8.2 Research Methods

Following the formulated research questions which are to be answered by the selected research methods a research design is implemented and consistent following the respective research method(s) in accord. The research design serves as a layout plan which explains how the identified research problem intends to be addressed to achieve the proposed research objectives. Comprehensive literature reviews, formal methodologies, model construction, simulations, and experiments are some of the methods used in this study.

The methodologies and research strategy utilized in this study are explored in further depth in Chapter 3.

1.9 Research Scope and Limitations

The research designed and implemented an efficient algorithm for addressing energy-aware controller placement in SDWSN. Furthermore, this research only focused on or carried out performance evaluation on simulations and the actual system implementation was not being done. Moreover, this research was limited by time, tools and finance. However, they did not affect the realization of this study.

1.10 Dissertation Organization

This section presents how this dissertation is organized:

Chapter 1: Introduction and Problem Statement: The introduction of SDWSN is discussed with the identification of the efficient CPP energy-aware problem described in this chapter. Moreover, the proposition of an efficient CPP energy-aware approach for improving network scalability and energy efficiency is discussed respectively.

Chapter 2: Literature review: This section looks at the WSN, SDN, SDWSN, CPP, and their various technologies and architectures. Furthermore, existing or related work forms part of the chapter discussion.

Chapter 3: Research Methodology and Design: This chapter discusses constructive methodology as a research technique and the numerous methods that can be employed to accomplish research goals. Furthermore, the research design followed in this research is also described.

Chapter 4: Comparative Clustering-based Meta-Heuristic Algorithm: The chapter explores and evaluates three clustering-based meta-heuristic algorithms. The chapter also performed experiments with results presented graphically.

Chapter 5: Comparative WSN-based Energy Efficiency Protocols: WSN-based energy efficiency protocols are compared in this chapter, with a focus on cluster-based routing methods. Experiments are performed and the results are presented graphically.

Chapter 6: Results and Analysis: In this chapter, performance results are assessed and described through data collected from the simulation method.

Chapter 7: Summary, Conclusions and Recommendations: The findings of this study are summarized in this chapter and a conclusion of our research objectives and research goal. Contributions to the research are highlighted following a discussion of open research opportunities and future work.

1.11 Research Outputs

The research outputs that resulted from this research are outlined as follows:

1. R. Molose, B. Isong, N. Dladlu and A. M. Abu-Mahfouz, "Analysis of Hierarchical Cluster-based Energy-Aware Routing Protocols in WSNs for SDWSN Application," *2021 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, 2021, pp. 1-8, DOI: 10.1109/ICECET52533.2021.9698557.
2. B. Isong, R. R. S. Molose, A. M. Abu-Mahfouz and N. Dladlu, "Comprehensive Review of SDN Controller Placement Strategies," in *IEEE Access*, vol. 8, pp. 170070-170092, 2020, DOI: 10.1109/ACCESS.2020.3023974.
3. Accepted for publication: R. Molose, B. Isong, N. Dladlu and A. M. Abu-Mahfouz, "Data Aggregation Schemes for Maximal Network Lifetime: review" 2022 *Conference on Electrical, Computer and Energy Technologies (ICECET)*. Prague, Czech Republic, July 20-21, 2022

1.12 Chapter Summary

Following the formation of our existing research problem that we are addressing or answering in our research, this chapter acts as an introduction base for our research topic. The research goal is discussed ensuing the different research questions that are alluded to or asserted to efficiently achieve our goal. The jurisdiction for doing the research is also covered, as well as research restrictions to limit the scope of our study. Additionally, the provision of research chapters and research outputs are highlighted. Lastly, the methodology study is provided, discussing how the problem was solved.

Chapter 2

Literature Review

2.1 Chapter Outline

This chapter discusses the literature on Controller Placement in the SDN-SDWSN. A description of various performance metrics that affect the placement of controllers in the SDWSN is highlighted and a clear description of existing systems is also presented.

2.2 Introduction

Communication networks are affected by many factors which consequently can lead to failure or the whole network collapsing. In the year 2019, Facebook and Instagram experienced an outage or network failure with consideration to reported news. According to CBS News [29], the failure had affected the company due to an alleged Distributed Denial of service (DDOS) attack. This happens when there is a great saturation of traffic that occurs in a network resulting in an overwhelming state within the network. With emerging domains like SDN, this can be avoided hence highlighting the importance of some of the technologies that can truly affect not only the daily users but for businesses that rely on social media platforms to put food on the table. This is important considering the benefits that SDNs/SDWSNs is the leading and primary contributor toward a reliable, energy-efficient, robust and more importantly secure network.

In this age of continuous growth in Information Communication Technology (ICT) including the 4^{th} industrial revolution, technologies like the Internet of Things (IoT), Artificial Intelligence (AI) and 5G, an expansion or increase of communication networks has been in demand. These technologies bring about many complexities within networks to keep an equilibrium between the end-user needs, demand and network manageability or performance. Traffic engineering in these heterogeneous and geographically distributed networks is the primary concern as these arising technological domains have increased networks into larger Wide Area Networks (WAN) due to the increase of traffic that these technologies bear [30, 31]. SDWSN is still an emerging paradigm, however, it is the primary contributor to an efficient approach to solving all the challenges that arise with these emerging technologies to bring Low-Rate Wireless Networks [6]. Considering various factors or attributes that arise when SDWSN is of interest, the programmability trait that it brings is of particular interest in ensuring a more reliable, energy-efficient, secure and robust network. With so much to

acknowledge to reach such a goal, researchers and professionals are at the peak of innovations, therefore, making SDWSN tools of importance as these are still early stages of implementation. This section will dwell much on what this domain (SDWSN) entails and identify the various challenges, architecture and applications.

2.3 Wireless Sensor Networks

This section succinctly introduces different aspects that serve as the constituents of WSN. This includes factors such as the architecture that highlight the communication of sensors as a collective network, respective protocol standards that administer the communication as well as the various challenges that are faced within the WSN discipline.

2.3.1 Overview

Various sensor nodes within a WSN consist of their sensing range that determines the communication with other sensor nodes through a wireless interface and sense surrounding phenomenon or an entity. Fig. 2.1 represent a wireless sensor network with various sensors that are visually dispersed.

Information is gathered from the environment monitored by the different sensors which communicate respectively. The sensors transfer all their data to the base station (BS) or sink node to enable remote management resulting in users acquiring the particular data through the use of the internet [32, 33]. The various types of WSN controls are illustrated in Fig. 2.2.

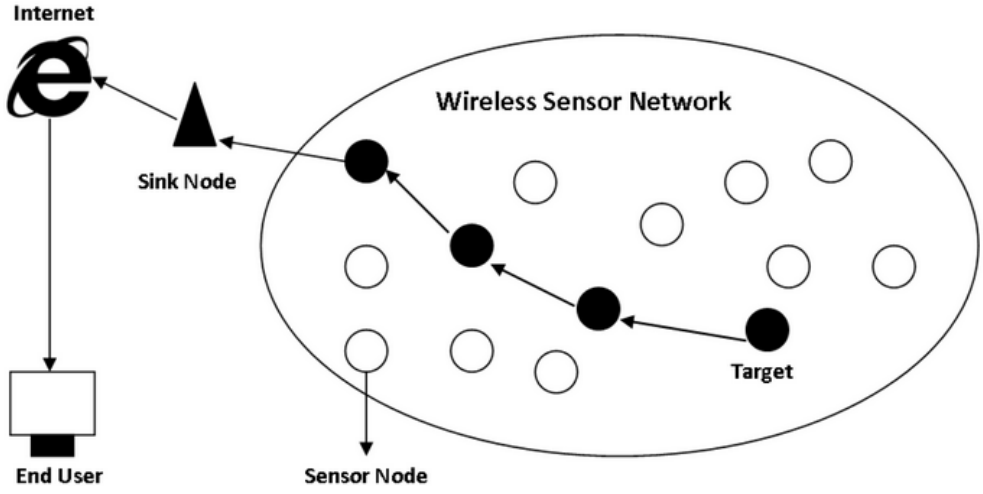

Figure 2.1: Wireless Sensor Network [32]

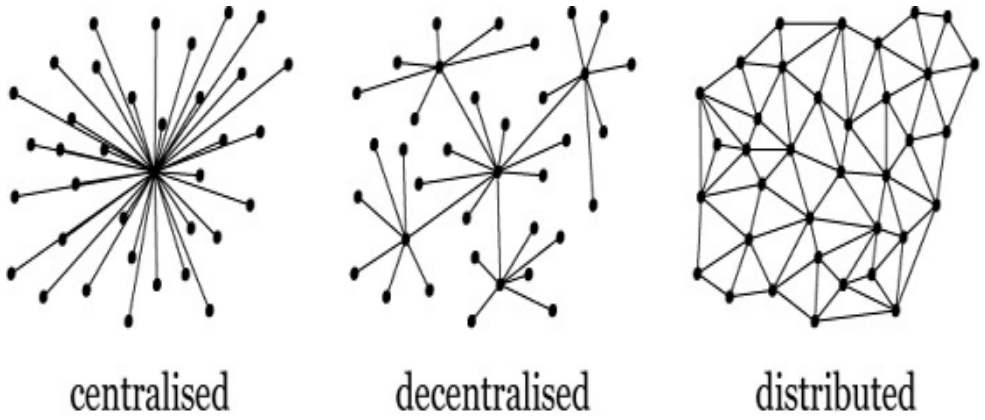

Figure 2.2: Types of WSN control [26]

Structured and unstructured WSN are the two main types of WSN. Structured WSN includes a small number of sensors which are easy to manage. In comparison, unstructured WSN is very much difficult to manage as it comprises a high number of sensors. WSN control is classified into centralized, decentralized and distributed control. A centralized control describes that there is only a single sensor node that has an overview of the entire network and is in control of the specific functionalities of other sensor nodes. In a decentralized control, there is the partitioning of various nodes into groups with a central node for each group that oversees all functionalities of nodes within a particular group. A distributed control has no central node hence no specific nodes delegates any functionalities to other nodes.

2.3.2 WSN Protocol Standards

Some of the proposed or developed WSN protocols are discussed as follows: *Mostafaei and Month* [26] describe the main design objective of WSN is to achieve a low power consumption as stated by *Yick et al.* [34]. Four main standards are included in WSN; ZigBee, IEEE 802.15.4, ISA100.11a and 6LoWPAN. The IEEE 802.15.4 is a standard that is stated by *Howitt and Gutierrez* [35] in [26] which aims at achieving low complexity, low power consumption and low-cost implementation, therefore, it is designed particularly for LR-WPAN (low-rate Wireless Personal Area Network). Additionally, under the physical layer, this standard offers bands between 868/915 MHz and 2.4 GHz. The main challenge of this standard is the maximization of sensor residual power. ZigBee [26], stated by *Kinney* [36], is described as a standard that supports larger networks that consist of about 65 thousand sensor nodes. The respective sensors can keep track of the environment for a lot of years due to the features that ZigBee possess including low power and low cost. The functions or operations of this standard occur on top of the IEEE 802.15.4 standard.

6LoWPAN is a standard that is outlined in [26] and defined by *Shelby and Bormann* [37] to be based on IPv6 Low Power Wireless sensor networks that are built on or are enabled over IEEE 802.15.4. This standard allows communication between low-power sensors and IPv6 speaking devices, which is achieved by the adaptation layer that has abilities to accommodate IPv6 packets into IEEE 802.15.4 frames. The ISA100.11a is explained by *Yick et al.* [34] as a standard which is designed to achieve automating and monitoring of applications using low-rate wireless communication. The main advantages include interaction capabilities with other devices, scalability and low energy consumption. Furthermore, ISA100.11a consist of a simple but more robust or strong security system for securing data or for data protection [26].

2.3.3 WSN Challenges

This subsection presents some of the challenges faced by WSN:

A. Security

Zhou et al. [38], Kgogo et al. [39] and *SW. Pritchard et al.* [40] assert security challenges or attacks that follow a similar classification or categorisation scheme. The attack technique is a type of security vulnerability that is based on eavesdropping, as the injection of false packets causes some sort of turmoil or confusion in the network [38-40]. Node compromising is a very malicious attack whereby attackers take advantage of nodes that are within a hostile environment that lacks continuous network monitoring [38-40]. Furthermore, passive and active attacks are types of attacks whereby an attacker extracts very sensitive WSN security information unknowingly or without any trace [38-40]. An external and internal attack occurs when authorization is granted to access the network and compromise an entity as network authority depends upon the legitimacy of entities (sensor nodes) [38-40].

B. Localization

Singh et al. [41] describe different factors or parameters that are considered when addressing localization as a challenge. (a) Accuracy is a very important concept when localization is considered especially in applications that highly depend on precision like military sensor networks, this is due to accurate intrusion detection. (b) Cost is another challenge as far as localization is concerned. Many localization algorithms deployed to address low cost ought to give rise to a low accuracy rate. (c) Power plays an important role in WSN and the nodes are to be of a fairly distributed localization as the power that is supplied by the battery is limited. (d) Static nodes have similar attributes as battery power

and (e) Mobile nodes have more battery power compared to static nodes/ devices. This describes that during the localization process, there should be less change during algorithm runtime in mobile nodes compared to static nodes which have limited battery power [41]. *Paul and Sato* [42] explain that during localization, accuracy depends on the number of anchor nodes (position and distance), therefore, the increase in anchor nodes results in higher accuracy but is consequent in a higher cost for the whole system [42].

C. Reliability

Mahmood et al. [43] describe and classifies reliability into two main factors: (a) Retransmission-based reliability and (b)Redundancy-based reliability. In a WSN, to ensure reliability in data transmission, retransmission is the main technique that is used. This can be performed using methods including end-to-end or hop-by-hop. However, even with the use of these techniques, the sensor nodes suffer a great all-back in terms of battery power limitations, limited memory and insufficient transmission range, which result in the data delivery being more difficult [43]. During retransmission of data, loss of bits in a packet can occur, and corrupted bits ought to emanate as well. However, there are different coding techniques to ensure retrieval or recovery of the corrupted or lost bits, this ensures reliability, hence describing redundancy-based reliability. Various protocols are proposed to address redundancy-based reliability. These protocols cause high traffic overhead which is consequent to congestion, as there may be large amounts of lost fragments or corrupted bits resulting in a degradation of the network [43]. *Dâmaso et al.* [44] describe the challenge(s) that ought to occur in a WSN as communication link failure and sensor node failure after failure of packet delivery to the sink node. Link failure is caused by various factors like weak signal due to distance, noise or interference and environmental state or condition like walls. Furthermore, sensor node failure possibly results from software deficiency or a breakdown of the hardware[44].

D. Routing

In terms of topological context, WSN suffers greatly as it experiences traffic congestion, limitation in memory capacity and loss of packets due to the unstructured design of the network. Hence, the traditional routing protocols are not of great help in addressing such great challenges. Failure to address routing challenges can lead to much greater challenges like QoS, security breaches and an increase in power consumption [6]. *Pantazis et al.* [45] assert that the two most important features to be considered when addressing the challenge

of efficiency of routing protocols are energy consumption and the life span of a network with consideration of all classified routing protocols, including reliable routing, network structure, communication model and topology-based routing [45].

E. Energy

Sensor nodes are the brains behind the activity of the network, however, they suffer from an energy life span because they run on batteries. Therefore, the lifespan of the network must be sustained by efficiently managing the battery power. Energy consumption can be addressed by implementing protocols but this becomes inefficient when the WSN becomes larger over time. Many solutions ought to be proposed but this can become difficult as an efficient solution highly depends upon the deployment application. Furthermore, sensor nodes are used for monitoring and tracking very high-priority applications like in the military. However, low power transmission is one of the factors that create a fall-back in terms of achieving energy efficiency in any WSN [6, 26].

Addressing energy efficiency in WSN challenges to be considered are classified into the *lifetime* and the *coverage* of the sensor nodes. Sensor nodes use batteries which in essence have a limited lifespan because they are deployed in a resource constraint application like agriculture and wildlife niche for tracking. To address these factors, an efficient routing-aware approach is one of the few efficient solutions to be considered to ensure a higher lifespan [46, 47]. In a coverage context, taking into consideration that sensor nodes have limited battery power, a challenge of node coverage arises whereby there can be an occurrence of network disconnection in certain parts of the network topology. Deploying many sensors can cause an increase in network cost, however, coverage sensor partitioning is efficient. Operation time switching of sensors in terms of network activities is another addressing challenge that occurs when all sensors are working [48, 49].

2.4 Software-Defined Networks

This section provides an overview and a brief introduction. An in-depth explanation of the SDN architecture is then further described in the section together with the respective protocol, OpenFlow. Different applications and challenges will finally conclude this section.

2.4.1 Overview

SDN is described as a more simplified approach to managing networks through decoupling the control plane and data plane to achieve a central control that oversees the entire network [1].

To obtain a more configurable, programmable, manageable and flexible network, the controller consists of policies that edict the network. In this manner, it is the central intelligence that controls the data plane, however, communication between these two planes is important. OpenFlow is the most important protocol that ensures communication between the planes, and it will be further discussed later in this section [50, 51]. With the efficiency of the decoupling of the data plane and control plane, using centralized control, the network still faces scalability and security challenges, hence, bringing a new term, decentralized or distributed SDN. These challenges are to be explained in-depth with the progress of this section. With the distribution of controllers and efficiency of the controller's attributes, many applications like IoT and WSN have taken advantage of this technology which is discussed later in the application in this section [1, 50].

2.4.2 SDN Architecture

This subsection describes and explains the different parts that are included in the SDN architecture which is presented in Figure 2.3.

Control Plane: The control plane is the “decision-making intelligence” amongst all the components of the entire network; meaning it is liable for the network control and management of the activity within the network. The control layer or plane serves as an intermediary and an entity that provides communication amongst the different network applications as well as the network data plane or simply the devices (switches). The controller enables physical distribution as it is logically centralized to eliminate challenges including, scalability and reliability. The mentioned attributes of the controller are supported by the Network Operating System which enables the controller to administer the network hence bringing about an abstract global view of the network. All these different activities are achieved through communication API namely, southbound interface (controller-data plane) and northbound interfaces (controller-application layer) [1, 6, 50].

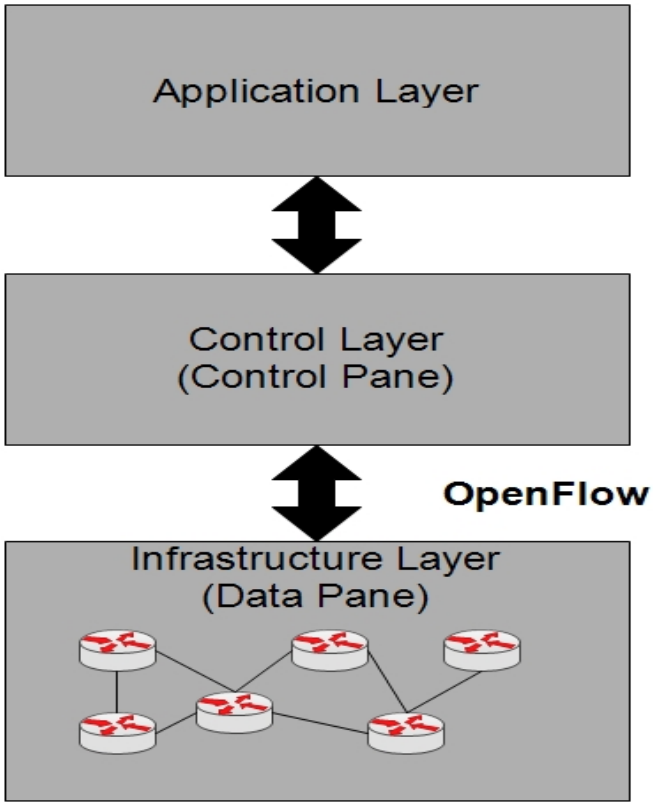

Figure 2.3: Three-layer SDN architecture [52]

The SDN control architecture can be classified into *physical classification* and *logical classification* as shown in Table 2.1. The figure is a summary of the classification which illustrates and broadens the scope of how the growth of SDN can handle large-scale services or wide area networks (WAN) and the respective controllers that are used to handle the scaling of SDN [53].

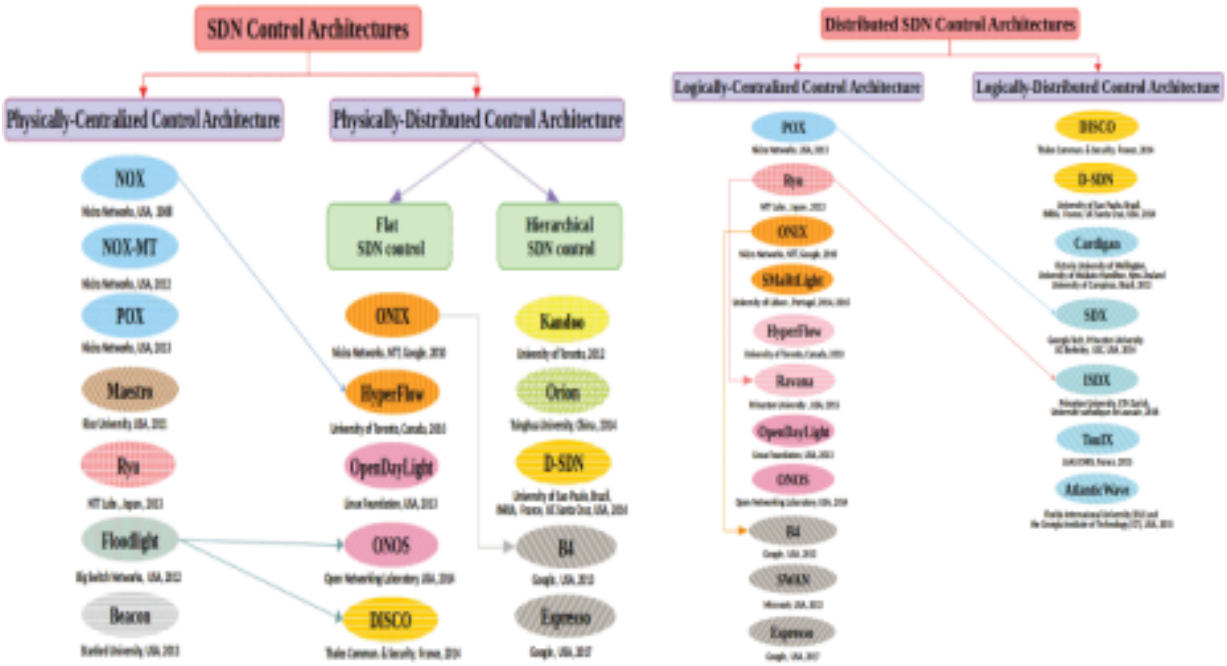

Figure 2.4: Classification of SDN Control Plane Architectures [54]

Physical Classification	**Logical classification**
Physically Centralized SDN Control	Logically Centralized SDN Control
Physically Distributed SDN Control 1) Flat SDN Control 2) Hierarchical SDN Control	Logically Distributed SDN Control

Table 2.1: Summary of SDN Control Plane Architecture Classification

Data Plane: This layer is made up of devices or nodes that are in charge of packet forwarding or transit through the network. The only thing that the devices consist of is the different rules which are legislated through or by the controller on the manner the packets are to be forwarded as the devices receive the respective packets. The data plane is isolated from the software-based control making the OpenFlow and ForCES communication protocols to be of great use as they are highly used. These are the protocols widely used in the southbound interface mainly for communication basis between the data plane and the software control. The two protocols are discussed in much detail as the section progresses [6, 50].

Application Layer: The application layer, as illustrated in Figure 2.3, is situated just above the control plane and it serves as the policy or rule container for the control plane for the

programmability of the physical network. It includes all specific implementation information or rules including Application-Layer Traffic Optimization protocol (ALTO) and eXtensible Session Protocol (XSP) amongst a few. It is connected to the control plane and communicates with it via the northbound interface to manipulate the physical layer using the control plane [55]. Furthermore, this brings about efficiency in controlling the entire network. However, the northbound interface that is utilized by the SDN is not sustained by an accepted standard, hence, it is classified as including [53, 55]:

a) *Simple Primitive APIs*: include low-level ad-hoc APIs as they are connected to the controller, they allow the developer to immediately implement the various applications within the controller. NOX using C++ and POX leveraging Python are a few controller examples that independently make use of their APIs. Web-based service controller APIs are another category of simple primitive APIs found in SDN controllers. These APIs functionalities and services are accessed outside of the controller i.e. through an external server. The floodlight controller serves as an example that utilizes this type of API [53].
b) *High-level API*: is the second classification of northbound interface which is described as being "domain-specific" utilizing programming languages. These APIs aim to achieve a more abstract network to bring about flexibility in application development in terms of network policies of higher-level [53].

2.4.3 SDN OpenFlow

As discussed in the previous section, SDN OpenFlow leverages the functionality of decoupling the data plane and control plane due to the centralized control of the entire network. Furthermore, this separation of data and control plane allows flexibility within telecommunication-based SDN networks. However, this brings about architectural management due to the decoupling of the data and control plane, hence, communication of the two planes is of high priority as the concept of OpenFlow arises. It is described to be a southbound interface protocol that communicates to the low level of the SDN and the data plane, which includes the network devices like the routers and switches. All devices within the SDN network are supported by the OpenFlow protocol comprises of two logical elements:

a) *Flow Table:* This component consists of rules or policies that govern and defines the procedure of how packets are to be forwarded and processed in the network [56].
b) *OpenFlow API*: The administration of data or communication between the network devices (switches or routers) and controllers, are handled by this component [56].

Figure 2.6 is the basic illustration of how OpenFlow works, and how communications occur between the control plane as well as the data plane. Moreover, there are different ways in which communication is reached through the OpenFlow protocol:

1) *Controller-to-Switch* is the type of communication that enables the configuration functionality through switches, flow tables and establishing handshakes [56].
2) *Asynchronous* is described and shown by an illustration in Figure 6a. The main functionality is the OpenFlow-complaint switch which starts by informing the controller through a series of steps like sending "packet-in messages, port status messages and flow removed message" [56].
3) *Symmetric* communication as compared to asynchronous only initiates communication by connection detection, hence, there is no restriction on the source of exchange. This serves as important employing switch-controller connection problem detection, hence, initiates message instances like hello and echo which is illustrated in Figure 2.5 b [56].

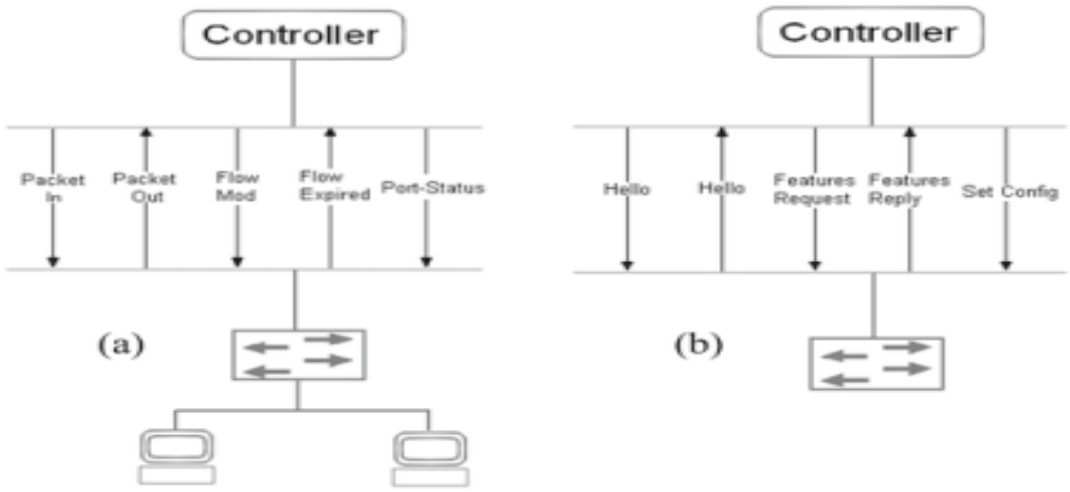

Figure 2.5: OpenFlow Message Exchange [56]

2.4.4 SDN Applications

A. Enterprise Networks: Large-scale networks are often comprised of enterprise networks. University networks are one of the instances that can be an example classified into this category. This type of network includes many different attributes, performance requirements and a vast or different community of users. The devices found in such networks are temporary, meaning they are not in control by the enterprise or in this case, university control. This brings about a challenge in security and allocation of resources. In this instance, sufficient administration is of out-most importance, therefore, SDN can be used to efficiently manage and organize the operation within the

network. The SDN brings about programmability; this functionality can be of good use to efficiently configure the reinforcement of administration policies and assist in network performance tuning and monitoring of network operations [1, 57, 58].

B. **Data Centre:** Data centres are the most critical part of an organization and the heart of everyday InfoTech operational functions which houses all the storage, management and processing of the important data. Furthermore, throughout the advancing years, data centres have kept a constant evolvement in the rapid and high demands in the network. However, the lack of policy enforcement and traffic management results in a decrease in organization productivity and a loss in profit brought by disruptions or delays within the network, especially in large-scale growing networks [57]. Much work has been investigated to optimize the functions and monitoring of data centres as they are susceptible to high demand in service workload. Moreover, the function or characteristics of a centralized programmable controller makes it efficient to serve the demanding network of data centres [1, 57].

C. **Wireless Access Networks***:* Much work has highlighted the need for simultaneous function in various wireless infrastructures while maintaining manageable aspects of a network. Deploying an SDN-based wireless architecture allows a user to utilize different wireless access networks like cellular and Wi-Fi. However, this type of architecture is based upon challenges faced when scalability is involved through mobility when considering the data plane of the wireless networks. Furthermore, the SDN-based architecture allows programmability functions that ensure there is the deployment of routing protocols and mobility-aware network designs that support wireless nodes including OpenRoad which supports Wi-Fi and LTE. This results in an efficient solution to ensure mobility, routing efficiency and control productivity of controllers [1, 57].

D. **Optical Networks:** SDNs (OpenFlow) allows the support for integration in multiple network technologies through data handling employing traffic flows. According to the Optical Transport Working Group (OTWG) as stated in [1], combining both OpenFlow standards in SDN with optical transport networks serves as the beneficiary utilizing control improvement and flexible management within the Optical transport network. There has been work that has been produced for the emphasis on the integration of SDN and optical network including SDON architecture asserted in [57] with further QoS - aware developments in control protocols. This work has shown optimization in the performance of the network as well as capacity improvement [1, 57].

E. **Home and Small Business:** Home or business-based networks are very difficult to manage as they are progressively becoming complex due to the devices that are of common availability in low-cost network access. Furthermore, this calls for an increase in more efficient network management and much-secured security measures to be considered. Consequently, if these measures are not considered, they could result in targets unknowingly, therefore, causing an outage in the network employing external malicious configuration to the business network; affecting the business heavily. Practically, there can't be a network administrator in every office, therefore, an SDN controller can address such challenges through a dedicated centralized controller by monitoring and preventing any attacks [57, 59].

2.4.5 SDN Challenges

Software-Defined Networking experiences a lot of challenges and differs over the progress of the years. Below are a few challenges amongst the many that impact the SDN, however, these challenges are the most popular in research.

A. **Reliability:** The controller which serves to be the brain of SDN is expected to increase network availability through the prevention of manual errors by utilizing efficient configuration and validation of the network topology. The absence of a backup or standby controller will consequently lead to a collapse in the network. Exploiting the centralized controllers' main functionalities would increase the reliability of the networks. There might occur a path/link failure within the network, therefore the controller is to respond quickly through a multi-path support solution and/or deploy a traffic rerouting mechanism to ensure active links [53, 56].
B. **Scalability:** The decoupling of the data and control plane has a great drawback; as such the growth and scaling of the number of switches and end hosts in a network the SDN controller can be left overwhelmed. Flow entries limit the controller, leading to an overwhelmed controller which results from an increase in queued requests due to the increase in bandwidth, number of switches and flows. The queued bottleneck is called overhead. Flow setup is another scalability limitation that additionally can hinder the performance of the network. Flow setup includes steps such as switch requests and new flow entry instructions from a controller [53, 56].
C. **Low-level Interface:** The simplicity of SDN provides communication between the control and data plane using effective and easy-to-use control applications (APIs) to create or determine the set of respective network policies that should govern the

network more efficiently and securely. However, the challenge with these APIs is configuration difficulty utilizing high-level network policy translation into a low level which can be processed or understood by the underlying switch nodes. Furthermore, there is now a controller improvement through the support of a low-level programming interface which requires constant coordination of tasks or events occurring on switches to perform even the simple processes [57-59].

D. **Performance and Security:** As the SDN controller is the central control of a network and serves to provide simplicity throughout the network, the network is still susceptible to security risks. These emerging risks are failure to integrate existing security developments or technology and not being able to monitor every packet. Improving the intelligence of the SDN controller results in the more efficient management of the network, however, this could lead to vulnerabilities for hackers to manipulate or exploit with access to the controller and, control over the entire network [57-59].

2.5 Software-Defined Wireless Sensor Networks

This section describes various factors that have an impact on the development of SDWSN. The various elements or factors include the specific application, respective challenges that are experienced within this discipline and the architecture that administers and illustrates the functionalities and particular communication elements that occur in the actual network.

2.5.1 SDWSN Architecture

Figure 2.6 describes the general architecture that is depicted in an SDWSN. The architecture consists of three main logical planes or layers namely: a) Application plane/layer, b) Control plane/layer, and c) Data plane or Infrastructure layer. Data is acquired through sensing by the sensor nodes, then the data is forwarded in the network, this process occurs in the data layer/plane. The controllers that are responsible for the governing of the entire network are found in the control layer/plane. The application layer/planes include the various laws or protocols that assist the control layer to govern the network efficiently such as routing protocols [6, 19, 60].

Sensor OpenFlow (SOF) is the communication protocol that is stated in [60] which is described to be the first communication protocol when the first SDWSN architecture was introduced. SOF is found between the infrastructure layer and the control layer which aims at maintaining the forwarding process of flow-based packets from the sensor nodes which will be processed by the controller as the smart or intelligent agent resulting in a decision-making phase. The

control plane and the sensor nodes (data plane) communicate using SOF as it supports both non-IP and IP-based communication. Its main goal is to utilize APIs to achieve the programmability of the data plane of the WSN [61, 62]. SDWSN is generally categorized into two control approaches:

a) *Directly Connected Controller:* The direct communication between the controller and all the sensor nodes within the infrastructure layer results in a separate channel requirement to maintain traffic control [61, 62].
b) *Indirectly Connected Controller:* Controller and sensor node communication occurs by leveraging other sensors' nodes multi-hop communication can be an example [61, 62].

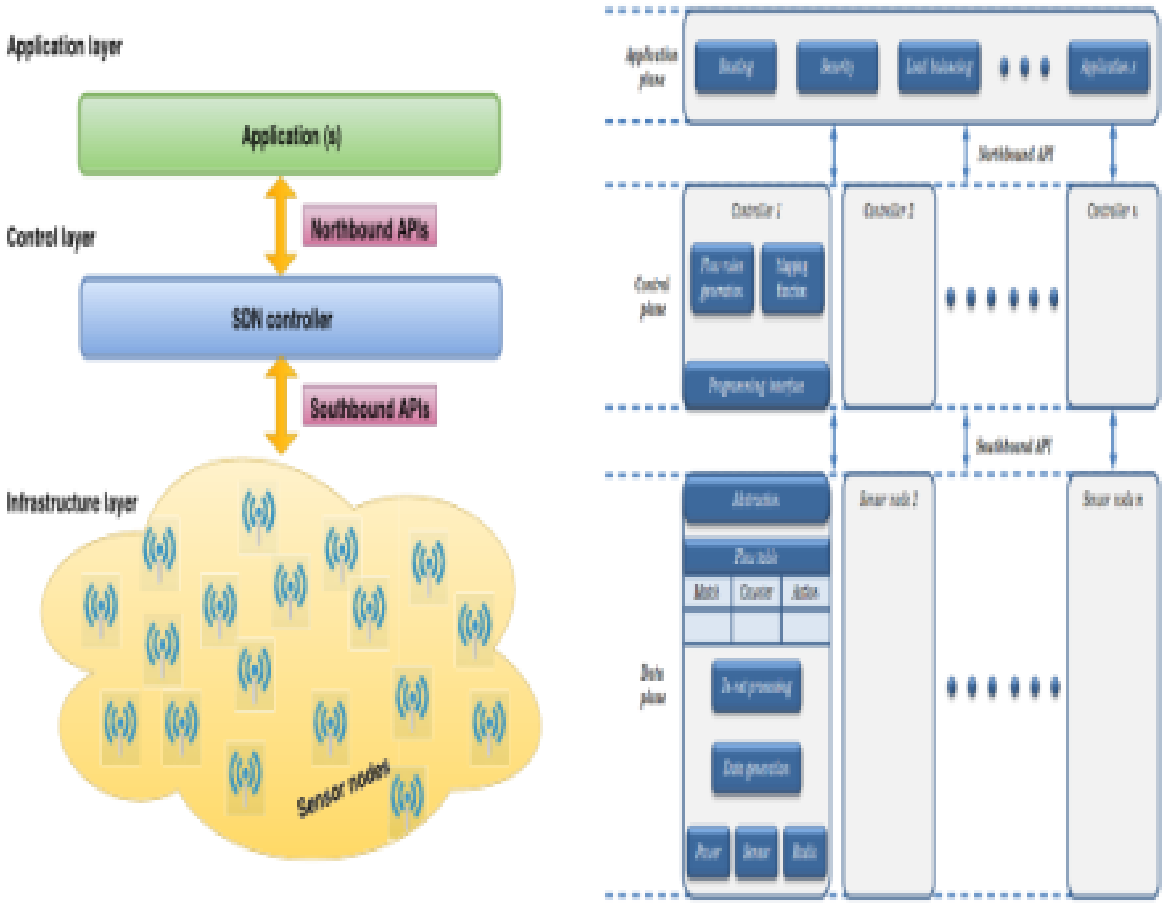

Figure 2.6: SDWSN Generic Architectures [6, 26]

2.5.2 SDWSN Applications

The following are all WSN applications that serve to benefit from SDN characteristics or technological implementation.

A. Health Care Application

Quality in the health of individuals with chronic illnesses as well as the elderly people is to benefit from WSN-based technologies including patient tracking within the hospital particularly wheelchairs, psychiatric patients and bed management. Deployment of sensor implants in patients ought to benefit patients and doctors for better health monitoring and better diagnosis. However, with these benefits, the energy supply of the sensor nodes has battery constraints and this can further be critical for the elderly suffering from memory loss, they might forget to charge the batteries. Additionally, reliability of packet delivery due to traffic overhead, QoS is affected by interference through the placement of multiple sensors nearby, and privacy and securing of medical data that is transferred through the network.

SDN controller firstly provides encryption techniques to eliminate security and privacy challenges. Controller functions that allow provisioning or separation in the sensor of proximity to eliminate node of destructive inferring. The central control in the SDN provides monitoring of the network, allowing flow and accuracy in traffic or packet delivery [26, 60].

B. State Protection or Military Applications

The use of WSN in the line of defence includes enemy presence and weapon or chemically developed weapons' tracking or detection. Furthermore, monetization of fleet or troops, fire control and efficiency in managing health and safety of troops during attacks. These technologies are less positive when susceptible to risks however, the SDN controller leverages the global view for configuration and management purposes of the network resulting in more security protocols to be deployed, yielding a secure and robust network [26, 60].

C. Wireless Home Networks

There is an increase in smart homes and this has been seen through IoT with the use of sensors that can be remotely controlled, including the lights of the house and the temperature of heaters and devices such as air conditioners. These devices are integrated via the internet through communication mediums and use communication technologies including power-line communications, Ethernet or Wi-Fi. The challenge of this scenario is that the network consists of a heterogeneous architecture which is not automated. The user manually selects the preferred communication medium resulting in a less optimized QoS

as a lack of knowledge in this case. SDN provides serves to manage and provision heterogeneous networks through ISP's resulting in a better QoS quality [26, 60].

D. Environment based Applications

The development of smart water grids for even distribution of water, monetization of forest fires and natural disasters like earthquake detection are all implementations that were motivated by irrigation of animals or monitoring of animals and disaster relief for rescue operations. This environment is driven by applications deployed through sensors making up a WSN with the placement of sensors in very hard-to-reach places and very large areas with harsh conditions. As such, the placement of sensors requires priority in the energy supply for the sensor nodes to increase the lifetime of the network and the ability to adapt to changes in sensing requirements as well as the failure of the node [26, 60].

2.5.3 SDWSN Challenges

The following are some of the problems that SDWSN faces:

A. **Network operation:** The previous section highlighted different applications which can benefit from SDN. There are, however, some issues to consider when developing techniques and in this subsection, we mention network operation-based challenges. *Re-clustering* is a drawback that is caused by the limitation of energy of cluster heads from the overwhelming number of communication from other nodes in the cluster. This calls for a dynamic reallocation of a new cluster head to regulate energy within the cluster and for the network. *Topology control* emphasizes the need for a complete control protocol for an SDWSN topology because control topology protocols try to maintain connectivity and network coverage. *Node mobility* is caused by sensor nodes changing position in a network intentionally, this causes a change in topology and affects the coverage in the network. This calls for a need for a more flexible and dynamic topology management mechanism or approach with consideration of resource limitations such as power limitation and bandwidth [6, 26].
B. **Routing:** is of importance particularly when routing protocols are the source to produce a topology in an SDWSN, hence, coming up with an improvement in routing is a challenge due to performance objectives that conflict with one another. SDN controller by a global overview of the network assists in creating multiple constraints optimal paths which may increase efficiency in routing. *Data traffic scheduling* in SDWSN is affected by data transmission that uses up to about 80% of energy in a sensor node. This

is a great challenge as an efficient scheduling approach to saving energy is to be investigated with consideration that the traffic scheduled or transferred to the node is done efficiently and nodes of higher energy residual are to be of alternative in the process of scheduling. *Network monitoring* arises due to unstable wireless links that cause packet loss as well as sensor nodes failing in the process of operation. This calls for real-time monitoring tools to ensure network availability and good network performance [6, 26].

C. **Network applications:** *Coverage* is a network application-based challenge that is influenced by the network topology area coverage with the goal of nodes covering the whole network area. There are two types of coverage challenges, and these are: (i) *Partial coverage* arises as it only considers just a minimal percentage of network area monetization, therefore, it is not enough. This results in more consideration in developing future work in the discipline of SDWSN. Due to the mobility of the sensor nodes, coverage algorithms are needed to evenly distribute coverage of nodes in a topology however, (ii) *Coverage holes* are likely to occur with an area in the network which is not covered by a node(s) or the chosen active node(s) in the network. Coverage holes are likely to cause poor QoS in the network and packet loss, hence, there is a need for future consideration and investigation [6, 26].

D. **Node mobility:** is a network application-based challenge that needs further investigation. However, node mobility eliminates the challenge posed by the above-mentioned coverage holes. It leverages the functionality of a moving node to cover the network region with a coverage hole area [6, 26].

E. **Control plane vs Data plane resilience:** Control plane resilience occurs when there is a need for multiple controllers and a challenge of inter-communication which affects controller-to-controller [26, 60]. Data plane resilience includes occurring failures among nodes or links which will therefore affect the packet to be delivered. However, the SDN repairs this by installing a set of new instructions (flow entries) [26, 60].

F. **Scalability:** SDWSN is a field that adopts both WSN and SDN challenges. Scalability challenges are SDN-based with emphasis on the robustness of the network. A robust network has a scalability challenge through an optimal controller placement employing and eliminating the single point of failure, considering multiple controller placements [19, 26, 60].

G. **Security:** The majority of work that serves to eliminate security issues is mainly SDN-based and considers switches and routers with less solution in sensor nodes or WSN-

based. The security vulnerabilities or threats are very important as indicated in the previous section of the *SDWSN Application*. The military application utilizes or serves to benefit from SDWSN through the efficiency of the functionalities that the SDN controller has [26, 40, 60].

2.6 Controller Placement Problem

In this section, a brief overview of CPP has been defined. Additionally, various performance metrics affecting SDN controller placement are explained with respective equations.

2.6.1 Overview

Decoupling of a network into the data plane and control plane faces a single point of failure through the use of a controller i.e. control plane, hence calls for a multiple of controllers given a network G with E edges or distance $d(v,s)$ between a set of V vertices modelled as $G(V, E)$. However, achieving optimal CPP is a challenge considering (a) the number of controllers needed, (b) wherein a topology is to be placed and (c) allocation of the controllers to switches to efficiently optimise network performance. The controller placement problem is modelled by minimizing and/or maximizing the various performance metrics that ensure a better quality of service and network requirements, given a network [18, 23].

2.6.2 Performance Metrics

Figure 2.7 presents the classification of the CPP in terms of four optimization objectives or metrics that are to be explained throughout the section. Figure 2.8 also includes a multi-objective which is not further justified in the section. However, the section explains the integration of the multiple optimized objectives. One of the performance metrics is load balancing.

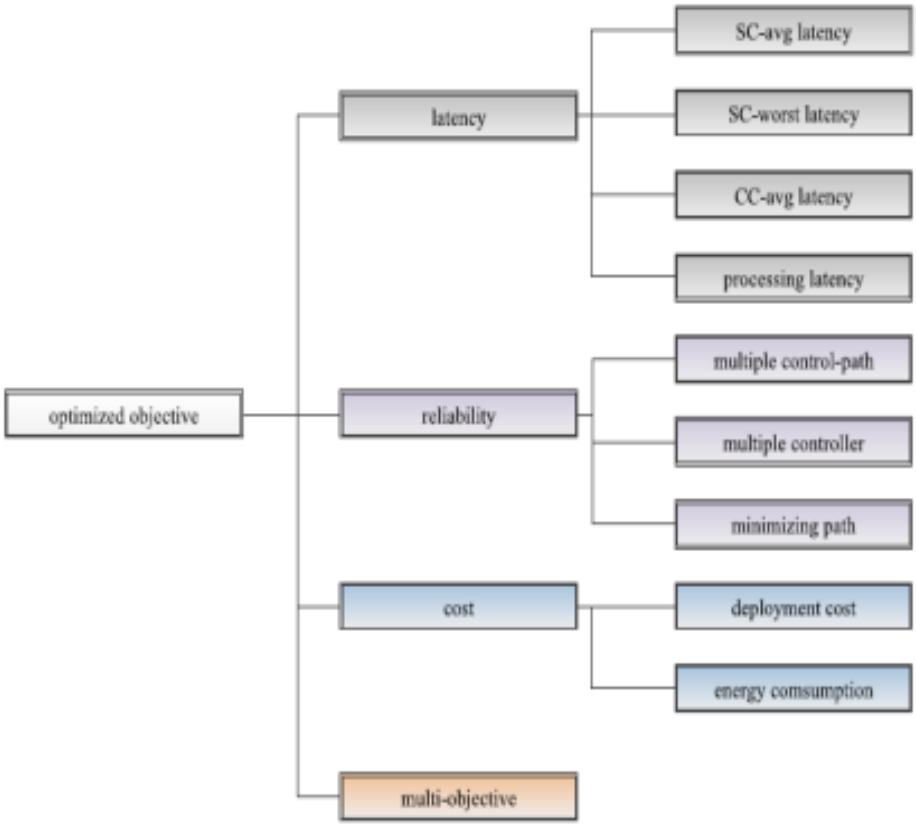

Figure 2.7: Classification of CPP Optimization Objectives [18]

I. Latency

For a network to function, there should be communication links between controllers and switches which ensures message exchange thus making latency important. With that being said, the distance between network nodes has a major impact on latency which is one factor that ensures the type of latency namely packet propagation latency. The controller processing latency, on the other hand, is believed to be greatly influenced by controller load. These two types are the most considered types of latency as they have a much higher effect in wide-area networks which is a research area that researchers are much interested in future optimal placement of controllers as networks become much larger in future. The optimality influences Controller-to-Controller (CC) and Switch-to-Controller latency (SC) [18, 23, 24].

- ***Minimization of Propagation Latency:*** This is considered a Controller placement problem by which the distribution of nodes ought to be optimized, giving rise to average and the worst-case scenario in propagation latency. These two scenarios find the optimal solution in controller placement with the average-case propagation latency

being the most efficient [18, 23, 24]. The propagation latency in consideration of the two scenarios is formulated as follows:

- *Average-Case Latency*: Given a network graph G (V, E) of which the edge (E) weights represent the propagation latency, with the shortest distance d (v, s) from node v ϵ V to s ϵ V, with the number of nodes being n = |V|. The average propagation latency for the placement of controllers S′ is shown as Equation 2.1:

$$\boldsymbol{L_{avg}(S') = \frac{1}{n} \sum_{v \epsilon V} \min_{(s \epsilon S')} d(v, s)} \qquad \boldsymbol{(2.1)}$$

- *Worst-Case Latency:* being an alternative metric, the WC latency finds the maximum Switch/Node-to-Controller propagation delay (Equation 2.2):

$$\boldsymbol{L_{wc}(S') = \max_{(v \epsilon V)} \min_{(s \epsilon S')} d(v, s)} \qquad \boldsymbol{(2.2)}$$

- *Minimization of Controller Processing Latency:* The multiple controller placement in a network is of utmost importance to eliminate a single point of failure. However, to ensure global view consistency of the network states among all controllers, communication between these controllers must be maintained more effectively. Considering the load capacity within the network, the controllers are to maintain an even distribution or communication overhead of load capacity resulting in a maintainable shared state of the controllers [18, 23, 24]. The controller-to-controller latency is modelled as a mathematical expression as shown in Equation 2.3:

$$\boldsymbol{L_{cc-avg} = \frac{1}{k} \sum_{s \epsilon S} \min_{(s' \epsilon S)} d(s, s')} \qquad \boldsymbol{(2.3)}$$

- ***Wireless Average latency:*** Is described as the round time taken from a node to another node in a network. Furthermore, with consideration of the wireless feature, an assumption of a Rayleigh fading channel with no line of sight is considered. Transmission from node *I* to node *j* is said to be successful if $SINR\gamma ij$ has reached an acceptable threshold (Θ). However, the interference signal created is not considered when the threshold is not reached. $SINR\gamma$ is formulated as shown in Equation 2.4. [63]:

$$\gamma = \frac{Q}{No + I} \qquad (2.4)$$

The noise power is denoted by No, I is the interference power (sum of all received power of undesired transmitters), and Q is the power received.

A Rayleigh fading network consisting of a slotted ALOHA and the nodes with an equal power transmission level with probability p. Given the desired distance d_o and n other nodes at distance di (I = 1,……,n) between transmitter-receiver, the probability of transmission is denoted by (Equation 2.5):

$$P_s \mid d_o \ldots d_n = exp\left(-\frac{\Theta N_o}{P_o d_o^{-\alpha}}\right).\prod_{i=1}^{n}\left(1 - \frac{\Theta p}{\left(\frac{d}{d_o}\right)^{\alpha} + \Theta}\right) \qquad (2.5)$$

Parameters included in the equation are described as

P_o: Transmission Power

N_o: Noise Power

Θ: *SINR threshold.*

In a proposed Carrier-sense multiple access with collision detection (CSMA) architecture, collisions are expected to occur through two network node transmissions. Moreover, wireless nodes or Access points (APs) with a much further reception radius of R=50 cause negligible interference. High transmission power used by the controller causes interference even though the reception radius is much further away. Considering all interfering nodes in a set of interferes A^I, the total of all interference received by the AP_i is formulated by [64, 65] (Equation 2.6):

$$\zeta^{ji} = \sum_{z \,\epsilon\, A^I, i,j \,\epsilon\, A \mid j \neq z, A^I \subset A} P_{zi} \qquad (2.6)$$

The parameters found in equation (3) are described as:

P_{zi}: represent the received APi signal power from a z_{th} interfering node at a distance of d_{zi}. The set of all network nodes is denoted by *A,* however, the resulting interfering signal power that is found at AP_i from interferes A^I use an SINR equation between the link node j and AP_i that is denoted by [64] (Equation 2.7):

$$\gamma_{ji} = \frac{P_{ji}^{r}}{(\zeta^{ji} + N_o)}, 1 \leq i, j \leq N \qquad (2.7)$$

N: suggest the total number of network nodes.

p^{r}_{ji}: Received signal power over node j at AP_i at a distance of d_{ji}.

Furthermore, the measure of received signal power from node j at AP_i over a distance of d_{ji} is formulated as shown in Equation 2.8 [64]:

$$P_{ji}^{r} = P^{t}(mW) G_{ji} d_{ij}^{-\alpha} (dBm) \qquad (2.8)$$

G_{ji} : Channel gain characteristics by an exponential distribution ($G_{ji} \sim \exp(P_t)$) for an account in fading and shadowing.

α = 3.5: Utilized for path loss exponent.

Equation 2.4 and Equation 2.5 are now utilized to calculate the latency i.e. total end-to-end delay of a packet to a certain station. The equation is formulated as follows [64] (Equation 2.9):

$$t_{ji} = \frac{F(bits)}{R_{ji}} \qquad (2.9)$$

R_{ji}: Rate of transmission, i.e. determined through SINRγ_{ji} experienced by AP_i with the association of controller *j*. In [64, 65] illustrates a mapping in terms of 802.11 between R_{ji} and SINRγji.

When a collision is experienced on a frame between AP_j and AP_i the time of transmission is extended therefore formulated as shown in Equations 2.10, 2.11 and 2.12) [64]:

$$\tilde{t}_{ji} = DIFS + t_{bf} + t_{ji} + SIFS + t_{ack} \qquad (2.10)$$

$$t_{bf} = \frac{CW_{max}}{2} \chi \qquad (2.11)$$

$$t_{ack} = \frac{1}{t} \qquad (2.12)$$

t_{ack} is the time taken to transmit ACK frame considering the basic data rate *r*

SIFS and DIFS: time intervals defined in the standard 802.11. The average latency of a WCPP is formulated as shown in Equation 2.13 [64]:

$$L_{average} = \frac{1}{N_{links}} \sum_{l \, \epsilon \, links} t(l) \qquad (2.13)$$

II. Load Balancing

A controller's load capacity is determined by the number of switches it controls. Each controller has a certain threshold which can handle its respective load capacity; if the threshold exceeds the processing capability, then the controller is susceptible to failure. If a controller fails within a network, it results in other controllers and switches being subjected to an increase in communication overhead. To maintain a load balance between controllers and switches, there is crucial consideration of the optimal placement of the controller with the efficient switch to controller assignment.

Much work has proposed load balance techniques which emphasize the even distribution of load in a network to avoid underloading and overloading of controllers. This could result in poor performance through packet loss among a few factors affected by degraded network performance. Load balancing is classified into 1) *Clustering* which is a process whereby nodes are partitioned into a pool of data plane devices within a certain proximity, with a dedicated controller assigned to manage the load of the domain or cluster. The cluster controller seeks to govern policy or efficient even distribution mechanisms to avoid overload from the global overview controller called the super controller. 2) *Switch Migration* is a technique that is used for a more dynamic network response to overloading. It explains the change in the switch-controller relationship in response to the overloaded controller to another controller that is underloaded as to processes or caters for the switches experiencing overloading from their respective controller [18, 24, 27].

III. Reliability

Assignment of multiple switches to control is susceptible to failure when the controller fails, this will cause a disconnection within the network which consequently result in packet loss. Multiple controller placement eliminates the single point failure, however, it does not justify to reliability contribution towards the controller or the entire network. *Link*

Connectivity is one of the two classifications of reliability as a performance metric in controller placement. With link connection between nodes (switch-controllers and controller-controller), every link in a network has a limited capacity requirement. When there is an occurrence of failure, disturbance or congestion, the switches and controllers fail to communicate efficiently, consequently resulting in packet delivery and isolation of devices. *Node congestion* is another reliability classification or objective that is ought to be minimized when considering reliability or a fault-tolerant controller placement approach. Switches or controllers are susceptible to overwhelming attacks by various vulnerabilities which causes the node to collapse. A more robust node congestion management scheme is needed to ensure a more optimal reselection of controllers when one crashes. Moreover, routing approaches to fast respond to link failures should be considered in controller placement in a CPP approach [18, 24, 27]. Equations 2.14 – 2.16 all form part of this section and they have been explained accordingly.

Link failure Probability: The assumption is that the AP is located in a star topology, meaning it has a single route to the assigned controller. The link or channel of the single route, assuming that it experiences a Rayleigh fading, the probability of link failure is the particular probability that ensures SINR would go below a threshold Θ, this is formulated by Equation 2.14 [64] $\boldsymbol{P_{r_{i,j}}}$:

$$\boldsymbol{P_{r_{i,j}} = 1 - P_{j \mid d_o \ldots\ldots d_n}} \qquad \boldsymbol{(2.14)}$$

As seen in equation 2.14, the formula $P_{j \mid d_o \ldots\ldots d_n}$ is defined with i,j being our source node and destination. Furthermore, the average probability for a link failure is denoted by Equation 2.15 [64]:

$$\boldsymbol{P_{r_{avg}} = \frac{1}{N} \sum_{s \,\epsilon\, C_t} \sum_{d \,\epsilon\, AP^s} P_{r_{s,d}}} \qquad \boldsymbol{(2.15)}$$

N: total number of AP's-controller links, it is equal to the number of APs.

C_t: Set of all controllers

APs: all sets of nodes assigned to respective controllers.

Since the objective (Equation 2.14 and 2.15) of this section is Link Failure Probability, as a wireless requirement or consideration, Figure 2.8 explains how it affects the network i.e. SDN, when not properly addressed. Network failure can highly occur between controllers and/or switches resulting in loss of packets and reduction in the performance of the network hence reliability is of utmost importance. Reliability includes various aspects in the network, firstly, multiple control-path is the alternative paths that assist against link failure and node failure. Multiple controllers are the second reliability metric that ensures switches are connected to multiple controllers to avoid a single point of failure. Finally, Minimizing Control-path includes the failures that occur in the physical components of links and/or nodes as this independent, reduction of physical units within the control promotes reliability [18].

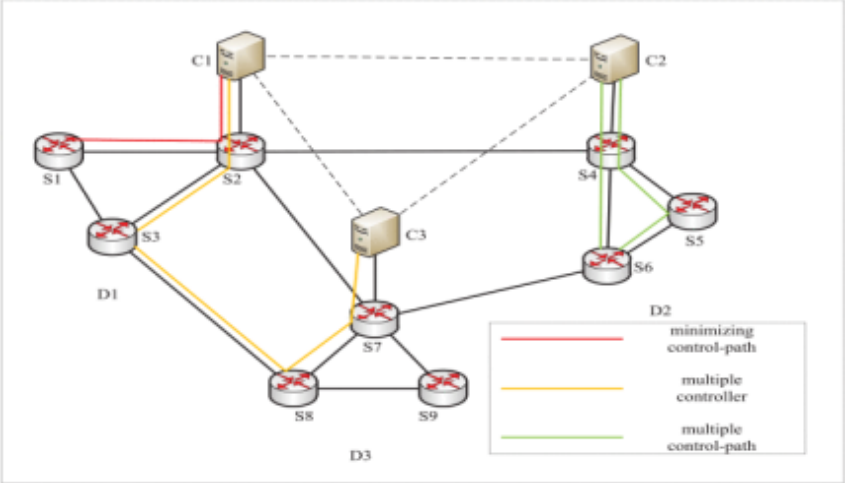

Figure 2.8: Different Types of Reliability [18]

IV. Cost

Deployment cost is the operational and cost expenses or expenditures in developing a convenient network utilizing switch to controller linkages and installation of a controller in a network. With a rapid increase in resource-demanding devices and an increase or growth within the network, this is a performance metric that is to be considered and to be minimized [18, 24].

This function will not be used to evaluate the algorithms in this chapter, however, it is of consideration in this research. It will be considered in benchmarking evaluation through experimentation together with other functions in chapter 6. The objective is based on

controller minimization which constitutes CAPEX and OPEX and is defined in Equation 2.16 as follows:

$$\boldsymbol{f_i = \lambda_i (U_{th} - U_i) + \delta_i(\Delta_{th} - \Delta_i)} \qquad \boldsymbol{(2.16)}$$

Where the upper bound and controller $\boldsymbol{i}$ utilization is represented as $\boldsymbol{U_{th}\ and\ U_i}$ moreover, the current delay and upper bound of controller $\boldsymbol{i}$ are presented as $\boldsymbol{\Delta_{th}\ and\ \Delta_i}$. The controller will delegate its load to another controller if the specific controller has a much greater value on the upper bound as defined by [66].

2.7 Related Works

This section presents the relevant literature that addresses the Controller placement problems using different metrics with specific objectives. The literature is categorized into SDN and SDWSN-based CPP approaches. In the preceding section, we discussed several metrics that are illustrated as sub-classification for each category. Table 2.2 and Table 2.3 illustrates a summary of the SDN CPP and SDWSN CPP algorithm and implementation literature that is explained.

2.7.1 SDN CPP Implementations

The various SDN approaches that serve to address CPP using respective or a particular metric are illustrated below.

I. Cost

This subsection presents different researchers that considered cost in the design of their CPP scheme. *Sallahi and St.-Hilaire* [67] developed an expansion model that covers the placement of controllers, focusing on reducing the cost to meet an increase in demand for users and services. The model uses or contributes to an existing proposed mathematical model as the primary basis to allow expansion scenario modification. The expansion scenarios consist of how many controllers are to be placed, wherein the topology of the controllers are to be placed and which type of controllers to install, given the various equipment as well as existing infrastructure. The model's objective is to reduce costs as much as possible with more consideration of other performance metrics including latency and traffic to find the feasibility of the approach or model. CPLEX was the tool that was used to solve the model. Results acquired from the experiment and the analysis or conclusion asserted that the model was of the general approach to planning a problem to achieve cost minimization. Therefore, a new

network can be designed from scratch or updated the existing one given the particular specification or attributes of the network.

Most work that addresses CPP don't usually follow one metric, the primary metric is achieved through the adoption of other performance metrics which will make the solution to be more feasible. This is seen by a Chaotic salp swarm algorithm (CSSA) proposed by *Ateya et al* [68] with consideration of cost and latency. With latency and cost-aware problems formulated, a Metaheuristic algorithm is proposed to solve the problem to find the optimum switch-to-controller allocation and number of controllers to minimize deployment cost and latency. With the use of Matlab for performance evaluation, the results of the proposed algorithm are asserted to perform better compared to the game theory-based and metaheuristic algorithms through reliability and execution time. Also, *Das and Gurusamy* [69] proposed a multi-objective optimization model utilizing Integer Linear programming (ILP) that is trialled in various network scenarios including an increase in network traffic and Capital and Operational Expenditure (OPEX and CAPEX) considering vendor competition and growth or maturity in technology. The approach was solved using a GUROBI optimizer with results indicating that by reaching a 70-80 % cost efficiency or saving, the latency of 35-45 % is to be compromised with a further multi-period schedule of controller placement.

II. Latency

As it has been explained above, the various performance metrics have types of objectives that make up the metric. Besides, controller-to-switch latency and inter-controller latency are two objectives that are introduced and addressed by *Sahoo et al* [10]. Survivability is the third objective that introduces and ensures reliability in terms of connectivity between the network and is the most explored in CPP as it will be seen when this section proceeds further. The multi-objective combinatorial optimization problem (MOCOP) is formulated and solved using two Meta-heuristic algorithms including the Firefly algorithm (FFA) and Particle Swarm Optimization (PSO). The experiments and results were acquired using Matlab R2014a and Topology Zoo library for various network topologies for evaluation purposes. Based on the results found, the performance of the proposed approach had efficiently performed well and achieves a better control path through the improvement of survivability.

Similar to *Sahoo et al* [10], metrics that are considered include latency and reliability, however, *Liao et al* [70] proposed a density cluster-based approach, particularly in SDN. This paper eliminates the drawback of heuristic algorithms that get trapped in locally optimal solutions.

Hence, the approach exploits the splitting technique in the network through clustering, meaning a sub-network has a delegated controller which can focus on the different phenomenon that occurs uniquely in that subnetwork. Problem formulation, experiments and results are acquired from the POCO framework and Internet topology Zoo library for various topologies to run evaluations. In consideration of time consumption, propagation latency and fault tolerance, the proposed Density-based Controller Placement approach show better results compared to other state-of-the-art approaches.

Density-Based clustering faces drawbacks, even though it has been used efficiently as seen by *Liao et al.* . One of the drawbacks includes the random initialization of k values in clusters and cluster heads. *Killi et al* [71] proposed a cooperative Game theory-based network partitioning algorithm which eliminates the random initialization as they are declared to be far from optimal. Since means adopts propagation latency in average and worst-case scenarios, game theory consequently adopts this performance metric as well. The results were obtained using Python using the Internet 2 OS3E topology library. The results suggest that the proposed game theory approach outperforms the standard K-means algorithm with close to optimal solutions considering the worst-case switch to controller latency.

Wang et al. [72] assert an approach of solving multiple controller placement in consequence of minimizing latency by proposing a network partition algorithm. A clustering algorithm is proposed to partition the network into subnetworks. A k-means algorithm is used for optimization purposes to ensure that the maximum latency is minimized between the nodes and specific centroid of a subnetwork. The approach was implemented using various topologies that are widely used, Internet2 OS3E topology and Chinanet obtained from Topology Zoo. The latency performance results were obtained by simulation, indicating that the proposed optimization K-means algorithm performs effectively compared to the standard k-means algorithm with the latency between centroid and nodes being minimized. However, this approach considers a single metric (latency minimization) within subnetworks and does not consider the overall latency including controller-to-controller latency.

Zhu et al. [73] describe how determining the location of controllers in a network has received light in research and is a critical challenge in Controller Placement Problem (CPP) while maintaining the QoS of a network. Authors in [72] consider only the latency minimization between centroid and nodes of a subnetwork, however, *Zhu et al.* [73] propose an enhancement towards the minimization of latency in consideration of the delay between controllers, as well

as between controllers and switches. A heuristic method is used which is based on k-means and the Dijkstra algorithm. The approach was evaluated using simulation and widely used network topologies. The results of the approach are described as being effective by delay performance (latency) as the algorithm was evaluated using the number of nodes (switches) and the increase of delay. The increase in switches increases delay. This approach does justice to the latency metric, however, considers a single objective. Therefore, it does not consider how for instance load balancing can affect or lead to an increase in control plane overall delay.

Previous authors considered different latency constraints. *Li et al.* [74] explain latency as a performance measurement metric that depends on the "maximum or average latency" as being bounded and "balanced cost-latency placement". The optimization of the mentioned constraints results in efficiency in minimization of latency and consequently an increase in the performance of the network (QoS) [74]. A few algorithms have been proposed such as Bounded Maximum Latency Placement (BMLP) and Bounded Average Latency Placement (BALP) which have an intent of identifying the controllers when evaluated using fixed latency requirement, and finally utilizes Balanced Cost-Latency Placement (BCLP) for finding the trade-off cost and latency. A simulation was used to obtain the performance results using Garr201103 network topology from Internet Topology Zoo. The results assert that the approximation algorithm (BCLP) being proposed shows good performance when balancing the latency to control the network and the cost of the respective controllers. With the authors aiming at minimizing latency and cost, the approach still utilizes or considers a single metric for performance measurement. Therefore, the approach arises gaps of an increase of load (controller overload) and how to dynamically address these challenges to ensure a reliable link between controllers and switches.

Overall, latency is made up of end-to-end latency, propagation latency and queueing latency. *Wang et al.* [75] explain that most authors only consider one out of the three contributory towards overall latency, namely propagation latency. However, in this paper, two of the contributory latencies are investigated and addressed. In addressing end-to-end latency and queueing latency, a clustering network partition algorithm (CNPA) is proposed to reduce the end-to-end latency. The CNPA partitions the network into sub-networks, ensuring a reduction in end-to-end latency between controllers and switches. Minimizing queueing latency is achieved by placing multiple controllers within the subnetworks that are partitioned. The evaluation of the proposed solutions is performed under the Chinanet topology and Internet2 OS3E topology. Matlab was used to simulate the proposed method and the results illustrate

that CNPA can minimize the end-to-end latency between the controllers in its respective switches compared to the existing solution including k-means and k-centre. By adding multiple controllers within the subnetworks, simulation results showed a great decrease in the overall latency. This proposed solution achieved an effective output, however, the increase in multiple controllers brings about a challenge of hardware costs in the development of the network. Hence, that can be of disadvantage to network service providers.

III. Load Balancing

This subsection presents the studies that considered load balance as a performance metric in their CPP design. *Khorramizadeh and Ahmadi* [14] proposed a Capacity and load-aware software-defined network controller placement approach that is evaluated in heterogeneous environments. Since the approach is addressed in a WAN, latency and scalability are to be optimized through the use of multiple controllers. However, location placement, an optimal number of controllers and load balancing are what is focused on as a challenge being particularly highlighted in this paper. A location-allocation model is proposed consisting of two phases that seek to determine the optimal number of controllers considering the minimization of cost. The optimal number of controllers from the first phase serves as input into the second phase to balance the controller load utilizing the proposed fair load distribution function furthermore reducing the inter-controller latency. The evaluation of the proposed approach was done using Matlab 2018a and CPLEX using the Topology Zoo topology library. The results of this approach showed that it's much more efficient, particularly for the various topologies as well as the heterogeneous outperforming homogeneous deployment considering the total cost as well as the fair load distribution within the controllers.

A Kuhn-Munkres algorithm-based solution is proposed in finding the optimal placement of controllers as well as optimal switch-controller assignment more particularly in a WAN environment. The approach acknowledges load balancing and uses a genetic algorithm to achieve accurate controller placement and balance the controller load. *Yaun et al* [76] propose this scheme using the evaluation of the Survivable fixed telecommunication Network Design library (SNDlib). The results acquired suggest that the scheme achieves good load balancing and shows good performance in terms of reduction in propagation delay.

A heuristic that is called Multi-Start Hybrid NSGA-II (MHNSGA- II) is proposed by *Jalili et al*. [77] as an extension of MHNSGA. The approach considers competing metrics that arise as drawbacks in terms of decision-making. These performance metrics include load balancing,

switch-controller latency and inter-controller latency. The approach incorporates trade-offs between these competing metrics and was evaluated and tested using network topologies from the Topology Zoo library. The approach achieved faster computation with more efficiency and performance of the overall algorithm in terms of the Pareto solution set.

Multi-controller placement in SDN as opposed to single placement is much preferred for large amounts of network flow since a single controller cannot handle such an amount of load, hence, multiple controllers ought to ease the workload. Even though multiple controllers ensure a decrease in the workload of controllers, overload can still occur as the higher the number of switches assigned to a controller the more the load that is processed by the controller. This is a load balance challenge that *Hakan et al.* [78] propose Cooperative Load Balancing Scheme (COLBAS) which chooses a super controller to manage the flow steps of controllers and provides allocation rules for switches to bring about load balance. The solution was evaluated using SimPy, which is a simulation framework with built-in Python programming. The results are then produced by comparing other algorithms including Random Reassignment Algorithm and Nearest Neighbour algorithm, therefore, illustrating that COLBAS performs more efficiently compared to the mentioned algorithms. The authors only considered two controllers, hence, this does not put a large-scale scenario that can be more realistic.

It is important to ensure an efficient load balancing technique so that none of the controllers is overwhelmed by the amount of load and the connectivity within the network. However, load balancing is to correspond to the traffic delay within the network. *Lin et al.* [79] propose an algorithm called Polynomial-Time Approximation Algorithm (PTAA) to bring about the effective delivery time in traffic control. Performance Evaluation was done on PTAA over an Internet OS3E network topology and results illustrate that the algorithm shows a decrease in delay by 80%, therefore, communication efficiency.

There is variability and sudden burst when there is a dynamic traffic flow. An increase in end-to-end flow setup time results from none adapted to change in traffic flow by the controller. Traffic conditions occur at a small-time scale hence reactive controllers are important to bring about optimization in the network performance. *He et al.* [80] propose a Mixed Integer programming from the linearization method through the transformation of dynamic traffic flow which is based on a combined controller placement model to achieve a reduction in the average flow setup time. Evaluation is performed on a python-based framework and optimization is reached using Gurobi 6.5. Results reveal a 50% reduction of the controller to switch average

flow setup time by adaptation to different flows. Future contributions include consideration of the controller migration cost with load dependency.

Much work has been done on multi-controller placement more particularly in a large-scale network or a Wide Area Network making the placement of the controller specifically its location on the topology and the number of controllers when propagation delay and load balance is of consideration. *Yuan et al.* [76] propose an approach that uses a bipartite graph by finding its minimum weight match, the approach correlates to that of a WAN topology. In this case, the Kuhn-Munkres algorithm is used as an approach to find the controller placement and also considers the controller assignment by mapping the optimal assignment of controllers and switches using propagation delay as a measurement to be kept at a minimum. Furthermore, a genetic algorithm is proposed to achieve the controller placement problem based on the assignment scheme of the Kuhn-Munkres algorithm. The approach was evaluated using three topologies which are found in Survivable Fixed Telecommunication Network Design Library (SNDlib) namely; TA2, GERMANY50 and INDIA35. The results show that the approach highlights a good network performance by achieving good load balancing within controllers and minimization in the average propagation delay.

IV. Reliability

This subsection presents the studies that considered reliability or fault tolerance as a performance metric in their CPP design. *Ros and Ruiz* [81] proposed a reliable controller placement approach that focuses mainly on fault tolerance. The main goal of this algorithm is to find an optimal number of controllers to be elected, the location of deployment in the network with a switch-controller assignment with control information to achieve reliability between controllers and nodes on the southbound interface. A heuristic algorithm is proposed to solve the problem of formulated fault-tolerant controller placement. The results for the solution were acquired using mininet and evaluation was done using network topologies from the Topology Zoo library. Results suggest feasibility insight in achieving fault tolerance in SDN with careful Controller placement.

Compared to [81], *Killi and Rao* [82] proposed a technique that minimizes propagation latency considering worst and average cases while achieving resilience due to the pre-specified number of controller failures, network operational cost and backup capacity reserved at controllers. Implementation and solving of the various proposed models are done using Matlab and IBM

CPLEX optimizer. Evaluation results suggest that resilience is achieved with a reduction in a backup capacity and a 50% in reduction of reserved backup capacity.

Killi and Rao [83] present a MILP technique that achieves capacitated controller placement approach which in the worst-case scenario dynamically recovers from link failures with no increase in the switch-controller propagation latency. With the use of CPLEX optimizer and Matlab for implementation as well as Internet Topology Zoo and Internet OS3E for evaluation, the results assert that the proposed model achieves a reduction in worst-case propagation latency with failure, and further achieves a reduction in the average and maximum controller-controller latency.

SDN constitutes a centralized control point in a network which has several limitations including a single point of failure and processing capabilities as more traffic is only concentrated on a single controller; this is a challenge that *Zhang et al.* [84] address. To achieve a more reliable network by ensuring there is no single point of failure or processing limitations, *Zhang et al.* [84] describe a distributed architecture as a contributory factor towards eliminating the mentioned challenges and how the choice of the controller is a master to a switch or data owner can affect the overall performance. Furthermore, a Pareto-optimal controller placement to find a Pareto-frontier is proposed which is used to evaluate the problem of controller placement delay trade-offs. A model is then developed to obtain an estimation of the reaction time that is perceived by the switch. An optimization problem is formulized and a novel approximation has used the minimization of the reaction time. Performance is then evaluated using Gurobi utilizing 58 ISP topologies with results describing a controller that constitutes data owning achieving an improvement of reaction time. However, the master controller of the closest switch to the controller achieves no minimization of reaction time perceived by the switches. The proposed work opens opportunities for understanding the importance of inter-controller communications, though the methodology remains complex.

As mentioned above, a distributed SDN eliminates some of the limitations that are faced by the SDN, among a few, achieves better robustness by controllers backing up each and reduction of the load computation of controllers. The seismic phenomenon is among the few events that can occur due to a large-scale network failure. *Hirayama et al.* [20] propose a reactive approach called the C-plane reconstruction method which aims at achieving recoveries from failures. When there is a certain node disconnection from the main controller, other nodes can detect and apply or deploy a substitute controller to the disconnected node ensuring a reconnection of

other nodes. The C-plane reconstructions receive connectivity signals through the substitute and the main controllers; the re-connectivity detection will form a C-plane merger between the two to limit conflict. Performance evaluation was performed through simulation and the results show that a 90% connectivity probability is recovered when a regional failure is experienced in a network topology with 100 nodes. According to the results, the approach is very much practical and efficient as it uses probability and the convergence time of the respective C-plane reconstruction system is in proportion to the size of a network.

Network performance relative to CPP has been well studied with little consideration under the maximization of fault tolerance. *Alshamrani et al.* [85] propose a model to measure the fault-tolerance of a network system compared entirely to the performance of the network. Error critical and non-error critical are two states that are defined and modelled in a manner that dictates if the system performance follows performance requirements, hence, the network system effect will occur accordingly. The performance requirements are dependent on the overall latency, latency is bounded by certain values, for instance, if a non-error-critical system occurs it means there is a minimization in latency, hence, a decrease in late transmission and failures. The approach is evaluated through several topologies found in the Internet Topology Zoo. Results describe that the approach performs better when the number of available controllers increases, but, due to the limited computational power, the experiment was performed on only 4 available controllers.

The load experienced within a network is always a problem, by this, it makes other issues faced by the SDN become more difficult. Traffic control or load balancing is crucial when adopting a fault-tolerant solution for a system. *Ramya and Manoharan* [86] proposed a Pareto Integrated Tabu Search(PITS) algorithm which is a multi-objective algorithm to obtain the appropriate number of controllers needed in a certain network and the optimal location placement of the respective controllers. Furthermore, the authors propose a migration scheme that will handle events of failure in a network through the adaptability of dynamic traffic. The approach is described as efficient in improving the performance; however, the approach does not include any simulation or experimental results.

V. Energy

This subsection presents the studies that considered energy as a performance metric in their CPP design. *Hu et al.* [87] propose an energy-aware CPP technique that is modelled using binary integer programming (BIP) through energy consumption problem formulation.

Furthermore, a genetic heuristic algorithm is adopted to address the BIP complexity and used for effective solutions in sub-optimal. The evaluation results are generated using IBM ILOG CPLEX Optimizer and SND-lib for the acquisition of network topologies. The results found that the proposed algorithm acquires solutions that are close to optimal.

An energy-aware algorithm, GreCo, is proposed by *Ruiz-Rivera et al.* [88]. The authors consider energy consumption that is based on energy-saving constraints including the delay between switch-controller, utilization of links and load on controllers. A BIP and a heuristic algorithm are proposed in this instance. The model is simulated using Matlab and evaluated by considering four different topologies. Results explain that 55% of energy-saving during off-peak with 20% utilization of links by the proposed heuristic is compared to the optimal solution.

Considering the increase in mobile application popularity, the volume and diversity of network traffic ought to increase, calling for the management of traffic. SDN provides the necessary traits for traffic management by utilizing routing methods that will serve to satisfy energy consumption. An efficient routing algorithm is then proposed by *Özbek et al.* [89] to minimize the power consumption in proportion to cost minimization. This is achieved by obtaining the active number of switches with consideration of the network's load requirements based on the link capacity constraints or limitations within the network. The proposed algorithm is similar to the shortest-path algorithm. Performance evaluation illustrates that the proposed algorithm achieved 20% more performance when compared to the shortest path. Performance evaluation was performed through simulation considering two Waxman-based topologies and the proposed architecture opens a gap in considering various factors like interference level. The results show positive outcomes but do not consider large-scale topology scenarios.

Carbon dioxide and an increase in energy consumption are a result of the large data networks. In light of the importance of modelling an energy-aware routing model, *Fernandez et al.* [90] propose an Integer Linear Problem route to reduce energy consumption. A heuristic mathematical model is also proposed for the aim of complexities when large-scale networks are considered. The proposed methods were designed and developed using Gurobi Optimizer and Python utilizing SND-lib topologies. The results show that the proposed methods achieve preservation of energy, however, the methods affect a slight reduction in delay in controller communications.

It is estimated that by 2020 global data centres will utilize 8% of the electricity utilization, hence, the seriousness of energy-saving mechanisms in SDNs. *Wu et al.* [91] propose a Control-path Based-saving Routing Algorithm (CERA) with energy consumption by path (ECP) contribution. The proposed approach was simulated using ta2, zib54 and Germany50 topologies which are found in SND-lib. The performance evaluation was carried out and the results showed that the proposed method achieves immense energy efficiency within an SDN network. However, limitations of the proposed approach arise when considering controller placement problems as well as the performance metrics like propagation delay which may be affected.

A. Multi-Objective

The controller placement problem is described as NP-hard because performance metrics conflict with one another when trying to achieve optimization of all metrics. It is less effective or efficient to evaluate for all placements and this is not practical when large-scale networks are considered. *Ahmadi and Khorramizadeh* [92] proposed a Multi-Start Hybrid Non-dominated Sorting Genetic Algorithm (MHNSGA) to obtain the trade-off between the conflicting metrics, load balancing, and controller-controller latency and node-controller latency. MHNSGA is firstly used for the solution of the multi-objective controller placement problem. In this case, optimization is of utmost importance, hence, proposing the Pareto front as one multi-objective solution might not be present, and this method allows for a set of possible solutions. Performance evaluation was performed using 40 graphs or topologies from the internet topology zoo and compared with other algorithms. The results from the comparisons show that the proposed approach is efficient according to the examined numerical evaluation. MHNSGA works well when considering large-scale networks, however, portrays complex methods of solution computation and only considers two metrics, load balancing and latency.

Scalability is the primary challenge that any SDN faces, meaning multiple controller placements are to be considered, however, this brings about distorted performance in the network. *Ksentini et al*. [93] propose a state-of-the-art approach using a bargaining game aimed at obtaining the trade-offs between the different objectives. The objectives include latency and communication overhead of controller-controller, latency and communication overhead of switches-controller and load balancing between controllers. Simulations were performed using IBM CPLEX, Matlab and CVX 2.0. The bargaining game approach includes an optimal

solution with equitable trade-offs amongst respective objectives. The results show that the proposed mechanism compared to mono-objective solutions illustrates a much more prominent output or trade-off of the selected objectives. Mathematical models are precise; however, computational complexity is to be considered more particularly as we head towards building intelligent networks.

Multiple controller placement in large networks is a challenge as many engineers have come forward with work that aims at giving the best possible solutions. Although latency is considered to be the main performance measurement, *Ramasamy and Pawar* [94] include a second objective in their proposed work, load balancing. The proposed multi-objective ant lion optimization (MO-ALO) considers propagation latency and load balancing as factors to find or obtain optimal controller placement in a network topology. Matlab and POCO tools are used in this case to obtain performance evaluations. A topology from Internet 2 topology with 34 nodes and 41 edges was used and simulations illustrate that the method used is of exceptional advent in obtaining solutions for determining multiple controller placement problems.

As mentioned before, latency plays a critical role in the network performance or quality of service (QoS) particularly the assigning of switches to controllers. *Jalili et al.* [95] introduced an AHP-based algorithm to achieve optimality in assigning switches to respective controllers in consideration of three objectives including hop count, propagation latency and link utilization. Additionally, location-allocation of controllers is achieved by using another proposed Controller Placement Genetic Algorithm (CPGA). Matlab2016a was used for implementation purposes as well as experimental evaluations and a basic network topology from Internet2 topology. The results explain that the proposed method produced efficiency in obtaining the optimal controller locations and optimal switch-controller assignment. Furthermore, the method performed way better compared to latency-based assignment when link load balancing is considered.

Table 2.2 Summary of SDN CPP Implementations

	Ref.	Method	Implementation	Evaluation Metrics	Network type
Cost	[67]	LP	Simulations: CPLEX optimizer 12.5, PC that has 2 Intel Xeon X5675 processors running at 3.07 GHz with a total memory of 96 GB, Topologies: 20, 25 and 30 switches	Number of controllers, Update Cost of controllers (installation, lining and linking)	WAN
	[68]	SSOA with chaotic maps	Simulation: MATLAB, Intel Core i5 processor Internet Topology Zoo	time utilization, cost-utility	CN

	[69]	ILP	Simulation: GUROBI optimizer	minimize the cost of controller placements as well as the control latency	CN
Latency	[10]	FFA	Simulations: MATLAB R2014a, Intel Core i5 with 4-Core processors	SC latency, CC latency, multi-path connectivity between the switch and controller	WAN
	[74]	LP rounding Primal-Dual	Simulations: Python, Internet 2 OS3E topology and Internet Topology Zoo	Latency Placement and Balanced Cost-Latency Placemen	CN
	[73]	K-means algorithm and Dijkstra algorithm	Simulation: switches randomly located in a rectangular region; 2000km×2000km	control-plane delay minimization	CN
	[72]	Kmeans algorithm	Simulations: Internet 2 OS3E topology and Internet Topology Zoo.	Latency minimization	CN
	[70]	Density-based cluster	Simulations: Internet Topology Zoo.	Reliability, SC latency, CC latency, time consumption	CN
	[71]	cooperative k-means/ Cooperative game	Simulations: Python, Internet 2 OS3E topology and Internet Topology Zoo	Worst-case latency, the controller load	WAN
	[75]	Clustering-based Network Partition Algorithm (CNPA)	Simulation: Matlab, Internet Topology Zoo	end-to-end latency, the queuing latency of controllers	WAN
Load Balancing	[14]	Heuristic Algorithm	Simulations: MATLAB, 2018a and CPLEX 12.6, 4-GH Intel Core i7 machine Internet Topology Zoo (160 topologies)	SC latency, CC latency, and a new load balancing the scheme was applied as objective functions	CN
	[76]	Kuhn-Munkres algorithm Genetic algorithm	Simulations: Intel Core i7 (2.9 GHz) Topologies: TA2, GERMANY50 and INDIA35	Propagation delay and controllers load balancing	WAN
	[77]	MHNSGA-II	Simulations: MATLAB, Intel Core i7 CPU, 4 GH. Internet Topology Zoo	SC latency, CC latency, load balancing	WAN
	[78]	Cooperative Load Balancing Scheme For Hierarchical SDN Controllers (COLBAS)	Simulation: SimPy framework based: Python programming language.	controller-switch assignment and load-aware	CN
	[79]	polynomial-time approximation algorithm (PTAA) and alternating direction method of multipliers (ADMM)	Simulation: Internet2 OS3E with 27 nodes and 36 links and Sprint GIP backbone network	Minimized average control traffic delay	CN
	[80]	Mixed Integer Programming (MIP)	Simulation: SimPy framework based: Python programming language and Gurobi 6.5 Network topologies are Abilene (11 nodes) and AttMpls (24 nodes)	Controller-Switch for different flow dynamics and the different number of controllers.	CN
Reliability	[81]	FTCP based Heuristic algorithm.	Simulations: Linux, Intel Xeon CPU at 2.67 GHz, Mininet and 124 Internet Topology Zoo	Number of controllers, Reliability, Load balancing	WAN
	[82]	Resilient Multi-Controllers Mapping	Simulations: MATLAB IBM CPLEX Optimizer. Small-scale AT&T network, medium-scale GEANT network of Internet Topology Zoo	Propagation latency, resilience, cost, Backup capacity	CN
	[83]	Simulated Annealing	Simulations: MATLAB -CPLEX optimizer 12.6.2, 2 Intel Xeon E5-2630 processors (2.4 GHz). Topologies: Internet Topology Zoo and Internet 2 OS3E.	SC latency with and without link failure	WAN

	[84]	ILP	Simulations: real ISP topologies	SC delays, CC delays Reaction time	WAN
	[20]	C-plane reconstruction procedure	Simulations: Ns-3.22 and Gabriel Graph (N = 50, 100, 200)	Emergency against catastrophic disasters or network failures	WAN
	[85]	Pareto-based tabu search and a migration algorithm	Simulations: Pareto Optimal Controller Placement (POCO)	Controller overhead from heavy traffic.	CN
	[86]	fault-tolerance controller placement model	Simulations: Internet Topology Zoo (Latent, Telcove, Uninett2010, Uninett2011)	fault-tolerance: available number of controllers	WAN
Energy	[87]	binary integer program (BIP)	Simulation: Matlab and IBM ILOG CPLEX Optimizer 2 Intel Xeon E5-2430 processors with 8 cores, and 16 GB of RAM. Four topologies from SND-lib: Abilene (12 nodes, 15 links), Janos-us (26 nodes, 42 links), Pioro (40 nodes, 89 links),	energy consumption, minimization of delay of control paths and a load of controllers	CN
	[88]	GreCo (green centralized controller)	Simulation: MATLAB Topology: Abilene (11 nodes, 28 links), AT&T (25 nodes, 112 links), GEANT (40 nodes, 122 links), and SURFnet (50 nodes, 138 links)	Reducing SDN energy consumption	CN
	[89]	Efficient routing protocol	Simulation: 2.5 GHz Intel Core i7 and 16GB RAM. Topology- SNDlib	minimize the cost: power consumption	CN
	[90]	ILP model and heuristic algorithm	Simulation: Python - 3.30 GHz Intel Core i7 and 16 GB RAM, real network topologies - SNDlib	minimizes the number of links following traffic demand	WAN
	[91]	Control-path-based Energy-saving Routing Algorithm (CERA)	Simulation: 2.5 GHz Intel Core i7 and 16GB RAM. Topology- SNDlib	Energy Consumption by Path	CN
Multi-Objective	[92]	MHNSGA	Simulations: MATLAB, 2014a, 4-GH Intel Core i7 Internet topology Zoo	SC latency, CC latency load balancing	WAN
	[95]	Genetic Algorithm hybridized by AHP technique	Simulations: MATLAB 2016a 4-GH Intel Core i7 machine, Internet topology Zoo	propagation delay, hop count and link utilization	CN
	[93]	Bargaining Game	Simulation: IBM CPLEX, Matlab and CVX 2.0	The latency and communication overhead between switches and controllers; the latency and communication overhead between controllers; the guarantee of load balancing between controllers	CN
	[94]	multi-controller placement problem (MPP) and multi-objective ant lion optimization (MO-ALO)	Simulation: POCO tool and Matlab, topologies - internet 2 topology	load capacity factor and the propagation latency.	CN

2.7.2 SDWSN CPP Implementations

This section includes all works that have been proposed to solve CPP, however, as SDWSN is a new discipline, CPP has not yet received much work and many performance metrics have been used in this instance. The approaches included in this section make use of a centralized controller in solving various metrics illustrated below but do not consider multiple uses of

controllers due to the scalability problem. Frameworks have been proposed to deal with such challenges and have been explained appropriately.

I. Frameworks

A wireless controller placement problem was introduced by *Abdel-Rahman et al.* [96]. A joint controller placement and assignment scheme was developed with deterministically known link delays. The scheme was further extended through the formulation of stochastic link delays. The model was evaluated and tested using CPLEX. Results indicated that the proposed joint scheme shows a reduction in the required number of controllers, therefore, asserted satisfying ability given response time constraints. WSN suffers greatly from a lot of factors including, resource underutilization, rigidity to policy change and lack of efficiency in the management of resources. *Lou et al.* [61] provided or proposed an SDWSN architecture with a Sensor OpenFlow as a southbound protocol that conforms to the architecture at hand. This approach serves the mentioned technical challenges. This implementation was developed to attract the research community for the opportunity for various innovative techniques based on this implementation.

De Gante et al. [97] proposed a general framework that makes use of a generic architecture with a base station in an SDWN that will serve as a controller or where the controller ought to be implemented. This approach was implemented to achieve smart management by using SDN as WSN suffers from inefficiency in resource management. TinySDN is a framework proposed by Oliveira et al. [98] for configuration and management complexity in WSN through the deployment of multiple controllers by leveraging SDN attributes. This framework was tested using the cooja simulator. However, results show that hardware is independent of a single OS with further delay as well as memory drawbacks.

Kobo et al. [99] utilize SDN-WISE which is an SDWSN framework with abstraction between sensors' nodes and controllers. The use of ONOS controller with virtualization instantiation as multiple controllers are in charge of clusters, meaning a logically distributed network. The results are generated using the cooja simulator. However, results assert that a centralized controller with a logically distributed architecture, even with the best security policy, is susceptible to scalability, reliability and performance drawbacks as the device is unable to handle such large data.

II. Reliability

Olivier et al. [100] introduce an approach that uses clustering, intending to place the controller in a cluster that includes a base station, similar to *Gante et al.* [101]. The author(s) assert that much work has been proposed on preserving connectivity between nodes. However, their implementation also supports security through cluster heads of each domain or cluster can implement its unique security policies, therefore, ensuring that each domain is protected or secured from external threats or attacks.

III. Energy

A base-station-based SDWSN is proposed by *Gante et al*. [101] which includes five layers, physical, medium, access, NOS, Middleware and application. The controller resides on the middleware layer with mapping functions and information as well as the various flow tables' definitions. For this implementation to efficiently serve energy, the controller uses monitoring messages to get information about the network like sensor node energy levels. Such information is processed by the mapping function to produce the topology view of the network. Furthermore, the mapping information is stored in a mapping information module to use the application layer to locate and produce accurate sensed data for the maintenance of the sensor node position information [6, 101]. A machine learning technique that achieves energy efficiency is proposed by *Huang et al.* [102].

Table 2.3 Summary of SDWSN CPP Implementations

Ref.	Method	Implementation	Evaluation Metrics	Network type
[61]	Software-Defined WSN architecture	Simulation: Contiki OS- MicaZ, TelosB, and Epic platforms Communication: Sensor OpenFlow	Traffic Generation, In-Network Processing, Backward and Peer Compatibility, Control Plane: SOF Channel	WSN
[99]	distributed control system for Software-Defined Wireless Sensor Networks	Simulation: Cooja simulator and VM-2GHz CPU and 2G RAM/ 2GHz CPU and 1G RAM Communication: SDN-WISE and ONOS controller	Round-Trip Time (RTT)	WSN
[96]	chance-constrained stochastic programming (CCSP) - MILP	Simulation: intel core i5 3.3 GHz core duo with 16 GB RAM. We used CPLEX to solve our optimization	QoS- Average Response Time, Per-link Response Time	WSN
[97]	Generic architecture-SDN based sink	Theoretical Methodology	Energy Saving, Sensor Node Mobility, Network Management, Localization Accuracy and Topology Discovery	WSN
[103]	MIQP and MILP	Simulation: i7 quad-core 3.4 GHz CPU, 8 GB memory and Python 2.7.5	energy efficiency, control overhead	SDSN

[102]	SDWSN Reinforcement Learning Prototype	Simulation: NS-2	energy efficiency	WSN

Energy efficiency and WSN adaptability from environmental change and monitoring are reached through value redundancy filtering and load balancing by employing a reinforced learning algorithm on a cognitive SDWSN framework. The results were gathered using the NS-2 tool with a conclusion that the proposed prototype effectively achieves an improvement of self-adaptability of environment monitoring and energy efficiency in WSN with further improvement in QoS.

Zeng et al. [103] proposed an online algorithm that includes local optimization. This algorithm was developed for the sole purpose of energy consumption of a multitasking SDWSN which was solved using MILP. The approach was tested and the results showed that the proposed online algorithm achieves an optimized network energy efficiency with a decline in rescheduling time and as well controls overhead.

This part presented a current literature review; it is a method that achieves RO1 and responds to our RQ1. The importance of this literature review is to gain insight and understanding of CPP practices and the challenges that impact its efficiency. We have gained much more than that, consequently, we can now propose a method to address our problem following the existing knowledge we have acquired in this section. The related works justify the existence of the problem understudy considering all the proposed methods that the author(s) asserted towards a similar problem. As a result of this section, this research can now support the application of a construct as well as the evaluation process on existing or related works.

2.8 Chapter Summary

The background of this chapter covered SDN, WSN and SDWSN with emphasis on the architecture for each, challenge and application. Furthermore, the discussion was based on Controller placement problems and performance metrics including latency, load balancing, reliability and energy. Lastly, related works are comprehensively considered.

Chapter 3

Research Methodology and Design

3.1 Chapter Outline

The research methodology and design employed in this dissertation are presented in this chapter. It begins with a brief introduction, followed by a discussion on philosophical assumptions and their relation to effectively producing an accuracy of the final research outcome research philosophy, and the methodology is described with significance to the collection of data. Furthermore, data collection and analysis are detailed in terms of the instruments and settings used to get the desired findings in Chapters 4, 5 and 6. Lastly, the ethical consideration and the chapter summary are presented.

3.2 Introduction

As discussed in Chapter 1 of this research, the dissertation employed the constructive methodology, which is defined by various phases, which are strictly followed throughout this research. The significance of this chapter is to outline the methodology and research design that were followed in this dissertation.

The methodology chosen follows a set of philosophical assumptions that explain the approach, research, research design and most importantly the chosen strategy. The outcomes of quantitative evidence signify and reflects the effectiveness of the research design as well as the methodology used for data collection. Any research aims to find a solution or better the contribution of new knowledge or an existing problem. Following the stated ROs, which involved designing an efficient CPP solution that is energy-aware and further evaluating it with existing solutions. Thus, a constructive approach was adopted to achieve the objectives outlined in the research.

The chapter is provisioned into sections including section 3.3, the research process, which discusses the traditional structural format of research for a more understanding of the significance and outcomes of each chapter. Sections 3.4 and 3.5 highlight the methodology and research philosophy which describes the design of the research and the strategy that it follows. In sections 3.7 and 3.8, the process of quantitative data collection and analysis are described,

motivating the significance of the research design. Ethical considerations and chapter summaries are presented in sections 3.9 and 3.10 to conclude the chapter.

As asserted in [104], a research process is essential to guide or give meaning and understanding to the structural format of the research dissertation. Additionally, we can assert that research seeks to answer questions that have not been answered before through a search for knowledge [105, 106]. Thus, a research framework in Figure 3.1 is adopted with the sole purpose of providing a conceptual plan of the research process to achieve a well-constructed research report. The framework shows the flow of the various steps and/or milestones that have been undertaken to accomplish our research. Initially, we formulated a practical problem from studied literature reviews to propose the research. A proposal defence and research ethical clearance followed in this regard.

Furthermore, constructive research is developed with mixed methods through the use of two main research designs; model design and simulation. However, validity is to be tested in this regard thus, verification of construct, the relevance of scope, research contribution and theoretical connections. They are assessed based on the fact that they are a collection of both qualitative and quantitative data. Finally, this is when a conclusion can be drawn together with possible future work that has been identified.

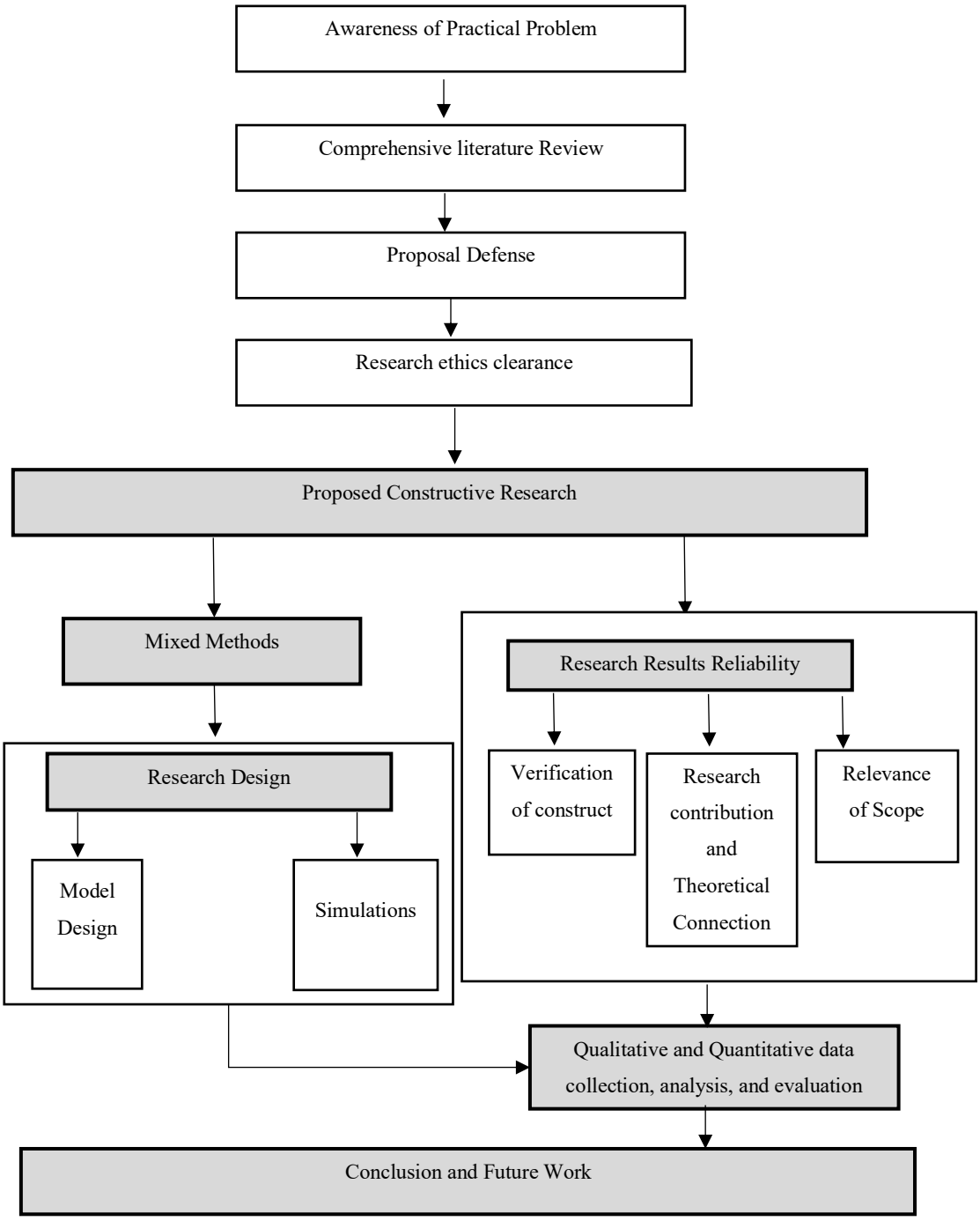

Figure 3.1 Research framework

3.3 Research Paradigm

This section provides a research paradigm that outlines the various philosophical views of respective groups of people concerning their conduct in research. A paradigm is a beneficiary about guiding research as well as the development and implementation of a method or system. Ontology, epistemology, methodology and axiology are the four main philosophical

groundings in ICT. Various paradigms follow these philosophical groundings or assumptions as can be found in [107]. However, our focus is on the most relevant to this research being the design or well known to be a pragmatic paradigm; which is, briefly discussed below concerning the respective philosophical assumptions.

As seen in our research framework indicated in Figure 3.1, the first component is awareness of a practical problem. As asserted in the paradigm adopted in this research, the pragmatism paradigm supports this element is the most imperative, unlike the epistemological, ontological and axiological components that can be possibly followed in research [108, 109]. The paradigm highly depends on the relationship between data or observations gathered and the theories proposed. Thus, making mixed-methods are used, and qualitative and quantitative are possible to an extent that there is a relation between data and theory, therefore, there is an iterative process between the components [109-111]. As a result, this paradigm makes use or enables the use of different methodologies in the same study as they satisfied the problem of our research. In addition, we verify our results to conclude research contributions and theoretical connections, making pragmatism the best paradigm for our research as it allows us to study and justify our verification through the search for effective ideas by studying our quantitative and qualitative data relationship [109, 112, 113]. Finally, we adopted comparison studies in Chapters 4 and 5 to evaluate the different methods suitable for our construct. This serves as the main focus of pragmatism asserted in [112] as it emphasises the importance to evaluate various methods to study their effectiveness [109, 112, 114].

We explain pragmatism using five different assumptions ontology, epistemology, axiology and typical methods. These are explained below.

a) **Ontology** is described as the study of the nature of reality or science, thus, it distinguishes how scientists report their findings [115]. The way we validate the accuracy of our findings is important and relies on how we describe them. Therefore, Chapter 3 serves to illustrate the steps taken to plan the dissertation and what methodology and methods have been considered to assert or explain the results of our study. Since we use both quantitative and qualitative, referring to the two respectively, we present how we can improve and contribute to other aspiring researchers in the field as well as the developed construct relevance in terms of the scope in the industry or real world.

b) **Epistemology** explains the nature of the relationship between a researcher and an object under study. Hence, reporting of the researcher is based on truth from the knowledge acquired or gained [115]. Designing the construct based on the gathered data or knowledge supports the assumption as we validate the knowledge acquired by the researcher about the construct of our research. RQ 2 best illustrates this process with a comparison study developed and seen in Chapters 4 and 5.

c) **Axiology** explains the beliefs or values of a researcher regarding the research environment [115]. Since we have to understand our depth research, our comprehensive literature supports this very well and answers RQ 1 with a clear problem formulated in Chapter 1.

d) **The methodology** is the development or the construction of the construct with its use [115]. This assumption is in alignment with our RQ 3, where we have illustrated an analysis for the developed construct. We further describe the impact of our findings in Chapter 7 as contributions and draw conclusions based on the qualitative and quantitative findings.

The highlighted in Figure 3.3 represent the philosophical assumptions of constructive methodology which is adopted in this research. As stated above, it is specifically used in designing and implementing constructs or artefacts or innovations with the sole purpose to address problems [107, 115]. Throughout the process of reporting the research following a constructive paradigm, there are various challenges faced including epistemology assumptions. The next section gives a thorough description of the constructive methodology.

3.4 Research Methodology

The methodology used in this research is discussed in this section. The process of data collection following a systematic manner is regarded as the research methodology. The data collection that we asserted is quantitative data.

3.4.1 Design Science Research

Following our previous section where we explained the research paradigm, asserting our philosophical views as researchers, this also affects the type of methodologies used. Therefore, constructive research is aimed at contributing to new knowledge about existing ones. Respectively, a pragmatic paradigm supports this even with its various assumptions which bring about how our research is conducted. Thus, the design science research (DSR) methodology, as compared to other methodologies aims at focusing on the contribution of new

knowledge hence, it is, for this reason, we have adopted the DSR. It supports existing knowledge as qualitative evidence and quantitative evidence as our verification process to support our construct and further present the existing knowledge.

A DSR methodology is imperative for the constructed improvement in accord with the approach being used in the study. Design Science Research explores or addresses the socio-technical effects organizations and users incur about particular operations or organizational management. Furthermore, it seeks to improve the construction of an artefact through the process of learning and/or invention [107]. However, this does not explain a demonstration of technical skills or knowledge but rather a process of thorough analysis, explanation, justification and evaluation of the artefact being studied [107, 115].

The process steps involved in this workflow to achieve the goal of this research are as follows:

1) ***Awareness of Practical Problem:*** A formulation of a problem may be sourced in different ways including a reference from new developments identified in the industry [107]. This is the first step involving comprehensive insights concerning energy consumption issues in SDWSN as well as CPP challenges facing SDN as adopted by SDWSN. In Chapter 1, we have included or formulated a problem statement from sources of readings that have been referenced. Awareness of the problem, being the initial step, puts effort into a proposal output.
2) ***Comprehensive Literature Review:*** Following the output of the awareness of the problem, the incorporation of new functionalities added to the proposal being formulated is a creative step called suggestion [107]. Literature review in Chapter 2 and a published paper [107, 116], assist in finding suggestions or new functionalities concerning existing methods that can be incorporated into the strengthening of our proposal. Additionally, the experiments conducted in Chapters 4 and 5 illustrate a further addition to the justification of the choice of methods used in the next step through investigation or a comparative study.
3) ***Construct Development:*** This phase uses the tentative design as input for implementation purposes to achieve an artefact output. However, there are various outputs in artefacts including constructs, concepts, frameworks or models, hence, implementation will vary. In this regard, our proposed energy-aware CPP algorithm is implemented with the assistance of the comparative study performed in Chapters 4 and 5, and by adopting a comprehensive literature review to produce an efficient energy-aware CPP algorithm. An energy-aware TEEN protocol, clustering PSO algorithm and a custom gateway for an SDN-enabled sink

to perform multi-local controller placement were chosen to model or design the proposed construct.

4) ***Construct Verification Evaluation:*** The behaviour of the artefact is of importance, which adopted criteria included in the proposal or the identified problem; a constructed prototype is designed and further implemented. The use of existing or traditional nature-based algorithms and cluster-based energy-efficient routing protocol has been greatly utilized. However, this research explores these approaches in a new research area by further adopting a custom gateway; an SDN-enabled sink node. The expectation of quantitative and/or qualitative evidence is crucial with further tentatively explained [107]. The outcomes of the evaluation process were performed through simulation to see the performance or effectiveness of the construct by validating its reliability by the collected simulation data. Furthermore, the completeness of the research is to be published in peer-reviewed conferences and/or journals for contribution to research knowledge.

5) **Theoretical Connections and Research Contribution:** This stage or phase of research assisted in identifying the research goal including energy efficiency, reliability and decline in delay or propagation latency. Primarily, the construct was measured or evaluated in comparison to existing literature.

6) **Scope Relation or Relevance:** The gathered results or quantitative evidence from the proposed efficient energy-aware CPP solution are based on simulation scenarios. However, the solution was recommended to a more generic and real-world outline to illustrate an effective and efficient outlook on network performance and management.

3.5 Research Design and Methods

This section explains the study design as well as the technique used to meet the research goals and objectives.

3.5.1 Research Design

According to Durrheim [117], a research design is described as a bridge formulated from an action-oriented strategic framework between achieving implementation or execution of ROs and answering RQs. Additionally, research design serves as a guideline for efficient arrangement in collecting and analysing the data [117]. This is of relevance to the quality of the dissertation reporting.

To satisfy the research goal together with the defined research objectives, a research design is proposed as illustrated in Figure 3.3. This framework illustrates the various phases following how the research was conducted from the initial phase of exploring literature to the final phase where we evaluated the validity of the evidence or results gathered. Additionally, specific stages or steps are outlined to deduce a method from the practical problem and achieve research objectives in support of the respective research questions. The main reason for a research design is how we have undergone the process of quantitative and qualitative evidence as well as validating these outcomes.

Chapters 1 and 2 explain awareness of the problem, background study and existing literature, thus, making up a comprehensive literature review as shown in Figure 3.3 as the first step in the framework. Following all the knowledge gathered from the qualitative data, a practical problem and a theoretical solution were proposed with further thorough analysis following a better understanding.

By achieving the research goal, an efficient energy-aware controller placement mechanism design was proposed to address energy, connectivity, and latency inefficiency in SDWSN. Data collection and analysis were for qualitative and quantitative data explained in sections 3.6 and 3.7. Furthermore, Chapters 4 and 5 provide qualitative evidence for comparison studies performed on routing protocols, and clustering algorithms are gathered and analyzed. Chapter 6 shows the proposed efficient energy-aware CPP mechanism which is designed and implemented. A simulation method is adopted to collect quantitative evidence, and different evaluation parameters or cases are used to measure the data accuracy and consistency. The link

between the collection, processing, and evaluation of qualitative and quantitative data is summarized in Table 3.1.

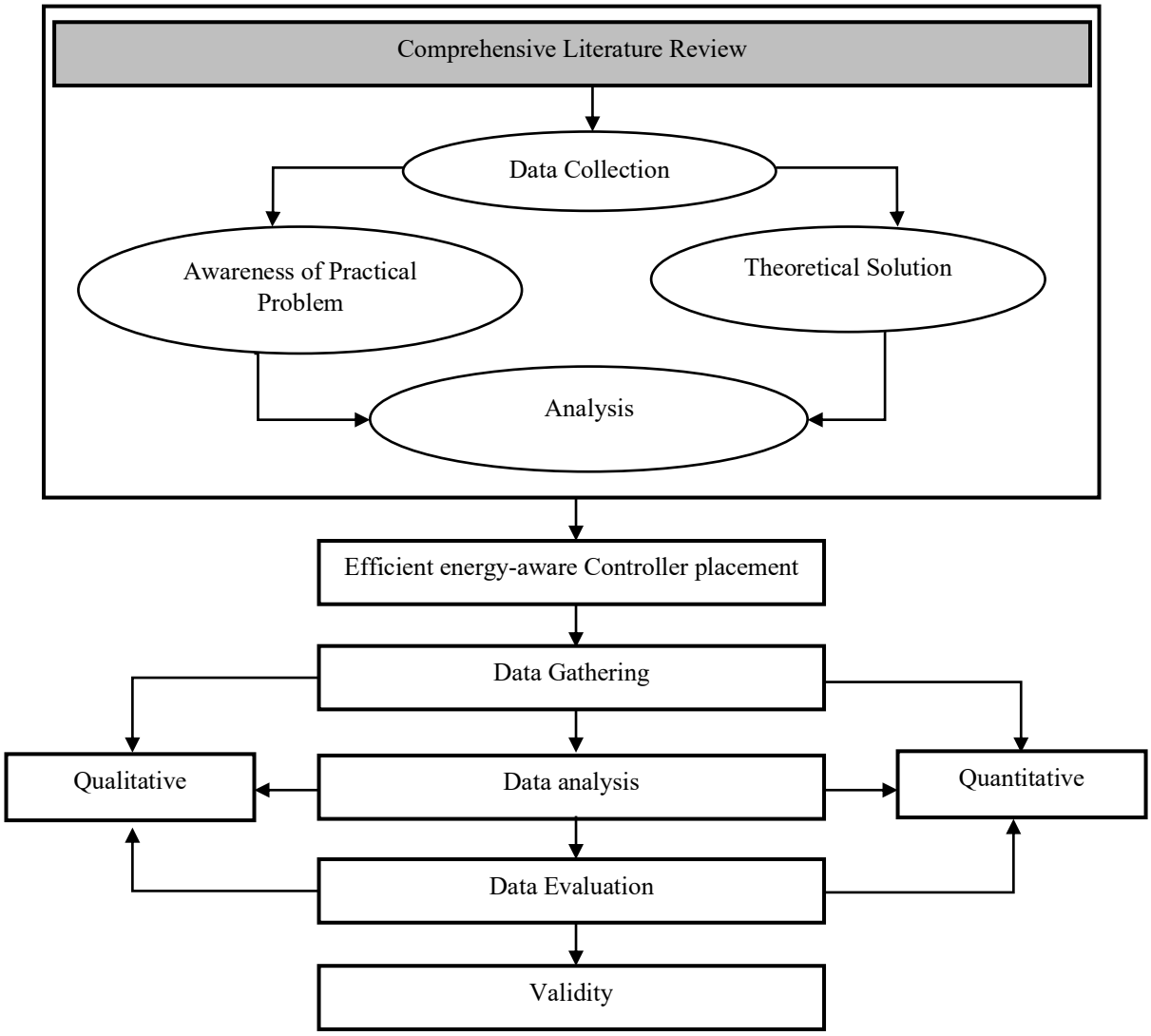

Figure 3.3 Research design and approach

Table 3.1 Relationship of Data Gathering, analysis and evaluation

Scheme	Qualitative	Quantitative
Data Gathering	• Existing Literature evidence • The premature proposition of drawback assumptions	• Simulations
Data Analysis	• Research connections and contributions	• Extension of existing mechanism for improvement • Observation of data changes
Data Evaluation	• Addressing how, why and what of quantitative data	• Data generalization • Summarization of Data • Precise and consistent data

In this research, design research is adopted as a DSR strategy as this research aims at designing an energy-aware CPP algorithm, thus, constructing an artefact instead of a socio-technical intervention. The design research strategy illustrated in Figure 3.5 uses a rigour cycle, design cycle and relevance cycle for justification in the process of artefact construction. The use of existing knowledge including design products, methods, artefacts experiments, theories and expertise is adopted by the rigour cycle. The relevance cycle explains the need for the research as well as the research problem. Finally, research activities and actions describe the evaluation or testing processes in the design cycle. In Figure 3.5, the design research cycle and research relevance and rigour are illustrated. The design research adopts all the steps followed in DSR including awareness of the problem, suggestion, development, evaluation and conclusion [107, 115].

3.5.2 Research Methods

In our introductory chapter, we briefly asserted the various methods that we will adopt for this research to best cater to or answer our research questions consequently achieving our research objectives. The significance of research methods and research design collaboratively is to enable some development and adjustment if necessary. Moreover, these are the methods supporting the research design on how the evidence is collected and analysed. The research design and technique employed in this study are depicted in Figure 3.3. The supporting methods include…

1) *Comprehensive literature review:* Problem Identification and Selection for the energy-aware controller placement problem statement was formulated based on the literature review that was conducted. Feasible research contributions required in SDWSN were as stated in [6, 18, 26].

   Additionally, by achieving a pre-understanding of the efficient algorithm, a thorough literature study in SDWSN energy-aware CPP was done to understand existing systems therefore efficiently constructing the problem statement and the performance comparison of multi-objective algorithms.

2) *Model Design:* As seen in Chapter 2, some of the existing knowledge includes frameworks, formal and mathematical proofs and/or models that were acquired to design the proposed solution accordingly. The energy-aware CPP algorithm was designed using three different algorithms, including k-means, PSO, and TEEN, based on the information gleaned through the review of the literature. This brought about the justification and significance of Chapters 4 and 5 as a comparison study to choose the best algorithms amongst the family or category of the particular algorithm i.e., clustering-based metaheuristic algorithms and WSN energy-efficient protocols.

   Additionally, the design of the proposed solution was implemented by adopting software engineering design models to illustrate the sequence of the algorithms and flow of data in the network including sequence diagrams. This was essential as it showed how the SDN-enable sink node act as a local controller, hence, enabling the distribution of the energy-efficient protocol through clustering.

3) *Simulation:* Simulation mimics real-world scenarios, hence, cost reduction was achieved for conducting this research. Following the model design, the data was

collected using an appropriate simulation program to analyse and assess the construct's performance effectiveness and dependability. After simulation and data gathering then performance evaluation between the proposed algorithm and existing multi-objective contributed towards optimal CPP and energy efficiency in SDWSN utilizing QoS/performance.

3.6 Data Collection

This section provides a brief explanation of how data evidence was acquired in the research reporting process. This research adopted quantitative evidence through simulation. The process of gathering these data required certain requirements which are discussed in this section, including parameters or participants, data gathering instruments, the procedure for acquiring the data and sources of the data.

3.6.1 Participants

This section describes the study participants i.e. WSN sensor nodes. We focus mainly on the energy consumption of sensor nodes and latency between nodes. As asserted in the previous section, we used the design research strategy as a DSR methodology strategy mainly to utilize existing knowledge, in this case, we carried out a comparison study in Chapters 4 and 5 to eliminate bias following the choice of algorithm or protocol best for our research algorithm. A sample of nodes was adopted and evaluated in a simulation tool for evaluation purposes as we will explore this instrument in the next section.

This study specifically brought focuses on preserving the energy of sensor nodes and further seeks efficient placement of controllers by minimizing end-to-end delay and avoiding data loss. The energy consumption of the sensor nodes is described as the network lifetime which is represented by the number of dead nodes (dependent variable) after a certain period i.e. iterations (independent variable) as shown in the evaluation methodology, section 5.5. of Chapter 5.

The distance between nodes is given as a mathematical equation called the Euclidean distance as asserted in Chapter 2. This equation was adopted as latency or end-to-end delay between sensor nodes or nodes in a network when minimized. Moreover, there can be little to plenty of solutions following the shortest Euclidean distance i.e. decrease of end-to-end delay. This set of solutions is described as the best cost (dependent

variable) acquired over a certain period i.e. iteration (independent variable) as illustrated in evaluation methodology, section 4.4 of Chapter 4.

3.6.2 Instruments

The data collection instrument used in this study is a simulation which was used on all participants. The simulation parameters are presented in Table 5.1 of Chapter 5 and Table 4.4 of Chapter 4 followed by their simulation setups respectively.

The simulation carried out in Chapters 4 and 5 was for evaluation purposes to measure the best performing algorithm and the routing protocol to serve as the best candidate for our energy-aware CPP algorithm. Matlab 2020b was used in this instance in both chapters as a simulation instrument. The tool used a set of nodes with a single performance objective i.e. energy consumption and propagation latency to measure the efficiency of all proposed algorithms and protocols. As asserted in Chapters 4 and 5, we proposed energy-efficient routing protocols and clustering-based meta-heuristic algorithms with justifications for each as described in the chapters in support of our ROs.

3.6.3 Procedure

With our RQ2.1 and RQ2.2, this study sought to answer these questions through the set of steps that were taken to achieve the evidence of this study. We used a large number of nodes as it was necessary for validity purposes when concluding the studied performance measure, moreover, to check for patterns.

A large number of nodes were important as we adopted a quantitative method for data collection, this explained the correlation of our independent variable(s) concerning our dependent variable(s). The procedure in Chapters 4 and 5 was for evaluation purposes and is described accordingly in their respective chapters. Moreover, the chapters serve important through the motivation of choosing the best approach for our algorithm. However, Chapter 6 differs slightly from Chapters 4 and 5 as its main purpose was to seek patterns to draw decisions and conclusions on our algorithm. This is explained thoroughly in Chapter 6 with the motivation of using a smaller to a much larger number of nodes.

3.6.4 Data sources

For the simulation of the research algorithm, primary data sources were gathered and evaluated through a comparison study. Since we used a simulated environment, all data

sources are embedded within all the utilized instruments as built-in tools. Concerning the theoretical evidence through literature review, academic-related information is widely available on scholarly sites as existing knowledge in our research area.

3.6 Data Analysis

This section presents an explanation of how the collected raw data was formatted into more meaningful information in line with the expected outcome of the research goal.

3.6.1 Simulation and Experimentation

Under the illustrated Figure 4.5, IS research framework justifies its developed artefact through experimental, analytical etc. However, simulation was the chosen method to achieve the evaluation of the research objective. Data were collected using a simulation method with no meaning in terms of meaning or format. This set of quantitative raw data was represented in form of tables and/or charts with the significant motivation behind its numerical format concerning the respective objective.

The numerical or quantitative data can be analysed or presented using various graphical representation methods like charts, tables and/or 2D/ 3D graphs. Following the representation of the collected data as a factor of analysis, the researcher's interpretation is the most significant factor in analysing the represented data.

In this research, we have adopted tables and graphs as the data representation method. The raw quantitative data is presented in form of tables as illustrated in Chapters 4 and 5, Table 4.5 and Tables 5.1-5.4; and graphs. This only represented the data collected from the comparison study of methods.

3.6.2 Literature Analysis

In this chapter we have adopted a DSR approach, the first phase explained the problem which made use of proposed conference papers, journals, articles and books. This can be seen in Chapters 1 and chapter 2 as problem statements and literature reviews. Furthermore, a review paper in support of this paper is proposed in [118]. This is imperative as it describes the existing knowledge.

The researcher's analysis is supported by the outcomes of the collected data and the documented or reported literature of existing knowledge. Mapping of existing knowledge to the quantitative evidence served as motivation for the conclusions.

Figure 4.5 explains thorough the rigour process or cycle how knowledge base data analysis and/or other methodologies can assist in motivating the construct being developed. The evidence of this construct is justified using the existing knowledge base, bringing about reassessing or refinement to achieve improvement.

3.6.3 Validity and Verification

After analysis of the data collected, the data or findings were verified and/or checked for validity [119]. True or trusted findings are regarded as the measure of validity or verification. Validity and verification checks of quantitative data are achieved using objective numerical and statistical measurement. However, validity is prone to threats, below we describe 5 factors including [119].

a) **History** of data collection poses a negative threat to the validity of research outcomes as incidents occurrence might distort or affect the format of the data [119].
b) The **maturity** of organization changes highly influence the research over time [119].
c) **Instrumentation** change over time and thus, produce different results [119].
d) **Experimental mortality** is caused by the loss of subjects or parameters that were used to evaluate the study [119].
e) **Selection-maturation interaction** occurs in comparison groups and interaction of maturation results in interpretation errors [119].

As stated in section 3.8.2 of the analysis of data, researcher data interpretation is one of the factors included in analysing the data and interpretation of data is a method that is considered in this research. Moreover, logical surges or distorted assumptions are a consequence of data interpretation. However, the validity of the usefulness of a construct can be presented in various ways [115]. *Informed arguments* and *scenarios* are validity-oriented evaluation methods that form the basis of the DSR descriptive evaluation method to ensure verification. These methods were adopted in our research [115, 119].

a) Using scenarios around the proposed construct establishes an evaluation environment to ensure the validity of the outcomes [115].
b) Informed arguments form the basis of information of existing knowledge base to structure the validity of the proposed artefact on the argument. Comparing the proposed argument of the artefact with the existing

knowledge base arguments strengthen the validity of the outcomes of the research [115].

3.7 Ethical Consideration

Ethical consideration in conducting research is described as being lawful or acceptable. Humans, animals and the environment are two applications of research ethics. Integrity and confidence or honesty is another application that is adopted in this research as our research does not include applications on humans, animals or the environment.

Ethics can be defined as disciplines that focus on the study of standards of conduct [120]. Significant harm to humans and animal subjects, and the public at large are prone to ethical lapses. As a result, there are research ethics codes and policies in place, such as; honesty, objectivity, integrity, carefulness, openness, respect for intellectual property, confidentiality, responsible publication, responsible monitoring, respect for colleagues, social responsibility, non-discrimination, competence, legality and animal care [120].

The principles that are mentioned above are mainly considered and adopted by researchers. Thus, our quantitative evidence collection in this research considered the following principles:

Respect for intellectual property: Credit for authorship is essential as it sets a record for academic relationships and is the most important to be adopted [120]. This research has a published paper in its name that supports its literature, the authorship of the paper had included all relevant people who took part in its publication.

Objectivity: Avoiding biased outcomes defines objectivity [120]. Reporting of Chapters 4 and 5 are evidence that this ethical principle was adopted as a comparison of methods were taken to choose the best suitable one by an evaluation technique.

Carefulness: Errors and negligence are to be avoided at all costs. During the collection of data, a detailed description of events and procedures is documented in this chapter.

Honesty and Integrity: In this research, ethical consideration has been adapted accordingly. Furthermore, with the quantitative evidence that has been collected using simulation instrumentation, no violation of ethical wrongdoing is linked to our dissertation concerning ethical principles.

3.8 Chapter Summary

This chapter outlines the dissertation's research design and methodology. Initially, an overview was provided on the layout of the chapter following 7 aspects of importance. In this chapter, a literature study of research methodology paradigms or beliefs were highlighted following their respective philosophical assumptions and approach.

A constructive methodology was adopted in this research, hence, an explanation was included as well as a research design following respective research methods and the research strategy. We further described the different aspects that go on in data collection and data analysis. Our chapter came to a close with ethical issues and a chapter summary.

Chapter 4

Empirical Analysis of Clustering-based Meta-Heuristic Algorithms

4.1 Chapter Outline

This chapter explores and evaluates three clustering-based meta-heuristic algorithms for controller placement suitability in the SDWSN model. It begins with a brief introduction about controller placements in WSN and proceeds to the different considered algorithms. The chapter also performed experiments and the results are presented graphically through a discussion in support of the choice of the algorithm, considering the research objective.

4.2 Introduction

Meta-heuristic clustering algorithms are used for the placement of controllers in an SDN environment efficiently. Several algorithms exist and some have already been explored and applied to solving CPP in the SDN. This chapter, therefore, explores some of these algorithms for CPP application in the context of this research. Firstly, we present some of the selected performance objectives in wireless CPP and the meta-heuristic clustering-based algorithms, discussing how they operate and function in the realm of CPP. These objectives are used to evaluate the performance and efficiency of the algorithms considered in this research concerning iteration times and randomly placed nodes in a topology (*x-y plane*).

The introduction of multiple placements of controllers in SDN brought about the question of how many controllers are needed and wherein the topology is the controllers going to be placed considering performance metrics that are or will be affected. According to Heller *et al.*[121], CPP formulation was first proposed about latency. The authors described CPP as an NP-hard problem and used a minimization optimization problem that considers latency being the distance between two nodes by adopting the Euclidean distance/equation. Moreover, a wireless CPP was also proposed by Dvir *et al.* [64] with the consideration of wireless connectivity establishment as the main factor, in the probability of connectivity between nodes. However, this research considers attributes or characteristics of both Heller *et al.* [121] through a clustering algorithm and Dvir *et al.* [64] through energy-aware protocols. Consequently, energy consumption is the second objective for this proposed work compared to the first, CPP. The proposed energy model is explained as the chapter precedes, thus, both CPP and energy-aware

methods were integrated and evaluated using a benchmark tool in the next chapter. Various objectives selected and described in this chapter are introduced again to evaluate the efficiency and performance of the proposed method. Considering the constraint of battery-powered sensor nodes, we propose how energy efficiency and CPP are achieved without altering some of the primary objectives in both SDN and WSN that combines to yield SDWSN.

Furthermore, clustering is the main focus in the context of this research as follows:

1. Clustering-based meta-heuristic algorithms are the first step toward achieving optimal controller placements based on clustering with consideration of the Euclidean distance between nodes being the objective. Accordingly, the centroids of each cluster after running the three meta-heuristic algorithms represent the possible controller placements which were assigned as containers to perform virtualization (NFV) to minimize cost (i.e. operational expenditure or capital expenditure (OPEX/CAPEX) [122, 123].
2. Ensuring a reduction in energy consumption using WSN routing protocols which are clustering-based algorithms. The protocols that perform better following the evaluations were chosen to run independently for each controller in the container.

With the choice of the methods to achieve optimal controller placement and energy efficiency, this chapter explains the methods and evaluates the algorithms and protocols to justify and support the choice. Finally, the chapter discusses the different objectives which were used to design the proposed system in this research. Through a careful and in-depth study of the literature, several approaches have been proposed to meet the objectives defined in this research. However, this research found that the existing approaches only solved these objectives independently and not in an SDWSN-integrated architecture. Moreover, we chose the clustering approach for both objectives because there has been proposed work in cluster-based SDN to achieve CPP and cluster-based routing protocols in WSN that achieve energy efficiency. Our contribution serves to take clustering as the main approach and solve it differently for each objective and finally integrate it to be one working mechanism. Hence, the significance of this chapter is to address the research objective of efficient CPP by evaluating the algorithms using clustering-based meta-heuristics. Furthermore, the evaluation gives a comparison of the algorithms, hence, eliminating bias in the choice of algorithms. Energy efficiency is discussed in the chapter that follows.

4.3 Controller Placement in Wireless Sensor Networks

There exist a lot of proposed work in solving SDN-based CPP with different performance objectives [124-127]. Heller *et al*.[121] was first to propose the objectives for CPP which is latency (i.e. both propagation and processing) in intra and intercommunication levels. Moreover, the work considered a wired CPP approach and the performance objectives differ from that of wireless CPP. This subsection presents the respective performance objectives that are adopted from [64] and have been used to determine an optimal solution using various nature-inspired clustering algorithms in this chapter.

4.3.1 CPP Performance Objectives

Several performance objectives have been adopted from previous works including Dvir *et al.* [64] and Heller *et al.* [121]. See Table 4.1 as a reference from chapter 2, to the performance objectives being used in this chapter for evaluation purposes. The evaluation method includes minimization in the objective function to evaluate algorithm efficiency, however other objectives presented in this section are already provisioned through experimentation/simulation when deployed in a network with communication protocols. Therefore, this is further described in the next chapter.

Table 4.1 Selected Performance Objectives

Performance objective	Equation of Performance objective
1. Propagation Latency (Average-Case)	$\boldsymbol{L_{avg}(S') = \frac{1}{n} \sum_{v \epsilon V} \min_{(s \epsilon S')} d(v, s)}$
2. Propagation Latency (Worst-Case Latency)	$\boldsymbol{L_{wc}(S') = \max_{(v \epsilon V)} \min_{(s \epsilon S')} d(v, s)}$

As already stated in Chapter 2, propagation latency is described as Euclidean distance minimization between a set of *E* nodes connected by *V* links in a certain network topology *G* given by G(V, E). This type of latency ensures optimal connection and placement of controllers by Average-case and worst-case, meaning intra-domain and inter-domain denoting controller-controller communication and switch-controller communication [18, 24, 121]. We have adopted a clustering approach using the average case to find optimal clusters of a given G topology of V nodes. The worst-case is further brought into play when considering the cluster heads (CH) as the optimal controllers for the network. This can be seen through other authors adopting the same approach. However, with different methods or mechanisms like the K-means

algorithm [128, 129], in this case, we use Nature-inspired meta-heuristic clustering algorithms [130, 131].

4.3.2 Theoretical Framework of Nature Inspired Clustering Algorithms

This subsection presents some of the nature-inspired algorithms for CPP which includes a variety of algorithms that make up some of the categories in this class. Evolutionary and Swarm Intelligence algorithms are the two categories of bio-inspired algorithms that have been chosen specifically for evaluation in this chapter. The algorithms specifically focused on partitioned clustering using Euclidean Distance as the objective function. Differential Evolution (DE) and Genetic Algorithm (GA) are population-based clustering algorithms in evolutionary-based algorithms. Particle swarm optimization (PSO) is another population-based clustering algorithm, which forms part of swarm intelligence-based algorithms.

All these algorithms make up the cluster-based meta-heuristic, therefore, a similar family being nature-based justifies the reason for use in this research. Initially, k-Means-inspired algorithms were highly used for partitioned clustering, however, they suffer from being trapped into local optimum; even though they provide a much simpler computation. Hence, this chapter presents an evaluation of the cluster-based nature-inspired meta-heuristics with a high probability of achieving optimality in the clustering process serving as one of the core reasons for being worthy to be used for this research [130, 131].

A. Nature-based algorithms: Evolutionary Algorithms

This subsection succinctly discusses how evolutionary algorithms can be used in this research to explore and/or exploit the search space efficiently. As a search-based algorithm, the search space of this research is modelled in an undirected graph of sensor nodes to find an optimal placement of controllers and the location of the respective controllers. The three most popular algorithms within the evolutionary paradigm are therefore introduced, which are single objective algorithms with the main purpose of automatic clustering.

Essentially, evolutionary algorithms are population-based algorithms that solve search space-related problems, adopted by Darwinian Evolution by natural selection [130]. However, in this research, there are applied in the context of clustering. They adopt reproduction, mutation, recombination and selection as the respective mechanism following a biological process of evolution as the population being its set of solutions that are generated in every iteration [130-132]. The *evolutionary strategy* follows an assignment of searchers as mutation and selection, representation is assigned as the normal parameters of the problem. In comparison,

evolutionary programming follows a fixed structure. This method does not follow a constant pattern because of the extension of the membership concept resulting in parents being the generators [130, 131]. Some of the selected evolutionary algorithms are discussed as follows:

Genetic Algorithm

The need for optimal solutions for search and optimization problems with the use of biological methods asserts the importance of the use of a genetic algorithm (GA). This biological process follows the process of natural selection, only the fittest individuals (solutions) will be regarded for reproduction to generate offspring for the next generation [130]. In GA, the search space parameters are encoded through strings which are called chromosomes and this collection of strings makes up a population[130, 133]. The process initiates by creating a random population that constitutes various points within the search space. The degree and goodness of each string depend on the associated objective and fitness function. With the simple biological law of survival of the fittest, a few selected strings are assigned and designated to participate in the mating pool which will undergo crossover and mutation biology-inspired operators resulting in a new generation [130, 133, 134]. This process will continue through a predetermined series of generations until a satisfactory termination condition is met. GA serves as the best candidate for a cluster-based method compared to many clustering techniques that produce different multiple solutions for a similar dataset. Similar to any other artificial intelligence technique, there is no assurance of a constant response time to optimization resulting in the limitation used in real-time applications [133-135]. GA process is captured in Figure 4.1 and its parameters are presented in Table 4.2.

Genetic Algorithm
Input: $\text{Initialization}_{\textbf{parameters}}$, $K_{\textbf{clusters}}$, $\text{Data}_{\textbf{count}}$
Function: Genetic Algorithm Clustering
Output: Sub-set Clustering, $K_{\textbf{centroids}}$
Steps: 1. Initialization of random chromosome for n individuals in the population. 2. Assignment for fitness conditions to each of the respective n individuals in the program 3. Create offspring for the next generation through recombination from the current population 4. Mutate offspring for this generation. 5. Tournament selection is performed resulting in parent selection to create the next generation 6. The next population of *n* individuals is chosen 7. Set a new population to the current population 8. Assess the fitness of each offspring in the generation If the stopping criterion is satisfied then halt the program and print solutions, else go to step 3.

Figure 4.1: Genetic Algorithm [136]

Table 4.2 Genetic Algorithm Parameters [136]

Parameter	Value
Length of individual string	6 bits
No. of individuals in the population	6
Probability of mutation	0.3333
Probability of recombination	0.5
The initial string of individuals	Random
Bit type of individual's string	Real-coded
Crossover type	N-point
Mutation type	N-bit flip
Selection type	Tournament

a) Differential Evolution

Differential Evolution (DE) uses state-of-the-art parameter vectors or genomes which denote the individual trials or population. Compared to the recombination of the solution of EA's offsprings, DE uses the scaling difference of two population vectors that have been randomly selected [137, 138]. This classical technique makes use of three crossover methods, binomial, exponential and arithmetic along with some basic mutation strategies [138]. Moreover, the algorithm includes a 4-stage step, population initialization through variable vectors, mutation, crossover or recombination and selection [139] (see Figure 4.2.). Compared to other algorithms that require specification of the absolute step size for each variable iteration, DE includes an automatic adaptation attribute which results in improvement of the search move of the algorithm [138-140]. Even though the algorithm seems to advocate the efficiency it possesses, it brings about several drawbacks including getting stuck in local optima the majority of the time, this is called stagnation [138]. Stagnation results from the algorithm failing to improve the solution of its respective population through a prolonged iteration of generations [137]. Table 4.2 presents DE's parameters and values.

Differential Evolution Algorithm
Input: Initialization$_{\textbf{parameters}}$, K $_{\textbf{clusters}}$, Data$_{\textbf{count}}$
Function: Differential Evolution Clustering
Output: Sub-set Clustering, K$_{\textbf{centroids}}$
Steps:
1. Initialization of random population vectors and individual size initialization.
2. Random selection of a single principal parent and three auxiliary parents
3. Generation of a vector through differential mutation
4. Generate a child trial vector through recombination of the mutated vector together with the principal parent.
5. Perform the "Knock-out" competition for next-generation survival selection
6. After the maximum iterations are achieved results in the satisfaction of the fitness criterion hence stopping the program and outputting the solution *Else*: Repeat step 2.

Figure 4.2: Differential Evolution Algorithm [136]

Table 4.2 DE Algorithm Parameters [136]

Parameter	Value
Individual Size (*N*)	6
Population Size (*P*)	7
Mutation Amplification factor (*F*)	0.3
Crossover Population (*CR*)	0.667

B. Nature-based algorithms: Swarm Intelligence

Swarm intelligence-inspired clustering approaches such as PSO uses swarm behaviour. Fish and bird schooling in nature, ant colonies and virtual ant algorithm are some of the algorithms that use this method. GA and virtual ant algorithm possess similar characteristics to PSO. However, PSO uses random generation of real numbers and global communication between particles, instead of mutation/crossover [63, 130, 141]. PSO uses a space search for the respective objective function by adjusting of trajectories of individual agents (particles). The particle in the algorithm can potentially move randomly and is attracted towards a global best (location) which has been updated as the new best current $\boldsymbol{i}$. For all the n particles, there is a current best, however, the goal of the algorithm is to achieve a global best throughout all the possible current best after the number of iterations has been completed or if the objective no longer improves [141].

PSO is a random search algorithm amongst the swarm intelligence family. It is one of the widely used meta-heuristic algorithms in clustering-based scenarios. It takes advantage of information from its neighbours to improve its potential solutions through the problem space. The respective equation used to update the solutions is illustrated in Equations 4.1 and 4.2.

$$\boldsymbol{v_i^{new} \leftarrow v_i + \varphi_1 \otimes (p_i - x_i) + \varphi_2 \otimes (p_g - x_i)} \qquad \mathbf{4.1}$$

$$\boldsymbol{x_i^{new} \leftarrow x_i + v_i^{new}} \qquad \mathbf{4.2}$$

In equations (s) 4.1 and 4.2, $\varphi_1 = c_1\mathbf{R}_1$ and $\varphi_2 = c_2\mathbf{R}_2$, $\mathbf{R}_1$ $\mathbf{R}_2$ represent individual functions that return random values ranging from [0,1] from a particular vector; c_1 and c_2 denote acceleration coefficients and $\otimes$ represent component-wise multiplication; vi denotes the velocity of a particle and the direction in which it has travelled. Finally, $\mathbf{P}_i$ denotes personal best position and $\mathbf{P}_g$ represents global best in its neighbourhood [142]. As described in [63], PSO is much dependent on a stochastic process similar to evolutionary algorithms including genetics which are amongst the clustering meta-heuristics which have been evaluated or compared [63, 142-144]. Figure 4.3 presents the PSO algorithm while Table 4.3 captures the parameters.

PSO Algorithm

Input: Initialization**parameters**, K **clusters**, Data**count**

Function: Particle Swarm Intelligence Clustering

Output: Sub-set Clustering, K**centroids**

Steps:

1. Initialization of random chromosome for n individuals in the population.
2. Assignment for fitness conditions to each of the respective n individuals in the program
3. Create offspring for the next generation through recombination from the current population
4. Mutate offspring for this generation.
5. Tournament selection is performed resulting in parent selection to create the next generation
6. The next population of *n* individuals is chosen
7. Set a new population to the current population
8. Assess the fitness of each offspring in the generation
9. If the stopping criterion is satisfied then halt the program and print solutions, else go to step 3.

Figure 4.3: Particle Swarm Intelligence Algorithm [136]

Table 4.3 PSO Algorithm Parameters [136]

Parameter	Value
Individual parameters ($c_1, c_2, r_1, r_2,\ w$)	(1, 1.2, 0.5, 0.5, 0.8)
Amount of particles	6
Initial social Influence ($s_1, s_2, s_3, s_4, s_5, s_6$)	(1.1, 1.05, 1.033, 1.025, 1.02, 1.017)
Initial personal Influence ($p_1, p_2, p_3, p_4, p_5, p_6$)	(3, 4, 5, 6, 7, 8)

4.4 Methodology

This section presents the methodology used to assess the performance of each algorithm to choose the most efficient algorithm. The algorithms were measured using the number of iterations with the best cost. After a static assigned iterative process, the algorithm that performs efficiently was influenced or determined by having a high number of best possible solutions in the search space, further finding the optimal solution with fewer iterations.

The performance comparison was performed on a windows 10 intel(R) Core(TM) i3-4005U CPU 1.70 GHz with 4 Gb ram. Matlab R2020b Tool was used to perform and present the results obtained. The data used includes sensor nodes located randomly in a topology (V, E), where V is the set of nodes and E represents the set of links that connects the respective nodes. Moreover, we used 50 nodes with 200 iterations for all the algorithms considered in the Chapter to eliminate inconsistency or redundant results in all algorithms.

The algorithms are executed to gather their numerical data and presented graphically as seen in Figures 4.4 to Figure 4.11. Table 4.5 shows our quantitative evidence from the automatic algorithms as will later be explained that it is more modified and efficient compared to the traditional ones. Furthermore, the method of data gathering used in this chapter is necessary as it best explains the performance of the evaluated algorithms to present its significance to avoid any bias in our outcomes. That being said, the significance of these results is to bring about an explanation of the underlying results to support our choice of algorithm. More of this explanation will follow as it adds emphasis to the significance of our chapter and the respective results.

Table 4.4 Evaluation Parameters

Parameters	Value
Nature-based Clustering algorithm(s)	**GA, DE, PSO**
Number of nodes/ Population	**50**
Number of Iterations	**200**
K Clusters	**3**
Area	**100m * 100m**

4.4.1 Performance Results and Analysis

In this section, we present the results obtained from MATLAB for the evaluation of the three algorithms: DE, GA and PSO being the leading clustering-based meta-heuristic and evolutionary algorithms in terms of time efficiency (best cost) to cluster a network after ***r*** iterations. We further evaluated the automatic-based DE, GA and PSO as the advancement of this meta-heuristic-based clustering algorithms under the same conditions. We selected Evolutionary Clustering (EC) and Evolutionary Automatic Clustering (EAC) solely because the automatic-based EA clustering is a much more recent development in this method to perform efficiently and much better compared to the earlier version of algorithms.

Figure 4.4, Figure 4.5 and Figure 4.6 are illustrations of the EA algorithms results which are for GA, DE and PSO respectively. The graphs show the performance of the respective algorithms shown as the best cost concerning the number of iterations with Figure 4.7 showing the comparison of all algorithms. Furthermore, when assessing the graph, the trend of the graph entails an increase in the number of iterations, showing a decline in the best cost. As illustrated in the graph, approximately between 0 – 20 iterations, there is a steep or rapid decline in the best cost. This explains that the Genetic algorithm performs clustering at fewer iterations, however, with less best possible solutions compared to PSO and DE in Figure 4.5 and Figure 4.6. Similar to PSO, they both perform in fewer iterations to find the best solution (best cost). However, PSO and DE find more best solutions compared to GA. DE is the least performing as it requires more iterations to find the best possible solutions in the search space, it takes approximately 0 – 85 iterations to find the best possible solution. PSO has both efficiencies in finding the best solution in the search space with less time or iterations, thus providing

equilibrium between GA and DE with two drawbacks by our dependent and independent variables.

Although PSO appears to be efficient, it provides a lot of possible solutions of above 570 best solutions. Thus, an additional modification or enhancement to the algorithm is needed to generate satisfying results, hence, automatic clustering was further evaluated of the very same PSO, GA and DE algorithms as seen in Figures 4.8 to Figure 4.11. Table 4.5 presents the quantitative results to support the analysis and the choice of the algorithm.

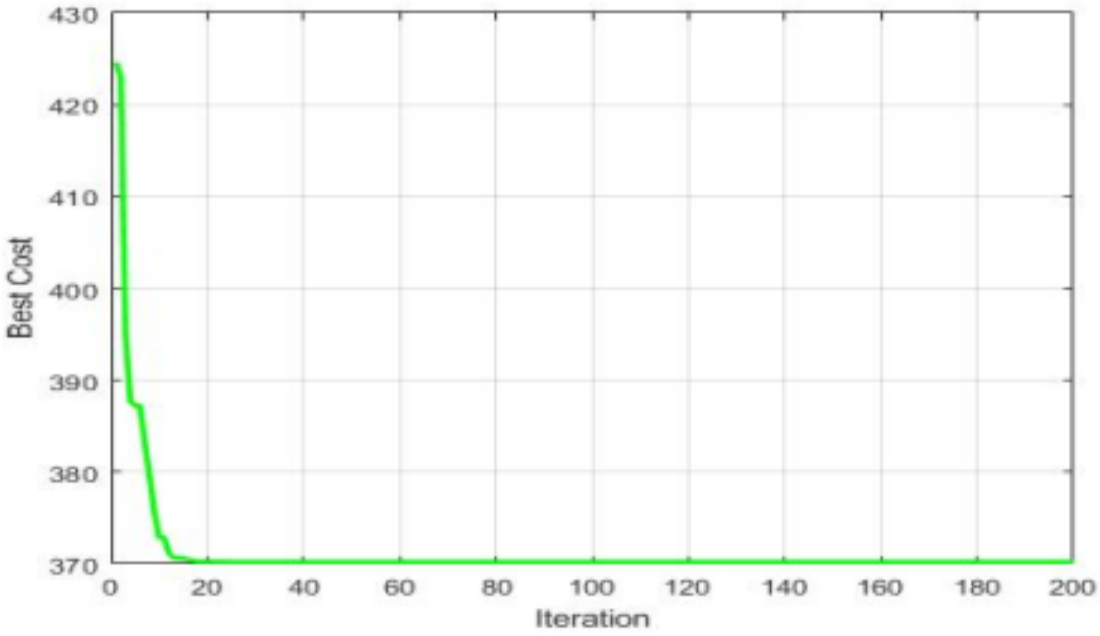

Figure 4.4: Best cost vs Number of iterations with GA

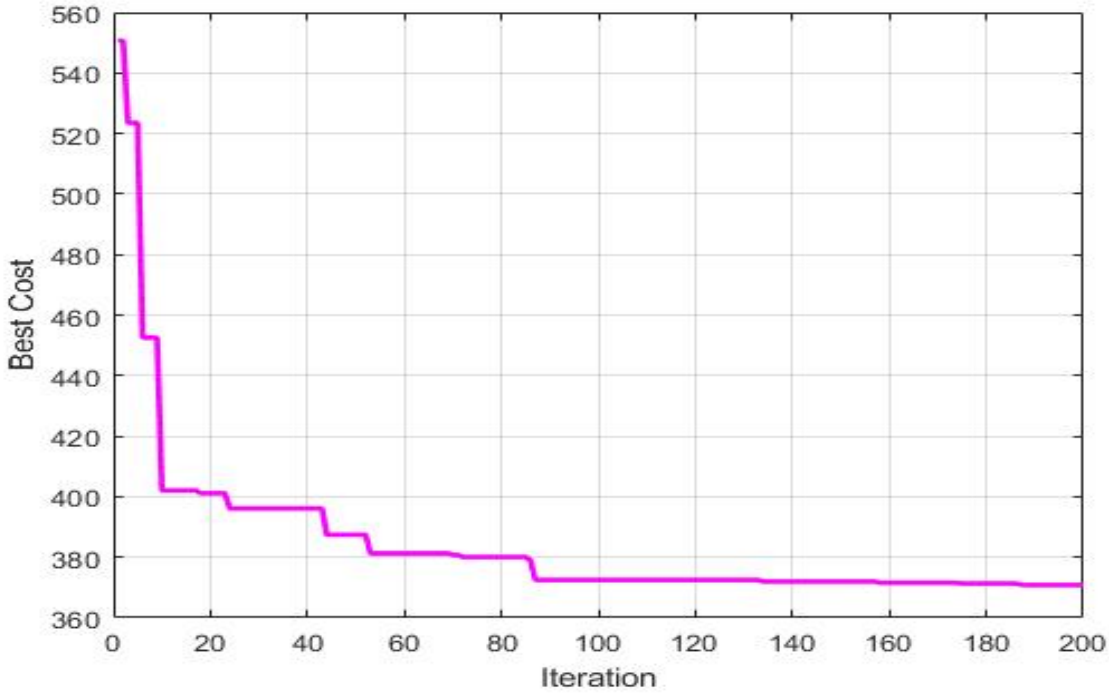

Figure 4.5: Best cost vs Number of iterations with DE

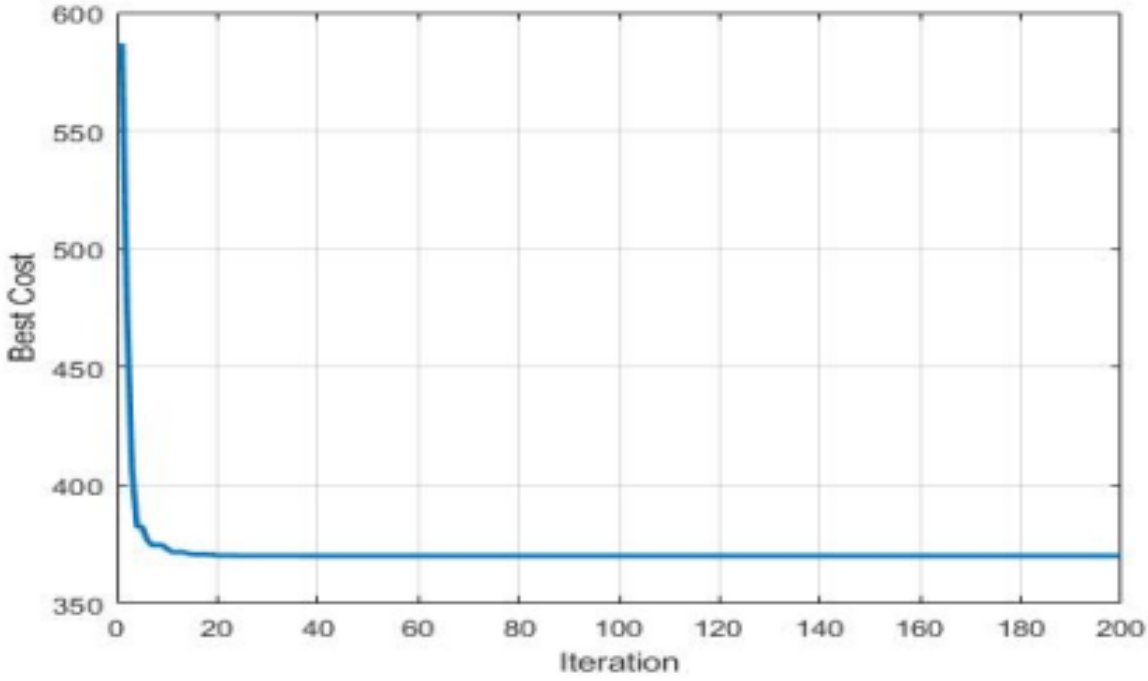

Figure 4.6: Best cost vs Number of iterations with PSO

Compared to Figures 4.4 to 4.6 which show the individual results for each algorithm, Figure 4.7 explores the comparison of all the algorithms. The results show efficiency in PSO and GA.

However, the results are not very satisfying to see the most efficient. Consequently, we explored the automatic approach which is presented next.

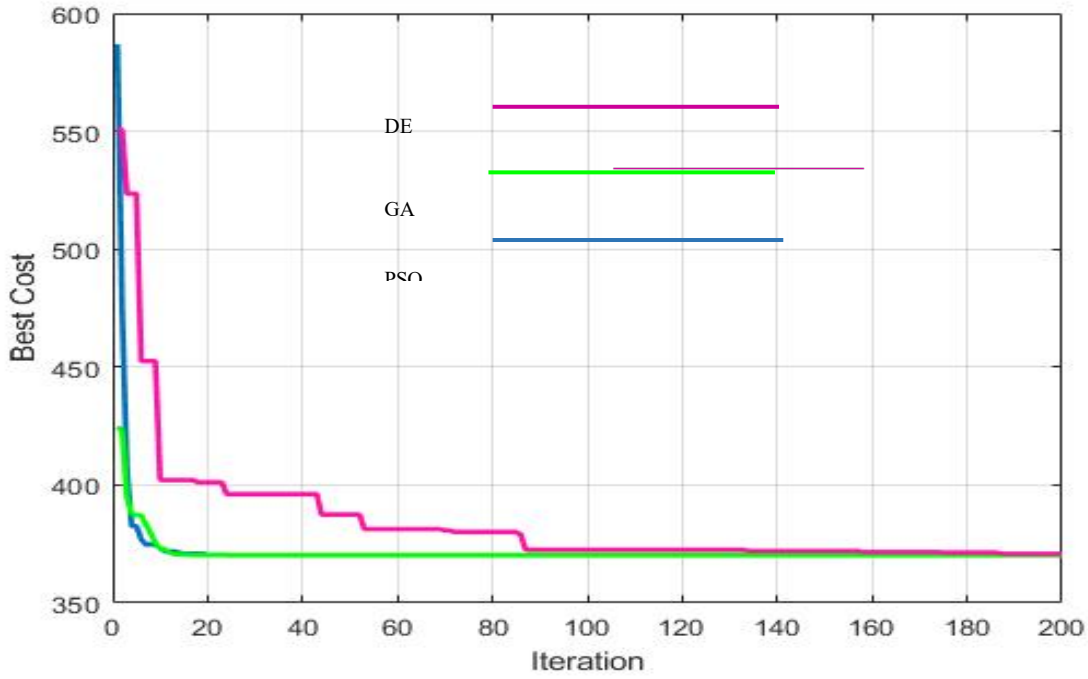

Figure 4.7: Comparative Best cost vs Number of iterations with DE, GA and PSO

The results that have been provided signify the best performing algorithm with the higher best possible solutions. In this case, PSO is seen as the best or most efficient candidate of the three nature-based clustering algorithms. This has been fully explained at the beginning of this section. Therefore, it is important to evaluate a more modified version of these algorithms and this is carried out in the next section and is illustrated in Figures 4.8 to Figure 4.11.

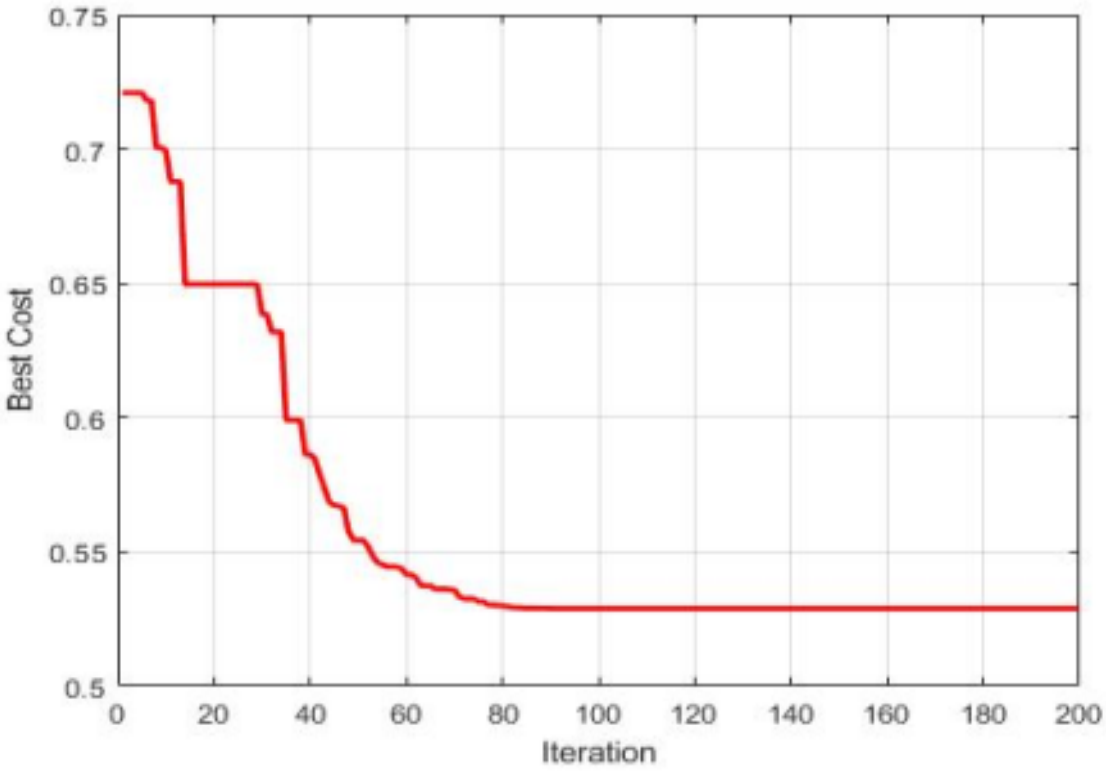

Figure 4.8: Best cost vs Number of iterations with Automatic GA

Figures 4.8 to Figure 4.10 present the respective EAC algorithms results including the automatic PSO, GA and DE. The respective graphs show the performance of the algorithms shown as the best cost concerning the number of iterations. As seen in the above-mentioned graphs, we explained the best cost as the pool of best solutions that are acquired in the search space following a minimization of the Euclidean distance between nodes to form an efficient or optimal cluster(s). Thus, this section shows the automatic PSO, GA and DE, illustrating a graphical representation of all the respective algorithms. Similar to the above-illustrated results, the DE is the least performing as it needs close to approximately 120 iterations throughout the search space to bring about 0.75 best possible solutions, greater best solutions compared to the GA. However, the GA outperforms the DE in about 40% difference in iterations as it completes the process at approximately 80 iterations, with much fewer best solutions. This leaves automatic PSO as the best or most efficient algorithm as it provides more best solutions throughout the search space at a very short iteration period. The quantitative results are provided in Table 4.4. These results are explained in great detail below as they further give accurate numerical data on the performance of the algorithms at hand.

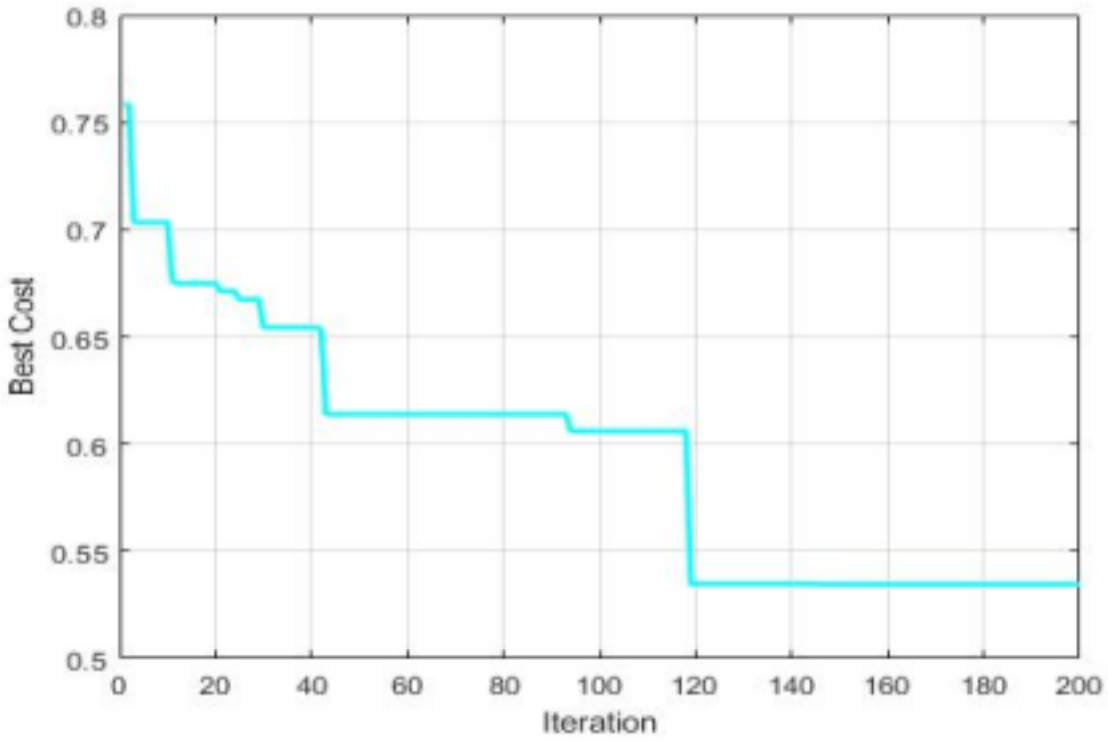

Figure 4.9: Best cost vs Number of iterations with Automatic DE

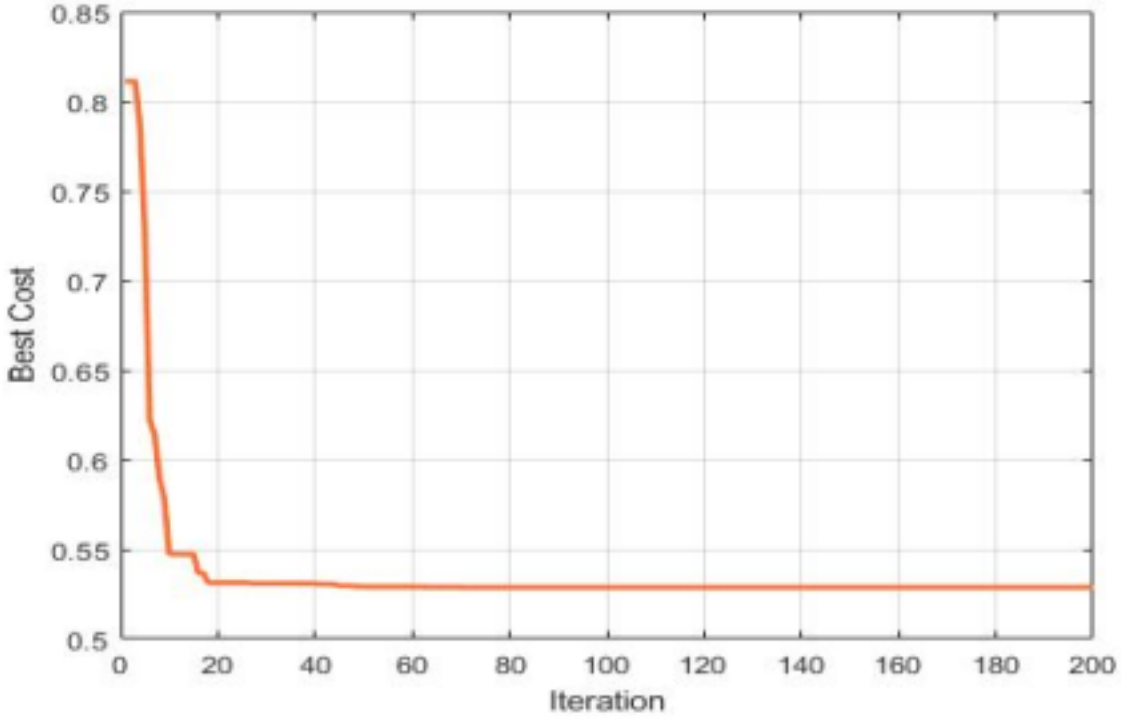

Figure 4.10: Best cost vs Number of iterations with Automatic PSO

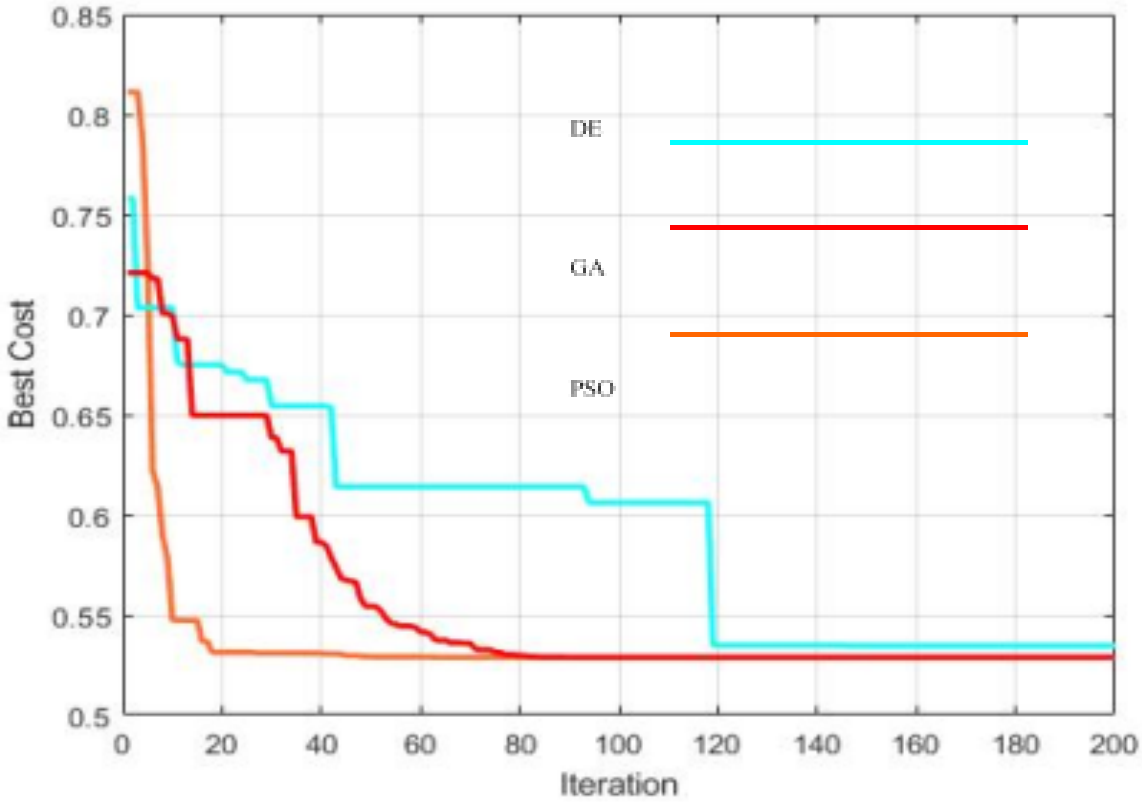

Figure 4.11: Comparative Best cost vs Number of iterations with Automatic DE, GA and PSO

In comparison to Figure 4.8 to 4.10, which shows results for each algorithm respectively, Figure 4.11 presents the comparison of all the algorithms in terms of automatic clustering concerning the best cost and the number of iterations. The results show the PSO algorithm as being more efficient than other considered algorithms. These results show the best performance of the algorithms in comparison to Figure 4.7 as the results are not very satisfactory in terms of efficiency. Therefore, an automatic approach presents many satisfying results and is adopted in the context of this research as the best performing; PSO to be specific.

Table 4.4 shows the quantitative results of this chapter as far as evaluation is concerned. These results are graphically illustrated in Figure 4.8 to Figure 4.11 which explains the best performing automatic algorithm for the minimization of the distance i.e. propagation latency, as being the primary performance objective. The algorithms gather all the best solutions throughout the search space or the G network of V nodes that are connected with E links and used 100 nodes with all the algorithms for evaluation purposes. As mentioned above, the graphical representation of the results explains the efficiency of the algorithm. Thus, the

Table 4.5 Results for Automatic PSO, GA and DE

Clustering-Based Meta-Heuristic					
Automatic Differential Evolution		**Automatic Genetic Algorithm**		**Automatic Particle Swarm Intelligence**	
Iteration	**Best Cost**	**Iteration**	**Best Cost**	**Iteration**	**Best Cost**
0.0523438695946335	0.759604359355479	0.703306336730314	0.726664176205135	0.790513833992084	0.812665061864147
1.51698459806328	0.748370673606345	4.64620503471702	0.664882250978752	1.97628458498024	0.760505265139133
1.88141644486363	0.724427797557615	7.77621708855085	0.640349959654524	2.76679841897235	0.730672283475488
9.36362693635282	0.692202244860672	8.53132502515365	0.608174441887571	3.95256916996047	0.604970430027639
10.9270296829245	0.676479825388752	10.8564199117381	0.568467768446849	4.74308300395258	0.545312886960485
18.4497326018360	0.674927286464172	13.6138589203351	0.564758225994442	4.74308300395258	0.430275292975522
22.8021747396386	0.671902205849674	22.1710847454250	0.449471025970533	4.74308300395258	0.359974541095823
29.5130538697428	0.655393146903564	28.4849028221910	0.451544335196199	5.53359683794467	0.298186672214132
41.3921494071810	0.653808016513010	33.9459868702868	0.392987358417262	7.11462450592886	0.238520708886842
42.9239483079597	0.614145683852904	34.2966438540390	0.351330653596725	8.69565217391306	0.185245723003160
71.4385181746804	0.613932357893801	39.7178805176175	0.254893756911031	9.09090909090910	0.114940760993393
92.0313865693532	0.613030166858430	46.3604395166512	0.194476405367443	14.6245059288538	0.110621167543365
95.1907835284707	0.606273363389908	50.6599723060678	0.156719763306536	19.3675889328063	0.0722247813208913
117.366807073336	0.604611198625233	62.8393253837801	0.109717282806850	30.8300395256917	0.0720100646874103
120.441268696885	0.533515878463461	71.5021467778409	0.0948118705358480	39.5256916996048	0.0699723617343756
139.847017633958	0.533515878463461	79.7744638036321	0.0836832431786261	52.1739130434783	0.0676146888961536
161.628980726592	0.533515878463461	90.0291882091590	0.0836832431786261	60.474308300395	0.0643307874429156
180.637706350892	0.533515878463461	180.759690385822	0.0836832431786261	180.237154150198	0.0643307874429156
200.836514293333	0.533515878463461	199.697159877669	0.0836832431786261	200.395256916996	0.0643307874429156

significance of providing these results is great to illustrate the most efficient algorithm of the nature-based clustering calibre. Furthermore, the meaning of these respective results is to avoid bias as these provided outcomes support our choice of algorithm that has been adopted as one of the primary techniques or methods utilized in our Efficient energy-aware CPP algorithm.

Therefore, an automatic PSO is seen in our above-mentioned analysis, as being efficient and the numerical data confirms or supports that. This is why it was our chosen algorithm for performing one of our key research objectives being efficient CPP. The numerical data explain the decrease in the acquired best solutions in a search space with the iterations as to how long can the algorithm choose the underlying solutions. The number of best solutions collected also plays a vital role in the capabilities of the algorithm(s) when given a particular search space.

As mentioned, the data shows the automatic PSO collecting higher best solutions compared to other algorithms at *0.762665061864147*. GE and DE accomplished less amount of best solutions throughout the search space at *0.759604359355479* and *0.726664176205135;* illustrating a weakness in the capabilities of the algorithms. The algorithms further show the number of iterations until termination or optimal solution. The automatic PSO terminates at *60.474308300395*, this can be seen as the best cost; started to be constant throughout until *200* iterations at *0.0643307874429156*. Compared to GA and DE, the two terminate or attain an optimal best solution at *120.441268696885* and *79.7744638036321* which are much slower compared to automatic PSO especially as they attained much higher best solutions.

4.5 Discussion

In this chapter, we have evaluated and compared three meta-heuristic-based algorithms that perform clustering which are GA, DE and PSO as potential candidates for the design of an efficient CPP model in the SDWSN. The results obtained are captured in Figure 4.7 and Figure 4.11. The evaluation was conducted using Matlab R2020b with randomly placed nodes in an ***nxm*** Cartesian plane. The nodes were set to 100 and the iteration for all the algorithms was assigned to 200.

Firstly, the three algorithms were respectively evaluated as shown in Figure 4.7. The comparison between the algorithms shows that DE and PSO have the same efficiency resulting in an extension of the evaluation using advanced techniques in the form of an automatic-based approach as shown in Figures 4.8 to Figure 4.10. In Figure 4.11, which is the overall comparison of the three algorithms using the automatic-based approach, it can be seen that the automatic PSO outperformed GA and DE algorithms in terms of time efficiency. PSO performed efficiently with consideration of the amount of time taken to cluster the network nodes (best cost) concerning the number of iterations. All the algorithms showed good results but from the findings, PSO performed much better as all algorithms were evaluated using one

objective, being the latency-aware, $\boldsymbol{d_{z_p,m_j}}$. Based on this result, this study adopted PSO in the design and implementation of an efficient controller placement for SDWSN.

4.6 Chapter Summary

This chapter introduced some of the important objectives, which were to achieve controller placement, and selected and evaluated three meta-heuristic-based algorithms that perform clustering based on a latency-aware approach to find the optimal placement of controllers in SDWSN. The investigated were the evolutionary clustering algorithms which are the GA and DE, and PSO. The evaluations were performed using the Matlab R2020b tool and the results obtained are presented in Figures 4.4 to Figure 4.11. The results show the time efficiency among all algorithms concerning the number of iterations applied. The overall results show the automatic PSO as the best performing algorithm as presented in Figure 4.11. The next chapter evaluates the protocols that achieve energy efficiency in wireless networks.

Chapter 5

Empirical Evaluation of WSN-based Energy Efficiency Protocols

5.1 Chapter outline

This chapter compares the energy efficiency protocols based on WSNs, with a focus on cluster-based routing techniques. The goal is to build an energy-aware controller placement algorithm in SDWSN using the best energy-efficient protocol. The chapter begins with a brief introduction, followed by a presentation of the various WSN objectives and an overview of each protocol of the mentioned calibre. Experiments were also performed and the results were obtained, analysed and presented graphically, through a discussion in support of the choice of protocols, given the objectives outlined in this research.

5.2 Introduction

Energy-constraint sensors in WSN ought to function autonomously for a long time. However, the replacement of batteries on such devices is very challenging considering sensors are located in unpleasant or hostile locations or environments. It is of importance to consider, among a few, primary objectives in WSN, the increase in network lifetime and energy efficiency. Such objectives can be achieved using different methods like routing, clustering, data aggregation, error control code and duty cycling. However, these methods depend on the application, type of topology, type of sensors etc. Attaining energy efficiency in WSN can be categorized into transmission power management techniques and battery management methods [145, 146]. Data transmission serves to be the main contributory factor in energy consumption, hence, this research sought to explore the respective methods. This research narrowed its scope to routing techniques for data transmission for the reduction in energy consumption. Hierarchical Cluster-based routing protocols were evaluated to compare and support the choice of the protocol as they are regarded or considered to be the best methods to achieve energy efficiency. These were used to answer *RQ2.2* and achieve the energy efficiency objectives i.e. RO2 outlined in Chapter 1 of this research [146, 147].

Similar to the CPP methods illustrated in Chapter 4, the algorithms perform clustering and acquire centroids. The hierarchical Cluster-based routing protocols illustrated in this chapter

present cluster heads (CH) for cluster management instead of controller placements. The energy model used to evaluate and test network lifetime is illustrated in Equations (5.1) and (5.2) with the representation of the required energy for *l-bit* message transmission through a distance *d* (Equation 5.1) and required energy by radio for receiving a *1-bit* message (Equation 5.2). $E_{elec}, \varepsilon_{fs}$ and ε_{mp} denote the required energy by the amplifier and electronic circuit concerning multipath (*mp*) and free space (*fs*). [146, 148-150].

$$E_t(l,d) = \begin{cases} lE_{elec} + l\varepsilon_{fs}d^2 & for\ d < d_o \\ lE_{elec} + l\varepsilon_{mp}d^4 & for\ d \geq d_o \end{cases} \quad (5.1)$$

$$E_r(l) = lE_{elec} \quad (5.2)$$

Therefore, this chapter reviewed three essential energy-aware routing protocols which use clustering as a technique to arrive at an increase in a network's lifespan. To do so, a comparative analysis of the individual protocols was carried out based on the number of iterations and the number of dead nodes. The protocols follow the minimization of energy consumption as the objective function and the performance comparison is presented graphically as an indicator of choosing an efficient protocol. The results were acquired by including randomly placed sensor nodes in the network topology. The threshold sensitive energy efficient sensor network (TEEN) protocol had shown to be the most efficient according to the comparison study performed. In general, the significance of this chapter is to address energy consumption in consideration of designing an energy-aware algorithm, eliminating bias and supporting the choice of protocol for the research algorithm.

5.3 WSN Objectives

This section explains how WSN's energy efficiency is affected by several performance objectives. The objectives discussed in this chapter have been adopted from [151, 152]. The comparison process includes the minimization of energy consumption as the objective function presented in 5.1 and 5.2. Other objectives were discussed in Chapter 2 and were used as a reference in the next chapter for their critical importance when evaluating the entire energy-aware CPP algorithm but were considered as performance objectives in this chapter.

5.3.1 Hop Count

For a packet to be able to move from its origin to its destination, it must first be able to travel from its source which in this case are called Hop counts (N_c). The packet keeps count or uses

TTL (Time to Live) to keep count of routes to compute the Hop count at the end of the packet's destination. TTL values are OS sensitive; meaning if nodes of the sender and receiver are heterogeneous, this will result in a mismatch in TTL values [151]. The Hop Count computation was done using the following Equation 5.3. for *n* packets:

$$N_c = t - t_0 \qquad (5.3)$$

Where t_0 denotes the initial TTL and t represent the end TTL from an IP address (i).

5.3.2 Transmission Count (ETX)

To ensure the successful delivery of data or packets from one node to the other, a significant amount of MAC layer transmission through a wireless channel or link with no error is needed. Throughput can be calculated using ETX to analyse the performance of a path resulting in a reduction in error rate. The Transmission Error rate uses Equation 5.4 where d_f (forward delivery ratio) denotes the probability of packets sent to a node or destination and d_r (reverse delivery ratio) represent the ACK packet of data being received from the sender to the receiver [151].

$$ETX = \frac{1}{(d_f * d_r)} \qquad (5.4)$$

5.3.3 Transmission Time (ETT)

An increase in bandwidth may potentially minimize the time it takes for a packet to be delivered or transmitted successfully. The processing of link transmission is known to be ETT with consideration of bandwidth [151, 152]. The ETT equation is defined as follows:

$$ETT_l = ETX_l \frac{s}{b_l} \qquad (5.5)$$

Where s denotes the packet size that is to be transmitted and b_l represents the bandwidth of a certain link *l*. to compute ETT, the ETX is computed first for dependability. The ETT of a link contention can be affected by the loss channel resulting in packet loss and a reduction in the available bandwidth. The Channel diversity index (CDI) which assists to overcome this drawback, is illustrated as Equation 5.6:

$$CDI = \frac{\min(N_a, N_g)}{2 * [N/2]} \qquad (5.6)$$

Where N represents hops of a certain path, N_a denotes the hops collected in link a and N_g hops have taken in g link [152].

5.3.4 Energy Consumption

The drawback of battery-powered sensor nodes results in nodes dying in a network. When a node dies, the network losses control over the network area being covered by the dead node, creating a hole in the sensing space. The network has the responsibility to avoid this type of phenomenon to occur through energy consumption monetization of each node. This will preserve the longevity of the network. The calculation of energy needed to transmit n bit over a distance of l (E_{Tx}) considers adding together energy consumed by the electronics of the transmitter radio (E_{elec}) and the transmitter radio amplifier, which consumes energy (E_{amg}). The equation is described in the opening paragraph as Equations 5.1 and 5.2 [151].

5.4 Energy-Aware Clustering Protocols Theoretical Framework

This section presents an overview of the energy efficiency protocols considered in this research. Routing is one of many techniques with the main goal in WSN to preserve the energy of sensor nodes. WSN routing techniques are also well-known for lowering the energy consumption of WSN sensor nodes. Routing protocols are classified according to their Network Structure including Flat networks and Hierarchical networks; Topology is another category with further classification including Location-Based [145, 153]. In the context of this research, the Hierarchical Based routing approach has been chosen since they are known for less interference and collision in the network.

The protocols considered include Low Energy Adaptive Clustering Hierarchy (LEACH), TEEN and Distributed Energy Efficient Clustering (DEEC) [154, 155]. Transmission makes up the majority of all factors which consume energy the most, therefore, a technique that assists in reducing the rate of transmission is needed to achieve energy efficiency. Therefore, this forms the basis of performing the comparison of energy-aware routing protocols. Efficiency is achieved when partnered with an SDN controller as the protocol will not be running on a sensor node but rather on a controller, which will take the master decision of increasing the energy lifetime of a network as the sensor node will not be doing the heavy lifting [155].

The three protocols considered are as follows:

5.4.1 LEACH

LEACH is a clustering mechanism with the express objective of lowering energy usage. Nodes within each cluster are potential members selected to be assigned as cluster heads (CH). This protocol uses the randomized rotation of a local CH in each cluster and CH is critical to prolonging the network lifetime. LEACH achieves reduced transmission, thus, preserving the energy of sensor nodes [153]. A drawback of single-hop routing by LEACH has motivated researchers to propose a modification of the protocol [156-158], as dynamic clustering promotes a shrink in overhead leading to an increase in energy consumption [153, 159]. The following are the steps followed by LEACH:

The *setup phase* includes clustering setup, CH advertisement to alert all potential CH to know the information of each CH, and then the creation of a transmission schedule on each CH to establish a connection to transmit data [159-161]. The formulas used when CH advertisement is broadcasted to other respective CH are shown as Equations 5.7 and 5.8:

$$T(n) = \frac{P}{1 - P\ (r\ .mod P^{-1})};\quad \forall\ n\ \in G \tag{5.7}$$

$$T(n) = 0;\qquad \forall\ n\ \notin G \tag{5.8}$$

Where *T(n)* denotes the threshold, G is a set of non-potential CH in the previous round, and P represents the cluster head probability.

The *steady phase* is described as the data transmission of members in a cluster who forward their data to the CH through a single hop, resulting in the CH transmitting the data to other CHs or directly to the base station. LEACH provides some advantages, the three-phase advantage includes distribution of CH which is energy consuming, aggregation of sensed data by the CH, and finally, TDMA allows LEACH to put sensor nodes in sleep mode to preserve the energy. The disadvantage of this protocol includes randomization of CH which ignores the energy residual of the CH and other nodes within the cluster resulting in an uneven distribution of CHs [159-161].

5.4.2 TEEN

TEEN is mostly utilized in the time-dependent application, hence, a threshold-based hierarchical routing protocol (like LEACH). This includes explosive detection systems, intruder detection etc. Sensor nodes in a network, with consideration to the TEEN protocol, function based on their energy levels or residual. The CH does not have a direct connection to the base station (BS) but instead employs a hierarchy to transport energy from a lower level

CH to a higher level CH and then to the BS. The TEEN protocol uses two types of functionality after processing of clustering:

1) Hard threshold has the sole purpose to allow the sink nodes to forward data strictly to the nodes that are in their range of interest to reduce the amount of transmission.
2) Soft threshold ensures reduction of transmission quantity through a comparison of variation in sensing attributes with relation to overcoming transmission. This means that if the threshold is not met or reached, the nodes will fail to communicate.

The main contribution of this protocol is the fact that it reduces energy consumption effectively through its main objective of data transmission which is the main contributory factor towards energy consumption [154, 155, 162]

5.4.3 DEEC

DEEC uses the functionality of energy residual to take advantage of a dynamic and probable CH selection process. Epoch rotation is used in this protocol to ensure extension in network lifetime through initial energy residual which increases the CH selection probability. The probability threshold for a certain sink node selected to be CH in each round is depicted as in Equation 5.9:

$$T(Si) = \begin{cases} \frac{P_i}{1 - P_i \left| r \bmod \left(\frac{1}{P_i}\right)\right|}, & if\ S_i \in G \\ 0, & Otherwise \end{cases} \tag{5.9}$$

The set of sink nodes can potentially be selected for being CHs considering at round r. Furthermore, a weighted election probability process for a 2-level heterogeneous sensor network is normal SN and advanced SN which are formulated following Equations 5.10 and 5.11 [154, 163]:

$$P_{adv} = \frac{P_{opt}}{1 + \alpha.m} \tag{5.10}$$

$$P_{nrm} = \frac{P_{opt}(1 + \alpha)}{1 + \alpha.m} \tag{5.11}$$

Therefore, considering the probability threshold, by substituting $T(Si)$ to P_i with *E(r)* as a factor.

$$\boldsymbol{P_i} = \begin{cases} \dfrac{P_{opt}E_i(r)}{(1+\ \alpha m)\bar{E}(r)}, & if\ S_i\ is\ the\ normal\ node \\ \dfrac{P_{opt}(1+\ \alpha)E_i(r)}{(1+\ \alpha m)\bar{E}(r)}, & if\ S_i is\ the\ advanced\ node \end{cases} \quad (5.12)$$

The E(r) is defined as the average energy at round r and is illustrated as follows:

$$\bar{E}\ (r) = \frac{1}{N}\sum_{i=1}^{N} E_i(r) \quad (5.13)$$

5.5 Methodology

This section presents the methodology employed in performing quantitative comparative analysis to evaluate the effectiveness of each protocol based on their performances. Results presented graphically were used as the basis to choose an efficient protocol. The protocols were evaluated or measured using the number of iterations concerning nodes with depleted energy or referred to as the number of dead nodes.

The performance evaluation was performed in Windows 10 Intel(R) Core(TM) i3-4005U CPU 1.70 GHz with 4 Gb ram. MATLAB R2020b software tool was used to perform and present the results obtained, the data used includes nodes located randomly in a topology (v, e). About 100 nodes with 10000 iterations for all protocols were employed to eliminate inconsistency or redundancy in the results. All the relevant parameters that have been used in this evaluation are seen in Table 5.1.

All routing protocols considered in this study were executed individually. After all the executions had been done, all the plotted graphs were obtained and presented. Each plot is explained and the numerical data is analysed individually. The numerical data is graphically represented by the graphs seen in Figures 5.1 to Figure 5.4. Table 5.4 presents the quantitative results obtained from the experiment of the evaluated protocols. This is a useful data gathering method, especially when considering experimentation. The results provide significant insight into the performance of all the respective protocols with the sole purpose of supporting our choice of protocol to achieve the second objective of our proposed efficient energy-aware CPP algorithm following the first part already fulfilled in Chapter 4. In the next section, an in-depth data analysis of this protocol evaluation is provided.

Table 5.1 Evaluation Parameters

Parameters	Value
Routing Protocol(s)	LEACH, DEEC, TEEN
Number of nodes	100
Number of rounds	2 000 – 9 000
Simulation time	[8 , 23] seconds
Area	100m * 100m
Initial node Energy	0.5 j

5.5.1 Performance Evaluation and Results

This section compares the outcomes of the three methods: LEACH, DEEC and TEEN using hierarchical-based WSN routing protocols that are energy efficient. The results are presented in Figures 5.1 to Figure 5.4. The graphs show the performance of the respective protocols shown according to the dead nodes concerning the number of iterations. We used quantitative evidence to support our choice in the protocol and they explain the residual energy of a sensor node in a network area after a certain period of iterations.

Firstly, the evaluation of LEACH shows a rapid or steep trend line by an increase in iterations with the increase of dead nodes. The increase in the number of dead nodes can be seen starting from 687.791572891360 iterations. DEEC and TEEN are more efficient as the respective number of dead nodes starts to be seen at *1271.85001378550* and *3826.77165354331* which is much more efficient compared to LEACH whereby the number of dead nodes starts increasing much earlier. TEEN is regarded as the most efficient, not only because it starts losing sensor nodes sooner than LEACH and DEEC but the increase of dead nodes is less rapid with an interval of [*3826.77165354331, 7263.77952755906*] which shows the increase in the lifetime of the network by a difference of approximately *3437.00787401575*. In comparison, LEACH and DEEC amount to differences of *698.40370828551* and *1845.87813620071*. These numbers show the efficiency of the TEEN protocol as the difference in the iteration of the first dead node until the last dead node suggests a prolonging network lifetime.

This analysis signifies that TEEN supports our choice of energy-efficient routing protocol. This was necessary to evaluate protocols under this calibre to avoid bias.

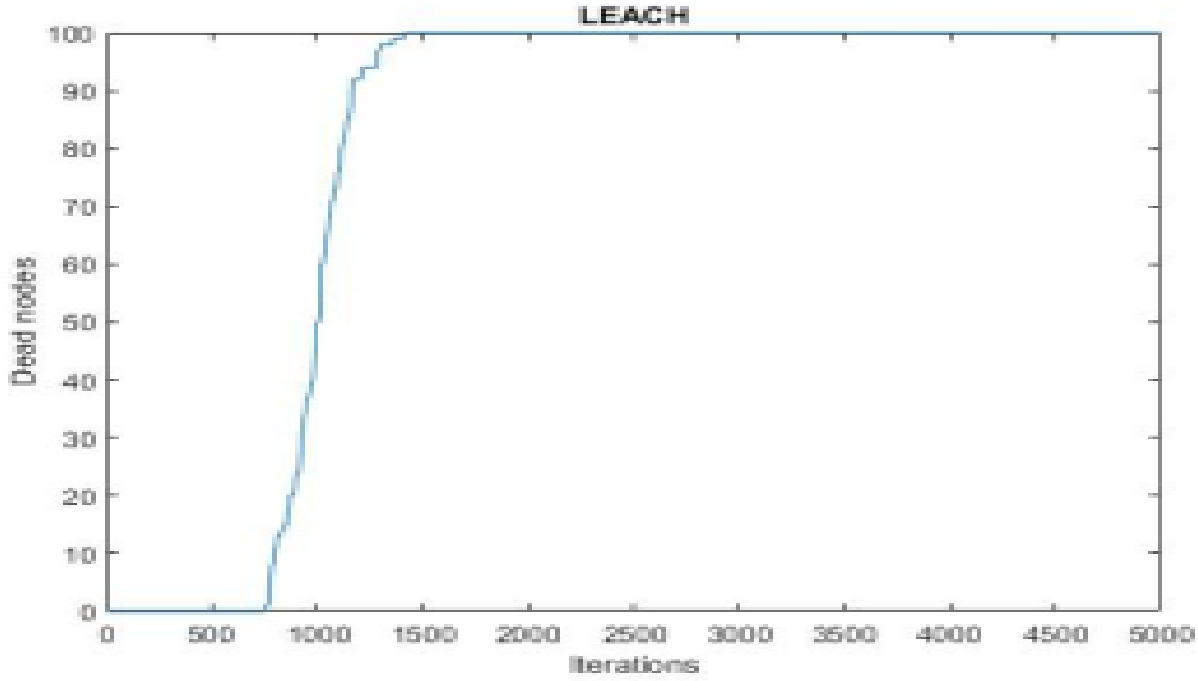

Figure 5.1: Dead Nodes Vs Number of iterations with LEACH protocol

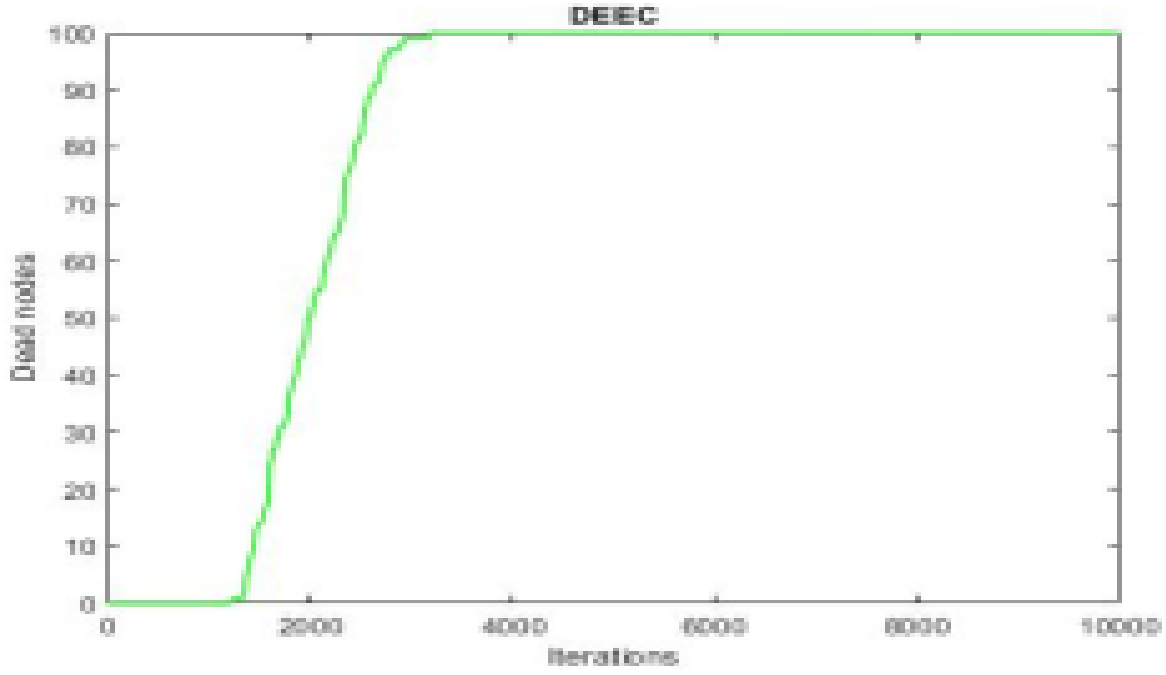

Figure 5.2: Dead Nodes Vs Number of iterations with DEEC protocol

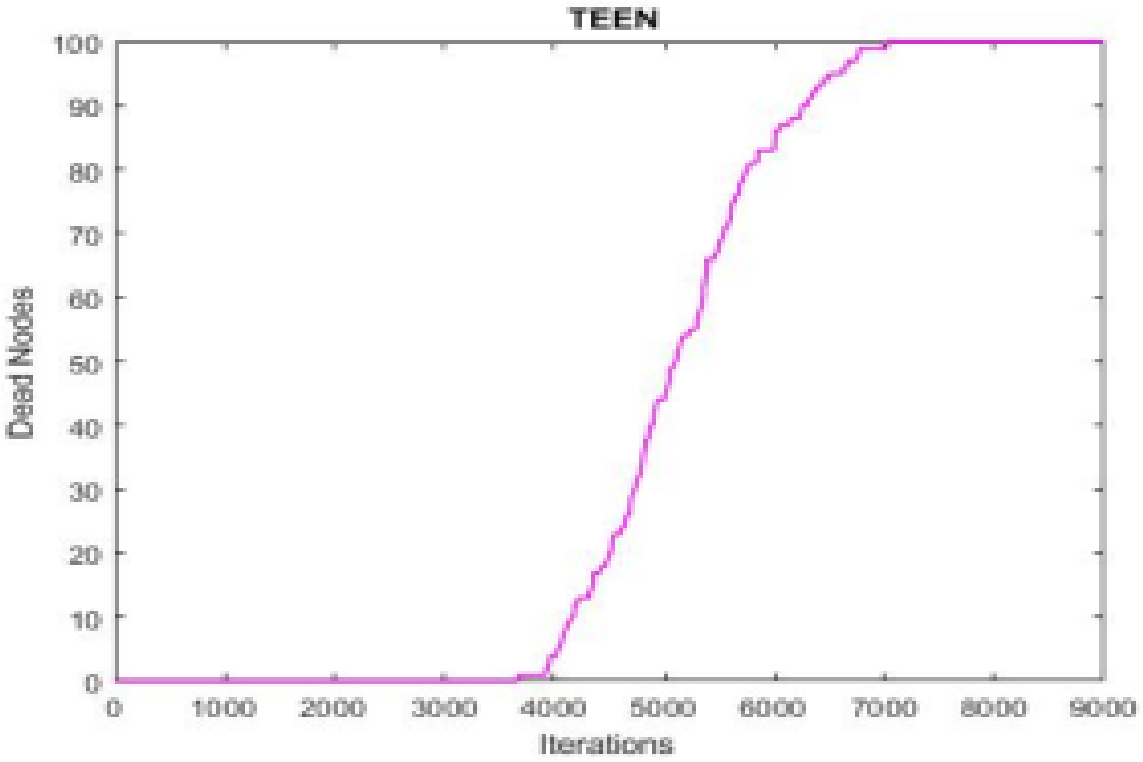

Figure 5.3: Dead Nodes Vs Number of iterations with TEEN protocol

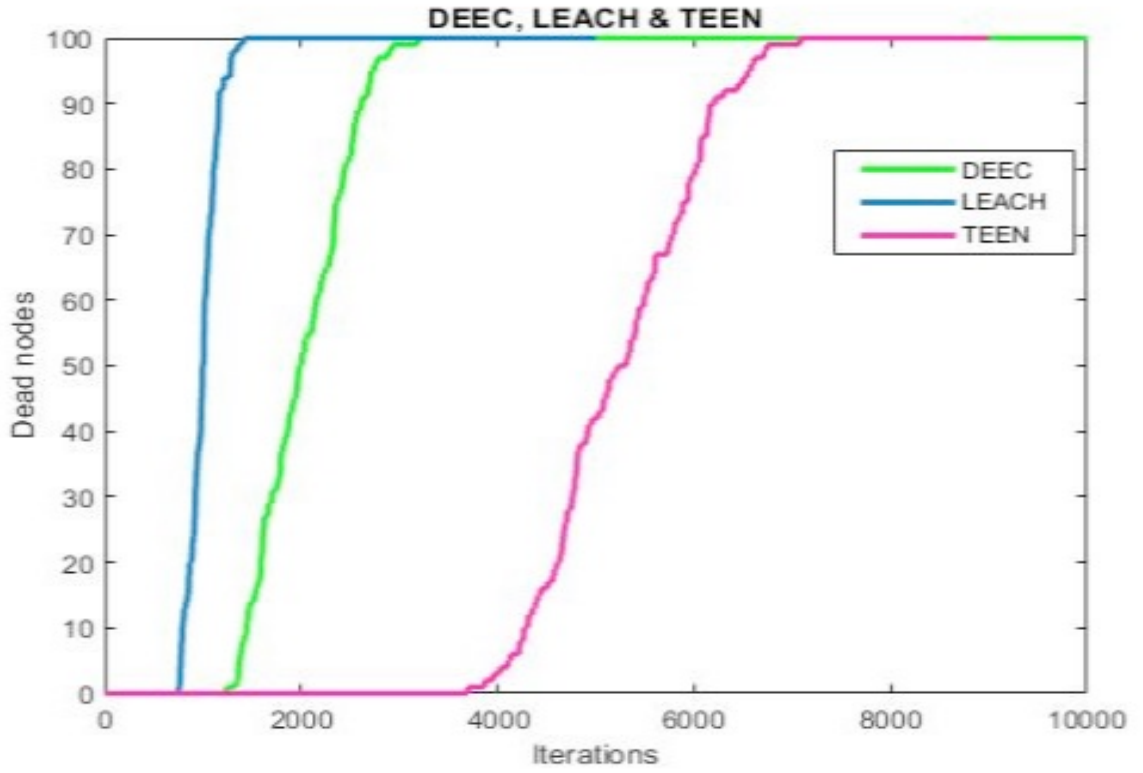

Figure 5.4: Comparative Dead Nodes Vs Number of iterations with LEACH, TEEN and DEEC Protocol

Figure 5.1 to 5.3 shows results for LEACH, DEEC and TEEN protocol respectively Figure 5.4 explores the comparison of all the algorithms. The results show the dead nodes concerning the number of iterations and present the TEEN protocol as efficient.

5.6 Discussion

This research aimed at constructing, not only an optimal controller placement algorithm but one which is efficient and energy-aware. Therefore, this chapter has addressed the energy-aware factor of the aim, additionally, it fully satisfies the RO2 by adopting an energy-efficient technique. However, many WSN energy-efficient techniques can be exploited, including routing protocols, MAC, data aggregation and more. Since data transmission is one of the leading elements that heavily influence energy consumption as asserted in the introduction of this section, we have adopted a routing technique. Routing performance on alternative routes during data transmission reduces the energy consumption of sensor nodes. Furthermore, clustering is another technique that adopts a data aggregation, meaning data is converged or is transmitted to a single point before being transmitted to a master node or base station, hence, we compared cluster-based routing protocols. TEEN, DEEC and LEACH are the three that use hierarchical clustering thus, comparing the best out of the three to answer RQ2.2 satisfying the aim of this research.

This section presents a discussion of the results obtained from the comparative analysis of three energy-efficient protocols shown in Figures 5.1 to Figure 5.4. The test was carried out with nodes that were put at random on an ***nxm*** Cartesian plane. Considering other protocols give responses after a greater number of iterations, the nodes were set to 100 and the iteration for all protocols was set to 10 000.

Figure 5.4 shows the comparison between all protocols and we can see that TEEN performs much more efficiently in comparison to other procedures when considering the number of nodes that start to deplete their energy depending on time or in this case iterations. The TEEN protocol shows a response after the >3900 iterations where the nodes start losing energy. The protocols produce good results, but the TEEN protocol outperformed them all since they were all constrained by the same goal, which was the energy consumption performance metric $E_t\ (l, d)$ which is explained above.

5.7 Chapter Summary

This chapter initially introduced a series of objectives and a brief background to achieving energy efficiency. In the evaluation of the protocols, an energy-aware approach to reduce the

energy consumption of the sensor nodes is illustrated using minimization of the objective function i.e. energy consumption. The chapter further investigated energy-aware WSN routing protocols between hierarchical-based protocols - DEEC, TEEN and LEACH. The results analysed and discussed showed that the TEEN protocol was efficient among all the protocols as shown in Table 5.4.

Table 5.4 LEACH, DEC, TEEN Quantitative Results

Energy-Efficient Routing Protocols					
LEACH		**DEEC**		**TEEN**	
Iterations	**No. of Dead Nodes**	**Iterations**	**No. of Dead Nodes**	**Iterations**	**No. of Dead Nodes**
126.987912300836	0	307.637165701682	0	513.779527559056	0
455.069828447081	0	533.278191342707	0	779.527559055119	0
592.550638665732	0	820.457678522194	0	974.409448818898	0
687.791572891360	0.366655239741852	1107.63716570168	0	2781.49606299213	0
762.106330019525	1.90779962142101	1271.85001378550	0.537634408602179	3082.67716535433	0
762.613089293965	6.69667476487861	1374.68982630273	1.88172043010754	3366.14173228347	0
805.329915192345	10.3706795045683	1375.07582023711	3.76344086021507	3614.17322834646	0
848.136169197979	14.8897798578093	1416.48745519713	5.64516129032259	3826.77165354331	0
880.360097178543	19.4058992741419	1437.49655362559	8.06451612903227	3933.07086614173	1.99501246882794
902.031508503123	24.2007362914164	1458.50565205404	10.4838709677419	4003.93700787402	3.49127182044890
923.792347934956	29.8406689222423	1479.51475048249	12.9032258064516	4039.37007874016	5.48628428927683
934.851623865381	34.3508264647579	1541.49434794596	15.0537634408602	4092.51968503937	7.48129675810477
966.896695631437	37.1767546539878	1583.12655086849	18.0107526881721	4163.38582677166	10.9725685785536
988.627725694186	42.5349887469632	1604.68706920320	23.1182795698925	4305.11811023622	12.9675810473816
999.508145410103	45.3549550623761	1646.48469809760	26.8817204301075	4358.26771653544	16.4588528678304
1010.47799323327	49.0200169913404	1770.55417700579	31.7204301075269	4464.56692913386	18.9526184538654
1010.86551503137	52.6820979833962	1833.08519437552	36.5591397849463	4517.71653543307	22.1945137157108
1011.28284619856	56.6258775133024	1895.61621174524	41.3978494623656	4659.44881889764	25.9351620947631
1022.13345654539	59.1641452908649	1978.60490763717	45.9677419354839	4694.88188976378	29.9251870324190
1033.07349499948	62.5475086819787	2021.00909842845	52.6881720430108	4801.18110236221	35.6608478802993

1054.68528758589	66.7789486235524	2124.62089881445	57.7956989247312	4872.04724409449	40.3990024937656
1055.13242812216	71.0044266913091	2186.87620623104	61.2903225806452	5828.74015748032	82.2942643391522
1108.64024562920	76.6533021328604	2310.94568513923	66.1290322580645	5970.47244094488	84.2892768079801
1130.31165695378	81.4481391501349	2373.47670250896	70.9677419354839	6094.48818897638	87.0324189526185
1141.19207666970	84.2681054655478	2374.41411634960	75.5376344086022	6253.93700787402	89.7755610972569
1162.74425051794	87.9361483314206	2457.23738626964	79.3010752688172	6448.81889763780	93.7655860349128
1184.32623373526	91.8858897351438	2498.81444720154	81.9892473118280	6590.55118110237	95.7605985037407
1226.83439405005	93.5880047098803	2540.66721808657	86.0215053763441	6750.00000000000	98.2543640897756
1279.86526165174	94.7297035458244	2623.54562999724	90.0537634408602	6944.88188976378	99.7506234413966
1301.20876991638	96.4258566467441	2706.14833195478	92.7419354838710	7263.77952755906	100.249376558604
1311.96995215596	98.1190288107552	2768.29335539013	95.6989247311828	7529.52755905512	100.249376558604
1343.80635833843	98.9730672350320	2891.75627240143	97.5806451612903	7866.14173228347	100.249376558604
1386.19528117687	99.5483880583667	2974.08326440584	98.9247311827957	8167.32283464567	100.249376558604
1439.16653004039	100	3117.72814998621	99.1935483870968	8521.65354330709	100.249376558604
1926.04295530085	100	3343.53460159912	100	8805.11811023622	100.249376558604
2000	100	4000.00000000000	100	8964.56692913386	100.000000000000

Chapter 6

Implementation and Results

6.1 Overview

The implementation and results of the energy-aware CPP algorithm discussed in Chapters 4 and 5 are presented in this Chapter. The design and description of the algorithm are presented first, followed by an introduction. The simulation setup and qualitative evidence are presented progressively. Finally, we present a summary of the chapter and conclude with an analysis and overall discussion based on evaluation use cases produced from the provided data.

6.2 Introduction

This research adopted a constructive methodology which is explained more in Chapter 3. This chapter introduced the implementation and evaluation phase presented in Figure 3.4. In section 3.7 of Chapter 3, we explained how data were collected and made reference to this chapter following the collection and analysis of the data as evidence. Following the evaluation of our suggested algorithm, this chapter provides quantitative evidence. The tools that we have chosen are simulation instruments and numerical data is, therefore, our primary evidence.

This chapter is key in achieving RO3 as the final piece of the research, thus, achieving our research goal. Furthermore, the instruments used for data collection have their respective drawback concerning hardware and processing power limitation. However, to ensure that we meet the verification and/or validity of the outcomes, regardless of instrumentation limitations, the constructive descriptive evaluation method in section 3.8.3 of Chapter 3 was adopted. By using different network scenarios and theoretical analysis of the existing knowledge base, the validity of our outcomes has been accomplished.

Following existing knowledge, researchers have been able to acquire quantitative results through simulation. However, data plane protocols in asserting sensor node information to the controller have shown as a drawback as OpenFlow has only been utilized in wired and not wireless networks. Furthermore, the use of a centralized single-running algorithm has shown to be a scarce field of contribution as most SDWSN implementations are framework based. In this chapter, we make use of one of the frameworks, SDN-WISE [164].

By adopting an existing energy-aware framework, we focused on exploiting one of the features that served as beneficiaries of our outcomes, a programmable sensor. This type of sensor is

described further in a later stage; however, this sensor is designed in a manner that it acts as a controller whilst remaining as a sensor node. Our contribution is focused on using a single running algorithm in a centralized programmable sensor node to reduce the amount of energy consumed and efficiency in the placement of controllers.

6.3 Methodology

This section is a summary of the methodology following the research objectives to achieve a design of the algorithm and architecture. A brief problem definition that is drawn from Chapter 1 is explained, following ROs that have been met and explaining how this chapter will be concluding the research goal.

6.3.1 Problem Definition

A centralized controller is proven to be ineffective since it is vulnerable to a single point of failure, which can paralyze a network [22, 23]. To accomplish an optimal controller placement without distorting some network parameters, multiple controllers might be used i.e., propagation latency. Moreover, CPP has fewer solutions developed specifically for SDWSN thus, an energy-aware CPP is imperative. The energy efficiency drawbacks of WSN are adopted by SDWSN, therefore, an energy-aware design is included to make it a single approach of energy-aware CPP in SDWSN [26]. To achieve the research goal, all ROs were attained.

6.3.2 Research Objective

In the above-mentioned problem definition, RO1 and RO2 have been achieved in Chapters 1 to Chapter 5 through the use of a DSR methodology. In this chapter, we give reference to the methodology in Chapter 3 and implement it under the representation of flowcharts. Additionally, the RO3 explains evaluation to highlight the effectiveness of the proposed approach by using existing knowledge including the adopted SDN-WISE architecture.

6.3.3 SD-WSN architecture

In this research, we adopted frameworks that make it possible for multiple controllers to exist and be connected within a WSN however, to run independently on their respective networks. For this to be possible, a module called CC2530 which allows a sink node to act as an SDN controller i.e., an SDN-enabled sink node is considered an assumption in this research. The module was used and adopted in other past research including SDN-WISE and IT-SDN frameworks. Following our proposed algorithm, contribution in accord, Figure 6.1 represents SDWSN architecture that was considered in the setup of our simulation following the validation method through the use of different network conditions or in this instance, scenarios

[165]. The formulation of the architecture used in this research is shown in section 6.4.3.1 which describes the applicability. Two objectives that make SDWSN such an efficient architecture to be adopted include the reduction of exchange in information between sensor nodes and network controllers and enabling a normal sensor node to perform functions that can be supported by a controller. By using the proposed controller placement cluster-based meta-heuristic, the SDN-enabled sink nodes were assigned as centroids of each cluster, thus, designated controllers of each cluster. Since SDWSN suffer from topology discovery drawbacks, the SDN-enabled sink node has a dedicated module to manage and store the state of the topology including i.e. CC2530 [164]. Additionally, since a local controller i.e. SDN-enabled sink node was designated a single cluster, and an energy-efficient routing protocol deployed on a local level compared to executing it at the master controller, the protocol performed its CH mechanisms as mentioned in section 5.4.2 of Chapter 5.

The architecture of the SDN-enabled sink node presents functionalities that served importance in the setup i.e., core functions representing topology discovery and protocol responsible for managing energy consumption. This networking component under core functions is key to enabling the proposed CPP algorithm in achieving a local network with a dedicated controller that will serve its network. Thus, a local controller can delegate CH using our energy-aware protocol, promising an opportunity for larger networks or a wide area network under SDWSN [164, 165].

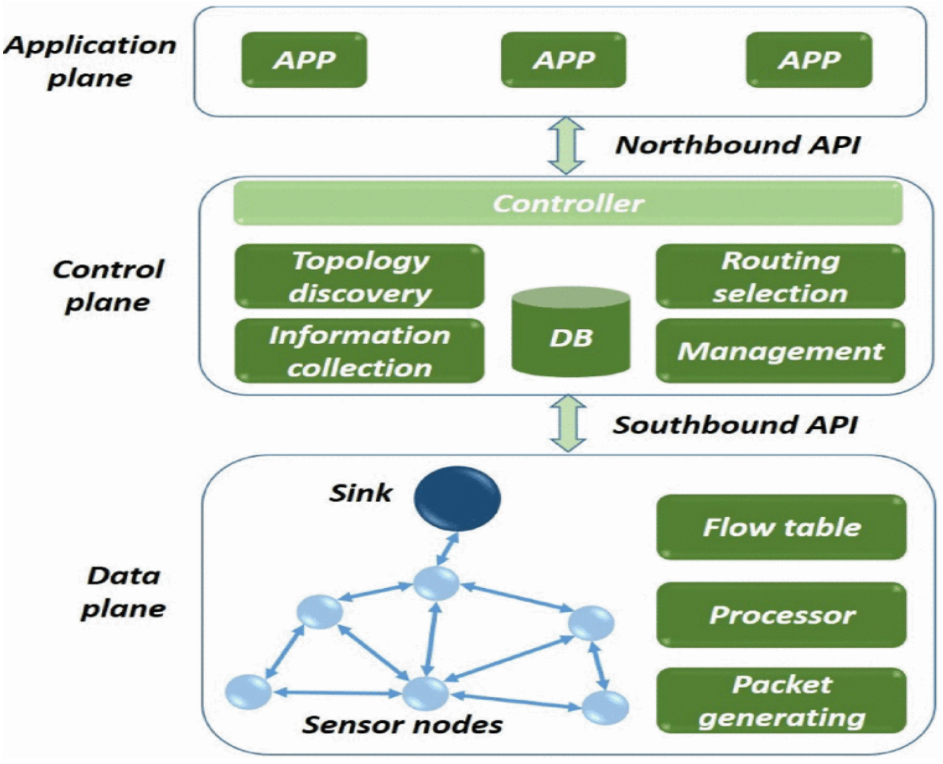

Figure 6.1: SDWSN architecture [166]

6.4 Energy-aware CPP

The designs of the two algorithms are shown in this section and a summary of the entire algorithm's results. The design of the adopted controller as well as the architecture used to generate the setup for generating the quantitative evidence in accordance to achieve an efficient energy-aware CPP construct is presented.

6.4.1 Phase I: Proposed Efficient CPP algorithm

The performance objectives considered in this algorithm are studied in Chapter 2, Chapter 4 and Chapter 5. We refer to all these objectives as the algorithm description and have been asserted in phases progressively below. The controller plays a critical part in our data collection goal with the simulation setup in accord. Furthermore, section 6.4.1.1 explains the latency - Cost aware CPP algorithm given its flow chart and pseudocode in how it was executed. In Table 4.1 we can see the Propagation latency represented mathematically.

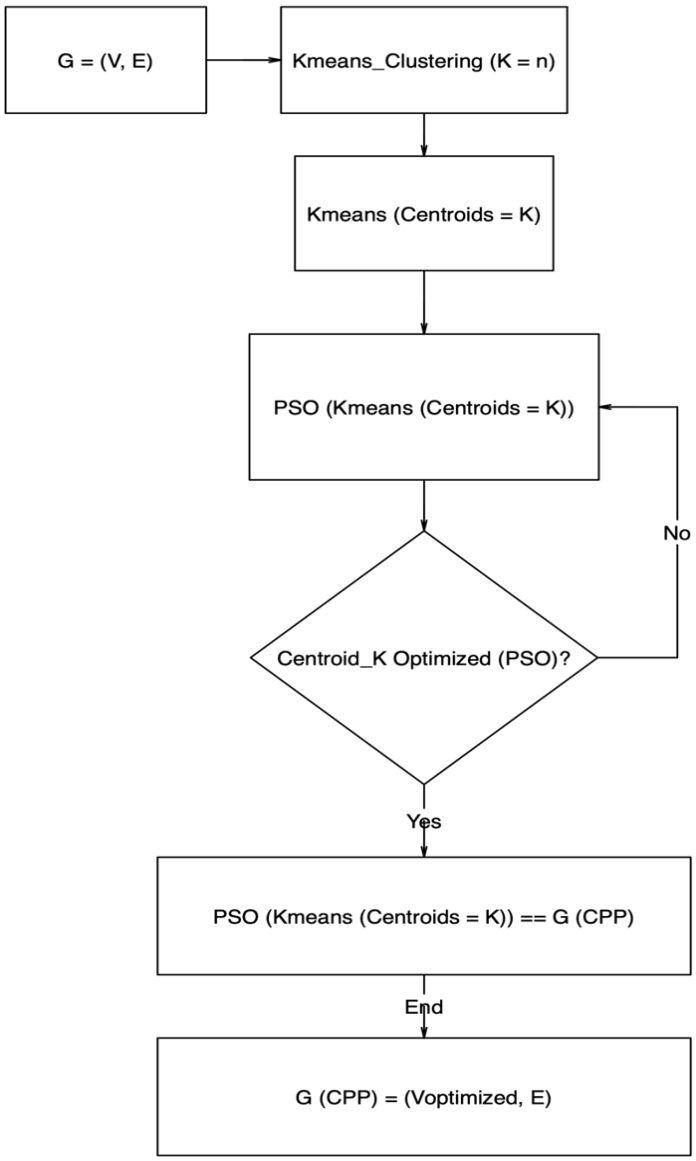

Figure 6.2: Proposed CPP Flowchart Algorithm

Efficient Controller Placement Algorithm

Input: G = (V, E)
Output: G (CPP) = ($V_{optimized}$, E)
Steps:

1. **Initialization**
2. PSO-kmeans (N_{SN} , $K_{clusters}$) // *hybrid function of PSO-kmeans*
   // *function accepts the number of nodes in the network topology and clusters of choice*
3. N_{SN} number of sensor nodes, $K_{clusters}$ number of clusters
4. **If** $K_{clusters}$ =! 0 **then**
   return $K_{clusters}$

   else
   PSO-kmeans (N_{SN} , $K_{clusters}$) // *execute/run function*
   return PSO-kmeans ($K_{centroids}$) // *output the centroids of each cluster*
5. $V_{optimized}$ = PSO-kmeans ($K_{centroids}$)
6. $G_{ControllerPlacement}$ = ($V_{optimized}$, E) // *New (x,y) topology of optimized placement location*
7. **end** // *termination of the algorithm (phase I)*

Figure 6.3: Phase I: Efficient CP Algorithm

Figure 6.2 and Figure 6.3 present the algorithmic processes in a flowchart and Pseudocode. This algorithm represents the first phase of the algorithm aimed at achieving Latency-Cost aware CPP. The algorithm starts with an input of an in-directed graph of sensor nodes within an n*n area network randomly placed. The nodes are clustered according to the proposed automatic PSO algorithm that is demonstrated in Figure 6.4, and parameters in Table 6.1 which are referenced from Chapter 4. The program divided the network into K clusters and determined the cluster's centre. The K value was assigned statically because it is an important part of data analysis and evaluation using different network conditions as shown in section 6.5. After clustering has been executed, the coordinates of these centres were returned as there are no controllers within the network initially. This is to consider the cost (CAPEX/OPEX) objective, by knowing the optimized location of a controller before being placed in a network minimizes CAPEX/OPEX. As asserted in Chapter 4, the algorithm used minimizes the propagation latency - Euclidean Distance.

PSO Algorithm
Input: Initialization$_{\textbf{parameters}}$, K $_{\textbf{clusters}}$, Data$_{\textbf{count}}$ **Function:** Particle Swarm Intelligence Clustering **Output:** Sub-set Clustering, K$_{\textbf{centroids}}$ **Steps:** 1. Initialization of random chromosome for n individuals in the population. 2. Assignment for fitness conditions to each of the respective n individuals in the program 3. Create offspring for the next generation through recombination from the current population 4. Mutate offspring for this generation. 5. Tournament selection is performed resulting in parent selection to create the next generation 6. The next population of *n* individuals is chosen 7. Set a new population to the current population 8. Assess the fitness of each offspring in the generation 9. If the stopping criterion is satisfied then halt the program and print solutions, else go to step 3.

Figure 6.4: PSO Algorithm [136]

Table 6.1 PSO Algorithm Parameters

Parameter	Value
Individual parameters ($c_1, c_2, r_1, r_2,\ w$)	(1, 1.2, 0.5, 0.5, 0.8)
Amount of particles	6
Initial social Influence ($s_1, s_2, s_3, s_4, s_5, s_6$)	(1.1, 1.05, 1.033, 1.025, 1.02, 1.017)
Initial personal Influence ($p_1, p_2, p_3, p_4, p_5, p_6$)	(3, 4, 5, 6, 7, 8)

6.4.1.1 CC2530 Node Design

The previous section shows the executable algorithm and steps taken for the latency-aware CPP objective to be achieved. The attention is drawn to the key functionality, similar to the description in Figure 6.3 of section 6.3.3, the topology manager (TM), functioning through a TCP/IP layer. This part was crucial as it highlighted the abstraction of the network logically, to ensure the management set of policies or applications delegated by the programmable sensor node or controller performed on all physical devices [164, 165].

Each controller is responsible for its network. This means that, if every cluster has a centroid assigned as a dedicated controller, then energy efficiency can be done for each cluster respectively. Thus, instead of preserving energy globally, the controller enables partitioning of the network and then executing the protocol under local controllers, therefore, saving much work being done by the master controller.

6.4.2 Phase II: Energy-Aware Routing Protocol

Following section 6.4.1, this section provides the second phase of the study which addresses energy efficiency. Similarly, we will provide a description or explanation of the algorithm as designed following Phase I being the input of the second Phase II explored in this section.

Figure 6.8 presents the Phase II approach thus, addressing energy consumption drawbacks encountered by sensor nodes being controlled by the set of K controllers i.e. sink nodes of Phase I. The algorithm accepts a set of K clusters from Phase I as input and then invokes a topology discovery (TD) protocol explained in the next section. The TD enables the local controller to execute an energy-efficient protocol on its cluster through the assignment and/or selection of Cluster Heads (CH).

The execution of the *TEEN* protocol allows an update of the dedicated CH in the cluster until the termination of *n* iterations. If a node of depleted energy is discovered in the cluster, data of the respective sensor node will be pushed to the current CH update to avoid data loss.

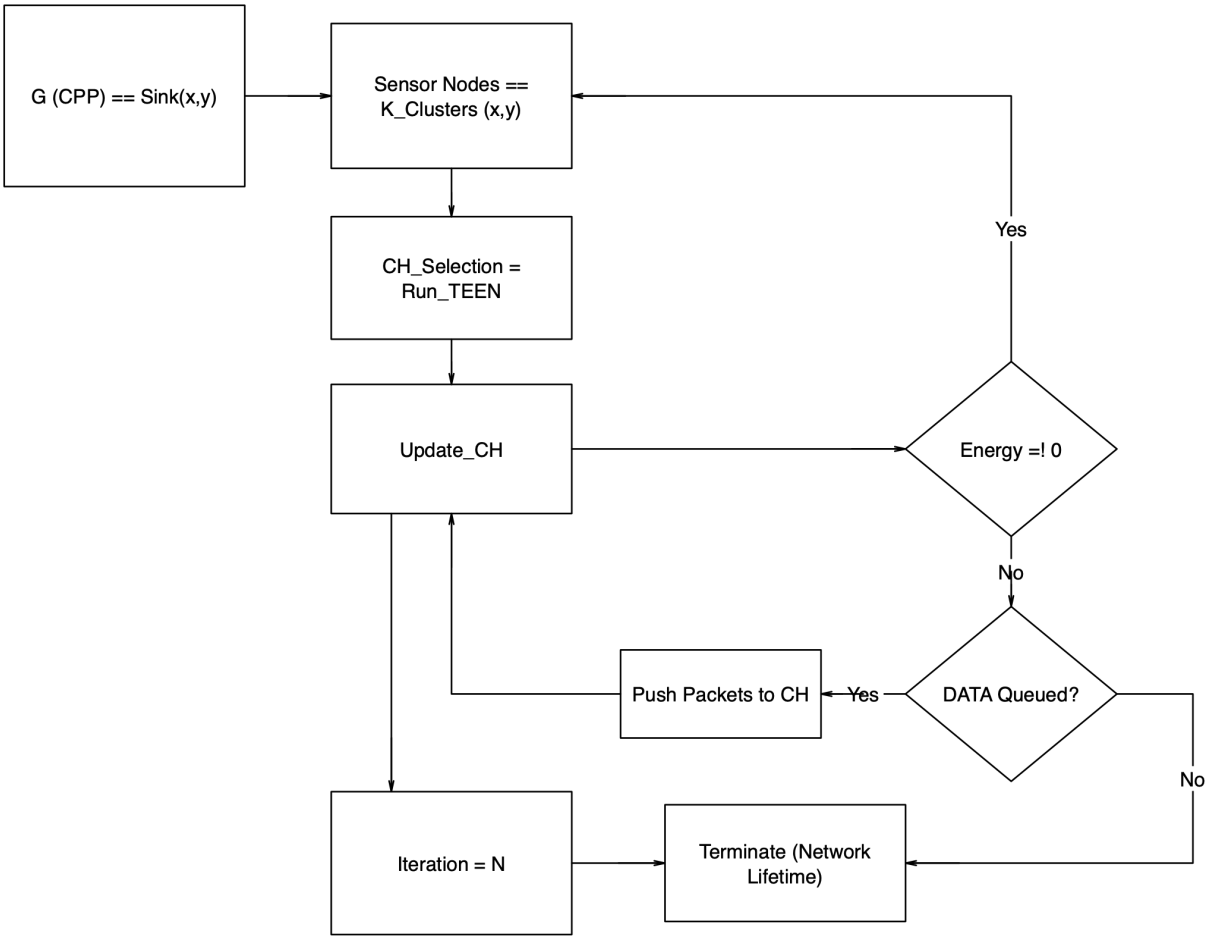

Figure 6.5: Proposed Energy-aware algorithmic process

Energy-aware Algorithm

Input: $G_{ControllerPlacement}$ = ($V_{optimized}$, E)
Output: Network Lifetime = $G_{AliveSensorNodes}$ (V, E)
Steps:

1. **Initialization**
2. TEEN ($n_{SenosrNodes}$, $Sink_{(x,y)}$) // *Routing Protocol function of Teen*
3. // *function accepts the number of sensor nodes of a cluster from Phase I and Sink node as the centroid(s) from Phase I*
4. **while** iteration =! 0 **do**
5. TEEN (*iteration = n*) // *teen routing protocol execution for n rounds*
   If *energy* (E_o) *= 0* then
   If *Data_Queued() == true* **then**
   return *push Data_Queued*(CH_{update})
   else
   terminate CH
   else
   insert (input ($G_{ControllerPlacement}$ = ($V_{optimized}$, E)))
   end
6. V = CH_{update} // *Number of alive nodes*
7. $G_{AliveSensorNodes}$ = (V, E)
8. **end** // *termination of the algorithm (phase II)*

Figure 6.6: Proposed Energy-aware algorithm

6.4.3 Algorithm Implementation: Phase I and Phase II

The initial step starts in Phase I with G (V, E) sensor nodes in a topology that are randomly placed. The nodes are partitioned into clusters concerning k centroids using a method called PSO clustering (illustrated in Figure 6.5). The program looks for data that needs to be sent to a specific node that is currently available. Therefore, this process has completed its output i.e. optimal latency aware-controller placements. TEEN ensure energy efficiency in every cluster used for calculating the CH, thus, the cluster head is updated to send information to the sink node. These steps will iterate for a certain time as a controlled variable for evaluation purposes with energy consumption information.

The topology will have to be updated and re-clustered using the initial step by using PSO when one of the dedicated sink nodes has depleted its energy. This work follows the SDN-WISE and IT-SDN frameworks to formulate our assumptions, we used python as the main programming language. We formulated an architecture in Figure 6.9 concerning the implementation of our

algorithm regarding SD-WSN architecture. Table 6.2 presents the elements present in the architecture.

Master Controller accepts G= (V, E), a set of V sensor nodes connected with a set of E wireless connections, and the Global controller executes PSO to find (x,y) coordinates for optimal controller placement location. The network is clustered into smaller networks and finding the centroid through the use of k-mans, therefore optimized using PSO.

PSO CLUSTERING allows Clustered Sensor Nodes to be dedicated as potential Controllers i.e., programmable sensor nodes, which will run *TEEN* individually for their respective cluster

TEEN is responsible for the CH selection of the sensor nodes from the k clusters and respective centroids. Therefore, the protocol will use a distributed approach as opposed to the central manner or traditional way in the WSN.

Table 6.2 Proposed Network Architecture Elements

Legend	Name
	Master/ Global Controller
	Local controller/ SDN-enabled Sink
	Clustered WSN
	(x,y) Coordinate of Local controller/ SDN-enabled Sink
	Direct Connections
	Inter-connections

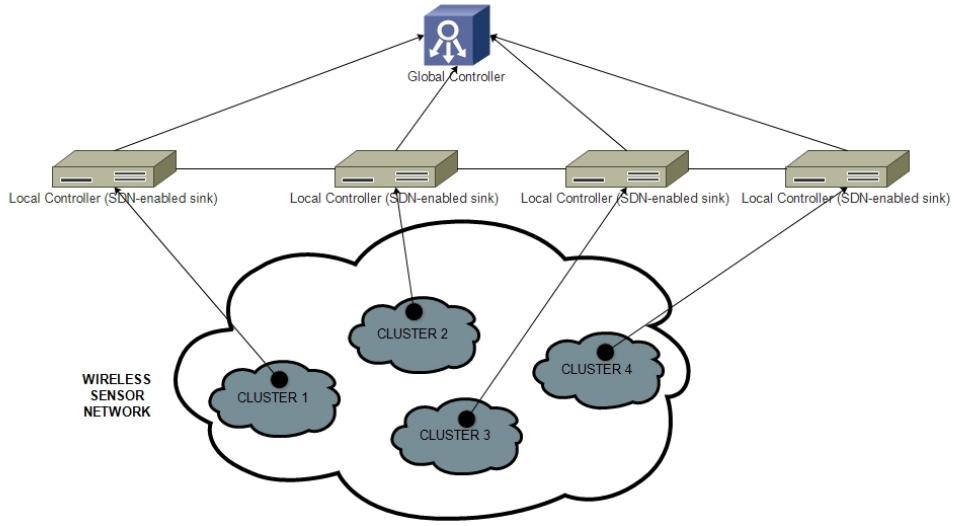

Figure 6.7: Proposed Network architecture

6.5 Experimentation and Evaluation

This section highlights the results, discussions and the simulation setup guided in acquiring the quantitative evidence provided in section 6.5.3 about the experimental setup and procedure. This is imperative for an unambiguous comprehension of how the discussed segments of the algorithm are used in practice to formulate and/or gather the expected numerical evidence.

6.5.1 Instrumentation

The use of an SDN-WISE framework in this research adopts all tools and requirements. The framework utilizes a simulation tool described in section 6.5.1.2 which is built on ContikiOS; this is imperative for the use of an SDN-enabled sensor node. We will give a brief description of the tool concerning the ONOS controller, and how we acquired the results shown in section 6.5.3.1.

Benchmarking

Benchmarking is a process where a lightweight tool is embedded in an emulated controller to acquire information about the network connected to the controller. Therefore, since we proposed using the control plane through a central/master controller, it is important to indicate how the data is extracted from the control plane. However, these results were alluded to by the simulator tool.

We have indicated that the adopted framework makes use of an SDN-enabled sink node which enables a gateway between the control plane (Master Controller, MC) and the data plane (sensor nodes). However, the SDN-enabled sink node is not just a mere intermediary between two planes, it played an essential role in acting as the benchmarking resource. Thus, the master controller provided an interface and control of the local controller (LC) sink nodes. Therefore, our evidence was collected using the interface provided by the simulation instrument in section 6.5.1.2.

6.5.2 Simulation Setup and Procedure

This section describes the setting used to carry out the suggested algorithm's implementation and an explanation of the algorithm through the evaluation considered to test the validity of the construct.

Table 6.3: System settings

Domains	Value
Software	Mac-MacBook Air operating system version 10.1
	Python: Jupyter Notebook, Matlab and Spyder IDE
	SD-WSN patch
	Matplotlib, NetworkX, Pandas and Numpy
Hardware	245.11 GB Hard Disk
	8 G RAM
	Processor: Apple M1

Table 6.4: Evaluation metrics

Metric	Description
Average Energy	After the simulation, the network's average overall consumption of energy.
Residual energy	The total quantity of energy left in the network once the simulation is finished.
Propagation Latency (Delay)	The difference in data transmission sending and receiving times

Table 6.5: Simulation settings

Domains	Value
SDN approach	PSO algorithm K-Means clustering
WSN Protocol	TEEN protocol
Simulator	Python: Jupyter and NetworkX

	Matlab (python kernel)
Patch	SD-WSN
Simulation Area	100*100 meters
Simulation Time	2000-9000 rounds/iteration
Topology	Palmetto (The Internet Topology Zoo)
Per-node initial energy	100 J
The total number of local controllers (s)	1 - 3
Number of Sensor(s)	30-45

6.5.2.1 Evaluation Scenarios

We simulated and evaluated the performance of our approach based on several assumptions that apply for the use of the controller, sensor node firmware and topology discovery being already a drawback for SDWSN.

A. Assumptions

1) Several SDN controllers can be used including ONOS, however, there has to be an intermediary which is the Sink node, through a TCP/IP connection. We assume that our master controller in this case is the sink node.
2) To allow multiple controllers, different approaches can arise including multiple sinks or virtual controllers. In this case, we assumed the use of multiple sink nodes as used or proposed for SDN-WISE framework as SDN-enabled sink i.e. local controller. However, this can be done through the use of the module CC2530 which allows a WSN node to adopt SDN attributes.
3) Our final assumption is the topology discovery. Since we designed our network, this assumption can hold as we can get much of the network information through the pre-defined function of our simulation tool.

B. Scenarios

The number of K clusters and nodes inside a network topology has a significant impact on the algorithm. The distribution or topology type is left to be randomly placed. We evaluated our algorithm using the number of K clusters about the number of sensor nodes:

Case I: K=2 with a network of 100 and 150

Case II: K=3 with a network of 100 and 150

Case III: K=4 with a network of 100 and 150

These can affect the energy as well as the cost of the network. Using multiple sinks to increase energy and/or network performance can lead to increase CAPEX/OPEX.

6.5.2.2 Simulation Procedure

The simulation process adopted the network parameters seen in Table 6.3. The first step is to get the number of clusters or aggregate the network before placing our local controllers, this helped in knowing beforehand as a measure to minimizing cost. PSO is the proposed method for clustering using K-means to get the centroids. The centroids were the geo-coordinates where our local controllers can be possibly placed, thus, every controller is responsible for their cluster. Now that we have got the placement location for the controllers we needed the number, this was left for evaluation purposes. A distributed routing method is proposed. In comparison to the traditional WSN routing process whereby a sink node is the main master of the network for routing protocols, we used our local controllers to execute the TEEN protocol for their respective clusters. The local controllers communicated directly with the sink as the master controller, thus, communicating with each other.

6.6 Results and Discussions

The outcomes of the simulations are presented in this section. Moreover, we provide an analysis based on the evaluation process to validate this work as described in section 6.5.2.1.

As asserted in Chapter 2 section 2.6.1, when considering CPP, there are a few questions to answer following network performance efficiency:

a) The number of controllers,
b) location and,
c) sensor assignments are the three questions or factors considered for efficient CPP.

Minimization of propagation latency (average and worst-case) and CAPEX/OPEX are considered to address these 3 factors. These objectives are explained in section 2.6.2 of Chapter 2. We proposed a PSO algorithm by minimizing the distance between the sensors to partition the network into sub-clusters i.e. reduction of delay or propagation latency between sensors. Multiple local controllers were deployed for these sub-clusters, k-means++ exploits this with the number of controllers being predefined to reduce CAPEX/OPEX. K-means++ minimizes the distance between sensors of a cluster to find the centre being the most efficient point to place the controller i.e. worst-case propagation latency. Assignment of sensors to respective

controllers was done by PSO as the initial clustering method, each cluster reports to a centroid or local controller. Now that clustering was the mechanism used to tackle latency, there was only one controller for each cluster, even though it is on a local network, however, it is prone to failure collapsing the sub-cluster resulting in a less reliable approach.

Following a more reliable method without sacrificing other objectives, backup local controllers are to be available when one fails. In the absence of the failing controller, re-clustering is an iterative strategy within the PSO K-means++ to reassign sensors and controllers to ensure that the initially available controllers can efficiently manage the resources.

Since these sensor nodes are susceptible to a low battery life span, we proposed a routing protocol that runs on a distributed network. Traditional WSN makes use of the sink node or base station as a global master to acquire all the information about the network, therefore, being the network manager. Using SDN as an advantage through the deployment of local controllers on sub-clusters, the proposed TEEN energy-efficient protocol can run on a local scale or sub-clusters making it a distributed method. This further reduces data transmission when considering its use on a traditional WSN i.e. global scale, therefore, energy efficiency is achieved.

6.6.1 Performance Results and Analysis: Energy efficiency

The quantitative evidence is illustrated graphically in the form of line graphs. The clustered wireless sensor network is seen in Figure 6.9 with multiple black-coloured centroids formed or generated using K-means++ clustering. These centroids are SDN-enabled sink nodes as explained in section 6.4.1.1 of this chapter, and have the ability of an OpenFlow switch or controller which served great importance in this phase as they are not normal sink nodes. They are connected to form an OpenFlow SDN network, allowing our proposed TEEN algorithm to function in a distributed manner. This topology of SDN-enabled sink nodes (OpenFlow SDN network), formed part of our second phase to find multiple controller placements in our network to improve performance.

For our proposed Distributed TEEN algorithm to be tested according to other existing methods, we considered extreme parameters in comparison to other methods. Our technique takes into account characteristics such as using additional nodes in the network and increasing the round time. This attribute puts the sensor nodes in more dyer conditions and hence are susceptible to energy consumption. The network conditions are described in Table 6.5 and Table 6.6 with the

various algorithms. These tables illustrate the comparison of different energy-efficient routing protocols including our proposed distributed TEEN. Tables 6.5 and 6.6 show the number of dead and alive nodes, respectively.

Figure 6.9 represents the graph that was adopted from an SDN network topology/dataset library called Topology Zoo. This dataset has been taken as input for our algorithm. However, WSN network topology libraries are limited, therefore we adopted this one and transformed it to be a WSN. Initially, there are sensors in Palmetto, meaning it is un-clustered as seen in Figure 6.9. But in this research, the distance between sensor nodes and their corresponding centroids/ centres was minimized using k-means/k-means++ clustering.

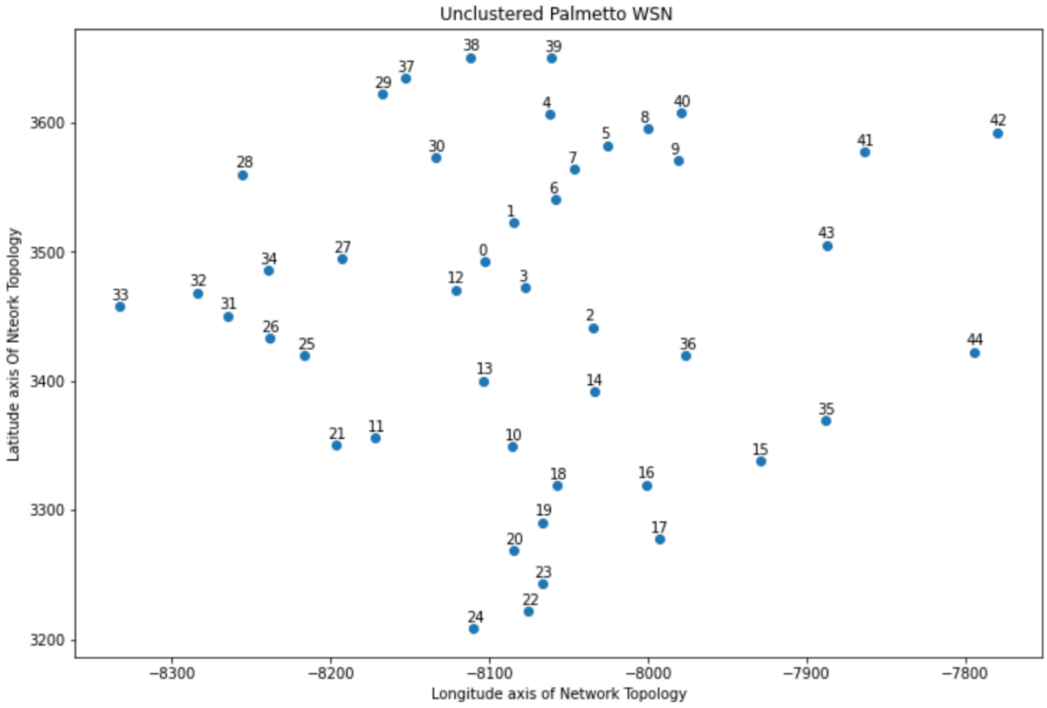

Figure 6.8: Palmetto network topology

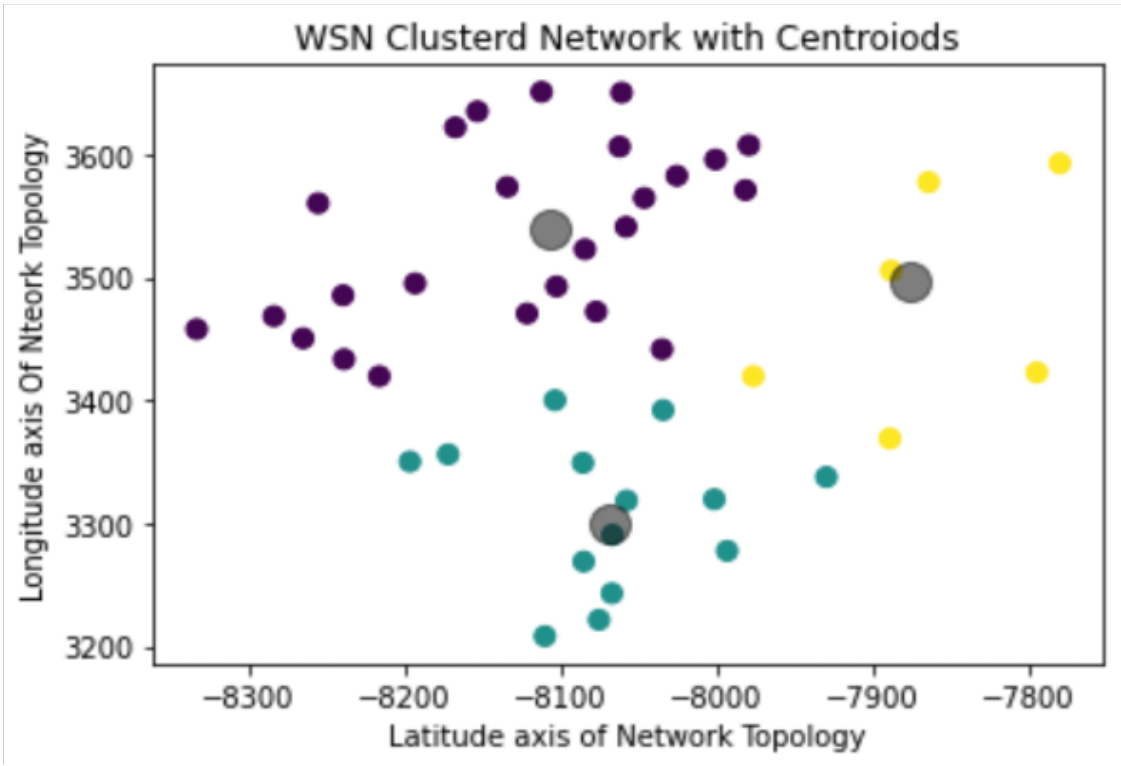

Figure 6.9: Clustered network topology

Table 6.4 SDN-enabled sink nodes coordinates and respective sensor nodes

	SDN-enabled sink nodes Coordinates	Sensor Nodes
Controller 1	[-8240.76346985, 3442.67053881]	25
Controller 2	[-8049.39337208, 3569.61436176]	14
Controller 3	[-8029.1636838 , 3323.74894768]	6

Table 6.5 The Number of Dead Nodes for each Algorithm

Algorithm	Iterations	Total nodes that have died	Number of Nodes in Total
LEACH	5000	30	30
DEEC	5000	30	30
TEEN	5000	18	30
ESSA	5000	21	30
PROPOSED Distributed TEEN (2 Clusters)	13000	43	45
PROPOSED Distributed TEEN (3 Clusters)	13000	14	45

Table 6.6 The Number of Survived/Alive Nodes for each Algorithm

Algorithm	Iterations	Total nodes that survived	Number of Nodes in Total
LEACH	5000	0	30
DEEC	5000	0	30
TEEN	5000	12	30
ESSA	5000	9	30
PROPOSED Distributed TEEN (2 Clusters)	13000	2	45
PROPOSED Distributed TEEN (3 Clusters)	13000	31	45

Figure(s) 10-11 illustrate our proposed distributed TEEN algorithm, LEACH, DEEC, ESSA and TEEN. We evaluated our proposed algorithm using 2 and 3 k clusters evaluation, however, take note that Figure 6.8 and Figure 6.9 are just for illustration purposes as they display 3 clusters.

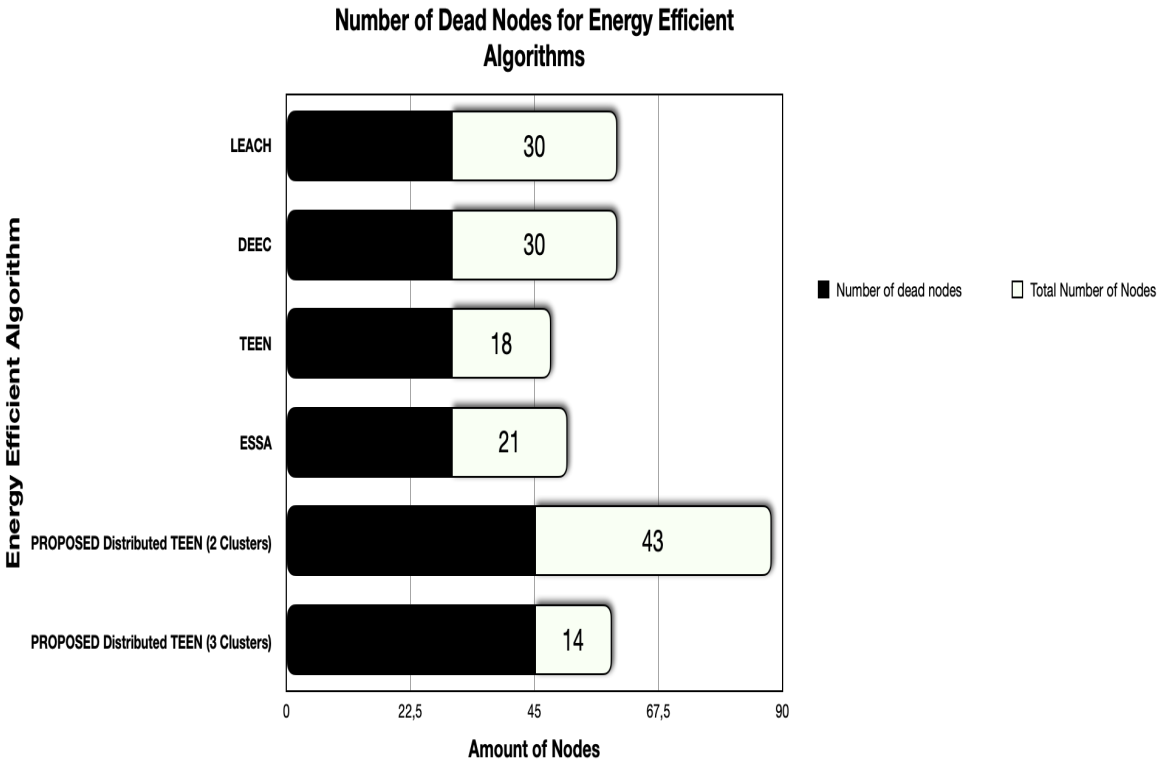

Figure 6.10: Number of Dead Nodes for Energy-efficient WSN routing protocols

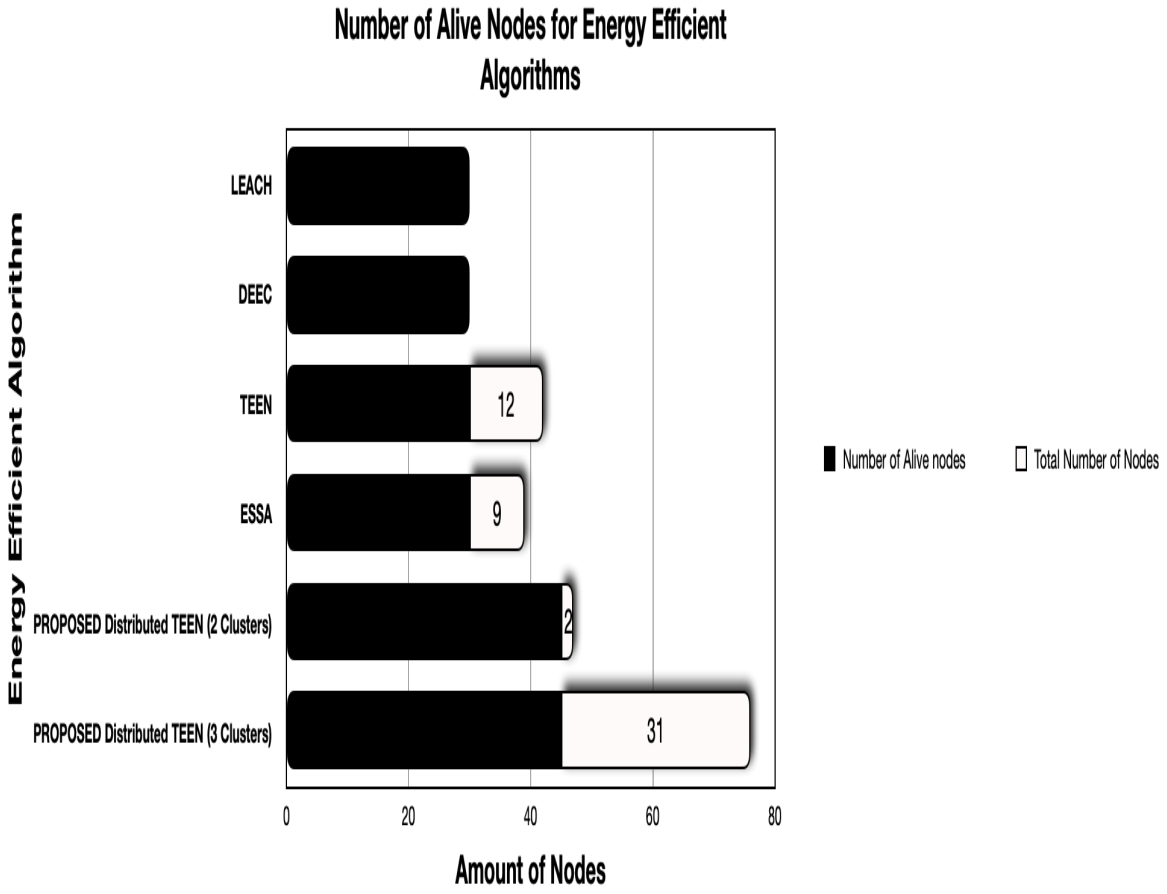

Figure 6.11: Number of Alive Nodes for Energy-efficient WSN routing protocols

Table 6.5 and Table 6.6 illustrate a comparison of our suggested algorithm utilizing the number of nodes and iterations shown in Figure 6.10. The number of dead nodes is the main energy efficiency objective measure we illustrate in this study. In addition, The number of live nodes is depicted in Figure 6.11. Our results show that our distributed TEEN algorithm does not perform well with two clusters. We further tested its efficiency using 3 clusters which shows to outperform all other algorithms. This can be seen as the graphic illustration of our results which shows the number of dead nodes is 14. This is the lowest amongst the other algorithms. We can see the number of alive nodes is 31, higher in comparison to every algorithm. This on average can assert that the proposed distributed TEEN prolongs the lifetime of a network.

Our proposed distributed TEEN only uses a subset of an SDN-enabled network because it's more than enough to evaluate and achieve our energy-aware objective of the first phase of our algorithm. In addition, we can visualize the SD-WSN on a much larger scale, making up an SDN with multiple SDN-enabled sink nodes which are connected to form a topology that can be extracted and utilized as input for our second phase. This topology is illustrated in Figure

6.12 with connected links. Furthermore, controller placement can be achieved using this topology.

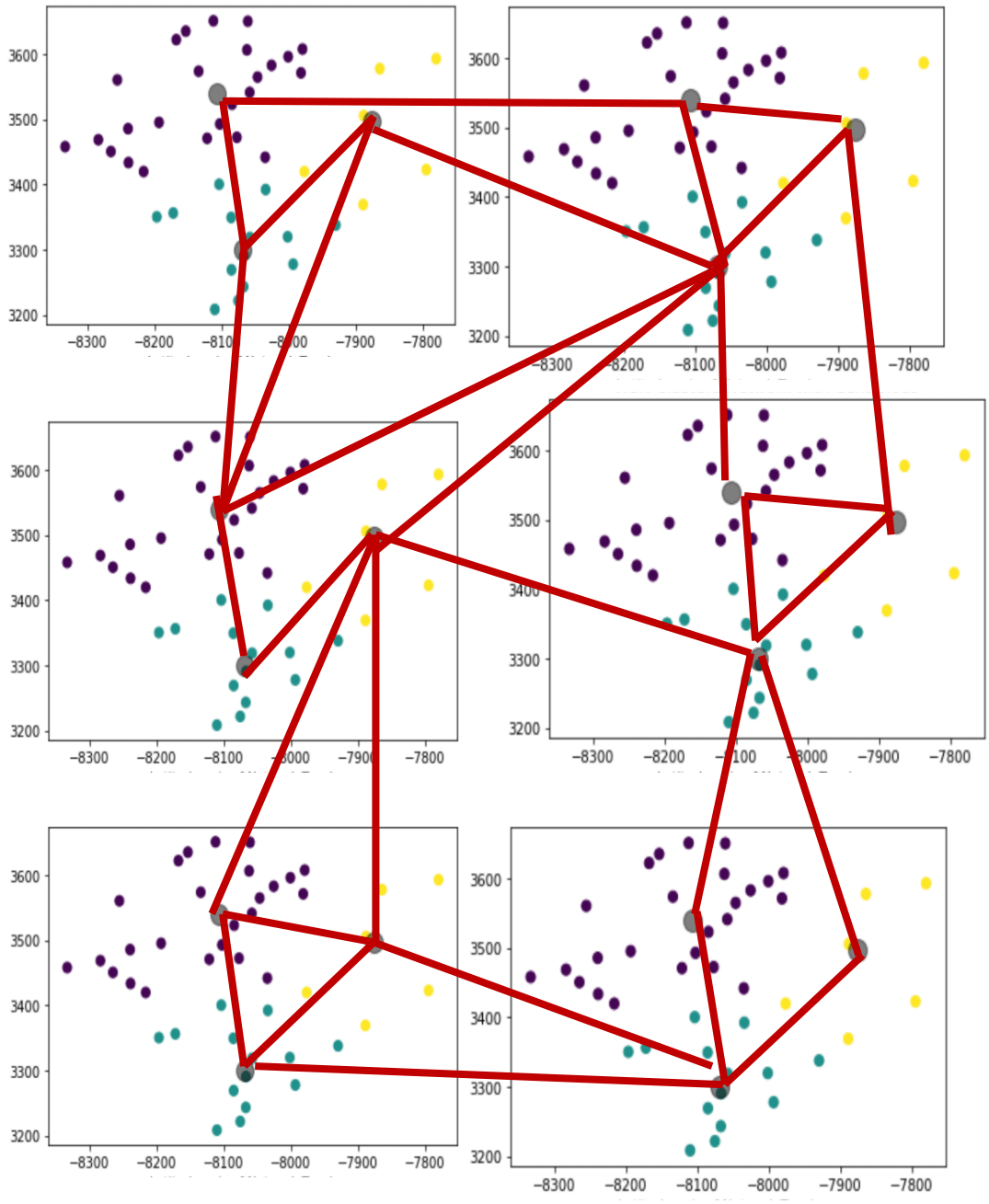

Figure 6.12: SDN-enabled network

6.6.2 Performance Results and Analysis: Propagation Latency and Cost

To achieve efficient controller placement, a few objectives should be considered to improve network performance. Propagation latency and cost are considered in this case. Before we can even place controllers, we clustered using an optimized PSO algorithm and find the optimal number of placements using silhouette, It is used to assess the effectiveness of the PSO algorithm as seen in Table 6.3. We aimed to achieve a hierarchical cluster to form sub-clusters. This is achieved using PSO K-means or PSO Hybrid in this instance which shows to be efficient as shown in our quantitative evidence in Table 6.3. Silhouette is a method that can validate whether classified data is consistent. It uses values between 1 and -1, the higher the value asserts an accurate match. **Silhouette_stdev** shows a higher value compared to other algorithms. This metric is noteworthy because it was used to determine the best number of controller placements after the network was aggregated using a Hybridized PSO.

Table 6.3 Benchmarks for CPP Optimization

Method	sse_mean	sse_stdev	silhouette_mean	silhouette_stdev	quantization_mean	quantization_stdev
K-Means++	522339,366600987	68547,5893089454	0,3798110099	0,053835321	92,9313285061	6,8859910912
PSO	658632,778924129	197387,659606645	0,3474796207	0,0603782243	88,6788242517	4,5556619775
PSO Hybrid	686103,008321649	271821,675863547	0,3293332962	0,1037320831	85,8449898284	4,719751469

We used 3 clusters from our first phase which we continued even within this phase. Silhouette inputted all clusters as initial topologies, further clustered them and find optimal placements. Topologies are derived from our SDN-enabled network which can be a collection of other existing generic topologies. In this case, we used a much larger topology that forms part of our cluster i.e. cluster 1 and we compared it with cluster 2 being the smaller cluster. Note that the two topologies are derived from our network clusters using the Hybrid PSO.

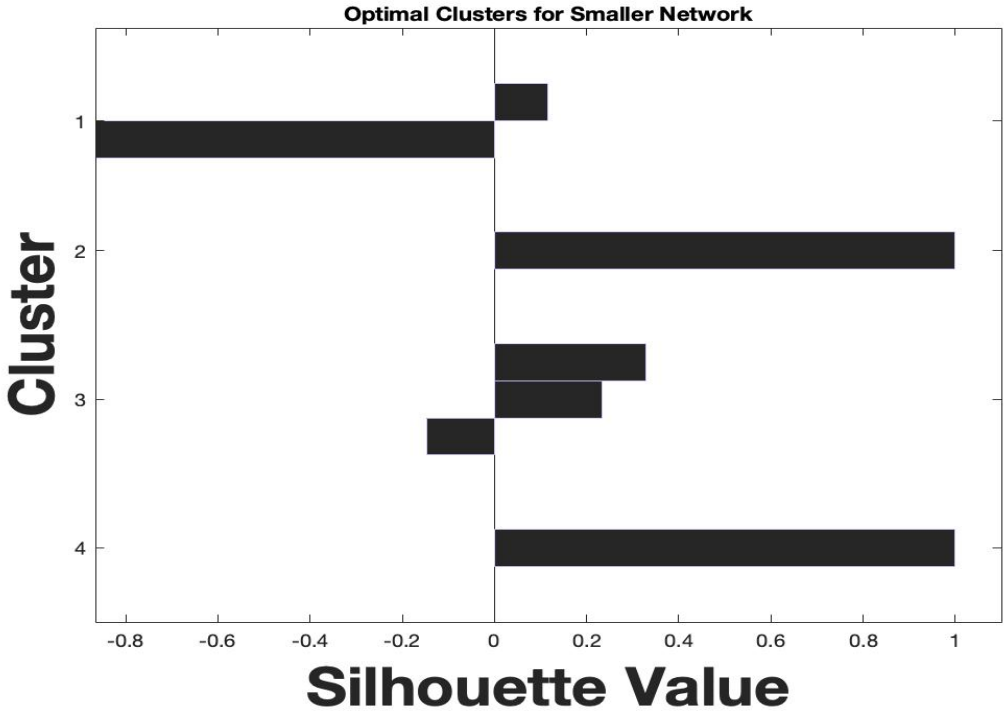

Figure 6.13: Optimal Cluster Using Silhouette for smaller cluster

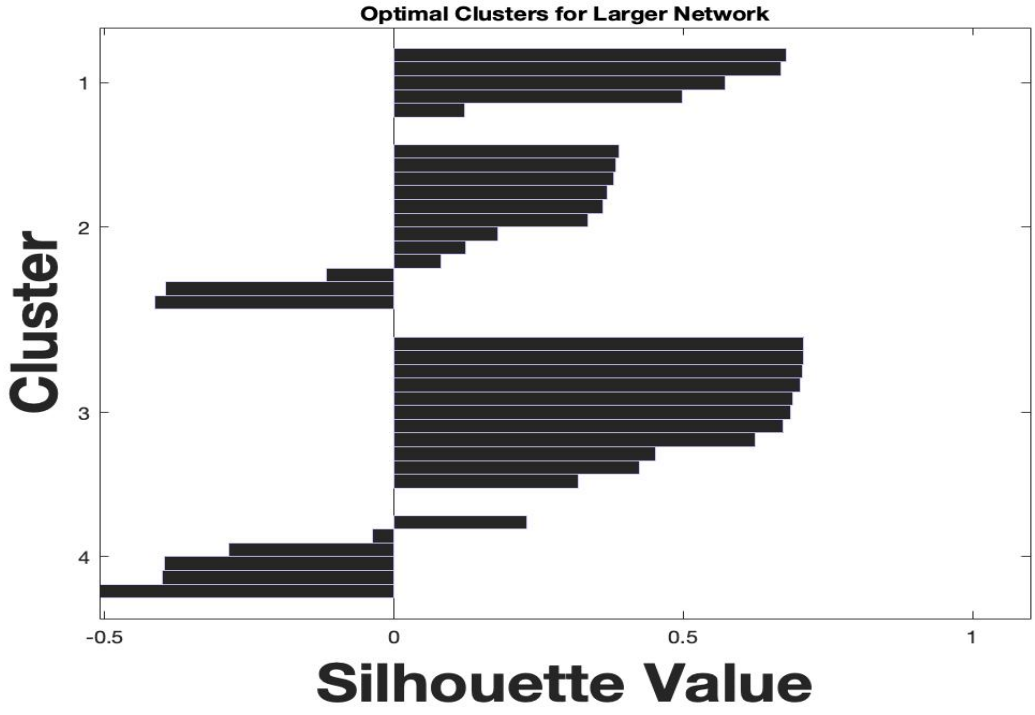

Figure 6.14: Optimal Cluster Using Silhouette for bigger cluster

Figures 6.13-6.14 show the optimal clusters for network topology, in this case, the biggest topology cluster from Hybrid PSO. We used Silhouette to measure the optimality. It explains that when the silhouette value increases for a certain cluster concerning the nodes, then the nodes are a good match for the cluster, thus, resulting in an optimal cluster. Because we were creating a hierarchical cluster, for the smaller cluster, we saw that two clusters were optimal as all nodes were positive matches compared to one of four clusters that had a negative match. Additionally, the bigger cluster was a positive match for one and three clusters. However, a single cluster was not feasible, hence we used a much-refined method of measure called Gap statistic. The method shows the exact number of controllers per cluster concerning the Gap value. The increase in Gap value shows accuracy with an increase in controller number.

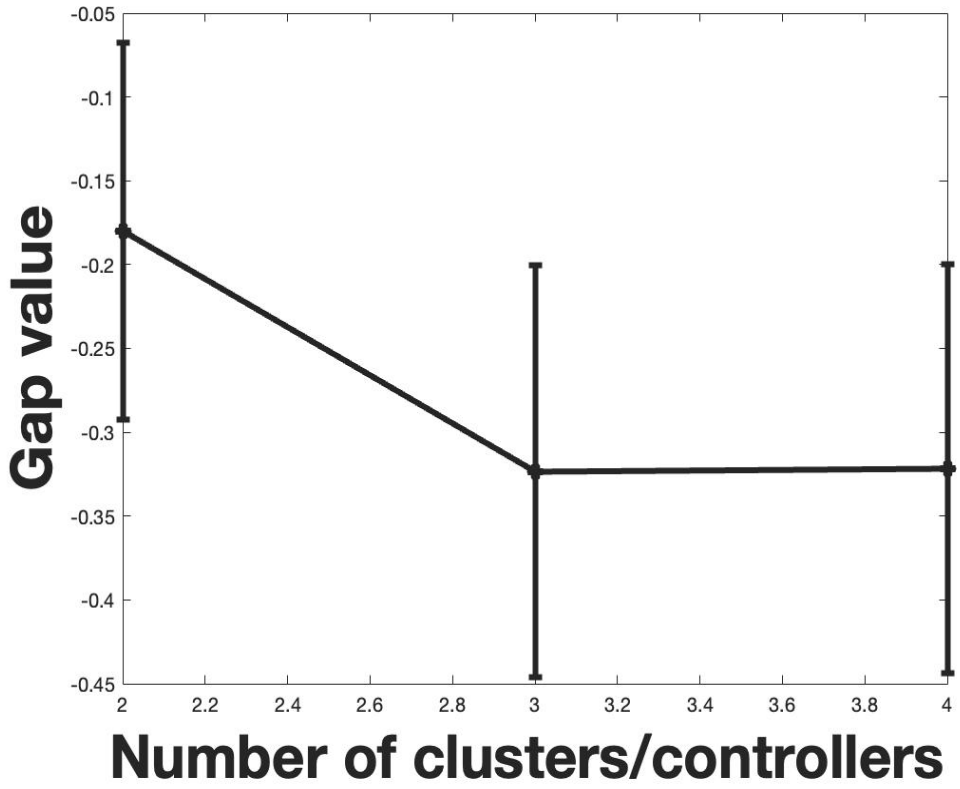

Figure 6.15: Optimal Controllers Using Gap statistic measure for bigger cluster

Figure 6.16 illustrates that three and four controllers are accurate for a larger cluster of our topology. However, following the silhouette value, we have discarded four clusters meaning three is optimal in comparison to both measures. Similarly, the smaller clustered topology shows that two controllers for two clusters are optimal using both measures. However, these optimal placements achieved propagation latency, and Figure 6.17 shows the average and worst-case latency which is explained in section 2.6.2.

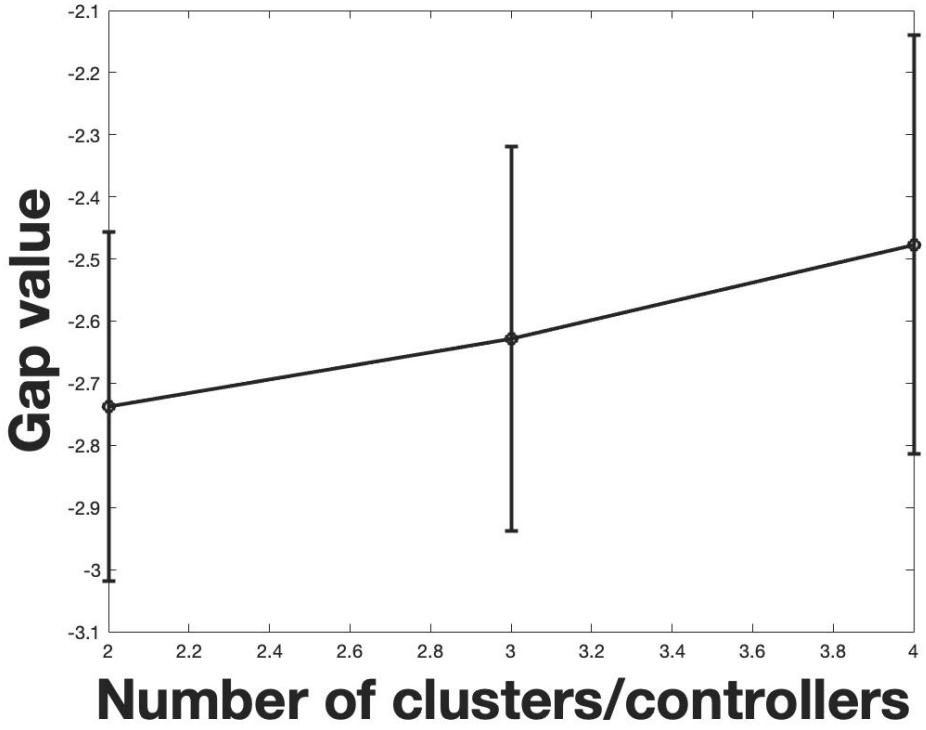

Figure 6.16: Optimal Controllers Using Gap statistic measure for smaller cluster

6.6.3 Discussion

The different graphs describe the final results of a distributed routing protocol that make use of clustering. We described the different case studies in terms of the number of clusters in terms of the number of nodes across the network, as stated in the evaluation section. The cluster number is important as it allowed us to perform. The results considered the lifetime of a network through the number of nodes that have survived or which are still alive. Since a traditional WSN performs the routing to the Sink, our approach localizes the control panel where, following our assumptions, all information is available at the local controllers. Therefore, the significance of this evidence is distributing a routing protocol to achieve prolonging the lifetime of a network i.e. this serves well when aggregation of a network is performed through portioning the network into sub-network (fewer nodes) which are controlled individually by a local controller by K-means.

In comparison to other existing energy-efficient routing techniques in WSN, we present our proposed method to explain its performance concerning other existing systems or methods. The number of alive nodes or nodes with residual energy in a network is referred to as network lifetime. Therefore, we used this as a benchmark to measure the efficiency of our method as energy is concerned. Our method outperforms other algorithms in this instance. Since we only have a collection of multiple SDN-enabled sinks which can be connected to form a topology, we used this idea to find controllers for the network. We used PSO hybrid to aggregate the topology, then used silhouette and Gap statistics through clustering, to find optimal clusters and controllers, this consequently forms a hierarchical approach.

The propagation latency is then tested using the average case (controller to controller) and worst-case (controller to switch/SDN-enabled sink). This is to examine if the delay between controllers and controllers and switches can be reduced, the results are shown in Figure 6.17. The results show that delay is minimized with the increase in controllers, in our case three optimal controllers considering our larger network or cluster. The latency between the switch and the controller, on the other hand, diminishes as the number of controllers increases. This might be caused by multiple factors like load imbalance between links and queueing of packets, this is one of the drawbacks or challenges of CPP as it is an NP-hard problem, not all factors or objectives will be optimized.

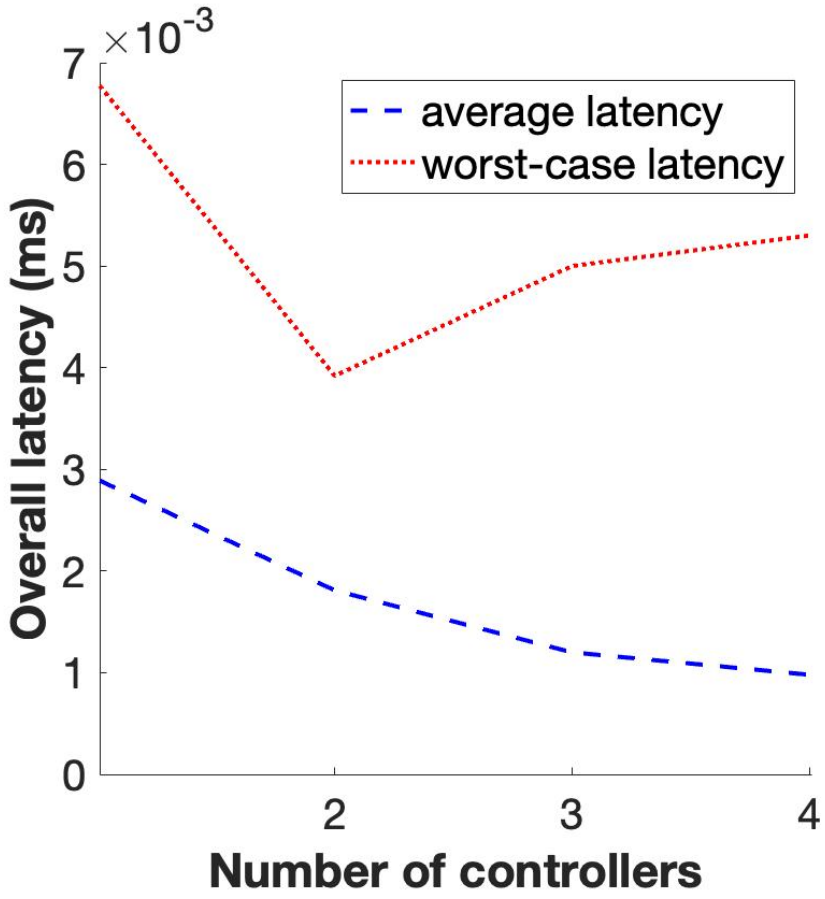

Figure 6.17: Average and worst-case latency of our proposed Efficient-CP

6.7 Chapter Summary

This chapter asserts a great deal about the algorithm structure as well as the tools that were used to gather and analyse the data. The pseudocode of both Phases of the algorithm is illustrated to describe energy-aware CPP. The optimization is also shown as quantitative evidence to solidify our approach. The data gathered shows that the use of a controller in an aggregated network consumes less energy compared to the traditional WSN.

Chapter 7

Conclusion and Recommendation

7.1 Chapter Overview

This chapter concludes our research objectives and research goal as it explains the research results further giving insights on future work and recommendations. Contributions to our research are highlighted in alignment to open research opportunities and future work discussed.

7.2 Research Conclusion

A proposal is initiated as Chapter 1 with all the relevant research objectives and research questions that serve as key primary to achieving the research goal. A problem statement is first formulated as it is highly referenced to give evidence of an existing problem. The respective research questions which are answered by our research objectives gave our research a clear outline on how to go about it and move forward including Chapter 2, as the literature chapter to give an understanding of the problem at hand and existing methods.

A clear system or methodology is put into place in Chapter 3 following the achievement of RO1 as in Chapters 1 and 2. In this chapter, an in-depth description of how the research was conducted is described comprehensively with more descriptions in Chapters 4 and 5 that highlight the evaluation of the choice in protocol and algorithm included in our research algorithm. Furthermore, Chapter 6 gives quantitative evidence supported by the literature review from Chapter 2 and quantitative data gathering in Chapters 4 and 5 to provide a solid and conclusive outcome for our dissertations' results. This was done through simulation in gathering the data and analysing the data through evaluation using different network conditions or scenarios to achieve rigorous outcomes. Finally, we provide a conclusion for our research with more detail on future research and research opportunities for potential researchers. Our research questions have been answered below to achieve our research goal or aim…

RQ1: What is the state-of-the-art practice of CPP and the challenges that impact its efficiency?

An in-depth analysis was carried out in Chapter 2 to answer RQ1 following proposed related works in section 2.7. Different mechanisms were proposed including algorithms, architectures and optimization problems to address the problem under study. A comprehensive discussion

was prescribed, which described these techniques following the identified objectives in section 2.6.2 i.e., minimization of energy consumption and maximization of network performance concerning latency and reliability. A published journal paper in this instance solidified the answer to RQ1 by comprehensively explaining the proposed techniques of other researchers related to works of Chapter 2, to appropriately justify the choice in schemes in this research through a comprehensive literature review.

RQ2: How can an efficient algorithm be designed and developed for SDWSN-based CPP which minimizes energy consumption and increase reliability?

An empirical analysis was conducted to answer RQ2 as depicted in Chapter 4 and Chapter 5 of this dissertation. To answer RQ2 the following sub-questions were addressed:

RQ2.1: How can a CPP algorithm be designed for optimal controller placement?

This research sub-question was answered through an empirical analysis in Chapter 4. This analysis supports the choice of an efficient scheme following the comprehensive literature review carried out in RQ1. A clustering-based meta-heuristic is justified to be an efficient technique in this instance. However, a comparison of various algorithms of this calibre was necessary hence sub-section 4.3-4.5 explained the relevant performance objective i.e. propagation latency and further illustrated the performance evaluation parameters to better understand the best algorithm for our design. The discussion sheds light on the best-performing algorithm with quantitative evidence. PSO algorithm was the best performing CPP algorithm which served as an element of our design to justify an optimal controller placement, therefore, answering RQ2.1.

RQ2.2: How can a reliable energy-aware CPP be achieved from RQ2.1?

This research sub-question was answered through an empirical analysis in Chapter 5. This analysis supports the choice of an energy-efficient scheme. Widely used energy efficient techniques include routing protocols to curb the most influential energy consumption factor i.e. data transmission. In this case, we conducted a comparison study for WSN cluster-based routing protocol to answer RQ2.2 and form part as an additional element of the energy-aware CPP design following RQ2.1. Furthermore, a comparison between energy-efficient WSN cluster-based routing protocols was carried out with sub-section 5.3-5.5 explaining the evaluation and discussion of parameters, maintaining a clear justification through the use of

quantitative evidence represented by graphs. TEEN routing protocol showed to be the best performing to be used in the design of the algorithm.

RQ3: How can we assess the effectiveness and performance of the proposed solution?

In Chapter 6, the suggested energy-aware CPP algorithm's performance and effectiveness are reviewed and assessed. The performance objectives that are used for evaluation and effectiveness are propagation latency (average and worst-case delay) and network lifetime (surviving alive node with residual). Our proposed distributed TEEN Hybrid Hierarchical PSO algorithm outperforms WSN energy-efficient protocols and significantly increases the propagation latency of controllers and between switches and controllers

7.3 Recommendation and Future Work

There has been a few constraints or limitations that we have encountered during this research. These limitations are a possibility to document possible recommendations to explore them as future work for potential researchers as well as to expand on our work. The recommendations include simulation tools, topology discovery protocols and the use of existing frameworks. Other areas that have shown interest and ought to be explored in future include virtualization, security vulnerabilities of sensors and lack of Northbound protocol implementation in SDWSN and other networks i.e. IoT and 5G.

7.3.1 Implementation tools

There have been a few simulation tools that have been proposed to carry out experimentation of SDWSN more particularly using a central customized algorithm. Much work has focused on implementing and evaluating frameworks instead of designed algorithms that are to be evaluated in a particular tool to test its efficiency against or under certain performance objectives or metrics. The lack of such tools makes it difficult for academics to design and deploy SDN-based networks, which is critical for network efficiency and hence limits the research contribution [26]. Therefore, it is important to explore the variety of tools that are incorporated with a controller and an easy-to-use interface for evaluation purposes. The widely used tools include NS2/NS3, Omnet++, mininet, Matlab and cooja with evaluation over proposed frameworks and less work done on centralized algorithms [26, 167-169].

7.3.2 Use of existing frameworks

The use of frameworks using a centrally based algorithm design is not only difficult to evaluate but very tedious to be incorporated with existing frameworks as they focus on specific goals. The combination of the two frameworks may be an argument worth investigating. However,

with the growth of networks, there is much efficiency in the design of a single algorithm that can conserve the cost of operation instead of designing a framework that requires custom OS and hardware. Therefore, it is crucial to explore the use of existing frameworks using designed algorithms to further better a network. TinySDN and SDN-WISE are the most used frameworks when considering CPP and energy efficiency. However, there is the limitation of cooperation in both when designing an algorithm with consideration of the two objectives as the key focus, hence, worth studying further or designing a new framework considering the two.

7.3.3 Topology Discovery Protocol

In this research, we limited our research to sensor nodes with no mobility. However, mobility is worth studying as it considers other applications with the sole purpose of dynamic topology. A dynamic topology changes over time due to sensors with mobility functions, therefore, the network topology is to be updated respectively. This calls for an efficient TD protocol making use of existing southbound protocols and neighbourhood protocols to exploit the efficiency of their features in keeping track of topology status. Therefore, a CPP TD protocol is to be considered in future, with consideration of security vulnerabilities or connectivity caused by holes in the network.

7.3.4 Northbound and Southbound protocols

The use of protocols in the control and data plane for SDWSN is still an issue when designing algorithms in combating the drawbacks of the network's nature. Furthermore, the use of traditional OpenFlow and Senor OpenFlow protocols is still a limitation as they fail to give allowance to certain requirements of the network like mobility and heterogeneity of sensors. The gathering of sensor node information of individual nodes is a constraint as all are not connected directly to a controller, hence, their information is to be known by other nodes before being known at a global controller. It is important to consider a protocol design or modification that can self-route the entire network i.e. neural network machine learning-based algorithm, and further make use of a MAC protocol to locally store information of the respective sensor to maximize connectivity, minimize end-to-end delay and prolong the lifetime of the network.

7.3.5 Virtualization

The use of virtualization is important for the management of resources and simplicity when considering large entities i.e. networks. With the growth of networks and mobility of networks, the consideration of topology change is challenging particularly when seeking optimal controller placement over the network, thus, calling for virtualization. The risk of using

virtualization in a control plane in SDWSN is security vulnerabilities, however, this may be another limitation worth studying. Therefore, designing an optimal controller placement algorithm using virtualization-like containers to achieve a reduction of expenditure cost and energy consumption in SDWSN is worth studying in future.

7.3.6 Security Vulnerabilities

Networks, both wired and wireless are always prone to security vulnerabilities. However, the use of certain technologies like virtualization and routing protocols can further expose weaknesses in the network whilst they aim to promote efficiency. Furthermore, heterogeneity in the network is adapted to limit operational cost thus of consideration in the growth of a network, particularly SDWSN. As a result, a method that can employ machine learning technologies to self-isolate a threat when picking up a sensor should be considered in the future to recover the state of the topology and avoid data loss in the network.

References

[1] H. Farhady, H. Lee, and A. Nakao, "Software-Defined Networking: A survey," *Computer Networks,* vol. 81, pp. 79-95, 2015/04/22/ 2015, doi: https://doi.org/10.1016/j.comnet.2015.02.014.

[2] A. Lara, A. Kolasani, and B. Ramamurthy, "Network Innovation using OpenFlow: A Survey," *IEEE Communications Surveys & Tutorials,* vol. 16, no. 1, pp. 493-512, 2014, doi: 10.1109/SURV.2013.081313.00105.

[3] M. Ndiaye, G. Hancke, and A. Abu-Mahfouz, "Software defined networking for improved wireless sensor network management: A survey," *Sensors,* vol. 17, no. 5, p. 1031, 2017.

[4] S. R. J. Ramson and D. J. Moni, "Applications of wireless sensor networks — A survey," in *2017 International Conference on Innovations in Electrical, Electronics, Instrumentation and Media Technology (ICEEIMT)*, 3-4 Feb. 2017 2017, pp. 325-329, doi: 10.1109/ICIEEIMT.2017.8116858.

[5] T. Tony and L. Hiryanto, "Software-Defi Wireless Sensor Networks: A Systematic Review, Architecture and Challenges," in *IOP Conference Series: Materials Science and Engineering*, 2020, vol. 852, no. 1: IOP Publishing, p. 012136.

[6] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, "A survey on software-defined wireless sensor networks: Challenges and design requirements," *IEEE Access,* vol. 5, pp. 1872-1899, 2017.

[7] D. Chalmers and M. Sloman, "A survey of quality of service in mobile computing environments," *IEEE Communications surveys,* vol. 2, no. 2, pp. 2-10, 1999.

[8] I. Deva Priya and S. Silas, "A Survey on Research Challenges and Applications in Empowering the SDN-Based Internet of Things," Singapore, 2019: Springer Singapore, in Advances in Big Data and Cloud Computing, pp. 457-467.

[9] S. K. Lee, M. Bae, and H. Kim, "Future of IoT Networks: A Survey," *Applied Sciences,* vol. 7, no. 10, p. 1072, 2017. [Online]. Available: https://www.mdpi.com/2076-3417/7/10/1072.

[10] K. S. Sahoo, D. Puthal, M. S. Obaidat, A. Sarkar, S. K. Mishra, and B. Sahoo, "On the placement of controllers in software-defined-WAN using meta-heuristic approach," *Journal of Systems and Software,* vol. 145, pp. 180-194, 2018.

[11] Y. Zhang, N. Ansari, M. Wu, and H. Yu, "On wide area network optimization," *IEEE communications surveys & tutorials,* vol. 14, no. 4, pp. 1090-1113, 2011.

[12] S. V. Rao, "Game Theory Based Network Partitioning Approaches for Controller Placement in SDN," in *Communication Systems and Networks: 10th International Conference, COMSNETS 2018, Bangalore, India, January 3-7, 2018, Extended Selected Papers*, 2019, vol. 11227: Springer, p. 245.

[13] B. Briscoe *et al.*, "Reducing internet latency: A survey of techniques and their merits," *IEEE Communications Surveys & Tutorials,* vol. 18, no. 3, pp. 2149-2196, 2014.

[14] M. Khorramizadeh and V. Ahmadi, "Capacity and load-aware software-defined network controller placement in heterogeneous environments," *Computer Communications,* vol. 129, pp. 226-247, 2018.

[15] J. Zhang, F. R. Yu, S. Wang, T. Huang, Z. Liu, and Y. Liu, "Load Balancing in Data Center Networks: A Survey," *IEEE Communications Surveys & Tutorials,* vol. 20, no. 3, pp. 2324-2352, 2018, doi: 10.1109/COMST.2018.2816042.

[16] R. Trivisonno, R. Guerzoni, I. Vaishnavi, and D. Soldani, "SDN-based 5G mobile networks: architecture, functions, procedures and backward compatibility,"

Transactions on Emerging Telecommunications Technologies, vol. 26, no. 1, pp. 82-92, 2015.

[17] A. Gupta and R. K. Jha, "A survey of 5G network: Architecture and emerging technologies," *IEEE Access,* vol. 3, pp. 1206-1232, 2015.

[18] J. Lu, Z. Zhang, T. Hu, P. Yi, and J. Lan, "A Survey of Controller Placement Problem in Software-Defined Networking," *IEEE Access,* vol. 7, pp. 24290-24307, 2019, doi: 10.1109/ACCESS.2019.2893283.

[19] M. Ndiaye, G. P. Hancke, and A. M. Abu-Mahfouz, "Software defined networking for improved wireless sensor network management: A survey," *Sensors,* vol. 17, no. 5, p. 1031, 2017.

[20] T. Hirayama, T. Miyazawa, H. Furukawa, and H. Harai, "Reconstruction of control plane of distributed SDN against large-scale disruption and restoration," in *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2017: IEEE, pp. 253-258.

[21] D. S. Rana, S. A. Dhondiyal, and S. K. Chamoli, "Software Defined Networking (SDN) Challenges, issues and Solution," 2019.

[22] Y. Zhang, L. Cui, W. Wang, and Y. Zhang, "A survey on software defined networking with multiple controllers," *Journal of Network and Computer Applications,* vol. 103, pp. 101-118, 2018.

[23] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proceedings of the first workshop on Hot topics in software defined networks*, 2012: ACM, pp. 7-12.

[24] G. Wang, Y. Zhao, J. Huang, and W. Wang, "The Controller Placement Problem in Software Defined Networking: A Survey," *IEEE Network,* vol. 31, no. 5, pp. 21-27, 2017, doi: 10.1109/MNET.2017.1600182.

[25] M. Karakus and A. Durresi, "A survey: Control plane scalability issues and approaches in software-defined networking (SDN)," *Computer Networks,* vol. 112, pp. 279-293, 2017.

[26] H. Mostafaei and M. Menth, "Software-defined wireless sensor networks: A survey," *Journal of Network and Computer Applications,* vol. 119, pp. 42-56, 2018.

[27] A. K. Singh and S. Srivastava, "A survey and classification of controller placement problem in SDN," *International Journal of Network Management,* vol. 28, no. 3, p. e2018, 2018.

[28] L. Lehtiranta, J.-M. Junnonen, S. Kärnä, and L. Pekuri, "The constructive research approach: Problem solving for complex projects," *Designs, methods and practices for research of project management,* pp. 95-106, 2015.

[29] C. News, "Facebook was down for hours on Wednesday, including Instagram and Messenger," ed. United State, 2019.

[30] I. T. Haque and N. Abu-Ghazaleh, "Wireless software defined networking: A survey and taxonomy," *IEEE Communications Surveys & Tutorials,* vol. 18, no. 4, pp. 2713-2737, 2016.

[31] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, and N. Venkatasubramanian, "A software defined networking architecture for the internet-of-things," in *2014 IEEE network operations and management symposium (NOMS)*, 2014: IEEE, pp. 1-9.

[32] A. Ali, Y. Ming, S. Chakraborty, and S. Iram, "A Comprehensive Survey on Real-Time Applications of WSN," *Future Internet,* vol. 9, no. 4, p. 77, 2017. [Online]. Available: https://www.mdpi.com/1999-5903/9/4/77.

[33] M. Pule, A. Yahya, and J. Chuma, "Wireless sensor networks: A survey on monitoring water quality," *Journal of Applied Research and Technology,* vol. 15, no. 6, pp. 562-570, 2017/12/01/ 2017, doi: https://doi.org/10.1016/j.jart.2017.07.004.

[34] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer networks,* vol. 52, no. 12, pp. 2292-2330, 2008.
[35] I. Howitt and J. A. Gutierrez, "IEEE 802.15. 4 low rate-wireless personal area network coexistence issues," in *2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003.*, 2003, vol. 3: IEEE, pp. 1481-1486.
[36] P. Kinney, "Zigbee technology: Wireless control that simply works," in *Communications design conference*, 2003, vol. 2, pp. 1-7.
[37] Z. Shelby and C. Bormann, *6LoWPAN: The wireless embedded Internet*. John Wiley & Sons, 2011.
[38] Y. Zhou, Y. Fang, and Y. Zhang, "Securing wireless sensor networks: a survey," *IEEE Communications Surveys & Tutorials,* vol. 10, no. 3, pp. 6-28, 2008.
[39] T. Kgogo, B. Isong, and A. M. Abu-Mahfouz, "Software defined wireless sensor networks security challenges," in *2017 IEEE AFRICON*, 18-20 Sept. 2017 2017, pp. 1508-1513, doi: 10.1109/AFRCON.2017.8095705.
[40] S. W. Pritchard, G. P. Hancke, and A. M. Abu-Mahfouz, "Security in software-defined wireless sensor networks: Threats, challenges and potential solutions," in *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, 2017: IEEE, pp. 168-173.
[41] P. Singh, B. Tripathi, and N. P. Singh, "Node localization in wireless sensor networks."
[42] A. K. Paul and T. Sato, "Localization in Wireless Sensor Networks: A Survey on Algorithms, Measurement Techniques, Applications and Challenges," *Journal of Sensor and Actuator Networks,* vol. 6, no. 4, p. 24, 2017. [Online]. Available: https://www.mdpi.com/2224-2708/6/4/24.
[43] M. A. Mahmood, W. K. Seah, and I. Welch, "Reliability in wireless sensor networks: A survey and challenges ahead," *Computer Networks,* vol. 79, pp. 166-187, 2015.
[44] A. Dâmaso, N. Rosa, and P. Maciel, "Reliability of Wireless Sensor Networks," *Sensors,* vol. 14, no. 9, pp. 15760-15785, 2014. [Online]. Available: https://www.mdpi.com/1424-8220/14/9/15760.
[45] N. A. Pantazis, S. A. Nikolidakis, and D. D. Vergados, "Energy-efficient routing protocols in wireless sensor networks: A survey," *IEEE communications surveys & tutorials,* vol. 15, no. 2, pp. 551-591, 2012.
[46] I. Dietrich and F. Dressler, "On the lifetime of wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN),* vol. 5, no. 1, p. 5, 2009.
[47] H. Mostafaei and M. R. Meybodi, "Maximizing lifetime of target coverage in wireless sensor networks using learning automata," *Wireless Personal Communications,* vol. 71, no. 2, pp. 1461-1477, 2013.
[48] B. Wang, *Coverage control in sensor networks*. Springer Science & Business Media, 2010.
[49] A. Sangwan and R. P. Singh, "Survey on coverage problems in wireless sensor networks," *Wireless Personal Communications,* vol. 80, no. 4, pp. 1475-1500, 2015.
[50] F. Bannour, S. Souihi, and A. Mellouk, "Distributed SDN Control: Survey, Taxonomy, and Challenges," *IEEE Communications Surveys & Tutorials,* vol. 20, no. 1, pp. 333-354, 2018, doi: 10.1109/COMST.2017.2782482.
[51] C. Trois, M. D. D. Fabro, L. C. E. d. Bona, and M. Martinello, "A Survey on SDN Programming Languages: Toward a Taxonomy," *IEEE Communications Surveys & Tutorials,* vol. 18, no. 4, pp. 2687-2712, 2016, doi: 10.1109/COMST.2016.2553778.
[52] O. Blial, M. Ben Mamoun, and R. Benaini, "An Overview on SDN Architectures with Multiple Controllers," *Journal of Computer Networks and Communications,* vol. 2016, p. 9396525, 2016/04/27 2016, doi: 10.1155/2016/9396525.

[53] F. Bannour, S. Souihi, and A. Mellouk, "Distributed SDN control: Survey, taxonomy, and challenges," *IEEE Communications Surveys & Tutorials,* vol. 20, no. 1, pp. 333-354.
[54] F. Bannour, S. Souihi, and A. Mellouk, "Distributed SDN control: Survey, taxonomy, and challenges," *IEEE Communications Surveys & Tutorials,* vol. 20, no. 1, pp. 333-354, 2017.
[55] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A Survey on Software-Defined Networking," *IEEE Communications Surveys & Tutorials,* vol. 17, no. 1, pp. 27-51, 2015, doi: 10.1109/COMST.2014.2330903.
[56] K. Benzekki, A. El Fergougui, and A. Elbelrhiti Elalaoui, "Software-defined networking (SDN): a survey," *Security and communication networks,* vol. 9, no. 18, pp. 5803-5833, 2016.
[57] B. A. A. Nunes, M. Mendonca, X. Nguyen, K. Obraczka, and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," *IEEE Communications Surveys & Tutorials,* vol. 16, no. 3, pp. 1617-1634, 2014, doi: 10.1109/SURV.2014.012214.00180.
[58] V. Patel and D. Vashi, "A Survey of Software-Defined Networking," 2017.
[59] S. Rowshanrad, S. Namvarasl, V. Abdi, M. Hajizadeh, and M. Keshtgary, "A survey on SDN, the future of networking," *Journal of Advanced Computer Science & Technology,* vol. 3, no. 2, pp. 232-248, 2014.
[60] K. M. Modieginyane, B. B. Letswamotse, R. Malekian, and A. M. Abu-Mahfouz, "Software defined wireless sensor networks application opportunities for efficient network management: A survey," *Computers & Electrical Engineering,* vol. 66, pp. 274-287, 2018.
[61] T. Luo, H.-P. Tan, and T. Q. Quek, "Sensor OpenFlow: Enabling software-defined wireless sensor networks," *IEEE Communications Letters,* vol. 16, no. 11, pp. 1896-1899, 2012.
[62] M. Jacobsson and C. Orfanidis, "Using software-defined networking principles for wireless sensor networks," in *SNCNW 2015, May 28–29, Karlstad, Sweden*, 2015.
[63] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, 27 Nov.-1 Dec. 1995 1995, vol. 4, pp. 1942-1948 vol.4, doi: 10.1109/ICNN.1995.488968.
[64] A. Dvir, Y. Haddad, and A. Zilberman, "Wireless controller placement problem," in *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, 2018: IEEE, pp. 1-4.
[65] P. B. Oni and S. D. Blostein, "Decentralized AP selection in large-scale wireless LANs considering multi-AP interference," in *2017 International conference on computing, networking and communications (ICNC)*, 2017: IEEE, pp. 13-18.
[66] B. P. R. Killi and S. V. Rao, "Controller placement in software defined networks: A Comprehensive survey," *Computer Networks,* vol. 163, p. 106883, 2019/11/09/ 2019, doi: https://doi.org/10.1016/j.comnet.2019.106883.
[67] A. Sallahi and M. St-Hilaire, "Expansion model for the controller placement problem in software defined networks," *IEEE Communications Letters,* vol. 21, no. 2, pp. 274-277, 2016.
[68] A. A. Ateya *et al.*, "Chaotic salp swarm algorithm for SDN multi-controller networks," *Engineering Science and Technology, an International Journal,* vol. 22, no. 4, pp. 1001-1012, 2019.
[69] T. Das and M. Gurusamy, "INCEPT: INcremental ControllEr PlacemenT in software defined networks," in *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, 2018: IEEE, pp. 1-6.

[70] J. Liao, H. Sun, J. Wang, Q. Qi, K. Li, and T. Li, "Density cluster based approach for controller placement problem in large-scale software defined networkings," *Computer Networks,* vol. 112, pp. 24-35, 2017.

[71] B. P. R. Killi, E. A. Reddy, and S. V. Rao, "Cooperative game theory based network partitioning for controller placement in SDN," in *2018 10th International Conference on Communication Systems & Networks (COMSNETS)*, 2018: IEEE, pp. 105-112.

[72] G. Wang, Y. Zhao, J. Huang, Q. Duan, and J. Li, "A K-means-based network partition algorithm for controller placement in software defined network," in *2016 IEEE International Conference on Communications (ICC)*, 2016: IEEE, pp. 1-6.

[73] L. Zhu, R. Chai, and Q. Chen, "Control plane delay minimization based SDN controller placement scheme," in *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*, 2017: IEEE, pp. 1-6.

[74] T. Li, Z. Gu, X. Lin, S. Li, and Q. Tan, "Approximation Algorithms for Controller Placement Problems in Software Defined Networks," in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, 2018: IEEE, pp. 250-257.

[75] G. Wang, Y. Zhao, J. Huang, and Y. Wu, "An effective approach to controller placement in software defined wide area networks," *IEEE Transactions on Network and Service Management,* vol. 15, no. 1, pp. 344-355, 2017.

[76] T. Yuan, X. Huang, M. Ma, and J. Yuan, "Balance-based SDN controller placement and assignment with minimum weight matching," in *2018 IEEE International Conference on Communications (ICC)*, 2018: IEEE, pp. 1-6.

[77] A. Jalili, M. Keshtgari, and R. Akbari, "Optimal controller placement in large scale software defined networks based on modified NSGA-II," *Applied Intelligence,* vol. 48, no. 9, pp. 2809-2823, 2018.

[78] H. Selvi, G. Gür, and F. Alagöz, "Cooperative load balancing for hierarchical SDN controllers," in *2016 IEEE 17th International Conference on High Performance Switching and Routing (HPSR)*, 2016: IEEE, pp. 100-105.

[79] S.-C. Lin, P. Wang, and M. Luo, "Control traffic balancing in software defined networks," *Computer Networks,* vol. 106, pp. 260-271, 2016.

[80] M. He, A. Basta, A. Blenk, and W. Kellerer, "Modeling flow setup time for controller placement in sdn: Evaluation for dynamic flows," in *2017 IEEE International Conference on Communications (ICC)*, 2017: IEEE, pp. 1-7.

[81] F. J. Ros and P. M. Ruiz, "On reliable controller placements in software-defined networks," *Computer Communications,* vol. 77, pp. 41-51, 2016.

[82] B. P. R. Killi and S. V. Rao, "Towards improving resilience of controller placement with minimum backup capacity in software defined networks," *Computer Networks,* vol. 149, pp. 102-114, 2019.

[83] B. P. R. Killi and S. V. Rao, "Link failure aware capacitated controller placement in software defined networks," in *2018 International Conference on Information Networking (ICOIN)*, 2018: IEEE, pp. 292-297.

[84] T. Zhang, P. Giaccone, A. Bianco, and S. De Domenico, "The role of the inter-controller consensus in the placement of distributed SDN controllers," *Computer Communications,* vol. 113, pp. 1-13, 2017.

[85] A. Alshamrani, S. Guha, S. Pisharody, A. Chowdhary, and D. Huang, "Fault tolerant controller placement in distributed SDN environments," in *2018 IEEE International Conference on Communications (ICC)*, 2018: IEEE, pp. 1-7.

[86] G. Ramya and R. Manoharan, "Enhanced Multi-Controller Placements in SDN," in *2018 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, 2018: IEEE, pp. 1-5.

[87] Y. Hu, T. Luo, N. C. Beaulieu, and C. Deng, "The energy-aware controller placement problem in software defined networks," *IEEE Communications Letters,* vol. 21, no. 4, pp. 741-744, 2016.
[88] A. Ruiz-Rivera, K.-W. Chin, and S. Soh, "GreCo: An energy aware controller association algorithm for software defined networks," *IEEE communications letters,* vol. 19, no. 4, pp. 541-544, 2015.
[89] B. Özbek, Y. Aydoğmuş, A. Ulaş, B. Gorkemli, and K. Ulusoy, "Energy aware routing and traffic management for software defined networks," in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, 2016: IEEE, pp. 73-77.
[90] A. Fernandez-Fernandez, C. Cervello-Pastor, and L. Ochoa-Aday, "Achieving energy efficiency: An energy-aware approach in SDN," in *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016: IEEE, pp. 1-7.
[91] Z. Wu, X. Ji, Y. Wang, X. Chen, and Y. Cai, "An Energy-aware Routing for Optimizing Control and Data Traffic in SDN," in *2018 26th International Conference on Systems Engineering (ICSEng)*, 2018: IEEE, pp. 1-4.
[92] V. Ahmadi and M. Khorramizadeh, "An adaptive heuristic for multi-objective controller placement in software-defined networks," *Computers & Electrical Engineering,* vol. 66, pp. 204-228, 2018.
[93] A. Ksentini, M. Bagaa, T. Taleb, and I. Balasingham, "On using bargaining game for optimal placement of SDN controllers," in *2016 IEEE International Conference on Communications (ICC)*, 2016: IEEE, pp. 1-6.
[94] M. Ramasamy and S. Pawar, "Pareto-Optimal Multi-Controller Placement in Software Defined Network," in *2018 3rd International Conference for Convergence in Technology (I2CT)*, 2018: IEEE, pp. 1-7.
[95] A. Jalili, M. Keshtgari, R. Akbari, and R. Javidan, "Multi criteria analysis of Controller Placement Problem in Software Defined Networks," *Computer Communications,* vol. 133, pp. 115-128, 2019.
[96] M. J. Abdel-Rahman, E. A. Mazied, A. MacKenzie, S. Midkiff, M. R. Rizk, and M. El-Nainay, "On stochastic controller placement in software-defined wireless networks," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, 2017: IEEE, pp. 1-6.
[97] A. D. Gante, M. Aslan, and A. Matrawy, "Smart wireless sensor network management based on software-defined networking," in *2014 27th Biennial Symposium on Communications (QBSC)*, 1-4 June 2014 2014, pp. 71-75.
[98] B. T. d. Oliveira, L. B. Gabriel, and C. B. Margi, "TinySDN: Enabling Multiple Controllers for Software-Defined Wireless Sensor Networks," *IEEE Latin America Transactions,* vol. 13, no. 11, pp. 3690-3696, 2015.
[99] H. I. Kobo, G. P. Hancke, and A. M. Abu-Mahfouz, "Towards a distributed control system for software defined Wireless Sensor Networks," in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 29 Oct.-1 Nov. 2017 2017, pp. 6125-6130.
[100] F. Olivier, G. Carlos, and N. Florent, "SDN based architecture for clustered WSN," in *2015 9th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, 2015: IEEE, pp. 342-347.
[101] A. De Gante, M. Aslan, and A. Matrawy, "Smart wireless sensor network management based on software-defined networking," in *2014 27th Biennial Symposium on Communications (QBSC)*, 2014: IEEE, pp. 71-75.
[102] R. Huang, X. Chu, J. Zhang, and Y. H. Hu, "Energy-efficient monitoring in software defined wireless sensor networks using reinforcement learning: A prototype," *International Journal of Distributed Sensor Networks,* vol. 11, no. 10, p. 360428, 2015.

[103] D. Zeng, P. Li, S. Guo, T. Miyazaki, J. Hu, and Y. Xiang, "Energy Minimization in Multi-Task Software-Defined Sensor Networks," *IEEE Transactions on Computers,* vol. 64, no. 11, pp. 3128-3139, 2015.
[104] B. J. Oates, *Researching information systems and computing*. Sage, 2005.
[105] C. R. Kothari, *Research methodology: Methods and techniques*. New Age International, 2004.
[106] Y. K. Singh, *Fundamental of research methodology and statistics*. New Age International, 2006.
[107] V. K. Vaishnavi and W. Kuechler, *Design science research methods and patterns: innovating information and communication technology*. Crc Press, 2015.
[108] M. Saunders, P. Lewis, and A. Thornhill, "Research methods for business students (6th ended.) Harlow," *England: Pearson Education,* 2012.
[109] P. K. Kankam, "The use of paradigms in information research," *Library & Information Science Research,* vol. 41, no. 2, pp. 85-92, 2019.
[110] D. L. Morgan, "Paradigms lost and pragmatism regained: Methodological implications of combining qualitative and quantitative methods," *Journal of mixed methods research,* vol. 1, no. 1, pp. 48-76, 2007.
[111] T. T. Tran, "Pragmatism and Constructivism in conducting research about university-enterprise collaboration in the Vietnamese context," *CIAIQ2016,* vol. 5, 2016.
[112] L. M. Scott, "Theory and research in construction education: the case for pragmatism," *Construction Management and Economics,* vol. 34, no. 7-8, pp. 552-560, 2016.
[113] G. Guthrie, *Basic research methods: An entry to social science research*. SAGE Publications India, 2010.
[114] C. Biddle and K. A. Schafft, "Axiology and anomaly in the practice of mixed methods work: Pragmatism, valuation, and the transformative paradigm," *Journal of Mixed Methods Research,* vol. 9, no. 4, pp. 320-334, 2015.
[115] A. Hevner and S. Chatterjee, "Design science research in information systems," in *Design research in information systems*: Springer, 2010, pp. 9-22.
[116] T. G. Gill and A. R. Hevner, "A fitness-utility model for design science research," *ACM Transactions on Management Information Systems (TMIS),* vol. 4, no. 2, pp. 1-24, 2013.
[117] K. Durrheim, "Research design," *Research in practice: Applied methods for the social sciences,* vol. 2, pp. 33-59, 2006.
[118] B. Isong, R. R. S. Molose, A. M. Abu-Mahfouz, and N. Dladlu, "Comprehensive review of SDN controller placement strategies," *IEEE Access,* vol. 8, pp. 170070-170092, 2020.
[119] C.-h. Yu and B. Ohlund, "Threats to validity of research design," *Retrieved January,* vol. 12, p. 2012, 2010.
[120] D. Gajjar, "Ethical consideration in research," *Education,* vol. 2, no. 7, pp. 8-15, 2013.
[121] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," *ACM SIGCOMM Computer Communication Review,* vol. 42, no. 4, pp. 473-478, 2012.
[122] J. DiGiglio and D. Ricci, "High performance, open standard virtualization with NFV and SDN," *Wind River,* 2013.
[123] Y. Hu, M. Song, and T. Li, "Towards" Full Containerization" in Containerized Network Function Virtualization," in *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems*, 2017, pp. 467-481.
[124] M. F. Bari *et al.*, "Dynamic controller provisioning in software defined networks," in *Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013)*, 2013: IEEE, pp. 18-25.

[125] D. Hock, M. Hartmann, S. Gebert, M. Jarschel, T. Zinner, and P. Tran-Gia, "Pareto-optimal resilient controller placement in SDN-based core networks," in *Proceedings of the 2013 25th International Teletraffic Congress (ITC)*, 2013: IEEE, pp. 1-9.
[126] Y. Hu, W. Wang, X. Gong, X. Que, and S. Cheng, "On reliability-optimized controller placement for software-defined networks," *China Communications,* vol. 11, no. 2, pp. 38-54, 2014.
[127] R. Sherwood and N. Mckeown, "The controller placement problem [C]," in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, 2012: Citeseer, pp. 7-12.
[128] L. Yue, C. Junyan, L. Chuxin, and L. Xiaochun, "Research on SDN Multi Controller Deployment based on K-means++," in *Journal of Physics: Conference Series*, 2020, vol. 1606, no. 1: IOP Publishing, p. 012004.
[129] H. Kuang, Y. Qiu, R. Li, and X. Liu, "A hierarchical K-means algorithm for controller placement in SDN-based WAN architecture," in *2018 10th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, 2018: IEEE, pp. 263-267.
[130] S. Das, A. Abraham, and A. Konar, *Metaheuristic clustering*. Springer, 2009.
[131] S. J. Nanda and G. Panda, "A survey on nature-inspired metaheuristic algorithms for partitional clustering," *Swarm and Evolutionary Computation,* vol. 16, pp. 1-18, 2014/06/01/ 2014, doi: https://doi.org/10.1016/j.swevo.2013.11.003.
[132] V. Guliashki, H. Toshev, and C. Korsemov, "Survey of evolutionary algorithms used in multiobjective optimization," *Problems of engineering cybernetics and robotics,* vol. 60, no. 1, pp. 42-54, 2009.
[133] R. H. Sheikh, M. M. Raghuwanshi, and A. N. Jaiswal, "Genetic algorithm based clustering: a survey," in *2008 First International Conference on Emerging Trends in Engineering and Technology*, 2008: IEEE, pp. 314-319.
[134] A. Thengade and R. Dondal, "Genetic Algorithm–Survey Paper."
[135] T. Ganesan, P. Vasant, and I. Elamvazuthi, *Advances in Metaheuristics: Applications in Engineering Systems*. CRC Press, 2016.
[136] X.-S. Yang, *Nature-inspired optimization algorithms*. Academic Press, 2020.
[137] F. Neri and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis," *Artif Intell Rev,* vol. 33, pp. 61-106, 2010.
[138] S. Das and P. N. Suganthan, "Differential Evolution: A Survey of the State-of-the-Art," *IEEE Transactions on Evolutionary Computation,* vol. 15, no. 1, pp. 4-31, 2011.
[139] K. R. Opara and J. Arabas, "Differential Evolution: A survey of theoretical analyses," *Swarm and Evolutionary Computation,* vol. 44, pp. 546-558, 2019/02/01/ 2019, doi: https://doi.org/10.1016/j.swevo.2018.06.010.
[140] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution – An updated survey," *Swarm and Evolutionary Computation,* vol. 27, pp. 1-30, 2016/04/01/ 2016, doi: https://doi.org/10.1016/j.swevo.2016.01.004.
[141] X. Yang, "Introduction to mathematical optimization," *From Linear Programming to Metaheuristics,* 2008.
[142] M. Gendreau and J.-Y. Potvin, *Handbook of metaheuristics*. Springer, 2010.
[143] W. Sun, M. Tang, L. Zhang, Z. Huo, and L. Shu, "A Survey of Using Swarm Intelligence Algorithms in IoT," *Sensors,* vol. 20, p. 1420, 2020.
[144] K. KAMEYAMA, "Particle Swarm Optimization–A Survey," 2009.
[145] S. M. Chowdhury and A. Hossain, "Different Energy Saving Schemes in Wireless Sensor Networks: A Survey," *Wireless Personal Communications,* vol. 114, pp. 2043-2062, 2020.

[146] J. Singh, R. Kaur, and D. Singh, "A survey and taxonomy on energy management schemes in wireless sensor networks," *Journal of Systems Architecture,* vol. 111, p. 101782, 2020/12/01/ 2020, doi: https://doi.org/10.1016/j.sysarc.2020.101782.
[147] G. Samara and K. M. Blaou, "Wireless sensor networks hierarchical protocols," in *2017 8th International Conference on Information Technology (ICIT)*, 2017: IEEE, pp. 998-1001.
[148] T. Rault, A. Bouabdallah, and Y. Challal, "Energy efficiency in wireless sensor networks: A top-down survey," *Computer Networks,* vol. 67, pp. 104-122, 2014/07/04/ 2014, doi: https://doi.org/10.1016/j.comnet.2014.03.027.
[149] P. Kuila and P. K. Jana, "Energy efficient clustering and routing algorithms for wireless sensor networks: Particle swarm optimization approach," *Engineering Applications of Artificial Intelligence,* vol. 33, pp. 127-140, 2014.
[150] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on wireless communications,* vol. 1, no. 4, pp. 660-670, 2002.
[151] A. R. Khan, N. Rakesh, A. Bansal, and D. K. Chaudhary, "Comparative study of WSN Protocols (LEACH, PEGASIS and TEEN)," in *2015 Third International Conference on Image Information Processing (ICIIP)*, 21-24 Dec. 2015 2015, pp. 422-427, doi: 10.1109/ICIIP.2015.7414810.
[152] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *Proceedings of the 10th annual international conference on Mobile computing and networking*, 2004, pp. 114-128.
[153] C. Nakas, D. Kandris, and G. Visvardis, "Energy efficient routing in wireless sensor networks: a comprehensive survey," *Algorithms,* vol. 13, no. 3, p. 72, 2020.
[154] A. S. Toor and A. K. Jain, "A survey of routing protocols in Wireless Sensor Networks: Hierarchical routing," in *2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, 23-25 Dec. 2016 2016, pp. 1-6, doi: 10.1109/ICRAIE.2016.7939555.
[155] J. Parmar and A. Pirishothm, "Study of wireless sensor networks using leach, teen and apteen routing protocols," *Int J Sci Res (IJSR) ISSN (Online),* pp. 2319-7064, 2015.
[156] Z. Beiranvand, A. Patooghy, and M. Fazeli, "I-LEACH: An efficient routing algorithm to improve performance & to reduce energy consumption in Wireless Sensor Networks," in *The 5th Conference on Information and Knowledge Technology*, 2013: IEEE, pp. 13-18.
[157] A. Razaque, S. Mudigulam, K. Gavini, F. Amsaad, M. Abdulgader, and G. S. Krishna, "H-LEACH: Hybrid-low energy adaptive clustering hierarchy for wireless sensor networks," in *2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, 2016: IEEE, pp. 1-4.
[158] N. Kumar, P. Bhutani, and P. Mishra, "U-LEACH: A novel routing protocol for heterogeneous Wireless Sensor Networks," in *2012 International Conference on Communication, Information & Computing Technology (ICCICT)*, 2012: IEEE, pp. 1-4.
[159] A. Singh, S. Rathkanthiwar, and S. Kakde, "LEACH based-energy efficient routing protocol for wireless sensor networks," in *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, 2016: IEEE, pp. 4654-4658.
[160] A. Braman and G. Umapathi, "A Comparative Study on Advances in LEACH Routing Protocol for Wireless Sensor Networks: A survey," 2014.
[161] R. I. Tandel, "Leach protocol in wireless sensor network: a survey," *International Journal of Computer Science and Information Technologies,* vol. 7, no. 4, pp. 1894-1896, 2016.

[162] M. Dener, "A new energy efficient hierarchical routing protocol for wireless sensor networks," *Wireless Personal Communications,* vol. 101, no. 1, pp. 269-286, 2018.
[163] T. Sharma, B. Kumar, and G. S. Tomar, "Performance Comparision of LEACH, SEP and DEEC Protocol in Wireless Sensor Network," in *Proc. of the Intl. Conf. on Advances in Computer Science and Electronics Engineering*, 2012.
[164] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for WIreless SEnsor networks," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, 26 April-1 May 2015 2015, pp. 513-521, doi: 10.1109/INFOCOM.2015.7218418.
[165] A.-C. Anadiotis, L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "SD-WISE: A Software-Defined WIreless SEnsor network," *Computer Networks,* vol. 159, pp. 84-95, 2019/08/04/ 2019, doi: https://doi.org/10.1016/j.comnet.2019.04.029.
[166] P. M. Egidius, A. M. Abu-Mahfouz, and G. P. Hancke, "Programmable node in software-defined wireless sensor networks: A review," in *IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society*, 2018: IEEE, pp. 4672-4677.
[167] A. Varga, "OMNeT++," in *Modeling and tools for network simulation*: Springer, 2010, pp. 35-59.
[168] M. L. Rajaram, E. Kougianos, S. P. Mohanty, and U. Choppali, "Wireless sensor network simulation frameworks: A tutorial review: MATLAB/Simulink bests the rest," *IEEE Consumer Electronics Magazine,* vol. 5, no. 2, pp. 63-69, 2016.
[169] J. Eriksson *et al.*, "COOJA/MSPSim: interoperability testing for wireless sensor networks," in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, 2009, pp. 1-7.