RESOURCE ORCHESTRATION IN SOFTWARE-DEFINED EDGE NETWORKS FOR IOT

Ph.D. Thesis

By

LALITA AGRAWAL

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

FEBRUARY 2026

RESOURCE ORCHESTRATION IN SOFTWARE-DEFINED EDGE NETWORKS FOR IOT

A THESIS

submitted to the

INDIAN INSTITUTE OF TECHNOLOGY INDORE

in partial fulfillment of the requirements for
the award of the degree of

DOCTOR OF PHILOSOPHY

by

LALITA AGRAWAL

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

FEBRUARY 2026

INDIAN INSTITUTE OF TECHNOLOGY INDORE

I hereby certify that the work which is being presented in the thesis entitled **Resource Orchestration in Software-Defined Edge Networks for IoT** in the partial fulfillment of the requirements for the award of the degree of **Doctor of Philosophy** and submitted in the **Department of Computer Science and Engineering, Indian Institute of Technology Indore,** is an authentic record of my own work carried out during the time period from January 2022 to February 2026 under the supervision of Dr. Ayan Mondal, Assistant Professor, Indian Institute of Technology Indore, Indore, India.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other institute.

Lalita 24/02/2026

Signature of the Student with Date
(Lalita Agrawal)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

24/02/2026

Signature of Thesis Supervisor with Date

(Dr. Ayan Mondal)

Lalita Agrawal has successfully given her Ph.D. Oral Examination held on **February 24, 2026**.

24/02/2026

Signature of Thesis Supervisor with Date

(Dr. Ayan Mondal)

ACKNOWLEDGEMENT

I wish to seize this opportunity with much gratitude in my heart to many of those who guided and inspired me in the accomplishment of this dissertation, which marks another important milestone in my life. I thank God for giving me the strength, wisdom, and perseverance to undertake and complete this journey.

First and foremost, I want to express my sincere thanks and heartfelt gratitude to my supervisor, **Dr. Ayan Mondal**, for his immense support, vision, and unwavering motivation throughout my PhD tenure. He has always given me the freedom to explore research ideas and develop my technical and analytical skills. His involvement of me in workshops and teaching activities significantly enhanced my technical skills. I am profoundly grateful to him for patiently correcting my writing mistakes and continuously keeping me engaged in my PhD work. His consistent guidance steered me in the right direction and helped transform me into an independent researcher.

My profound thanks to **Prof. Suhas Joshi**, Director, IIT Indore, for providing a competitive research environment in the institute. Besides, I extend my gratitude to my PhD student's progress committee (PSPC) members, **Prof. Neminath Hubballi** and **Dr. Saptarshi Ghosh**, for their keen observation, valuable comments, insightful suggestions, encouragement, and validation of the research work. I would also like to express my appreciation to **Dr. Ranveer Singh**, Head of the Department of Computer Science and Engineering for all his extended suggestions and support. I also convey sincere thanks to **Dr. Soumi Chattopadhyay**, DPGC Convener, Department of Computer Science and Engineering, **Prof. Aruna Tiwari**, and **Dr. Aniruddha Singh Kushwaha** for their support and motivation all the times. I also thank all the faculty members from the Department of Computer Science and Engineering for their helpful advice and constant encouragement. I am also grateful to the department technical staff members for their unfailing support and assistance. Special gratitude goes to the IIT Indore for providing me with an opportunity to pursue a PhD under a state-of-the-art research environment and the Ministry of Education (MoE), formerly the Ministry of Human Resource Development (MHRD), Government of India, for granting financial aid to carry out this doctoral study successfully.

I would also like to express my sincere gratitude to **Prof. M. S. Obaidat** for providing valuable suggestions and guidance for my research work. I am also thankful to the Anusandhan National Research Foundation (ANRF) and IIT Indore for providing the financial grant that enabled me to attend and present my work at the LANMAN 2025 conference. It was an enriching experience to interact with researchers across the globe and to gain exposure to the cultural heritage of France.

I would like to express my sincere gratitude to **Dr. Erkki Harjula** for supporting my research visit at the University of Oulu, Finland. I deeply admire the research culture, hospitality, and academic environment that I experienced during my stay. I would also like to thank my seniors, **Dr. Chandan Kumar Singh** and **Dr. Deepak Kumar**, and my juniors cum friends, **Smriti Uniyal** and **Ashutosh Prajapati**, for their constant support and encouragement throughout the visit. The time I spent in Finland was truly memorable; experiencing the Northern Lights, immersing myself in Finnish culture, and exploring the country made the research visit even more enriching. I will cherish this experience for my entire life, as it was my first international trip, and it would not have been possible without the support of all of you.

The best part of this special journey has been the company of cheerful doctoral fellows as well as postgraduate students, including **Dr. Anuj Rai**, **Col. V. K. Datta**, **Aditya Kanade**, **M. Harsh Kumar**, **Antim Jain**, and **Saket Jain**. My friends have played a significant role in bringing out the best in me and have been a strong support system, encouraging me to take brave decisions irrespective of the outcome. I would like to express my heartfelt thanks to **Md Sajid**, **Mushir Akhtar**, **Mayank Ramnani**, **Doli Uppal**, and **Dr. Nabasmita Phukan**.

Nothing would have been possible without the moral support of **my parents**, who have been the pillars of strength in all my endeavors. Without their unconditional love, sacrifice, support and encouragement I would have never come this far. I am always deeply indebted to them for all that they have done for me. I am grateful to **my mother** for always reminding me to eat well and take care of my health and **my father** for his support during difficult moments. I also thank my younger brother **Yamnesh Agrawal**, without his help this journey would not have been possible. Short technical discussions with him would solve weeks worth of confusion and bring more clarity to work. Also, I clearly remember that whenever I was scared to visit a new city for work or an exam, he always accompanied me. I also thank my elder brother, sister-in-law, and all siblings for their constant support. I also express my love to my nieces and nephews, whose innocence brought me immense joy throughout this journey.

Lalita

Lalita Agrawal
Indian Institute of Technology Indore
February 2026

Dedicated to

Almighty God

and

My Parents

ABSTRACT

In recent years, the proliferation of Internet of Things (IoT) applications has resulted in a substantial increase in heterogeneous and dynamic data traffic, demanding stringent quality-of-service (QoS) guarantees such as high throughput, low delay, strong reliability, and energy efficiency. Traditional network architectures with static configurations and vendor-specific constraints are not capable of handling dynamic traffic patterns. Software-Defined Networking (SDN) integrated with edge computing offers a programmable and flexible networking paradigm with centralized control. However, achieving effective resource orchestration across Software-Defined Edge Networks (SDEN) and softwarized 5G/6G environments remains challenging due to heterogeneous traffic characteristics, dynamic IoT workloads, and limited network resources.

We propose a comprehensive set of analytical and heuristic orchestration frameworks to address throughput optimization, delay reduction, traffic management, and energy efficiency in SDEN and softwarized 5G/6G networks. The first work introduces T-RESIN, a throughput-aware resource orchestration framework that uses an *evolutionary game* to determine equilibrium-driven flow distribution across SDN switches and edge nodes. Building on this, D-RESIN presents a delay-aware orchestration mechanism based on an *evolutionary game-theoretic approach* to minimize processing delay for delay-sensitive IoT applications. For next-generation wireless networks, TRON is proposed as an SDN-based traffic management architecture that leverages *OpenFlow Group Tables* to dynamically balance heterogeneous traffic and improve link utilization. Further, we propose FALCON, an energy-efficient bandwidth orchestration scheme along with its variants — S-FALCON and D-FALCON that optimizes *meter table* configurations and transmission strategies to reduce retransmission energy while ensuring high throughput.

All proposed frameworks are implemented and evaluated using Mininet integrated with the Ryu SDN controller and Open vSwitch. Experimental results demonstrate substantial improvements over baseline and state-of-the-art approaches, showcasing gains in network throughput, reduction in processing delay, improved load balancing, enhanced link utilization, and significant energy savings. The collective contributions of this thesis provide a unified foundation for adaptive, scalable, and QoS-aware resource orchestration in SDENs for 5G/6G-enabled IoT environments. The findings further offer valuable direction for future research on intelligent, autonomous, and data-driven network management.

Keywords: Internet of Things, Software-Defined Networking, Edge Computing, Software-Defined Edge Networks, Evolutionary Game, 5G/6G Network, Traffic Management, Mininet Emulator.

PUBLICATIONS FROM THE THESIS

The following mentioned publications have evolved from this doctoral dissertation:

In Refereed Journals

Published / Accepted:

1. **L. Agrawal**, and A. Mondal, "Energy-Efficient Bandwidth Orchestration for Industrial IoT in Softwarized 6G Networks," *IEEE Internet of Things Journal*, vol. 12, no. 19, pp. 40440-40448, October 2025. DOI: 10.1109/JIOT.2025.3591029

2. **L. Agrawal**, A. Mondal, M. S. Obaidat, and E. Harjula, "Delay-Aware Dynamic Resource Orchestration for IoT-Enabled Software-Defined Edge Networks," *International Journal of Communication Systems*, Wiley, vol. 38, no. 7, pp. e70072, March 2025. DOI: 10.1002/dac.70072

3. **L. Agrawal**, A. Mondal, and M. S. Obaidat, "T-RESIN: Throughput-Aware Dynamic Resource Orchestration for IoE-Enable Software-Defined Edge Networks," *International Journal of Communication Systems*, Wiley, vol. 37, no. 12, pp. e5802, April 2024. DOI: 10.1002/dac.5802

In Refereed Conferences

1. **L. Agrawal**, and A. Mondal, "TRON: Traffic Management and Resource Allocation for SDN-Enabled 5G/6G Networks," in *Proceedings of the 31st IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, Lille, France, July 7–8, 2025, pp. 1-2. DOI: 10.1109/LANMAN66415.2025.11154616 (Poster Paper)

The publication other than this doctoral dissertation is mentioned below:

In Refereed Book Chapters

1. T. Kumar, J. Partala, T. Nguyen, **L. Agrawal**, A. Mondal, A. Kumar, I. Ahmad, E. Peltonen, S. Pirttikangas, and E. Harjula, "Secure Edge Intelligence in the 6G Era," in *Security and Privacy for 6G Massive IoT*, G. Mantas, F. Saghezchi, J. Rodriguez, and V. Sucasas (Eds.), Wiley-VCH Verlag, 2025, pp. 35–52, ISBN(Electronic): 9781119988007. DOI: 10.1002/9781119988007.ch2

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations and Symbols

List of Abbreviations

5G	Fifth-Generation Wireless Networks
6G	Sixth Generation Wireless Networks
AI	Artificial Intelligence
AIoT	Artificial Intelligence of Things
API	Application Programming Interface
B5G	Beyond 5G Network
BWA	Bandwidth Allocation
D2D	Device-to-Device
DNN	Deep Neural Network
DRL	Deep Reinforcement Learning
eMBB	Enhanced Mobile Broadband
HAS	HTTP Adaptive Streaming
HFC	Hybrid Fiber-Coaxial Network
ICPS	Industrial Cyber-Physical Systems
IIoT	Industrial Internet of Things

IoE	Internet of Everything
IoT	Internet of Things
LAN	Local Area Network
LTE	Long Term Evolution
MAN	Metropolitan Area Network
MDP	Markov Decision Process
MEC	Multi-Access Edge Computing
MEMS	Micro-Electro-Mechanical Systems
mMTC	massive Machine-Type Communication
NFV	Network Function Virtualization
NP	Non-Deterministic Polynomial Time
OAI	OpenAirInterface (5G Platform)
OVS	Open vSwitch
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
REST	Representational State Transfer
SD-IoV	Software-Defined Internet of Vehicles
SDEN	Software-Defined Edge Networks
SDN	Software-Defined Network
SFC	Service Function Chain
TCAM	Ternary Content-Addressable Memory

TSN	Time-Sensitive Networking
UAV	Unmanned Aerial Vehicle
UDP	User Datagram Protocol
URL	Uniform Resource Locator
URLLC	Ultra-Reliable Low-Latency Communication
VM	Virtual Machine
VoIP	Voice over Internet Protocol
WiFi	Wireless Fidelity

List of Symbols

$\mathcal{N}$	Set of IoT/IoE/IIoT devices/things
$\mathcal{S}$	Set of SDN switches
$\mathcal{E}$	Set of Edge nodes
f_n	Data flow generated by IoE device $n \in \mathcal{N}$
B_s	Bandwidth associated with switch $s \in \mathcal{S}$
$P_n(t)$	Processes required by IoE end-device $n \in \mathcal{N}$
d_i	Data generation rate of flow $i \in (\mathbb{Z}^+ \cap [0, f_n])$
C_e	Computational capacity of edge node $e \in \mathcal{E}$
M_e	Memory capacity of edge node $e \in \mathcal{E}$
P_s	Processes associated with switch $s \in \mathcal{S}$
R_s^{max}	Maximum number of flow rules in TCAM of switch $s \in \mathcal{S}$
λ_p	Total delay for process $p \in P_n(t)$

λ_n	Total processing delay for IoE device $n \in \mathcal{N}$
P_e	Processes associated with edge node $e \in \mathcal{E}$
E^e	Total energy associated with edge node $e \in \mathcal{E}$
ω	Evolutionary iteration for SDN switches
ϕ	Evolutionary iteration for edge nodes
F_n	Set of flows generated by each IoT device $n \in \mathcal{N}$
$V(s)$	Volume of data associated with each switch $s \in \mathcal{S}$
$Pd_s^{avg}(f, n, s)$	Average processing delay of SDN switch $s \in \mathcal{S}$ for any flow f
μ_s	Processing rate for SDN switches
$V(e)$	Volume of data associated with each edge node $e \in \mathcal{E}$
$C_{use}(e)$	Computational power used by edge node $e \in \mathcal{E}$
$Pd_e^{avg}(f, s, e)$	Average processing delay of edge node $e \in \mathcal{E}$ for any flow f
μ_e	Processing rate for edge nodes
M_{xm}	Meter table of each SDN switch $s \in \mathcal{S}$
$P_{drop}(s)$	Packet drop at each switch $s \in \mathcal{S}$
E_{drop}	Retransmission energy required for dropped packet
F_s	Number of flow rules associated with each switch s
E_{Tx}	Energy cost to transmit/retransmit one unit of data traffic
$\mathcal{L}$	Set of physical links between SDN switches
C_l	Capacity of physical link $l \in \mathcal{L}$
$\mathcal{B}_l$	Set of buckets in the Group Table of link $l \in \mathcal{L}$
w_b	Weight assigned to each bucket $b \in \mathcal{B}_l$ in a Group Table

$\lambda_b(t)$	Traffic allocated to bucket $b \in \mathcal{B}_l$ at time t
θ	Congestion threshold for link utilization

Chapter 1

Introduction

With the advent of Internet of Things (IoT), the number of devices increased significantly. IoT represents a transformative evolution in the digital landscape, where interconnected devices and sensors seamlessly integrate to collect, exchange, and process data in real-time [1–4]. Moreover, Internet of Everything (IoE) considers not only the collection of raw data and interconnecting things, but also the decision-making process and people [5, 6]. Hence, IoE is a superset of the IoT by providing multiple forms of communication to IoT [7, 8]. In IoE, the process analyses the real-time data and decides the activity to make IoE an automated intelligence system. However, the exponential growth of connected devices has introduced critical challenges in network management, scalability, and Quality of Service (QoS). Conventional network architectures rely on static configurations, vendor-specific hardware, and lack the flexibility to dynamically adapt the varying traffic demands and heterogeneous data flows generated by IoT and IoE ecosystems. For that purpose, Software-Defined Networking (SDN) has emerged as a programmable and adaptive networking paradigm that decouples the control plane from the data plane, enabling centralized control, dynamic configuration, and efficient utilization of network resources [9–13]. As per the current scenario, traditional IoT network infrastructures often struggle with latency issues, inefficient bandwidth utilization and cannot meet the dynamic resource demands of IoT devices effectively. Additionally, the data generated by IoT devices are heterogeneous in terms of the volume and variety. Therefore, there is a need for an efficient resource allocation mechanism that can accommodate heterogeneous IoT devices and services while maximizing bandwidth utilization and improving overall network performance.

The rest of this chapter is organized as follows. Section 1.1 presents the necessary background concepts relevant to IoT, networking, and programmable infrastructures. Section 1.2 discusses the motivation and scope of this thesis, emphasizing the challenges and requirements in resource orchestration for SDN-enabled edge and IoT environments. Section 1.3 states the key objectives of the research conducted. Section 1.4 summarizes the major contributions of the thesis. Finally, Section 1.5 concludes the chapter by briefly describing the overall organization of the thesis.

1.1 Background

This section provides an overview of the key technologies that form the foundation of this research. The subsequent subsections outline the evolution of SDN, advancements in 5G and 6G networks, and the significance of edge-enabled IoT in achieving reliable and adaptive industrial communication systems.

1.1.1 Software-Defined Edge Networks (SDEN)

Traditional IoT architectures relying on centralized cloud computing suffer from high latency, increased response time and inefficient bandwidth utilization, making them unsuitable for delay-sensitive and mission-critical applications [14–18]. In contrast, edge computing offers a distributed architecture that enables computation and storage closer to the data sources [19–25]. This significantly reduces network congestion and response delay while improving bandwidth utilization and scalability. Edge computing distributes networking load to multiple edge nodes to offer reliability, security, and privacy protection for IoT applications. The edge devices are vendor specific and cannot be integrated seamlessly. To overcome these interoperability and resource-management challenges, this research proposes a SDN-enabled edge architecture, named as Software-Defined Edge Networks (SDENs). For SDEN networks, the SDN controller[1] has two types of application programming interfaces (APIs) for three-tier architecture — northbound and southbound APIs. Open-source Apache 2.0 [2] has licensed Open vSwitches as a high-quality multilayered switch for network or-

[1]https://opennetworking.org/software-defined-standards/
[2]https://www.apache.org/

chestration. The OpenFlow protocol is a southbound interface between Open vSwitches and the SDN controller. The REST API is a northbound interface between the SDN controller and applications. The Ryu controller is also actively developed for research activities like SDN OpenFlow rules placement management in its Ternary Content Addressable Memory (TCAM) compared to other SDN controllers like Pox. The generic architecture of the SDEN is illustrated in Figure 1.1. It shows the interaction among IoT devices, SDN switches, and edge nodes coordinated by the SDN controller, which manages policies through northbound and southbound APIs.

In existing literature, the researchers focused on the task offloading in the edge networks and resource allocation in SDN. However, the task offloading in the SDEN remains a challenging task due to several factors such as limited bandwidth and flow space of the SDN edge switches and processing capacity of the edge devices. Therefore, there is a need for resource allocation schemes for SDENs in the presence of heterogeneous IoT devices and services, while maximizing the bandwidth utilization of the edge networks. Enhancing the scalability and efficiency of edge-computing environments requires dynamic resource orchestration based on real-time traffic and application demands, ensuring optimal bandwidth utilization and improved performance and reliability of IoT-enabled SDENs.

1.1.2 Next-Generation Wireless Networks: 5G and 6G

Local Area Networks (LANs) and Metropolitan Area Networks (MANs) increasingly support diverse and dynamic services that demand fine-grained resource control, and adaptive traffic management. These capabilities are crucial for meeting modern service requirements. An adaptive framework empowers these networks with real-time flow optimization, intelligent traffic splitting, and enhanced link utilization — delivering superior performance, resilience, and service quality beyond traditional static approaches. The emergence of SDN and network slicing has revolutionized modern network architectures by enabling dynamic, flexible, and programmable network management [26, 27]. Network slicing [28] is a fundamental concept in 5G and emerging 6G networks to facilitate the creation of multiple virtual network instances over a shared physical infrastructure. Each slice is specifically configured to meet the distinct performance requirements of diverse applications, including ultra-reliable low-latency communication (URLLC), massive machine-type communication

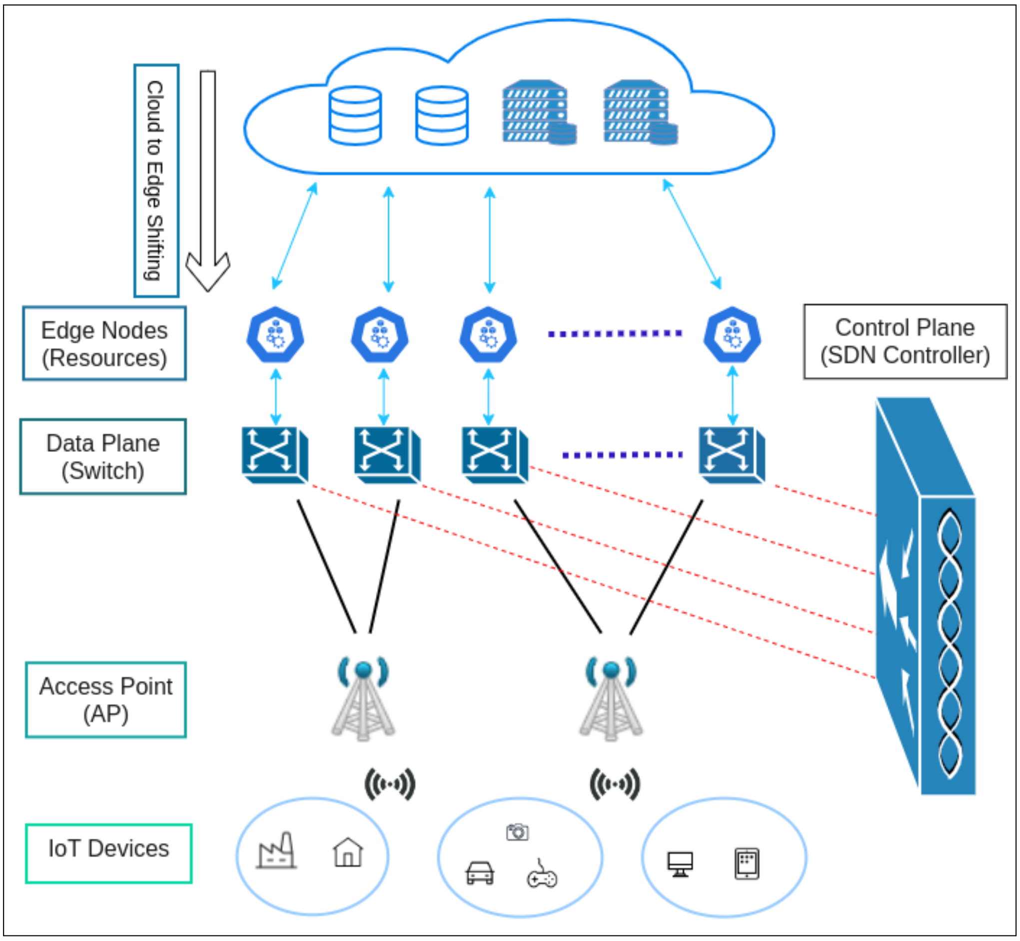


Figure 1.1: Generic Architecture of Software-Defined Edge Networks

(mMTC), and enhanced mobile broadband (eMBB) [29–31]. Additionally, in IoT network, where heterogeneous devices generate massive amounts of data, the need for intelligent and adaptive slicing strategies becomes even more critical. One of the key challenges in SDN-based network slicing is ensuring efficient resource allocation, adaptive traffic handling, and network resilience [11,32,33]. An extensive network of IoT devices can be deployed to continuously monitor traffic patterns, optimize channel delays, and mitigate congestion in real time. For instance, during a major public event, such as the Olympics, a dense deployment of 6G-enabled IoT sensors and cameras across the city can provide real-time data on traffic conditions. In such scenarios, edge networks play a crucial role by dynamically allocating communication and computational resources to manage the rapidly varying traffic load. This real-time adaptability significantly reduces latency and ensures uninterrupted connectivity even during sudden data surges or emergency situations. Leveraging the ultra-low latency and high responsiveness of 6G technology, the network can autonomously adjust routing decisions, regulate traffic signals, and perform intelligent rerouting to prevent bottlenecks — thereby enhancing overall efficiency, reliability, and quality of service across the urban network.

1.1.3 Industrial Internet of Things (IIoT)

The rapidly evolving telecommunications landscape has witnessed significant advancements with the introduction of fifth-generation (5G) networks [4, 26, 29]. However, as the demand for higher data rates, ultra-low latency, and improved network reliability continues to grow, researchers are making an effort toward the development of sixth-generation (6G) wireless networks. 6G promises to offer unprecedented capabilities, while surpassing its predecessors [30, 31]. We envision that the advancement of 6G networks will also have a high impact on the Industrial Internet of Things (IIoT). The presence of IIoT in Industry 5.0 highlights the need for energy-efficient solutions, where intelligent resource allocation aligns energy consumption with operational demands [34, 35]. It also emerges as a pivotal aspect of future network design, aligning with global sustainability goals and addressing climate change concerns. Though several schemes are proposed for IIoT applications in the existing literature, there is a need to address the challenges posed by heterogeneous bandwidth management and the integration of emerging Industry 5.0 technologies, such as network bandwidth optimization, SDN, and energy-efficient mechanisms, to achieve sustainable and high-performance 6G networks [11, 20, 36, 37]. Figure 1.2 depicts the generic architecture of edge-enabled IIoT within 5G/6G networks, where edge nodes act as intermediaries between IIoT devices and the core network to ensure low-latency and efficient communication.

For instance, an industrial smart factory deploying IIoT devices connected through a softwarized 6G network. The factory generates diverse types of traffic, including real-time video surveillance, Voice over Internet Protocol (VoIP) communications, and periodic data from sensors. For these heterogeneous traffic in dynamic IIoT environments, the existing work often results in either underutilized link capacity or congestion, leading to increased retransmission energy. This necessitates a dynamic SDN-enabled orchestration framework that can adapt bandwidth provisioning in real time while minimizing energy consumption in softwarized 6G networks [38–40]. However, to further enhance the adaptability and intelligence of such decentralized edge architectures, there is a growing need to integrate Artificial

Intelligence (AI) techniques into the edge computing paradigm [41]. AI technologies are envisioned to serve as a support system for fast data processing, i.e., analyzing and extracting large amounts of data generated by IIoT devices. Additionally, the integration of AI with edge computing supports multiple heterogeneous applications in terms of data types, data handling, and offloading, while ensuring minimal latency [42,43].

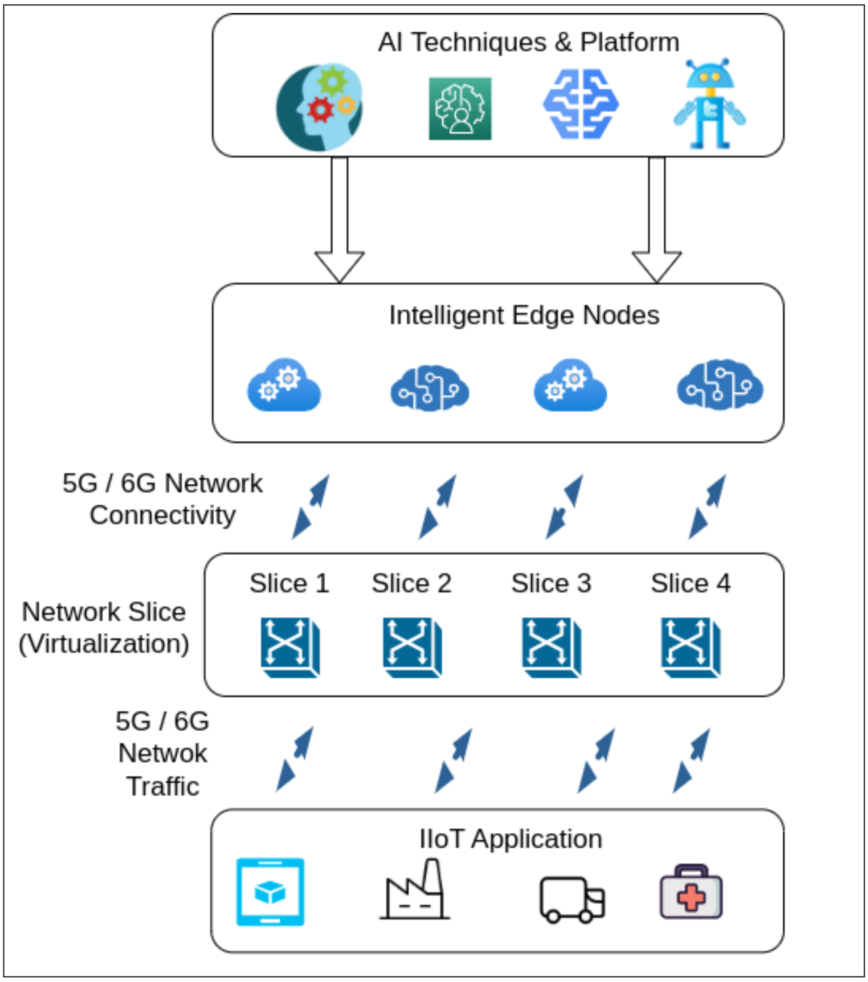


Figure 1.2: Generic Architecture of Edge-Enabled IIoT in 5G/6G networks

1.2 Motivation and Scope of the Work

This study aims to design adaptive and energy-efficient resource orchestration frameworks for SDENs and 5G/6G-enabled IIoT environments that jointly optimize throughput, latency, and energy efficiency. The research has been motivated by the following observations drawn from the state-of-the-art works in SDN, edge computing, and IIoT resource management.

- Existing literature does not provide a unified resource-orchestration method that satisfies all IoE network features. There is requirement for dynamic resource allocation method for IoE-enabled SDEN with heterogeneous data flow management.

- The existing SDN-based bandwidth allocation models are static or heuristic and operate mainly at the controller level, ignoring real-time flow variations. This leads to inefficient link utilization, degraded throughput, and limited responsiveness to dynamic IoT traffic. Moreover, the existing multi-tier frameworks do not have intelligent decision mechanisms for self-adaptive bandwidth and flow-space management.

- Delay-sensitive IoT applications require low-latency access to essential computing and networking resources. However, the current SDN-enabled edge systems lack effective QoS-driven mechanisms that can appropriately prioritize traffic and orchestrate resources based on application-specific deadlines.

- The emergence of 5G and 6G architectures introduces ultra-reliable low-latency and massive machine-type communication services. However, the current resource-allocation schemes lack adaptive orchestration across heterogeneous traffic such as data, VoIP, and video.

- The existing work do not fully address the complex requirements of 6G networks, especially in terms of dynamic bandwidth allocation and real-time spectrum management. Optimizing the energy consumption at SDN switches and edge nodes is overlooked in the existing literature; retransmissions and redundant flows increase operational cost and carbon footprint in large-scale IIoT deployments.

These limitations collectively motivate the need for a unified SDN-driven orchestration framework that integrates throughput-aware, delay-aware, traffic-adaptive, and energy-efficient mechanisms suitable for SDENs in the context of next-generation IoT, 5G, and 6G infrastructures.

1.3 Objectives

The primary objective of this research is to design a unified and adaptive resource-orchestration framework for SDENs and 5G/6G-enabled IoT environments to achieve optimal throughput, delay, and energy efficiency under heterogeneous conditions. The objectives, as illustrated in Figure 1.3, are outlined as follows.

1. Design of a throughput-aware resource-orchestration scheme for SDENs to maximize bandwidth utilization and fairness among IoE devices.

2. Design of a delay-aware orchestration scheme to minimize end-to-end latency for delay-sensitive applications in SDENs.

3. Design of an adaptive traffic-management scheme for heterogeneous network traffic in SDN-enabled 5G/6G networks.

4. Design of an energy-efficient bandwidth-orchestration framework for IoT in softwarized 6G to enhance throughput and energy sustainability.

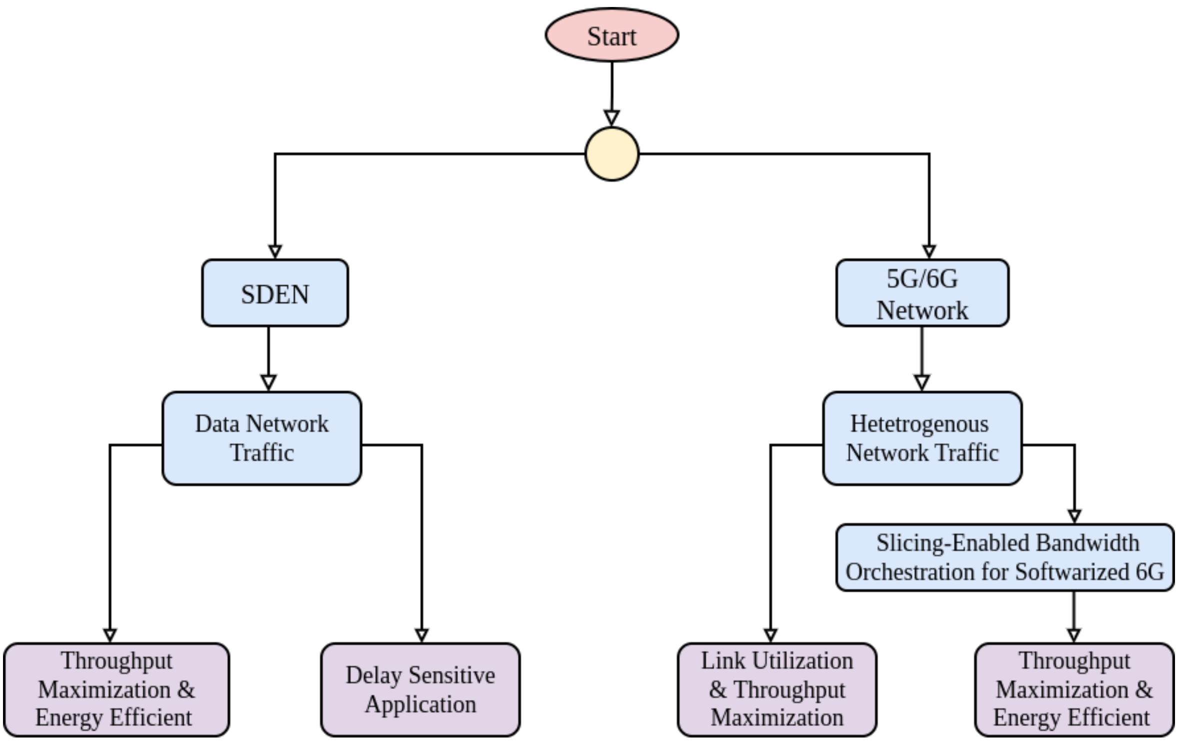


Figure 1.3: Overview of the Thesis Objectives Across SDEN and 5G/6G-Enabled IoT

1.4 Contributions of the Thesis

The major contribution of this work are listed as follows:

1. We propose T-RESIN, a throughput-aware dynamic resource allocation scheme, for IoE-enabled SDENs. The proposed scheme addresses the challenges of allocating heterogeneous IoE data flows across both the access tier and the edge tier, while satisfying bandwidth and computational constraints. The interactions among the entities — IoT devices, Ryu SDN controller/Open vSwitches, and the edge computing nodes — are modelled using evolutionary game-theoretic approach. We evaluated the performance of the proposed scheme, T-RESIN, theoretically and evaluated the performance of T-RESIN using Mininet network emulator while comparing with the existing state-of-the-art schemes.

2. We present D-RESIN, a dynamic resource allocation scheme for delay-sensitive IoT applications in the SDEN network. The proposed scheme ensures optimal processing delay between IoT devices, Open vSwitches, and Edge nodes— are programmed by the Ryu SDN controller. We evaluate the existence of evolutionary equilibrium for D-RESIN in access and edge tiers. Furthermore, we implemented the proposed schemes using the Mininet emulation environment. The proposed approach significantly reduces delay compared to existing schemes and achieves superior QoS for delay-sensitive applications.

3. We propose TRON, a heuristic-driven adaptive traffic management and resource allocation framework for SDN-enabled 5G/6G networks. The proposed framework leverages OpenFlow's Group Tables to dynamically optimize link utilization and traffic distribution across network paths. To effectively handle heterogeneous traffic patterns, TRON introduces dynamic traffic redirection using bucket weight adjustments for VoIP and data flows originating from IoT devices. We integrate the proposed TRON mechanism with the Ryu SDN controller to enable real-time flow redirection and validate it's effectiveness through a practical testbed using Mininet and Open vSwitch. Furthermore, the experimental results demonstrate substantial improvements in network throughput and overall resource utilization over the existing schemes.

4. We introduce FALCON, a novel energy-efficient bandwidth orchestration for SDN-enabled 6G networks, which integrates SDN, network slicing, and a heuristic-based bandwidth allocation strategy to address the challenges of heterogeneous traffic demands and sustainability. The proposed framework utilizes programmable meter tables to perform dynamic bandwidth distribution at runtime, thereby reducing packet loss and enhancing network throughput for IoT applications. The FALCON framework employs network bandwidth slicing to manage different traffic types, such as video, VoIP, and IoT data, within distinct slices. The performance of FALCON is bench marked against the existing schemes for diverse IoT traffic scenarios using Mininet network emulator and the Ryu SDN controller.

1.5 Organization of the Thesis

The rest of the thesis is organized as follows:

Chapter 2 – Literature Survey

This chapter presents a comprehensive review of the related literature on resource allocation in IoT, IoE, and edge computing environments. It highlights state-of-the-art approaches addressing scalability, latency, and energy efficiency challenges in SDN, IIoT, and 6G environment.

Chapter 3 – T-RESIN: Throughput-Aware Resource Orchestration Scheme

This chapter introduces *T-RESIN*, a throughput-aware and energy-efficient dynamic resource orchestration scheme, for SDEN. The evolutionary game formulation, theoretical analysis, and performance evaluation are examined comprehensively.

Chapter 4 – D-RESIN: Delay-Aware Resource Orchestration Scheme

This chapter presents *D-RESIN*, a delay-aware dynamic resource orchestration scheme. It analyzes delay-sensitive bandwidth allocation using an evolutionary game for SDEN.

Chapter 5 – TRON: Traffic Management and Resource Allocation for SDN-Enabled 5G/6G Networks

This chapter develops *TRON*, a traffic-management and resource-allocation framework, for SDN-enabled 5G/6G networks. It uses OpenFlow's Group Table mechanisms for traffic handling to improve throughput, link utilization, and overall network performance.

Chapter 6 – Energy-Efficient Bandwidth Orchestration for SDN-Enabled 6G Networks

This chapter presents the *FALCON* framework, an energy-efficient bandwidth orchestration scheme for IoT within softwarized 6G networks. It leverages programmable meter tables for adaptive bandwidth distribution and incorporates two bandwidth allocation schemes, *S-FALCON* and *D-FALCON*, as elaborated in the chapter.

Chapter 7 – Conclusions and Future Research Directions

This chapter concludes the thesis by summarizing the key findings and contributions of the research. It also outlines potential directions for future exploration toward intelligent, adaptive, and sustainable resource orchestration in SDN-enabled edge and IoT ecosystems.

Chapter 2

Literature Survey

In this chapter, with the expansion of the IoT and IoE, the need for efficient resource allocation across heterogeneous and large-scale networks is heightened. With the integration of SDN and edge computing, we aim to focus on improving scalability, latency, and energy efficiency through novel allocation models, and delay-aware mechanisms. The existing works cover a wide range of directions including resource allocation for SDN-enabled IoT/IoE environments, solutions for delay-sensitive applications, as well as dynamic strategies for multi-tier edge architectures, and orchestration techniques in 5G and 6G networks. Together, these studies provide the groundwork for developing adaptive and intelligent approaches that support reliable and efficient IoT systems.

The rest of the chapter is organized as follows. Section 2.1 reviews resource allocation strategies for IoT and IoE. Section 2.2 discusses resource orchestration schemes in SDN-enabled IoT networks, while Section 2.3 focuses on delay-sensitive IoT applications. Section 2.4 highlights the dynamic allocation methods in three-tier edge architectures, and Section 2.5 presents resource orchestration approaches for 5G and 6G networks. Finally, Section 2.6 concludes the chapter with a summary of insights and open challenges.

2.1 Resource Allocation for IoT and IoE Ecosystems

In this section, we focus on two broad areas. Firstly, we study the IoE, how it differs from IoT, and the components and enabling techniques for IoE. Secondly, we survey the existing work related to resource allocation techniques for IoT and IoE; and bandwidth distribution among SDN switches. So, we can analyze the network's overall performance and QoS using different approaches. To highlight the broader implications of IoE, Langleya *et al.* [44] studied the business model for IoE at three-level that are micro, meso, and macro. The authors described the potential impact of IoE on current business models and the value creation processes by firms and their customers. Liu *et al.* [45] presented the Unmanned aerial vehicle (UAV) enabled IoE to enhance the critical aspect of IoE that are scalability, intelligence, and diversity. Addressing the computational requirements of intelligent IoE ecosystems, Singh and Gill [46] surveyed edge computing techniques for AI-driven IoE platforms. From a hardware integration perspective, Iannacci et al. [47] explored Micro-Electro-Mechanical Systems (MEMS) technologies to support communication and energy harvesting in IoT/IoE and tactile internet devices.

In terms of resource management, Manogaran *et al.* [48] proposed a deep learning-based mutable service distribution model to reduce service overlap and improve QoS in wireless network-assisted IoE environments. In the context of healthcare-oriented IoE systems, Wang *et al.* [49] introduced a deep-reinforcement-learning framework that jointly models patient health states and optimizes IoE sensor resource allocation in near-field healthcare networks for improved monitoring accuracy and efficiency. At the communication layer, Chae *et al.* [50] proposed a joint sub-band assignment and power allocation strategy to maximize IoT cellular network throughput while mitigating inter-band interference with Long Term Evolution (LTE) systems. Additionally, Kim *et al.* [51] proposed a DNN partition placement and resource allocation framework that leverages heterogeneous IoT and edge devices to collaboratively execute multiple deep neural network applications with reduced delay and energy consumption. To improve access efficiency in massive IoT deployments, Guo *et al.* [52] presented a QoS-aware joint random access control and resource allocation scheme that reuses colliding preambles to improve access efficiency and reduce delay in prioritized massive IoT networks. Furthermore, Yang *et al.* [53] developed a UAV-assisted IoT resource allocation and control model that reduces transmission delay and energy consumption while improving communication robustness under dense user scenarios. However, existing schemes do not adequately address dynamic bandwidth allocation and flow control. This necessitates SDN-enabled orchestration mechanisms to achieve scalable and QoS-aware resource management.

2.2 Resource Orchestration for SDN-Enabled IoT Network

SDN has emerged as a critical solution for improving network resilience, QoS, and efficient resource utilization in IoT environments. Son and Buyya [54] introduced two algorithms — Priority-Aware virtual machine allocation (PAVA) and bandwidth allocation for priority applications (BWA), ensuring that high-priority applications meet QoS requirements despite network congestion in SDN-enabled cloud data centers. In hybrid fiber-coaxial (HFC) networks, Bentaleb *et al.* [55] presented SDN-based bandwidth broker for HTTP Adaptive Streaming (HAS). It employed convex optimization to centrally manage bandwidth allocation to improve video streaming stability, and resource utilization. In addition, Wang and You [56] proposed route management framework designed for SDN-based at-tree data center networks to achieve efficient load balancing and overhead reduction. The framework monitored network traffic, computed a dynamic load-deviation parameter to detect potential congestion, and rerouted flows adaptively. Misra *et al.* [57] studied the data broadcasting approach for SDN while incorporating the SDN-enabled edge and core networks. In another work, Yuan *et al.* [58] proposed the network management scheme for software-defined networking in terrestrial and non-terrestrial networks.

In the context of next-generation wireless networks, Jhaveri et al. [59] addressed the challenge of maintaining bandwidth reliability in Industrial Cyber-Physical Systems (ICPS). They developed an SDN-based framework named SDN-RMbw, which utilizes bandwidth contracts and resilience management to dynamically adjust resource allocation and sustain performance under varying network conditions or link failures. Ren *et al.* [60] proposed an orchestration scheme for IoT service function chains in SDN-IoT networks, optimizing both function composition and placement to reduce control latency and bandwidth consumption. Similarly, Bagaa *et al.* [61] proposed a QoS- and resource-aware security orchestration framework for SDN/NFV-enabled IoT systems that optimizes service function chain deployment while ensuring end-to-end performance and security compliance. Beyond static IoT networks, Pokhrel [62] extended SDN concepts to vehicular environments by presenting a Software-Defined Internet of Vehicles (SD-IoV) framework. This architecture enhances vehicular automation, improves scalability, and enables adaptive resource orchestration to maintain network stability and Quality of Experience (QoE) in dynamic mobility scenarios. Naithani *et al.* [27] presented an SDN-enabled distributed access architecture for cable networks that enables centralized control of remote edge nodes to improve load balancing, topology management, and failure handling. Despite these advancements, effective coordination of bandwidth, flow management, and processing across heterogeneous IoT infrastructures remains limited. This gap highlights the necessity of delay-aware and adaptive resource orchestration mechanisms.

2.3 Resource Orchestration in SDN-Enabled IoT Networks for Delay-Sensitive Application

SDN maintains the consistency for distributed control plane architecture as features are no longer limited to what the vendor provides. In this section, we explore strategies and solutions proposed in recent literature to address the inherent trade-offs between real-time network performance and the consistency required for reliable SDN operations. For the purpose mentioned above, Mondal and Misra [63] designed a FlowMan scheme for managing data flows in the presence of elephant and mice flows (Heterogeneous environment). Using the bisection method, the authors used the bounded knapsack approach to associate data flows from IoT devices to SDN switches and ensured optimal network delay and throughput. Another work by Zhang and Zhu [33] proposed an SDN-based framework integrating network function virtualization (NFV) with WiFi and device-to-device (D2D) offloading to enhance statistical QoS for multimedia services in 5G. Balasubramanian *et al.* [64] presented an SDN-based architecture, named TSN_u, aimed at enhancing the performance of time-sensitive industrial IoT networks. The framework is grounded in the IEEE Time Sensitive Networking (TSN) standards, particularly leveraging the IEEE 802.1Qbv and IEEE 802.1Qcc standards for improving transmission time-slot allocations, congestion mitigation, and network stability. Mondal *et al.* [65] proposed an analytical model named OPUS, alongside a specific queuing scheme, to theoretically determine the minimal buffer size requirements of an OpenFlow switch. Extensive simulations demonstrated that an optimized buffer size can significantly enhance packet handling rates while maintaining acceptable packet waiting times. Saha *et al.* [66] proposed a traffic-aware QoS routing scheme within a Software-Defined IoT network framework. It specifically addressed delay-sensitive and loss-sensitive routing strategies for incoming packets from IoT devices.

To effectively handle dynamic network conditions and varying workload demands, Fan *et al.* [67] developed a reinforcement learning-based delay-aware resource allocation scheme that jointly allocates radio and computation resources in fog-assisted IoT networks to minimize task latency under dynamic network conditions. Additionally, Lakew *et al.* [68] proposed an adaptive partial computation offloading and wireless resource harmonization scheme based on deep reinforcement learning to minimize latency and energy consumption in wireless edge computing-assisted IoE networks. Sellami *et al.* [69] presented a deep reinforcement learning-based task scheduling and offloading scheme in SDN-enabled fog–IoT networks to minimize latency and improve energy efficiency. For the same, Xavier *et al.* [70] addressed the challenge of managing computing resources in cloud, edge, and IoT environments to meet the latency and energy efficiency demands of time-sensitive applications. The proposed algorithm is evaluated against non-collaborative and collaborative offloading approaches for edge node utilization. Manogaran *et al.* [71] presented a novel resource allocation method with optimal fog node placement in IoT-Fog-Cloud architecture, aiming to minimize service delays and resource exploitation. These approaches emphasize the value of intelligent offloading and adaptive scheduling. However, end-to-end orchestration across network and edge layers remains an open challenge.

2.4 Dynamic Resource Allocation in Three-Tier Edge Network

The edge computing environment provides a decentralized approach to resource allocation and data processing at the network's periphery/edge. Here, we review recent advancements in dynamic resource allocation methods and integrate a three-tier architecture—consisting of cloud, edge, and end devices — to optimize data processing and latency for SDN-enabling technology. For the same purpose, Misra *et al.* [72] proposed pricing based resource allocation model (FogPrime) and clustered the fog nodes. Authors used dynamic coalition-formation game approach for resource allocation locally within a cluster. In another work, Deb *et al.* [73] proposed a distributed load management scheme using edge platforms for IoT-enabled smart grid environment. Deng *et al.* [74] presented two aspects of edge intelligence: AI for edge and AI on edge. According to the authors, AI for edges provides a better and more intelligent solution to data collection and computation offloading for the mobile environment. AI on edges deals with integrating and executing AI techniques in the edge computing environment. Zhang *et al.* [75] proposed a framework, named OpenEI. The authors constructed a model using lightweight packages of the deep learning algorithm. OpenEI framework aims to find appropriate AI techniques corresponding to edge environments to remove the inconsistency. In another work, Zhu *et al.* [76] identifies the importance of distributed learning, i.e., federated learning, for edge nodes. Using federated learning, the authors addressed many issues using the amalgamation of AI and edge computing. Singh *et al.* [77] proposed a collaborative machine learning-based resource allocation scheme for SDN-enabled fog environments that improves latency, execution time, and energy efficiency. He *et al.* [78] introduced a blockchain-enabled edge computing resource allocation framework that uses a deep reinforcement learning (A3C) algorithm to securely and efficiently assign IoT tasks to edge nodes. Ranjbaran *et al.* [79] proposed an online-learning-based digital twin placement framework for the cloud-edge continuum that dynamically migrates digital twins to minimize latency and improve resource utilization. Additionally, Byeon *et al.* [80] proposed a cloud–edge task offloading and resource allocation framework for Artificial Intelligence of Things (AIoT) systems using deep reinforcement learning and auction-based strategies to improve computational efficiency and edge resource utilization.

Existing research has predominantly focused on optimization-based and learning-driven approaches, including game theory, deep reinforcement learning, and graph-based models for traffic management. Although such methods can achieve strong performance under controlled conditions, these approaches often incur significant controller overhead and exhibit limited adaptability under dynamic traffic patterns in real-time environments.

2.5 Resource Orchestration for 5G and 6G Network

The evolution of wireless communication technologies towards 6G networks highlights the critical need for advanced resource management techniques that meet stringent requirements for energy efficiency, ultra-low latency, and high reliability. Various researchers proposed methods and frameworks targeting optimized resource allocation, network slicing, and energy efficiency. In the context of 5G edge networks, Bera and Mehta [32] addressed the challenges of managing network slices in 5G edge networks. The authors proposed a heuristic approach, RESET, to optimize the allocation of resources, including bandwidth, computing, and storage, while maximizing operator rewards and minimizing the costly redistribution of active slices. Huang *et al.* [81] introduced a novel 4D conflict graph model and utilized a Hasse traversal algorithm to achieve optimized resource allocation within network slicing environments. Trivisonno *et al.* [26] explored network resource management and QoS provisioning in SDN-enabled 5G systems. In another work, Lyu *et al.* [82] proposed a 5G-enabled framework for energy-efficient transmission and state estimation in IIoT systems. The framework used a hierarchical approach that integrates adaptive resource allocation and state estimation strategies to enhance transmission reliability and accuracy under constrained energy and communication resources. Liyanage *et al.* [83] explored the integration of MEC in 5G networks to enhance IoT applications. The study addressed the challenges and future directions for realizing a cohesive MEC-IoT ecosystem in 5G infrastructures.

Beyond 5G networks, Cao *et al.* [39] formulated energy-cost models for vehicle-assisted B5G networks to support both NFV and network slicing. The authors aimed to minimize total energy cost while maintaining high slice acceptance by prioritizing active nodes and jointly allocating both wireless and wired resources. In another work, Li *et al.* [84] proposed a framework to efficiently manage resources across multiple scenarios in 6G networks using a Time-Expanded Graph (TEG). Energy efficiency has been a key focus in IIoT. Shukla *et al.* [85] proposed 6G-SDI, an SDN-based green communication framework that predicts uplink collisions and mobile-node locations to enable energy-efficient, low-latency flow-rule placement in large-scale 6G-IoT networks. Manogaran *et al.* [86] presented a deep learning-based concurrent resource allocation method for 6G Network-in-Box (NIB) architecture. The authors leveraged attuned slicing and deep neural networks to optimize resource allocation and improve service response for UAV-assisted communications. In a similar context, Thantharate and Beard [87] introduced ADAPTIVE6G, a novel adaptive learning framework for resource optimization in network slice architectures for beyond 5G and 6G systems by leveraging transfer learning. Further building upon intelligent resource allocation, Mei *et* *al.* [88] presented a hierarchical deep reinforcement learning (DRL) approach designed for intelligent Radio Access Network (RAN) slicing. Their approach utilized a twin-timescale Markov Decision Process, significantly enhancing spectrum efficiency in 6G network scenarios. Addressing the critical issue of energy efficiency, Baptiste *et al.* [89] evaluated the energy consumption associated with 5G network softwarization using an experimental OpenAirInterface (OAI)-based platform. They highlighted significant energy consumption concerns and underscored the urgent need for optimized solutions. Complementing these approaches, Sasan *et al.* [40] proposed a comprehensive joint optimization framework focusing on network slicing, routing, and in-network computing in 6G networks. Their approach aimed to maximize user acceptance while minimizing energy consumption, emphasizing sustainable resource utilization. Sami *et al.* [90] introduced IScaler, a deep reinforcement learning-based solution for intelligent and proactive resource scaling and service placement in Mobile Edge Computing (MEC) environments. In another work, Jain *et al.* [91] presented a metaheuristic approach with a blockchain-based resource allocation technique (MWBA-RAT) for cybertwin-driven 6G on the IoE environment, addressing the challenges posed by increasing mobile Internet traffic and service demands. The proposed technique utilizes a quasi-oppositional search and rescue optimization (QO-SRO) algorithm for enhancing the network's ability to effectively monitor, manage, and share limited spectrum resources.

2.6 Conclusion

In this Chapter, we reviewed the state of the art in resource allocation and orchestration across IoT/IoE ecosystems, SDN-enabled networks, multi-tier edge computing, and 5G/6G infrastructures. We encompassed a wide range of architectures and paradigms—from edge and fog computing frameworks to SDN-enabled orchestration and delay-sensitive IoT environments—highlighting their individual contributions toward improving scalability, latency. Initially, we outlined the distinctions between IoT and IoE, highlighting their enabling technologies and operational requirements. Subsequently, we explored diverse bandwidth allocation strategies, QoS-aware scheduling, and control frameworks adopted in SDN-enabled IoT infrastructures — emphasizing the growing role of deep reinforcement learning, edge intelligence, and network slicing in ensuring dynamic adaptability.

Despite significant progress, challenges persist in achieving scalable, delay-aware, and energy-efficient orchestration across heterogeneous and distributed environments. In this regard, the present work contributes a unified and self-adaptive orchestration framework that jointly manages communication, computation, and storage resources in real time. The proposed approaches leverage SDN-enabled control and edge intelligence to enhance dynamic bandwidth allocation, flow management, and overall network efficiency.

Chapter 3

T-RESIN: Throughput-Aware Resource Orchestration Scheme

In this chapter, we address the problem of resource allocation in IoE-enabled software-defined edge networks. In the existing literature, the researchers considered optimizing the performance of the SDN platform using a single-tier architecture, where the IoT devices are in the same tier. However, with the advent of edge computing, we can explore the two-tier architecture of edge networks — access and edge tiers — in the presence of SDN. Hence, we propose a throughput-aware dynamic resource orchestration scheme, named T-RESIN, for software-defined edge networks. The scheme aims to efficiently distribute heterogeneous IoT/IoE flows across SDN switches and allocate processes to edge nodes under bandwidth, computation, and memory constraints. T-RESIN leverages an evolutionary game-theoretic framework for dynamic resource orchestration at access and edge tiers.

This chapter is organized as follows. Section 3.1 describes the SDEN system model, including entities, notation, and resource constraints at access and edge tiers. Section 3.2 presents the proposed T-RESIN scheme using an evolutionary game-theoretic formulation [92]. Section 3.3 details the experimental setup, benchmark schemes, performance metrics, and results discussion. Finally, Section 3.4 concludes this chapter.

3.1 System Model

We consider an SDEN architecture with a single controller, multiple SDN switches, and multiple IoE devices computing edge nodes, as shown in Figure 3.1. The IoT devices generates the data from the IoE process and applications that are to be processed by the edge nodes[1]. However, the devices are connected through the SDN switches at the access tier. $\mathcal{N}$ and $\mathcal{S}$ represent the sets of IoE devices and SDN switches, respectively. Bandwidth associated with each SDN switch $s \in \mathcal{S}$ is represented as B_s. Therefore, the total bandwidth $\mathcal{B}$ distributed among all the switches is as follows:

$$\mathcal{B} = \sum_{s \in \mathcal{S}} B_s \tag{3.1}$$

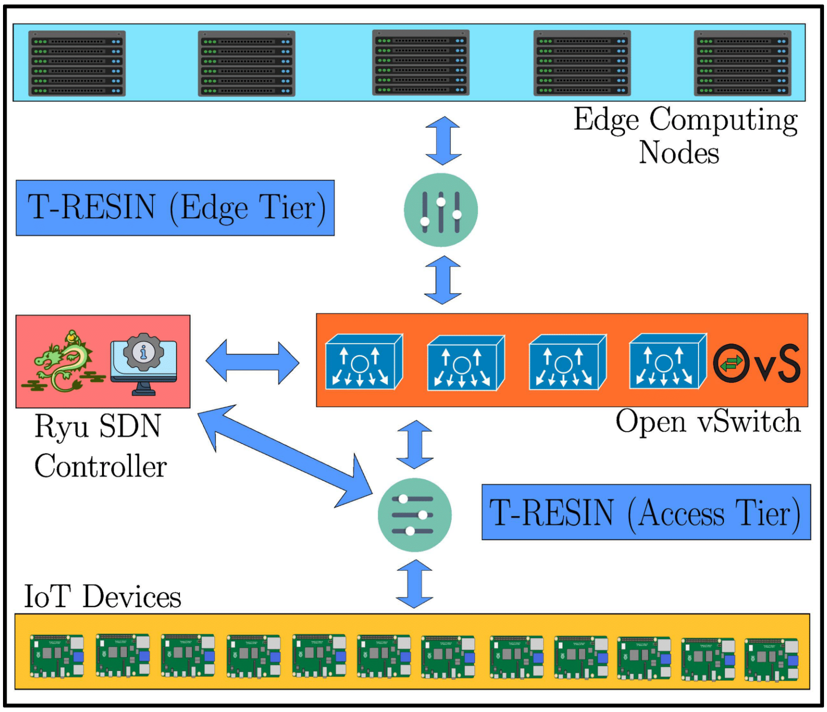


Figure 3.1: Schematic Diagram of Software-Defined Edge Networks

Each IoE device $n \in \mathcal{N}$ generates f_n number of data flows and the data generation rate of each data flow i is denoted as d_i, where $i \in (\mathbb{Z}^+ \cap [0, f_n], \forall n)$. Flows are heterogeneous in terms of bandwidth depending on flow type — scalar or multimedia, and the volume generated by flow. We consider that the SDN switches are capable to forward the generated data

[1]We clarify the use IoT for the devices and IoE for the associated services/processes.

and support the network requirement of the IoE applications. However, the flows might need to distributed among the multiple SDN switches as these switches have physical limitations in terms of flow rule capacity and bandwidth. Accordingly, each switch needs to satisfy the following constraints.

$$V_s = \sum_{n \in \mathcal{N}} \sum_{i \in \mathbb{Z}^+ \cap [0, f_n]} x_{i,n,s} d_i \leq B_s \tag{3.2}$$

$$F_s = \sum_{n \in \mathcal{N}} \sum_{i \in \mathbb{Z}^+ \cap [0, f_n]} x_{i,n,s} \leq R_s^{max} \tag{3.3}$$

where V_s and F_s denote the volume of data and the number of flow-rules to be handled by switch s. We consider that each switch $s \in \mathcal{S}$ is capable of installing maximum R_s^{max} number of flow rules in its TCAM memory. Here, $x_{i,n,s}$ is a binary variable and defines the association among flow i, where $i \in (\mathbb{Z}^+ \cap [0, f_n]$ and switch s and defined as follows:

$$x_{i,n,s} = \begin{cases} 1, & \text{if flow } i \text{ of } n \in \mathcal{N} \text{ is associated switch } s \\ 0, & \text{otherwise} \end{cases} \tag{3.4}$$

In other words, we consider that the set of P_s process is associated with switch $s \in \mathcal{S}$, where each process $q \in P_s$ requires f_q number of flows. Hence, we get:

$$\sum_{q \in P_s} f_q \equiv \sum_{n \in \mathcal{N}} \sum_{i \in \mathbb{Z}^+ \cap [0, f_n]} x_{i,n,s} \tag{3.5}$$

In the edge tier, we consider that there is $\mathcal{E}$ set of edge nodes. Each edge node $e \in \mathcal{E}$ has computational and memory capacities of C_e and M_e, respectively. Hence, while allocating the processes to the edge nodes, we need to satisfy the following constraints.

$$\sum_{s \in \mathcal{S}} \sum_{q \in P_s} \sum_{j \in \mathbb{Z}^+ \cap [0, f_q]} y_{j,q,e} m_q \leq M_e \tag{3.6}$$

$$C_{use}^e = \sum_{s \in \mathcal{S}} \sum_{q \in P_s} \sum_{j \in \mathbb{Z}^+ \cap [0, f_q]} y_{j,q,e} c_p \leq C_e \tag{3.7}$$

where m_q and c_q represent the required memory and CPU resources for each process $q \in P_s$, respectively. We denote the memory and computational capacities of edge node $e \in \mathcal{E}$ using M_e and C_e, respectively. Here, $y_{j,q,e}$ is a binary variable and is evaluated as follows:

$$y_{j,q,e} = \begin{cases} 1, & \text{if process } q \in P_s \text{ of switch } s \in \mathcal{S} \text{ is allocated to edge node } e \\ 0, & \text{otherwise} \end{cases} \tag{3.8}$$

We consider that the overall delay for each process q is denoted as λ_q and evaluated as follows:

$$\lambda_q = \lambda_{q,s} + \lambda_{q,e} \tag{3.9}$$

where $\lambda_{q,s}$ and $\lambda_{q,e}$ represent the incurred delay at the access and edge tiers, respectively. We consider that each IoE process is to be served within a threshold delay λ_q^{th}. Hence, we need to satisfy the following constraint:

$$\lambda_q \leq \lambda_q^{th} \tag{3.10}$$

We also consider that the edge devices have limited energy, i.e., maximum energy E_e^{max} for each edge device e. Considering that process q requires E_q amount of energy, the edge node e needs to ensure that it satisfies the following constraint for serving process q.

$$\sum_{s \in \mathcal{S}} \sum_{q' \in P_s \cup \{q\}} \sum_{j \in \mathbb{Z}^+ \cap [0, f_{q'}]} y_{j,q',e} E_{q'} \leq E_e^{max} \tag{3.11}$$

Therefore, considering the incoming process q is allocated to edge node e, the remaining amount of energy, E_e^{res}, is represented as follows:

$$E_e^{res} = E_e^{max} - \sum_{s \in \mathcal{S}} \sum_{q' \in P_s \cup \{q\}} \sum_{j \in \mathbb{Z}^+ \cap [0, f_{q'}]} y_{j,q',e} E_{q'} \tag{3.12}$$

3.2 T-RESIN: The Proposed Resource Orchestration Scheme

For modeling the interactions to ensure dynamic resource allocation in SDEN, i.e., a three-layer architecture, we use the evolutionary game-theoretic approach [92]. In the subsequent sections, we discuss the use of the evolutionary game-theoretic approach for the proposed scheme, T-RESIN.

3.2.1 Justification for Using Evolutionary Game

For optimal resource allocation of IoE processes in SDEN, we rely on Equations (3.2), (3.4), (3.5) and (3.6). These equations are functions of binary variables $x_{i,n,s}$ and $y_{j,q,e}$ defined in Equations (3.3) and (3.7), respectively. Hence, we argue that the problem mentioned above is a *multiple binary integer programming problem* [93] that can be mapped to *0-1 knapsack problem* [93, 94]. It is an NP-complete problem. Hence, to get a sub-optimal resource allocation in polynomial time, we use an evolutionary game-theoretic approach.

3.2.2 Game Formulation

We design the proposed scheme, T-RESIN, as a two-stage game — access and edge tiers. In the access tier, the IoT devices or the IoE processes act as the players and choose the set of forwarding SDN switches, i.e., strategies to forwarding generated data from the processes at the user end to the computing edge nodes, with the help of the SDN controller. The generated data by the IoE processes is considered as the population share at access tier in T-RESIN. Hence, the population share $y_s(\omega)$ of each switch $s \in \mathcal{S}$, where ω is the evolutionary iteration, is defined as follows:

$$y_s(\omega) = \frac{V_s(\omega)}{\sum\limits_{s \in \mathcal{S}} V_s(\omega)} \tag{3.13}$$

On the other hand, the population share $x_e(\phi)$ of each edge node $e \in \mathcal{E}$ is defined as follows:

$$x_e(\phi) = \frac{C^e_{use}(\phi)}{\sum\limits_{e \in \mathcal{E}} C^e_{use}} \tag{3.14}$$

where ϕ is the evolutionary iteration for edge nodes. Here, $x_e(\phi)$ signifies the computation power contribution of edge node e. C^e_{use} represents the used computation power of edge node e. We identify the selection of edge nodes for processing the data generated by the IoE devices and forwarded through the SDN switches. Based on the population shares, we define the utility functions of each SDN switch and edge node as given in the subsequent sections.

Utility Function of Each SDN Switch: Utility function $U_s(\omega)$ signifies the fitness function for switch s for evolutionary iteration ω. We consider that $U_s(\omega)$ varies linearly with the population share of switch s. Similarly, with the increase in the processes P_s associated with the switch s, the utility value increases with the increase in the number of flow rules installed in the switch. We define utility function $U_s(\omega)$ for each switch as follows:

$$U_s(\omega) = \frac{y_s(\omega) F_s(\omega)}{R^{max}_s} \tag{3.15}$$

Therefore, the average payoff $\overline{U}(\omega)$ of the SDN switches is evaluated as follows:

$$\overline{U}(\omega) = \sum_{s \in \mathcal{S}} y_s(\omega) U_s(\omega) \tag{3.16}$$

Utility Function of Each Edge Node: Utility function $W_e(\phi)$ signifies the fitness function for edge node e. We consider that $W_e(\phi)$ varies linearly with the population share and total energy consumed of edge node e. Additoinally, we consider that $W_e(\phi)$ decreases with the increase of residual energy E^e_{res} of each edge node $e \in \mathcal{E}$. The utility function $W_e(\phi)$ for each edge node is designed as follows:

$$W_e(\phi) = x_e(\phi) \left(1 - \frac{E^e_{res}(\phi)}{E^e}\right) \tag{3.17}$$

We define the average payoff $\overline{W}(\phi)$ of the edge nodes $\mathcal{E}$ as follows.

$$\overline{W}(\phi) = \sum_{e \in \mathcal{E}} x_e(\phi) W_e(\phi) \tag{3.18}$$

Replicator Dynamics: As evolutionary games are dynamic and have two mechanisms that are *mutation* and *selection*. The mutation mechanism modifies the characteristics of the population whenever new players come into the population and choose different strategies. The selection mechanism determines the strategy with high fitness and promotes that strategy in the population. In an evolutionary game, players replicate themselves by changing the strategies, called as *replicator dynamics*. Based on the significance of replicator dynamics in T-RESIN, we define replicator dynamics for SDN switches $\dot{y_s}(\omega)$ and edge nodes $\dot{x_e}(\phi)$ as follows:

$$\dot{y_s}(\omega) = \alpha y_s(\omega) \left(U_s(\omega) - \overline{U}(\omega) \right) \tag{3.19}$$

$$\dot{x_e}(\phi) = \beta x_e(\phi) \left(W_e(\phi) - \overline{W}(\phi) \right) \tag{3.20}$$

where $\{\alpha, \beta\} > 0$ and act as the evolutionary control factors.

Evolutionary Equilibrium: Through evolution, the population adapts higher utility value strategies, eventually leading to an evolutionary stable strategy or evolutionary equilibrium. The fractions of the population choosing different strategies cease to change at evolutionary equilibrium [92]. In T-RESIN, evolutionary equilibrium is determined at $\dot{y_s}(\omega)$ for SDN switches and $\dot{x_e}(\phi)$ for edge nodes.

3.2.3 Theoretical analysis

In T-RESIN, the SDN switches and the edge nodes execute Algorithms 3.1 and 3.2, respectively, to achieve the evolutionary equilibrium. We analyze the existence of evolutionary equilibrium in T-RESIN in this section.

Algorithm 3.1 T-RESIN Algorithm for SDN Switches

INPUTS:
$\mathcal{N}, \mathcal{S}, f_n, F_s, d_i, R_s^{\max}, \alpha$
OUTPUT:
y^*, U^*
PROCEDURE:
$\omega \leftarrow 0$
Randomly assign each flow $0 \le i \le f_n,\ \forall n$, to switch $s \in \mathcal{S}$
do
    $\omega \leftarrow \omega + 1$
    for Each $s \in \mathcal{S}$ **do**
        Calculate $V_s(\omega)$ using Equation (3.2)
        Calculate $F_s(\omega)$ using Equation (3.3)
        Calculate population share $y_s(\omega)$ using Equation (3.13)
        Calculate utility value $U_s(\omega)$ using Equation (3.15)
    end for
    Calculate average utility value $\overline{U}(\omega)$ using Equation (3.16)
    for Each $s \in \mathcal{S}$ **do**
        Calculate replicator dynamic $\dot{y}_s(\omega)$ using Equation (3.19)
        $y_s(\omega+1) \leftarrow y_s(\omega) + \dot{y}_s(\omega)$
    end for
while $(\dot{y}_s(\omega) \not\approx 0)$
$y_s^* \leftarrow y_s(\omega)$
Calculate utility value U_s^* using Equation (3.15) at evolutionary iteration ω
return y^*, U^*

Evolutionary equilibrium for SDN switches: At evolutionary equilibrium, we get.

$$\dot{y}_s(\omega) = \alpha y_s(\omega)\left(U_s(\omega) - \overline{U}(\omega)\right) = 0 \tag{3.21}$$

Considering that the population share of each switch $s \in \mathcal{S}$ is $y_s(\omega) \ge 0$ and evolutionary control factor α is a positive constant i.e. $\alpha > 0$. we get:

$$U_s(\omega) - \overline{U}(\omega) = 0 \tag{3.22}$$

By solving Equation (3.21), we get:

$$(y_s^*)^2 - y_s^* + \frac{\sum\limits_{s' \in \mathcal{S}/\{s\}} (y_{s'}^*)^2 \left(\frac{F_{s'}}{R_{s'}^{max}}\right)}{\frac{F_s}{R_s^{max}}} = 0 \tag{3.23}$$

Algorithm 3.2 T-RESIN Algorithm for Edge Nodes

INPUTS:
$\mathcal{S}, \mathcal{E}, C^e_{use}, E^e, P_e, W, \beta$
OUTPUT:
x^*, W^*
PROCEDURE:
$\phi \leftarrow 0$
Randomly map processes P_s, $\forall s$, to edge node $e \in \mathcal{E}$ for computational resources
do
    $\phi \leftarrow \phi + 1$
    for Each $e \in \mathcal{E}$ **do**
        Calculate $C^e_{use}(\phi)$ using Equation (3.7)
        Calculate $E^e_{res}(\phi)$ using Equation (3.12)
        Calculate population share $x_e(\phi)$ using Equation (3.14)
        Calculate utility value $W_e(\phi)$ using Equation (3.17)
    end for
    Calculate average utility value $\overline{W}(\phi)$ using Equation (3.18)
    for Each $e \in \mathcal{E}$ **do**
        Calculate replicator dynamic $\dot{x}_e(\phi)$ using Equation (3.20)
        $x_e(\phi + 1) \leftarrow x_e(\phi) + \dot{x}_e(\phi)$
    end for
while $(\dot{x}_e(\phi) \not\approx 0)$
$x^*_e \leftarrow x_e(\phi)$
Calculate utility value W^*_e using Equation (3.17) at evolutionary iteration ϕ
return x^*, W^*

At evolutionary equilibrium, we yield an optimal population share y^*_s for each switch $s \in \mathcal{S}$ as follows:

$$y^*_s = \frac{1 \pm \sqrt{1 - 4\psi}}{2} \tag{3.24}$$

where $\psi = \left[\frac{\sum_{s' \in \mathcal{S}/\{s\}} \left(y^*_{s'}\right)^2 \left(\frac{F_{s'}}{R^{max}_{s'}}\right)}{\frac{F_s}{R^{max}_s}} \right]$.

Evolutionary equilibrium for edge nodes: At evolutionary equilibrium, the change in the population share of the edge nodes reaches zero. Hence, we get.

$$\dot{x}_e(\phi) = \beta x_e(\phi) \left(W_e(\phi) - \overline{W}(\phi)\right) = 0 \tag{3.25}$$

Considering that $x_e > 0, \forall e$, and $\beta > 0$, we get.

$$W_e(\phi) - \overline{W}(\phi) = 0 \tag{3.26}$$

At evolutionary equilibrium, we get that the optimal population share x_e^* for each edge node e is as follows:

$$(x_e^*)^2 - x_e^* + \frac{\sum\limits_{e' \in \mathcal{E}/\{e\}} (x_{e'}^*)^2}{\left(1 - \frac{E_{res}^{e'}}{E^{e'}}\right)} \left(1 - \frac{E_{res}^e}{E^e}\right) = 0 \tag{3.27}$$

At evolutionary equilibrium, we yield optimal population share x_e^* for each edge node $e \in \mathcal{E}$ as follows:

$$x_e^* = \frac{1 \pm \sqrt{1 - 4\kappa}}{2} \tag{3.28}$$

where $\kappa = \left[\frac{\sum\limits_{e' \in \mathcal{E}/\{e\}} (x_{e'}^*)^2 \left(1 - \frac{E_{res}^{e'}}{E^{e'}}\right)}{\left(1 - \frac{E_{res}^e}{E^e}\right)} \right]$

3.3 Performance Analysis

We emulated the proposed scheme, T-RESIN, using the Mininet network emulator [2]. The performance of T-RESIN is evaluated while comparing with the existing competing schemes. The detailed experimental setup with simulation parameters and the yield results are discussed in the subsequent sections.

3.3.1 Experimental Setup

To evaluate the performance of T-RESIN, we emulated a SDN-enabled edge platform in Mininet. We consider that SDEN is equipped with a Ryu SDN controller[3] and Open vSwitch (SDN switches)[4]. The detailed experimental setup is mentioned in Table 3.1.

[2] https://mininet.org/
[3] https://ryu-sdn.org/
[4] https://www.openvswitch.org/

We consider that there is a single Ryu SDN controller for SDEN Mininet topology in T-RESIN in the presence of multiple SDN switches and edge nodes. The detailed simulation parameters are shown in Table 3.2.

Table 3.1: Experimental Setup

Hardware	Intel® Core™ i7-9700 CPU @3.00GHz × 8
Operating System	Ubuntu 20.04.6 LTS
Network Emulator	Mininet (Version 2.31b1)
SDN Controller	Ryu Controller (Version ryu 4.34)
SDN Switch	Open vSwitch (Version ovs-vsctl 2.13.8)
Programming Language	Python3 (Version 3.8.10)
Benchmarks	Random Flow, FlowMan

Table 3.2: Simulation Parameters

Parameter	**Value**
Number of Open vSwitches in Mininet topology	2, 5, 10
Number of Edge Nodes in Mininet topology	10, 20, 30
Number of IoE Devices in Mininet topology	50, 100, 200
Maximum Energy of each Edge node	20 Joule [95]
Energy Consumption at Transmitter side (Tx)	50 nJ/bit [96]
Evolutionary Control Factor	$\alpha = 0.01, \beta = 0.1$

3.3.2 Benchmarks

We compare the performance of T-RESIN with two schemes – RandomFlow and FlowMan [63]. In RandomFlow, we consider that the resource allocation for the switches and the edge nodes is random. To ensure unbiased result, we took 50 runs of the emulated platform for each topology and evaluated the 95% confidence interval result. On the other hand, in FlowMan, Mondal and Misra considered the presence of heterogeneous flows and flow-rules are placed to ensure high throughput. We argue that these schemes consider resource allocation in SDN-enabled platforms. However, the optimal edge resource allocation in provisioning IoE-enabled services are not considered in the existing literature.

3.3.3 Performance Metrics

To evaluate the performance of the proposed scheme, T-RESIN, we considered the following performance metrics.

- *Network Throughput:* It signifies the amount of data delivered and processed by the SDEN. We aim to achieve high network throughput.
- *Flow Count at Switch:* It is calculated as the total number of flow rules associated with TCAM memory for each switch.
- *Switch Delay:* It is calculated as the latency incurred by the flows at the SDN switches in Mininet.
- *Average Energy Consumption at Edge Node:* It signifies the amount of energy consumed to provision the IoE-enabled services/processes.
- *Average Computation Overhead at Edge Node:* We aim to achieve moderate utilization of the edge nodes and ensure low failure probability while provisioning the IoE-based services.
- *Computation Delay at Edge Node:* It is calculated as the processing delay the edge devices in provisioning the IoE-enabled services.

3.3.4 Results and Discussions

From Figures 3.2, we observe that the average throughput per IoE devices increases by 27.98-31.84% using T-RESIN than using other schemes — RandomFlow and FlowMan. This is due to the fact that T-RESIN ensures that the flows are optimally distributed among the available SDN switches. Additionally, in FlowMan, the flow association is decided based on the one hop networks and in RandomFlow, the flows are allocated randomly. Moreover, we observe that with the increase in the number of SDN switches the network throughput outperforms the other existing competing schemes while ensuring a high throughput. Moreover, using T-RESIN, average flow count per Open vSwitch increases by 10.73-13.03% than using other competing schemes — RandomFlow and FlowMan, as observed in Figure 3.3.

This eventually helps in distributing the network load optimally at the SDEN and ensures in achieving high throughput. However, in Figure 3.4, we observe that delay in the provisioned IoE-enabled services are mostly random using T-RESIN, as we did not consider delay parameters while modeling the game theoretic model. Though we argue that T-RESIN ensures the delay threshold values while provisioning services in IoE-enabled SDEN.

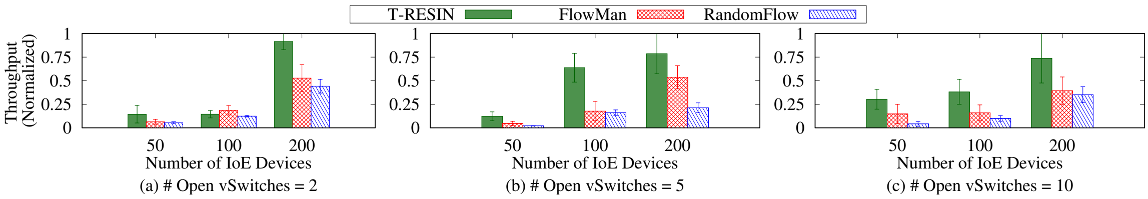


Figure 3.2: Network Throughput

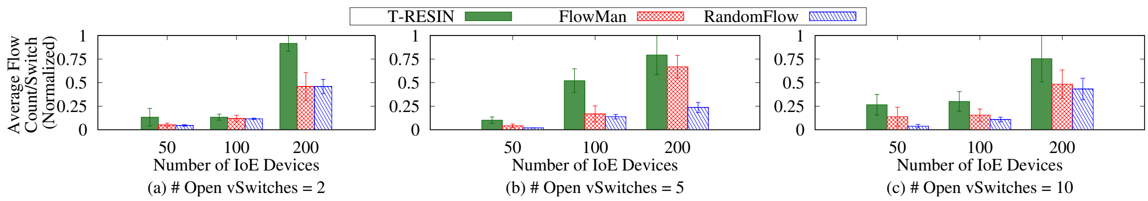


Figure 3.3: Flow Count per Open vSwitch

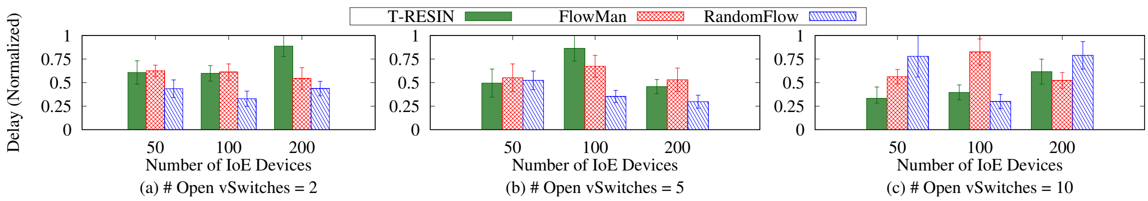


Figure 3.4: Switch Delay at Access Tier

On the other hand, Figure 3.5 depicts that using T-RESIN, energy consumption at edge tier decreases by 4.59-14.57% using other scheme — RandomFlow. This is due to the fact that the requested processes/applications are allocated optimally among the available edge nodes. Hence, we also observe that computation overhead of the edge nodes reduces by 9.5-22.29% using T-RESIN than using RandomFlow, as shown in Figure 3.6. However, we observe from Figure 3.7 that the delay incurred at the edge nodes varies randomly using T-RESIN, as we did not consider the delay parameters while designing the mathematical model of T-RESIN, as mentioned earlier.

Hence, we argue that T-RESIN ensures a high throughput while reducing the energy

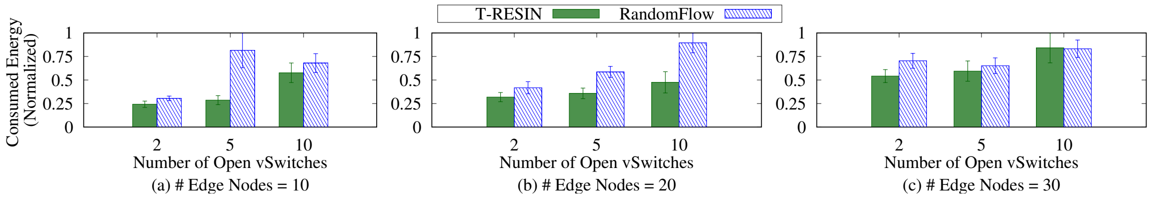


Figure 3.5: Energy Consumption of the Edge Nodes

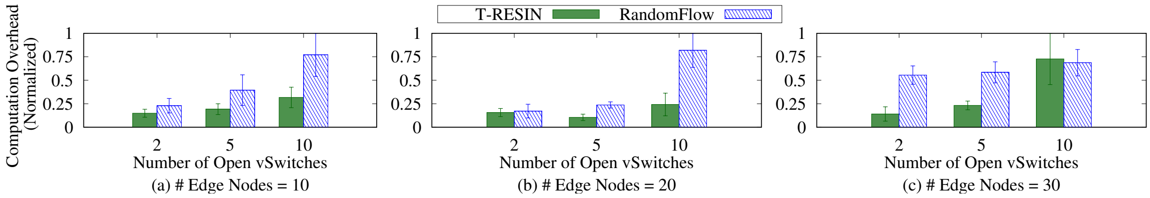


Figure 3.6: Computation Overhead of the Edge Nodes

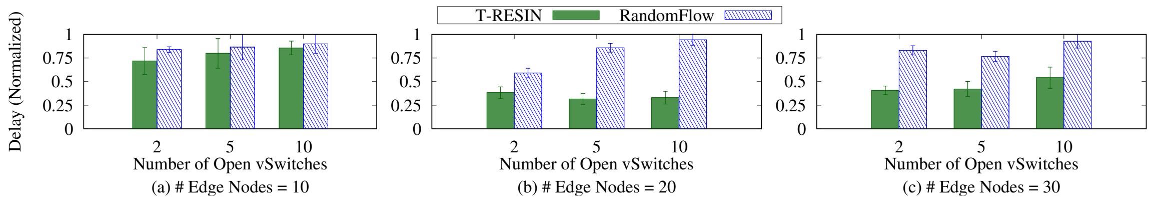


Figure 3.7: Computation Delay at the Edge Nodes

consumption and computation overhead than using the competing existing schemes. We plan to extend this work and optimize the delay performance for IoE-enabled SDEN.

3.4 Conclusion

In this chapter, we studied the problem of optimal resource allocation in an IoE-enabled SDEN. We designed a multi-tier dynamic resource allocation scheme, named T-RESIN, using evolutionary game theory. In the access tier, the volume of data generated by the IoE devices defined the population. Population share of each SDN switch is evaluated as the volume of data associated with the corresponding switch. On the other tier, the population share of each edge node is evaluated as amount of data processed by it. While following the replicator dynamics principle, we theoretically analyze the existence of evolutionary equilibrium in T-RESIN. We also evaluated the performance of T-RESIN while emulating on Ryu controller-based Mininet platform and observed that T-RESIN outperforms the competing existing schemes in terms of achieved network throughput.

Chapter 4

D-RESIN: Delay-Aware Resource Orchestration Scheme

In this chapter, we introduce Delay-Aware Dynamic Resource Orchestration for IoT-Enabled SDEN, named D-RESIN, a novel framework to orchestrate resources in SDENs, focusing on minimizing computation delays. As shown in Figure 4.1, functions FlowStats() and PortStats() are used for the statistics collection of data at the Access tier and Edge tier, respectively. Similarly, MeterStats() is used for bandwidth establishment between IoT devices to Open vSwitches and Open vSwitches to Edge nodes. D-RESIN addresses the latency issues inherent in IoT deployments and significantly enhances the scalability and efficiency of edge computing environments.

This chapter is organized as follows. Section 4.1 introduces the system model for IoT-enabled SDENs, highlighting delay modeling and resource constraints at access and edge tiers. Section 4.2 presents the proposed D-RESIN framework, covering the game-theoretic formulation and the theoretical analysis of proposed algorithms complexity analysis. Section 4.3 details the performance analysis of D-RESIN scheme using Mininet network emulator with Ryu SDN controller and Open vSwitches. Finally, Section 4.4 concludes the chapter with the findings and remarks.

4.1 System Model

We consider an IoT-enabled SD-edge network infrastructure. It has one SDN controller in the control plane and multiple switches in the data plane. The computation of data generated from IoT devices/things occurs at edge nodes. Data packets/flows from IoT devices to edge nodes go through switches via access points. We consider $\mathcal{N}$, $\mathcal{S}$, and $\mathcal{E}$ as a set of IoT devices, SDN switches, and edge nodes, respectively. Bandwidth associated with each switch $s \in \mathcal{S}$ is represented as B_s such that total bandwidth $\mathcal{B}$ distributed among the switches is as follows:

$$\mathcal{B} = \sum_{s \in \mathcal{S}} B_s \tag{4.1}$$

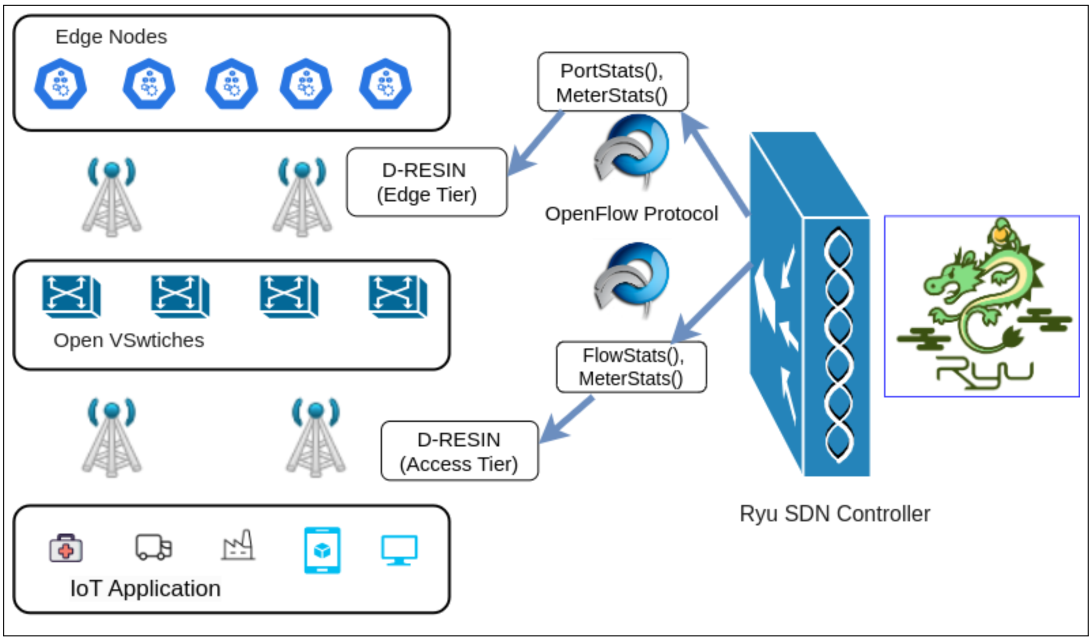


Figure 4.1: Schematic Architecture of IoT-Enable SD-Edge Networks

Each IoT device $n \in \mathcal{N}$ generates F_n set of flows. Each flow $f_i^n \in F_n$ starts and ends its transmission from the IoT device $n \in \mathcal{N}$ at $t_s(f_i^n)$ and $t_e(f_i^n)$, respectively, where $i \in (\mathbb{Z}^+ \cap [0, F_n])$ and $t_e(f_i^n) > t_s(f_i^n)$. Flows are dynamic and can be processed at any SDN switch. R_s^{max} denotes the maximum number of flow rules that switch $s \in \mathcal{S}$ can install in its ternary content addressable memory (TCAM). F_s represents the number of flow rules associated with each switch $s \in \mathcal{S}$ that satisfies the constraint $F_s \leq R_s^{max}$.

The processing delay[1] of any flow depends on queue discipline and volume of the data associated with a particular flow. For queue discipline, we consider first-come-first-serve, where flows from the queue are selected for servicing based on their arrival. The processing rate μ_s represents how much volume of data can be processed per unit of time at switch s, and it is constant and does not depend on network parameters. The volume of data associated with each SDN switch $s \in \mathcal{S}$ is represented in Equation (4.2). Definition 1 describes the switch processing delay for active flow rules F_s, which helps in determining the average processing delay of associated flows represented by Equation (4.4).

$$V(s) = \sum_{f \in \bigcup_n F_n} a_{f,n,s} V_f \leq B_s \tag{4.2}$$

where $a_{f,n,s}$ is defined as follows:

$$a_{f,n,s} = \begin{cases} 1, & \text{if flow } f \text{ of } n \in \mathcal{N} \text{ associated to a switch } s \\ 0, & \text{otherwise} \end{cases} \tag{4.3}$$

Definition 1. *We define $stat.duration\_sec$ parameter of AggregateStats() function of the Open vSwitches $s \in \mathcal{S}$ to capture the overall processing delay for F_s set of flow rules that stay active at any time instant t in its TCAM memory.*

$$Pd_s^{avg}(f,n,s) = \frac{V(s)}{\mu_s \sum_{f \in \bigcup_n F_n} a_{f,n,s}} \tag{4.4}$$

We consider that flow $f_j^s \in F_s$ is transmitted from the associated SDN switch towards the edge node for resource allocation when it completes its processing at the switch $s \in \mathcal{S}$ where $j \in (\mathbb{Z}^+ \cap [0, F_s])$. Each edge node $e \in \mathcal{E}$ has resource constraints in the form of computation C_e and memory M_e for processing the flows received from IoT devices. The volume of data and computational power used by an edge node $e \in \mathcal{E}$ for processing the

[1] We have not considered channel delay (transmission and propagation delay).

flows are represented in Equations (4.5) and (4.6).

$$V(e) = \sum_{f \in \bigcup_s F_s} b_{f,s,e} V_f \leq M_e \tag{4.5}$$

$$C_{use}(e) = \sum_{f \in \bigcup_s F_s} b_{f,s,e} C_f \leq C_e \tag{4.6}$$

where $b_{f,s,e}$ is defined as follows:

$$b_{f,s,e} = \begin{cases} 1, & \text{if flow } f \text{ of } s \in \mathcal{S} \text{ associated to edge node } e \\ 0, & \text{otherwise} \end{cases} \tag{4.7}$$

Definition 2. *We define $stat.duration\_sec$ parameter of PortStats() function to capture the overall processing delay for F_e set of flow rules that are associated with the edge nodes $e \in \mathcal{E}$ for resource allocation (memory and computation).*

The average processing delay of an edge node $e \in \mathcal{E}$ is presented in Equation (4.8) where μ_e is the processing rate for every edge node.

$$Pd_e^{avg}(f, s, e) = \frac{V(e)}{\mu_e C_{use}(e) \sum_{f \in \bigcup_s F_s} b_{f,s,e}} \tag{4.8}$$

Total energy consumption E_{use}^e for computing and storing any flow associated with an edge node e is represented in Equation (4.9). At a particular time instant t, the residual energy E_{res}^e of edge node e is written as follows:

$$E_{use}^e = \sum_{f \in \bigcup_s F_s} b_{f,s,e} E(f, s, e) \leq E^e \tag{4.9}$$

$$E_{res}^e = E^e - \sum_{f \in \bigcup_s F_s} b_{f,s,e} E(f, s, e) \tag{4.10}$$

where E^e denotes the total energy of each node e. Each edge node is constrained by its residual energy for the association of incoming flows.

4.2 D-RESIN: The Proposed Delay-Aware Resource Orchestration Scheme

We use the *evolutionary game-theoretic approach* [92] to model the interaction among IoT devices and SDN switches, and SDN switches to edge nodes. D-RESIN continuously monitors network conditions and adjusts data flow between IoT devices, Open vSwitches, and Edge nodes based on real-time demands. The framework distributes workloads to minimize congestion and ensure faster processing at access and edge tiers.

4.2.1 Justification for Using Evolutionary Game

We observe that it requires a binary variable $a_{f,n,s}$ in Equation (4.4) for minimizing the overall processing delay between IoT devices and SDN switches. Similarly, for SDN switches to the edge node, it requires a binary variable $b_{f,s,e}$ in Equations (4.8), (4.9), and (4.10) for minimizing the delay. So the problem mentioned above is a multiple *binary integer programming problem* [93]. It can be mapped to the variant of the *0-1 knapsack problem* [93, 94], a well-known NP-complete problem. Traditional brute-force or conventional optimization approaches are impractical for large-scale IoT deployments due to their exponential complexity. These methods can not efficiently handle the real-time resource allocation challenges posed by fluctuating IoT device demands and network traffic. To address this issue, we use *evolutionary game theoretic approach* to provide a computationally feasible solution with polynomial-time complexity while enabling the system to adapt dynamically according to network conditions. The *evolutionary game* becomes a method for exploring and optimizing the delay of the *0-1 knapsack problem*, which in turn represents the optimized solution for the original binary integer programming problem.

4.2.2 Game Formulation

We aim to minimize the processing delay $Pd_s^{avg}(f,n,s)$ of any flow from IoT devices to switches and the processing delay $Pd_e^{avg}(f,s,e)$ of any flow from SDN switches to edge nodes, which is represented in Equations (4.4) and (4.8), respectively. We consider each IoT end-device $n \in \mathcal{N}$ to act as a player and select the SDN switch to forward its data flows from the IoT end user to the Edge node with the help of the SDN controller. The population share $x_s(\cdot)$ of each switch $s \in \mathcal{S}$ is represented in Equation (4.11), which is a function of average processing delay and flows associated with the switch $s \in \mathcal{S}$. Similarly, the population share $y_e(\cdot)$ of edge nodes $e \in \mathcal{E}$ is represented in Equation (4.12).

$$x_s(\cdot) = \frac{\sum\limits_{f \in F_s, \forall n} a_{f,n,s} Pd_s^{avg}(f,n,s)}{\sum\limits_{f \in \bigcup\limits_s F_s, \forall n} a_{f,n,s} Pd_s^{avg}(f,n,s)} \tag{4.11}$$

$$y_e(\cdot) = \frac{\sum\limits_{f \in F_e, \forall s} b_{f,s,e} Pd_e^{avg}(f,s,e)}{\sum\limits_{f \in \bigcup\limits_e F_e, \forall s} b_{f,s,e} Pd_e^{avg}(f,s,e)} \tag{4.12}$$

Utility Function of SDN Switches and Edge Nodes

For delay-sensitive applications, we need to minimize the population shares $x_s(\cdot)$ and $y_e(\cdot)$ of SDN switches and Edge nodes, respectively. Hence, the utility function of SDN switches $U_s(\cdot)$ is considered a negative payoff of SDN switches. This means that switches with lower processing delays and fewer flow rules will have higher utility or payoff values, while switches with higher processing delays and more flow counts will have lower utility values. We define the utility function $U_s(\cdot)$ for each switch and the average payoff $\overline{U}(\cdot)$ of the population for SDN switches as follows:

$$U_s(\cdot) = x_s(\cdot)\left(1 - \frac{F_s}{R_s^{max}}\right) \tag{4.13}$$

$$\overline{U}(\cdot) = \sum_{s \in \mathcal{S}} x_s(\cdot) U_s(\cdot) \tag{4.14}$$

Processing delay and consumed energy of each Edge node define the negative fitness or payoff for the Edge nodes. A lower fitness or payoff value indicates a better outcome for edge nodes. The utility function $W_e(\cdot)$ for each edge node and the average payoff $\overline{W}(\cdot)$ of the population for edge nodes are represented as follows.

$$W_e(\cdot) = y_e(\cdot)\left(1 - \frac{E^e_{use}}{E^e}\right) \tag{4.15}$$

$$\overline{W}(\cdot) = \sum_{e \in \mathcal{E}} y_e(\cdot) W_e(\cdot) \tag{4.16}$$

Hence, the objective functions of each switch $s \in \mathcal{S}$ and edge node $e \in \mathcal{E}$ are as follows:

$$\arg_{x_s} \min U_s(\cdot) \tag{4.17}$$

$$\arg_{y_e} \min W_e(\cdot) \tag{4.18}$$

Replicator Dynamics

Replicator dynamics is used to model the evolution of strategies in a population of Edge nodes and SDN switches to optimize network performance. D-RESIN dynamically adjusts the population shares of SDN switches and edge nodes via replicator dynamics, which balances the load and minimizes processing delays even under high traffic. Replicator dynamics for SDN switches $\dot{x_s}(\cdot)$ and edge nodes $\dot{y_e}(\cdot)$ are defined as follows:

$$\dot{x_s}(\cdot) = \alpha x_s(\cdot)\big(U_s(\cdot) - \overline{U}(\cdot)\big) \tag{4.19}$$

$$\dot{y_e}(\cdot) = \beta y_e(\cdot)\big(W_e(\cdot) - \overline{W}(\cdot)\big) \tag{4.20}$$

where $\alpha > 0$ and $\beta > 0$ are evolutionary control factors.

4.2.3 Existence of Evolutionary Equilibrium for D-RESIN Scheme

Theorem 4.2.1. *For a given set of flow rules F_s and the average processing delay $Pd_s^{avg}(f, n, s)$ of a switch, there exists an evolutionary equilibrium for SDEN network switches at the Access tier.*

Proof. We aim to minimize $Pd_s^{avg}(f, n, s)$, which ensures optimal data traffic distribution among SDN switches based on processing delay. To identify the existence of evolutionary equilibrium, the replicator dynamics equation for switches should be as follows:

$$\dot{x_s}(\cdot) = \alpha x_s(\cdot)\big(U_s(\cdot) - \overline{U}(\cdot)\big) = 0 \tag{4.21}$$

Here, the population share of each switch $s \in \mathcal{S}$ is $x_s(\cdot) \geq 0$, and the evolutionary control factor α is a positive constant, i.e. $\alpha > 0$. Therefore, Equation (4.21) is represented as follows:

$$U_s(\cdot) - \overline{U}(\cdot) = 0 \tag{4.22}$$

By expanding the utility and average utility function of switches in Equation (4.22), we get:

$$(x_s)^2 - x_s + \frac{\sum\limits_{s' \in \mathcal{S}/\{s\}} (x_{s'})^2 \Big(1 - \frac{F_{s'}}{R_{s'}^{max}}\Big)}{\Big(1 - \frac{F_s}{R_s^{max}}\Big)} = 0 \tag{4.23}$$

In order to find the evolutionary equilibrium, we first identify the coefficients of the quadratic Equation (4.23). By solving using a quadratic formula, we yield optimal population share x_s^* for each switch $s \in \mathcal{S}$ as follows:

$$x_s^* = \frac{1 \pm \sqrt{1 - 4\psi}}{2} \tag{4.24}$$

where $\psi = \left[\frac{\sum\limits_{s' \in \mathcal{S}/\{s\}} (x_{s'})^2 \left(1 - \frac{F_{s'}}{R_{s'}^{max}}\right)}{\left(1 - \frac{F_s}{R_s^{max}}\right)} \right]$. □

Theorem 4.2.2. *For a given set of flow rules F_e associated with edge nodes, there exists an evolutionary equilibrium at the edge tier for the average processing delay $Pd_e^{avg}(f, s, e)$.*

Proof. At evolutionary equilibrium, replicator dynamics reach zero at the edge tier. Hence, Equation (4.20) is written as follows:

$$\dot{y_e}(\cdot) = \beta y_e(\cdot)\big(W_e(\cdot) - \overline{W}(\cdot)\big) = 0 \tag{4.25}$$

Similarly, for edge nodes, we consider the population share is $y_e(\cdot) \geq 0$, and the evolutionary control factor is $\beta > 0$. Hence, we get:

$$W_e(\cdot) - \overline{W}(\cdot) = 0 \tag{4.26}$$

By placing the payoff and average utility values of edge nodes in the above Equation (4.26), we get:

$$(y_e)^2 - y_e + \frac{\sum\limits_{e' \in \mathcal{E}/\{e\}} (y_{e'})^2 \left(1 - \frac{E_{use}^{e'}}{E^{e'}}\right)}{\left(1 - \frac{E_{use}^{e}}{E^{e}}\right)} = 0 \tag{4.27}$$

At evolutionary equilibrium, we yield optimal population share y_e^* for each edge node $e \in \mathcal{E}$ as follows:

$$y_e^* = \frac{1 \pm \sqrt{1 - 4\kappa}}{2} \tag{4.28}$$

where $\kappa = \left[\frac{\sum\limits_{e' \in \mathcal{E}/\{e\}} (y_{e'})^2 \left(1 - \frac{E_{use}^{e'}}{E^{e'}}\right)}{\left(1 - \frac{E_{use}^{e}}{E^{e}}\right)} \right]$ □

4.2.4 Proposed Algorithms

For the proposed SDEN network architecture, we designed two D-RESIN algorithms. As shown in the workflow illustrated in Figure 4.2, the decision point (access tier) ensures access and edge tiers executing Algorithms 4.1 and 4.2, respectively, with the help of the Ryu controller. The Knapsack algorithm identifies the association among IoT devices, SDN switches, and Edge computing nodes. To optimize the processing delay, each switch and edge node need to strategize $x_s(\cdot)$ and $y_e(\cdot)$, respectively, to reach evolutionary equilibrium. We set the evolutionary control factor $\alpha = 0.01$ at the access tier and $\beta = 0.1$ at the edge tier. We terminate the execution of both algorithms when replicator dynamics are $\dot{x_s}(\cdot) \not\approx 0$ and $\dot{y_e}(\cdot) \not\approx 0$ ($\leq 10^{-6}$ practically). At evolutionary equilibrium, we evaluate all the SDEN network parameters.

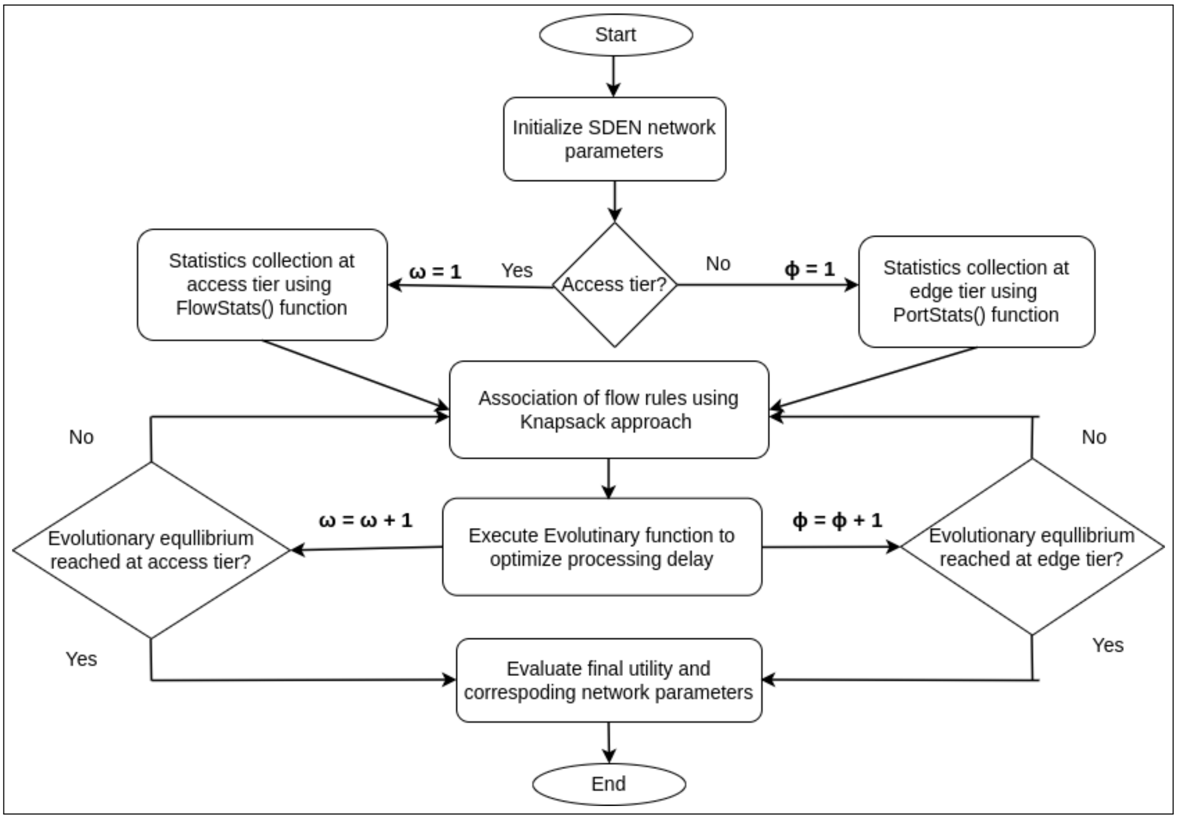


Figure 4.2: Workflow Diagram of D-RESIN

4.2.5 Complexity Analysis

The main part of the Algorithm 4.1 involves a *do-while* loop, from lines 4-16, which continues until an evolutionary stable state ω is reached. Within the outer loop, line 5 takes constant time $O(1)$. For lines 7-9, each line evaluates an equation that takes constant time $O(1)$; thus, the entire block from lines 6-10 takes $O(|\mathcal{S}|)$ time complexity for all switches. Evaluating an average utility for switches in line 11 takes $O(|\mathcal{S}|)$ time complexity. Evaluation and updating of the population share are assumed $O(1)$ per switch; thus, the entire block from lines 12-15 takes $O(|\mathcal{S}|)$ time complexity for all switches. Hence, the overall time complexity of the Algorithm 4.1 is $O(\omega|\mathcal{S}|)$. Similarly, the overall time complexity of the Algorithm 4.2 $O(\phi|\mathcal{E}|)$ for all the edge nodes $\mathcal{E}$, where ϕ is an evolutionary stable state for Algorithm 4.2.

Algorithm 4.1 D-RESIN Algorithm at Access Tier

INPUTS:
1: Network and SDN switch parameters: $\mathcal{N}, \mathcal{S}, F_s, \mu_s, V(s), R_s^{max}, \alpha$
OUTPUT:
2: Optimized population share and utility value vector for SDN switches: x^*, U^*
PROCEDURE:
3: $\omega \leftarrow 0$
4: Distribute flows $0 \leq i \leq f_n,\ \forall n \in \mathcal{N}$ randomly across switches $s \in \mathcal{S}$
5: **do**
6:     $\omega \leftarrow \omega + 1$
7:     **for** each $s \in \mathcal{S}$ **do**
8:         Evaluate $Pd_s^{avg}(f, n, s)$ using Equation (4.4)
9:         Determine population share $x_s(\cdot)$ as per Equation (4.11)
10:         Evaluate utility value $U_s(\cdot)$ according to Equation (4.13)
11:     **end for**
12:     Evaluate average utility $\overline{U}(\cdot)$ using Equation (4.14)
13:     **for** each $s \in \mathcal{S}$ **do**
14:         Evaluate replicator dynamic $\dot{x}_s(\cdot)$ using Equation (4.19)
15:         Update population share for the next iteration as: $x_s(\cdot) \leftarrow x_s(\cdot) + \dot{x}_s(\cdot)$
16:     **end for**
17: **while** $(\dot{x}_s(\cdot) \not\approx 0)$
18: Get evolutionary equilibrium at iteration ω
19: Finalize population share as: $x_s^* \leftarrow x_s(\cdot)$
20: Evaluate final utility value U_s^* using Equation (4.13) at evolutionary iteration ω
21: **return** x^*, U^*

4.3 Performance Analysis

We consider Mininet[2] the most suitable network emulation tool for the proposed SDEN architecture. Mininet provides a realistic virtual environment for custom topology and an SDN controller with OpenFlow compatibility. In the subsequent section, we discuss experimental setup, performance metrics, and results for our proposed D-RESIN algorithms in detail.

[2] https://mininet.org/

Algorithm 4.2 D-RESIN Algorithm at Edge Tier

INPUTS:
Network and Edge Nodes parameters: $\mathcal{N}, \mathcal{S}, \mu_e, C_{use}(e), V(e), E^e, \beta$
OUTPUT:
Optimized population share and utility value vector for Edge nodes: y^*, W^*
PROCEDURE:
$\phi \leftarrow 0$
Distribute flows $0 \leq i \leq f_s$, $\forall s \in \mathcal{S}$ randomly across edge nodes $e \in \mathcal{E}$ for computational resources.
do
    $\phi \leftarrow \phi + 1$
    for each $e \in \mathcal{E}$ **do**
        Evaluate $Pd_e^{avg}(f, s, e)$ using Equation (4.8)
        Evaluate E_{use}^e using Equation (4.9)
        Determine population share $y_e(\cdot)$ using Equation (4.12)
        Evaluate utility value $W_e(\cdot)$ using Equation (4.15)
    end for
    Evaluate average utility value $\overline{W}(\cdot)$ using Equation (4.16)
    for each $e \in \mathcal{E}$ **do**
        Evaluate replicator dynamic $\dot{y}_e(\cdot)$ using Equation (4.20)
        Update population share for the next iteration as follows: $y_e(\cdot) \leftarrow y_e(\cdot) + \dot{y}_e(\cdot)$
    end for
while $(\dot{y}_e(\cdot) \not\approx 0)$
Get evolutionary equilibrium at iteration ϕ
Finalize population share as: $y_e^* \leftarrow y_e(\cdot)$
Evaluate final utility value W_e^* using Equation (4.15) at evolutionary iteration ϕ
return y^*, W^*

4.3.1 Experimental Setup

The proposed SDEN network architecture is experimentally set up according to Table 4.1. Table 4.1 provides detailed information on the hardware, memory type, Mininet emulator, SDN switches, and SDN controller, along with their respective versions. To evaluate the performance of the proposed D-RESIN algorithm, we designed a multi-switch custom-based typology with varying Open vSwitches[3], IoT devices, and Edge nodes as depicted in Table 4.2. The network topology has varying numbers of IoT devices (50, 100, 200), Open vSwitches (2, 5, 10), and edge nodes (10, 20, 30) to analyze performance under different network scales. This Mininet topology is controlled by the single Ryu SDN controller[4]. The

[3]https://www.openvswitch.org/
[4]https://ryu-sdn.org/

Table 4.1: Experimental Setup

Hardware	Intel® Core™ i7-9700 CPU @3.00GHz $\times$ 8
Operating System	Ubuntu 20.04.6 LTS
RAM	24 GB DDR4
Disk Space	1.0 TB
Network Emulator	Mininet (Version 2.31b1)
SDN Controller	Ryu Controller (Version ryu 4.34)
SDN Switch	Open vSwitch (Version ovs-vsctl 2.13.8)
Network Interface Standard	Ethernet
Programming Language	Python3 (Version 3.8.10)
Benchmarks	T-RESIN, FlowMan, RandomFlow

Table 4.2: Simulation Parameters

Parameter	Value
Number of Switches in Mininet Topology	2, 5, 10
Number of Edge Nodes in Mininet Topology	10, 20, 30
Number of IoT Devices in Mininet Topology	50, 100, 200
Initial Energy of Each Edge Node	20 Joule [96]
Network Energy Consumption	50 nJ/bit [96]
Ethernet Frame Size	1518 Byte [97]
Processing Rate (μ_s, μ_e)	0.35 flows per second
Evolutionary Control Factor	$\alpha = 0.01, \beta = 0.1$

detailed emulation parameters are shown in Table 4.2.

In this work, we consider the following parameters for simulation:

1. For the resource orchestration at the edge node, we consider an initial energy of 20 joules [96] for each edge node. The edge node consumes 50 nj/bit [96] for incoming data traffic for computation.

2. The processing rate is considered 0.35 flows per second at both the access and edge tiers.

3. Ethernet is considered for networking standard; hence, we have taken 1518 Bytes for frame size.

4. The evolutionary control factors are set as $\alpha = 0.01$ at the access tier and $\beta = 0.1$ at the edge tier.

4.3.2 Benchmarks

We assess D-RESIN Algorithm 4.1 by comparing it with the existing schemes — T-RESIN, FlowMan, and RandomFlow. Additionally, D-RESIN Algorithm 4.2 is evaluated against T-RESIN and RandomFlow schemes. In RandomFlow, the data traffic is randomized at both access and edge tiers, leading to unpredictable outcomes in network throughput, computational delay, and energy consumption within the SDENs network. Alternatively, FlowMan [63] strategically placed flow rules to establish a trade-off between throughput and delay in two-tier SDN architecture. FlowMan is not specifically tailored for SDEN networks; it does not address resource allocation at the edge tier, which is a critical component for optimizing performance in these environments. On the other hand, In T-RESIN, Agrawal *et al.* [98] optimally allocated resources to achieve high throughput and data flows, contributing to a sustainable SDEN network. However, T-RESIN is unsuitable for meeting the SDEN network's delay-sensitive requirements.

4.3.3 Performance Metrics

- *Average processing delay at switch:* It is calculated as the ratio of the total volume of data and total processing time of all the flows associated with the particular switch. It depends on the flow's processing rate as well.

- *Per-Switch Flow:* Per-Switch Flow is calculated as the number of flow rules associated with TCAM memory for each switch.

- *Per-Switch Throughput:* Per-switch throughput is calculated as the average amount of data processed for each SDN switch individually. It also depends on the flow association of each switch in the SDEN.

- *Average processing delay at edge node:* The average processing delay at an edge refers to the time spent processing the data flows after arrival. This metric is crucial for assessing the performance and efficiency of SDEN.

- *Average energy consumption at edge node:* Each edge node serves the data traffic generated from IoT devices. Average energy consumption is the ratio of total energy consumed in the SDEN network and the number of edge nodes.

4.3.4 Result and Discussion

This section presents the yield results from emulating Algorithm 4.1 at the access tier and Algorithm 4.2 at the edge tier in the proposed SDEN architecture. We evaluated the performance of our scheme under various topological configurations, including different numbers of Open vSwitches (2, 5, 10), IoT devices (50, 100, 200), and Edge Nodes (10, 20, 30), and compared these results to an existing competing scheme. The delay reduction percentages vary across different scenarios based on factors such as network topology, data traffic distribution, and resource allocation strategies. It also highlights the impact of evolutionary game theory-based dynamic resource orchestration in optimizing network performance. For each performance metric, we conducted 20 iterations per topology and calculated the variance of these results at the 95% confidence intervals. The values in the result sections are normalized and scaled to a standard range, likely between 0 and 1, for easier comparison across different scenarios.

From Figure 4.3, we observe that the average processing delay reduces as the number of switches increases for the D-RESIN topology since data traffic is distributed among switches based on optimized processing delay. Using D-RESIN, the average processing delay at the access tier decreases by 52.43-88.82%, 32.71-87.91%, and 25.50-94.76% than using the existing schemes— T-RESIN, FlowMan, and RandomFlow, respectively. However, T-RESIN does not consider the delay parameter for the mathematical modeling of SDEN architecture at all. FlowMan also focuses on data management, specifically addressing scenarios of very high or very low data traffic, and in RandomFlow, data traffic is very random among switches. Figure 4.4 shows D-RESIN performance in handling the flows per switch as the number of IoT devices increases. Compared to the existing scheme, D-RESIN associates the data flows with SDN switches, ensuring that every flow should be processed optimally. Figure 4.5 provides insights into the D-RESIN switch throughput performance, crucial for assessing its capability to manage traffic as the number of connected devices grows for delay-sensitive applications.

On the other hand, from Figure 4.6, we observe that using D-RESIN, the average processing delay at the edge tier decreases by 35.44-85.10%, 55.41-84.89% than using the ex-

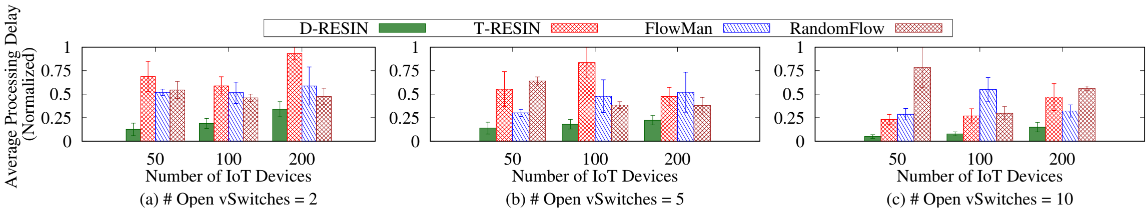


Figure 4.3: Average Processing Delay at Switch

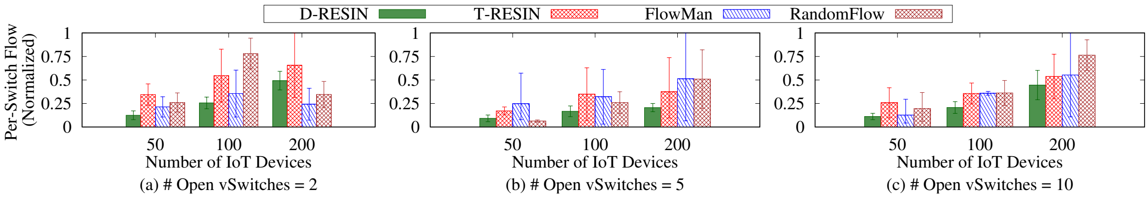


Figure 4.4: Per-Switch Flow

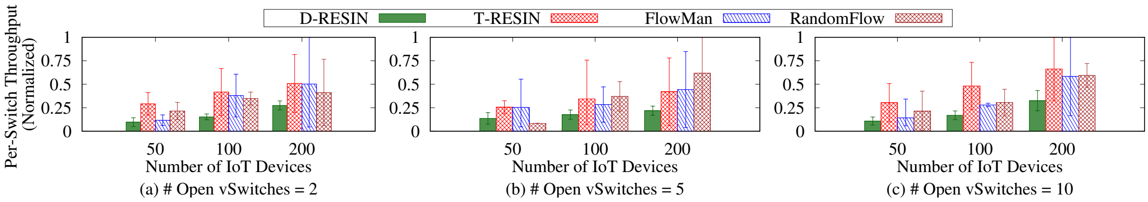


Figure 4.5: Per-Switch Throughput

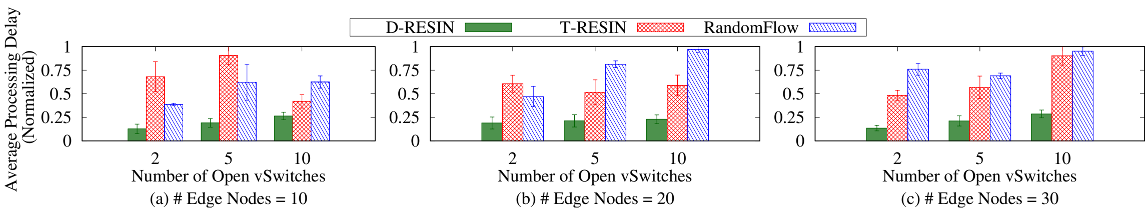


Figure 4.6: Average Processing Delay at Edge Node

isting schemes — T-RESIN and RandomFlow, respectively. This is because variation in the processing delay incurred at the edge nodes appears random for the T-RESIN scheme, and the data association is random for RandomFlow scheme at the edge tier. However, Figure 4.7 depicts the efficiency of D-RESIN at the edge tier regarding energy consumption as the network configuration scales with an increasing number of Open vSwitches. As the number of switches and edge nodes increases, D-RESIN effectively distributes data traffic and optimizes resource allocation to reduce average processing delay at both the access and edge tiers. The D-RESIN framework demonstrates improved scalability even in larger and more complex network environments. Hence, we argue that D-RESIN ensures a deduction in average processing delay at access and edge tiers for the proposed SDEN architecture.

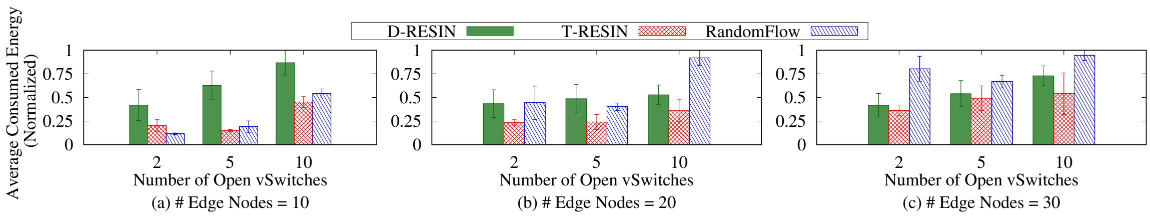


Figure 4.7: Average Energy Consumption at Edge Node

4.4 Conclusion

In this chapter, we proposed a novel framework, named D-RESIN, that addresses the challenges of dynamic resource orchestration in SDEN, ensuring high QoS for delay-sensitive applications. We leveraged the evolutionary game theoretic approach to ensure optimal processing delays between IoT devices, Open vSwitches, and edge nodes managed by the Ryu SDN controller. We demonstrated the existence of evolutionary equilibrium for D-RESIN at both access and edge tiers by proof. We observed that D-RESIN significantly reduces computation delay at both access and edge tiers to enhance the scalability and efficiency of SDEN. D-RESIN optimizes resource allocation in various IoT scenarios, such as smart traffic management, industrial IoT, and real-time healthcare monitoring, by dynamically adjusting network resources using an evolutionary game-theoretic approach.

Chapter 5

TRON: Traffic Management and Resource Allocation for SDN-Enabled 5G/6G Networks

In this chapter, we introduce TRON, an adaptive traffic management and resource allocation framework designed for SDN-Enabled 5G/6G networks. TRON leverages OpenFlow Group Tables to dynamically optimize link utilization across network slices for heterogeneous traffic such as data and VoIP. Unlike traditional static allocation strategies, TRON provides real-time adaptability by adjusting bucket weights within group tables, thereby ensuring balanced traffic distribution and efficient use of available network capacity. Figure 5.1 depicts the proposed Group Table-based traffic management architecture. It demonstrates how multiple hosts are interconnected through SDN-enabled switches, controlled centrally by an SDN controller. The Group Tables at various switches dynamically manage outgoing and incoming traffic of buckets for link utilization.

This chapter is organized as follows. Section 5.1 presents the system model and problem formulation. Section 5.2 introduces the TRON framework and its adaptive traffic management algorithm. Section 5.3 evaluates TRON through experimental analysis, discussing benchmarks, performance metrics, and results. Finally, Section 5.4 concludes the chapter with key insights.

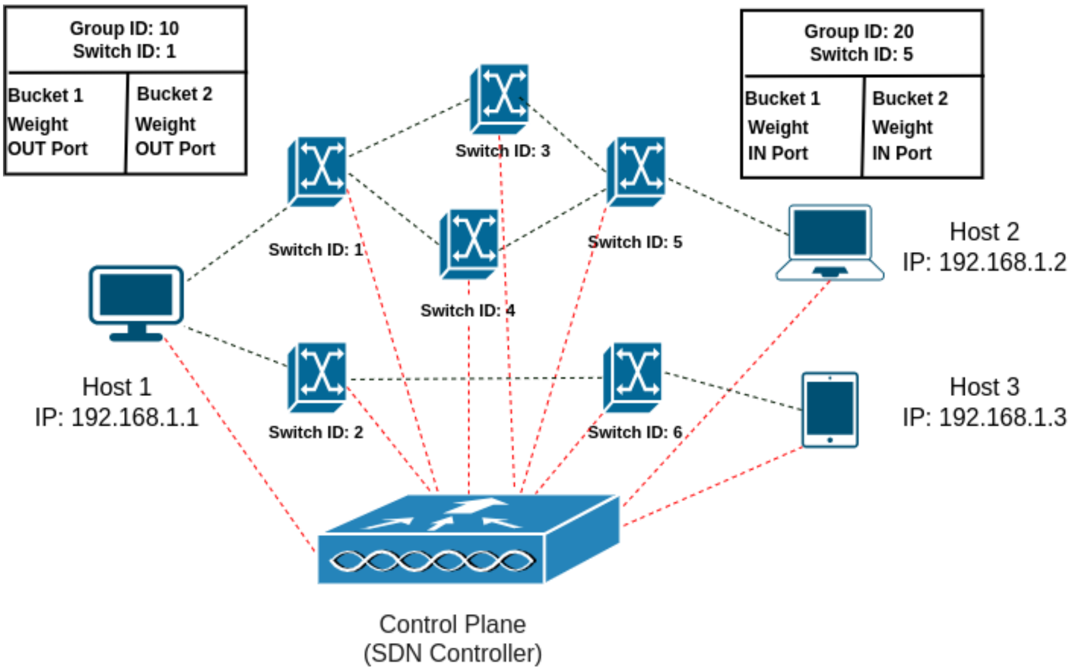


Figure 5.1: Schematic Architecture of Traffic Management using Group Tables in SDN-Enabled 5G/6G Networks.

5.1 System Model

We consider set of IoT devices $\mathcal{N}$ as hosts that generate the data traffic passing through the set of SDN switches $\mathcal{S}$. Each switch maintains OpenFlow's Group Table and each Group Table has multiple buckets with different weights and traffic splitting policies. The SDN Controller continuously monitors link utilization in real time and dynamically programs the switches using OpenFlow APIs (application programming interface), and intelligently redirects network flows as needed. Additionally, it updates the Group Tables within the switches based on current traffic loads. Incoming flows are split across different paths based on bucket weights to support heterogeneous traffic. To maintain feasibility, the number of flow rules installed at SDN switches must satisfy the constraint mentioned in Equation (5.1), where F_s represents the number of active flow entries and $R_s^{\max}$ is the flow table capacity of each switch $s \in \mathcal{S}$.

$$F_s \leq R_s^{max} \tag{5.1}$$

Let $\mathcal{L}$ be the set of physical links between switches and λ_i denotes the traffic arrival rate from an IoT device $i \in \mathcal{N}$. Each link $l \in \mathcal{L}$ has a capacity denoted by C_l, and its utilization at time t is represented by $U_l(t)$. Associated with each link $l \in \mathcal{L}$, there is a Group Table consisting of a set of buckets $\mathcal{B}_l$. Each bucket $b \in \mathcal{B}_l$ has an assigned weight w_b, satisfying the following conditions:

$$\sum_{b \in \mathcal{B}_l} w_b = 1, \quad \forall l \in \mathcal{L} \tag{5.2}$$

$$0 \leq w_b \leq 1, \quad \forall b \in \mathcal{B}_l \tag{5.3}$$

Definition 3. *The incoming traffic to each switch $s \in \mathcal{S}$ is split across buckets based on their weights. Thus, we define the traffic allocated $\lambda_b(t)$ to a bucket $b \in \mathcal{B}_l$ at time t as follows:*

$$\lambda_b(t) = w_b \times \sum_{i \in \mathcal{N}} \lambda_i(t) \tag{5.4}$$

Definition 4. *We define link utilization $U_l(t)$ of each link $l \in \mathcal{L}$ at time t as the ratio of the total incoming traffic on the link to its capacity C_l. Mathematically,*

$$U_l(t) = \frac{\sum\limits_{b \in \mathcal{B}_l} \lambda_b(t)}{C_l} \tag{5.5}$$

Expanding the traffic distribution across buckets $b \in \mathcal{B}_l$, the link utilization is expressed as follows:

$$U_l(t) = \frac{\sum\limits_{b \in \mathcal{B}_l} \left(w_b \times \sum\limits_{i \in \mathcal{N}} \lambda_i(t) \right)}{C_l} \tag{5.6}$$

Given that the bucket weights satisfy the normalization condition $\sum_{b \in \mathcal{B}_l} w_b = 1$, the link utilization simplifies to:

$$U_l(t) = \frac{\sum_{i \in \mathcal{N}} \lambda_i(t)}{C_l} \tag{5.7}$$

If $U_l(t) \leq 1$, the link can accommodate the current traffic load. Conversely, when $U_l(t) > 1$, congestion occurs, the SDN controller redirects flows dynamically by adjusting bucket weights.

5.2 TRON: The Proposed Traffic Management and Resource Allocation Framework

We propose a lightweight heuristic framework, *named TRON,* for adaptive traffic load balancing in SDN-enabled 5G/6G networks. The framework is based on real-time monitoring of link utilization and dynamic adjustment of OpenFlow's Group Table bucket weights. The detailed procedure is outlined in Algorithm 5.1. Initially, the controller assigns predefined bucket weights $w_b(t)$ across the available paths to distribute the traffic load. The controller collects traffic statistics from each switch including flow counts and byte counts for each group table by sending OpenFlow `GroupStatsRequest` and `GroupDescStatsRequest` messages. The controller adjusts bucket weights to manage flow distribution and splits the incoming flows across different paths based on Definition 3. The controller objective is presented in Equation (5.8) to dynamically update the bucket weights $w_b(t)$ of group table of each switch $s \in \mathcal{S}$ over time t while satisfying the constraints mentioned in Equations (5.2) and (5.3).

$$\min_{w_b(t)} \max_{l \in \mathcal{L}} U_l(t) \tag{5.8}$$

Using the collected statistics, the controller estimates the link utilization $U_l(t)$ for each link $l \in \mathcal{L}$ based on Equation (5.7). If the link utilization exceeds a predefined threshold θ, the link is marked as congested (see Definition 4). In response, the controller updates the bucket weights. Specifically, it decreases $w_b(t)$ for heavily loaded paths and increases $w_b(t)$

for lightly loaded paths according to the rule as follows:

$$w_b(t+1) = w_b(t) \times (1 - \alpha(U_l(t) - \theta)) \tag{5.9}$$

where α is a positive constant to control the magnitude. Following the weight adjustment, the controller normalizes the bucket weights which ensure the following constraint.

$$\sum_{b \in \mathcal{B}_l} w_b(t+1) = 1 \tag{5.10}$$

Finally, the controller reprograms the switches by sending updated `OFPGroupMod` messages containing the new bucket weights $w_b(t+1)$. The network throughput $T(t)$ at time t represents the total amount of data successfully received from all IoT devices. It is represented as the sum of the received traffic rates $\lambda_i^{recv}(t)$ at the receiver hosts. Mathematically,

$$T(t) = \sum_{i \in \mathcal{N}} \lambda_i^{recv}(t) \quad \forall i \in \mathcal{N} \tag{5.11}$$

This monitoring and adaptation process is repeated periodically, enabling real-time dynamic load balancing and improved link utilization across the network.

5.3 Complexity Analysis

The computational complexity of the proposed Algorithm 5.1 is primarily depends on its core operations within each control cycle. Initially, the controller collects group table statistics from each switch which requires $\mathcal{O}(|\mathcal{S}|)$ time. For every physical link $|\mathcal{L}|$, the algorithm 5.1 estimates link utilization, checks for congestion, and updates bucket weights, resulting in $\mathcal{O}(|\mathcal{L}| \times B)$ time, where B is the maximum number of buckets per group table. Subsequently, group table updation at switches contributes an additional $\mathcal{O}(|\mathcal{S}|)$ time, and network throughput calculation across IoT devices adds $\mathcal{O}(|\mathcal{N}|)$ time. Therefore, the total time complexity of Algorithm 5.1 is $\mathcal{O}(|\mathcal{S}| + |\mathcal{L}| \times B + |\mathcal{N}|)$. Since $|\mathcal{L}|$ typically dominates $|\mathcal{S}|$ and $|\mathcal{N}|$, the overall time complexity simplifies to $\mathcal{O}(|\mathcal{L}| \times B)$. This linear complexity ensures that the TRON framework remains lightweight and scalable, capable of supporting real-time

Algorithm 5.1 TRON: Adaptive Traffic Load Balancing Using Group Tables

INPUTS:
$\mathcal{N}$: IoT Devices, $\mathcal{S}$: SDN Switches, θ: Congestion Threshold
OUTPUTS:
$w_b(t+1)$: Updated bucket weights, $T(t)$: Network Throughput
PROCEDURE:
Deploy the network topology (For Data and VoIP traffic) using Mininet.
Assign initial bucket weights $w_b(0)$ at switches using Group Tables.
Initialize the Ryu controller to manage switches and install initial flow entries.
while network is operational **do**
  for Each switch $s \in \mathcal{S}$ **do**
    Collect Group Statistics (flow and byte counts) via OpenFlow messages.
  end for
  for Each link $l \in \mathcal{L}$ **do**
    Estimate link utilization using Eq. (5.7).
    if $U_l(t) > \theta$ **then**
      Mark link l as congested.
      for Each bucket $b \in \mathcal{B}_l$ **do**
        Update bucket weight of the group table using Eq. (5.9).
      end for
      Normalize bucket weights to satisfy the constraint in Eq. (5.10).
    end if
  end for
  for Each switch $s \in \mathcal{S}$ **do**
    Update Group Tables with new weights $w_b(t+1)$.
  end for
  Compute network throughput $T(t)$ using Eq. (5.11).
  return $w_b(t+1),\ T(t)$
end while

adaptive traffic management.

5.4 Performance Analysis

5.4.1 Experimental Setup

This section outlines the experimental setup to evaluate the adaptive load balancing framework, named TRON for link utilization in an SDN-enabled 5G/6G network. The setup leverages Mininet[1] for network simulation, Ryu[2] as the SDN controller, and Open vSwitches[3] are

[1] https://mininet.org/
[2] https://ryu-sdn.org/
[3] https://www.openvswitch.org/

Table 5.1: Experimental Setup

Hardware	Intel® Core™ i7-9700 CPU @3.00GHz × 8
Operating System	Ubuntu 20.04.6 LTS
RAM	24 GB DDR4
Disk Space	1.0 TB
Network Emulator	Mininet (Version 2.31b1)
SDN Controller	Ryu Controller (Version ryu 4.34)
SDN Switch	Open vSwitch (Version ovs-vsctl 2.13.8)
Network Traffic Generator	iPerf tool (Version 2.0.13)
Programming Language	Python3 (Version 3.8.10)
Benchmarks	D-RESIN [99], RandomFlow

Table 5.2: Simulation Parameters

Parameter	Value
Number of IoT Devices	50
Number of Open vSwitches	5, 10
Maximum Link Capacity	10 Mbps
Congestion Threshold	$\theta = 0.85$
Simulation Duration	120 Seconds

served as the virtual switches to facilitate communication between hosts, as mentioned in Table 5.1. We evaluated the performance across two topological configurations, consisting of 50 IoT devices distributed among either 5 or 10 Open vSwitches, as mentioned in Table 5.2. Each host is connected to its corresponding switch using TCLink links configured with a maximum link capacity of 10 Mbps. Traffic generation is performed using the iPerf[4] tool, creating both TCP flows for data transfer and UDP flows marked with specific QoS parameters to simulate VoIP traffic. Simulation time is a duration of 120 seconds to evaluate how the system handles traffic load.

5.4.2 Benchmarks

To evaluate the performance of the proposed TRON framework, we perform a comparative analysis against two benchmark schemes: D-RESIN [99] and RandomFlow. D-RESIN is a dynamic resource orchestration method based on evolutionary game theory that minimizes processing delays in IoT-enabled SDN. It dynamically adjusts resource according

[4]https://software.es.net/iperf/

to network conditions to enhance efficiency. In contrast, RandomFlow represents a baseline strategy where flows are randomly distributed across available paths without considering link utilization and congestion levels. These benchmarks enable a comprehensive evaluation of TRON's capability to improve link utilization and network throughput under heterogeneous traffic conditions.

5.4.3 Performance Metrics

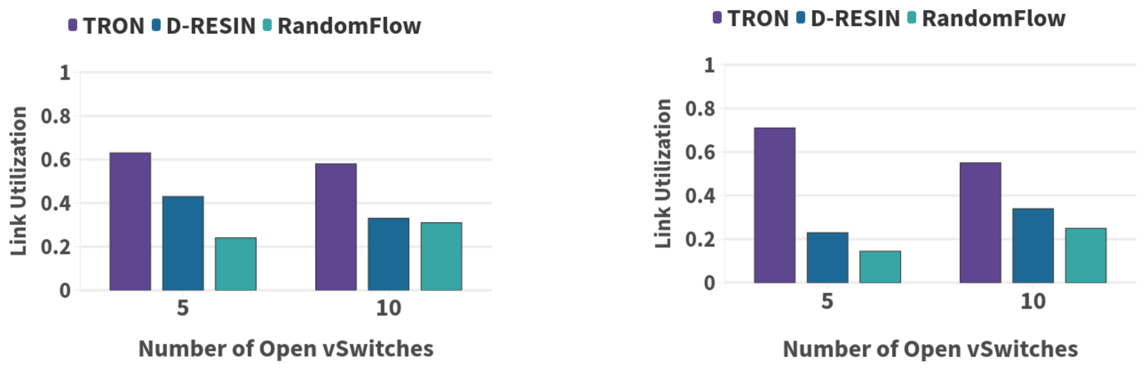
- *Link Utilization:* Link utilization metric provides insights how traffic is distributed and processed within the SDN switches, highlighting the effectiveness of the TRON mechanism in balancing network loads. Specifically, it measures the efficiency of available link usage by evaluating the ratio of total incoming traffic on a link to its maximum capacity.

- *Network Throughput:* Network throughput examines the overall data transfer efficiency of the network with focusing on how well heterogeneous traffic types — data and VoIP are handled. It is computed as the total amount of data successfully received from all IoT devices at the receiver side. This metric reflects the ability of the dynamic traffic management scheme.

5.4.4 Result and Discussion

The performance of the proposed TRON framework is evaluated in terms of link utilization and network throughput for heterogeneous traffic scenarios involving data and VoIP traffic. Figure 5.2 shows that TRON achieves significantly better link utilization compared to D-RESIN and RandomFlow across both traffic types. For data traffic, TRON improves link utilization by approximately 22.5% compared to D-RESIN and by 33% compared to RandomFlow as shown in Figure 5.2(a). Similarly, for VoIP traffic, TRON achieves an improvement of 34.5% over D-RESIN and 43.25% over RandomFlow as shown in Figure 5.2(b). These results demonstrate that TRON dynamically adjusts the traffic distribution more efficiently than the existing work. It leads to better utilization of available link capacities and maintaining network stability under varying load conditions.

Furthermore, TRON achieves substantial gains in network throughput as well. Figure 5.3

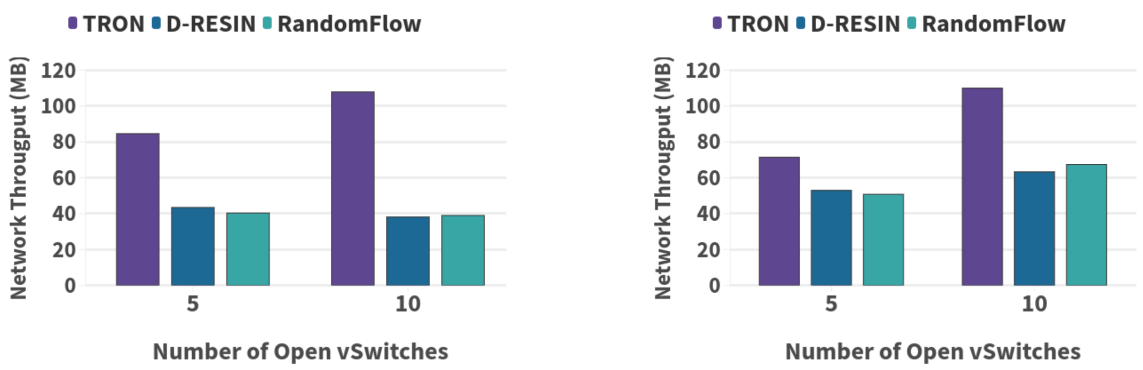


(a) Number of IoT Device: 50 and Traffic Type: Data Traffic

(b) Number of IoT Device: 50 and Traffic Type: VoIP Traffic

Figure 5.2: Link Utilization for Heterogeneous Network Traffic — Data, VoIP.

illustrates that TRON consistently achieves higher network throughput than D-RESIN and RandomFlow. As depicted in Figure 5.3(a), TRON improves network throughput by 55.44% and 56.55% compared to D-RESIN and RandomFlow, respectively for data traffic. Similarly, for VoIP traffic, TRON improves network throughput by 32.6% and 31.64% in comparison to D-RESIN and RandomFlow, respectively as shown in Figure 5.3(b). It demonstrates that TRON not only distributes network traffic more effectively but also significantly increases the end-to-end delivery of data under heterogeneous traffic loads. In summary, the evaluation results confirm that TRON effectively enhances link utilization and network throughput under heterogeneous traffic conditions in SDN-enabled 5G/6G networks.



(a) Number of IoT Device: 50 and Traffic Type: Data Traffic

(b) Number of IoT Device: 50 and Traffic Type: VoIP Traffic

Figure 5.3: Network Throughput for Heterogeneous Network Traffic — Data, VoIP.

5.5 Conclusion

In this chapter, we presented TRON, an adaptive traffic management and resource allocation framework for SDN-enabled 5G/6G networks. By leveraging OpenFlow's Group Tables and integrating real-time monitoring through the Ryu SDN controller, TRON dynamically adjusts traffic splitting to optimize link utilization and network throughput for heterogeneous traffic types. The proposed approach offers a lightweight, scalable, and effective solution for dynamic resource management in modern network slicing environments.

Chapter 6

Energy-Efficient Bandwidth Orchestration for SDN-Enabled 6G Networks

The advent of the sixth-generation (6G), wireless networks promises to revolutionize the telecommunications landscape by offering significantly high data rates, low latency, and enhanced reliability compared to the predecessors, i.e., 5G and 4G. For heterogeneous traffic in dynamic IoT environments, the existing work often results in either underutilized link capacity or congestion, leading to increased retransmission energy. This necessitates a dynamic SDN-enabled orchestration framework that adapts bandwidth provisioning in real time while minimizing energy consumption in softwarized 6G networks. In this chapter, we propose a novel architecture, *named FALCON,* that aims to manage bandwidth efficiently while addressing energy efficiency and sustainability concerns of IoT. By leveraging the advantages of SDN and bandwidth slicing, FALCON ensures an adaptive, resilient, and user-centric wireless network infrastructure. A key feature of FALCON is that it dynamically allocates available bandwidth in real time while significantly reducing packet loss, enhancing throughput, and optimizing energy consumption. Figure 6.1 depicts the integration of edge nodes and bandwidth slicing for ensuring optimal QoS. It highlights three network slices, where each meter table is used for managing different traffic types — video, VoIP, and data originating from IoT devices. The Ryu SDN Controller at the edge oversees the traffic distribution, ensuring optimal resource utilization across the slices.

This chapter is organized as follows. Section 6.1 describes the system model of the proposed framework, outlining the bandwidth and energy constraints of SDN-enabled IoT networks. Section 6.2 presents the FALCON architecture, including its justification for using a heuristic approach and detailed discussions of the S-FALCON and D-FALCON schemes. Section 6.3 provides performance analysis based on experimental evaluation and benchmarks, followed by a discussion of results. Finally, Section 6.4 concludes the chapter with key findings and implications for energy-efficient IoT deployments in softwarized 6G environments.

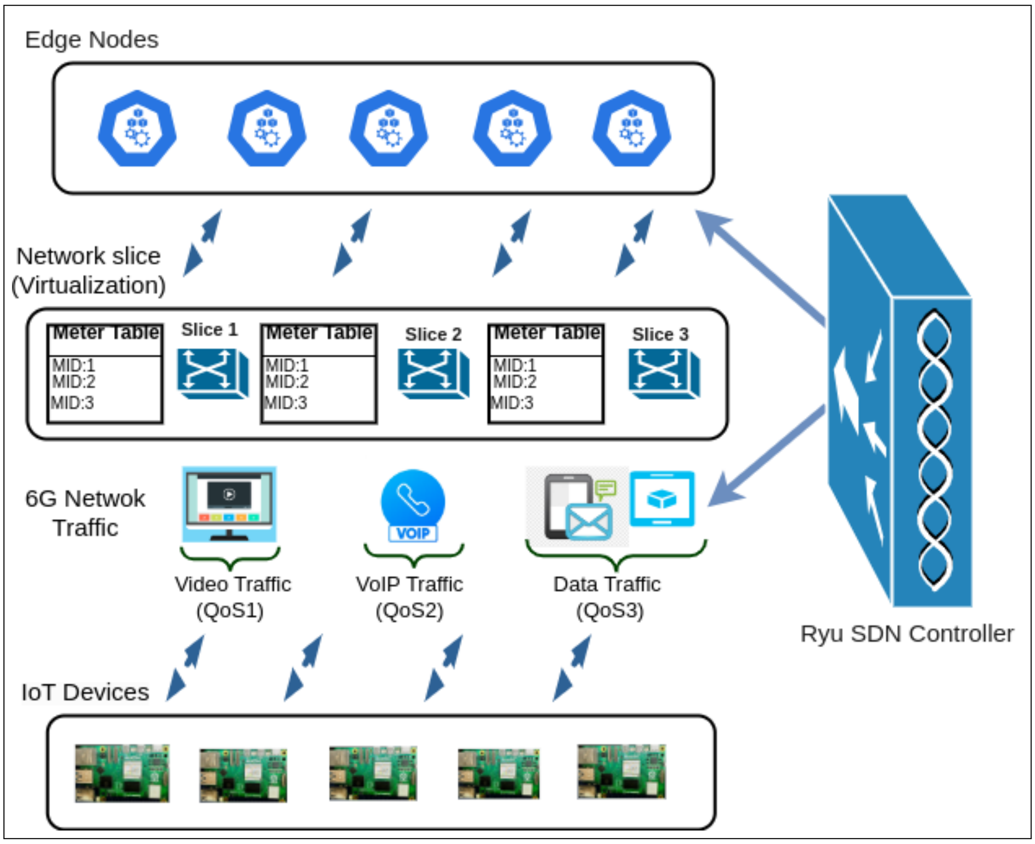


Figure 6.1: Schematic Architecture for Traffic Management in IoT-Enabled Softwarized 6G Networks

6.1 System Model

We consider that $\mathcal{N}$, $\mathcal{S}$, and $\mathcal{E}$ represent a set of IoT devices, SDN switches, and edge nodes, respectively. Each IoT device $n \in \mathcal{N}$ generates D_n data traffic. The bandwidth associated with each switch $s \in \mathcal{S}$ is represented as B_s. Each switch has a meter table [100] consisting of meter entries as a rate limiter to enable OpenFlow to implement various simple QoS operations. The meter table is represented with meter ID and associated bandwidth for each switch $s \in \mathcal{S}$. Let b_i represents the bandwidth requirement for traffic t_i, where $i \in (\mathbb{Z}^+ \cap [0, D_n])$ associated with meter m_j having a bandwidth b_j of Meter Table M_{xm} for the switch $s \in \mathcal{S}$. Thus, b_j represents a fraction of the total switch bandwidth B_s, such that $\sum_j b_j \leq B_s$. This indicates that b_j is a local bandwidth allocation under the global bandwidth constraint B_s. If incoming traffic t_i exceeds the bandwidth b_j of the meter, the overflow traffic is considered *dropped traffic*. Hence, the traffic throughput T_i is measured as:

$$T_i = \begin{cases} b_i, & \text{if } b_i \leq b_j \\ b_j, & \text{otherwise} \end{cases} \tag{6.1}$$

Each switch $s \in \mathcal{S}$ has a ternary content addressable memory (TCAM) with capacity of maximum flow rules R_s^{max}. We consider that F_s represents the number of flow rules associated with each switch $s \in \mathcal{S}$ and needs to satisfy the constraint —

$$F_s \leq R_s^{max} \tag{6.2}$$

The energy consumption at an edge node depends on the amount of data processed, the bandwidth allocation, and any overheads associated with maintaining QoS (e.g., packet drops, throughput optimization). Based on the work by Heinzelman *et al.* [101], the energy required to transmit a $k-$bit packet over a distance d depends on energy consumed per bit by the transmitter and energy consumed by the transmitter's amplifier for distance-based signal propagation. We do not consider amplifier energy, as the edge nodes and IoT devices are within a single hop range of SDN switches. Hence, energy consumption E_{use}^e of each edge node $e \in \mathcal{E}$ is as follows:

$$E_{use}^e = E_{\text{Tx}} \sum_{n \in \mathcal{N}} \sum_{i \in [0, D_n]} x_{t_i, n, e} T_i \tag{6.3}$$

where an edge node $e \in \mathcal{E}$ processes T_i data transmitted from the IoT devices; and E_{Tx} denotes the energy cost to transmit one unit of data traffic. Here, the binary variable $x_{t_i,n,e}$ is defined as follows:

$$x_{t_i,n,e} = \begin{cases} 1, & \text{if traffic } t_i \text{ of IoT devices } n \in \mathcal{N} \text{ is} \\ & \text{associated with an edge node } e \\ 0, & \text{otherwise} \end{cases} \tag{6.4}$$

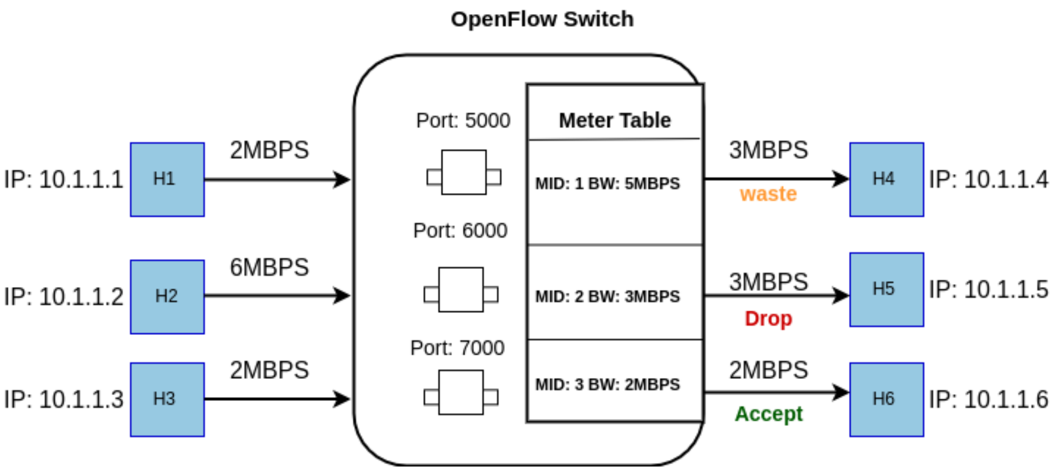


Figure 6.2: Bandwidth Allocation in an OpenFlow Switch with Static Meter Table.

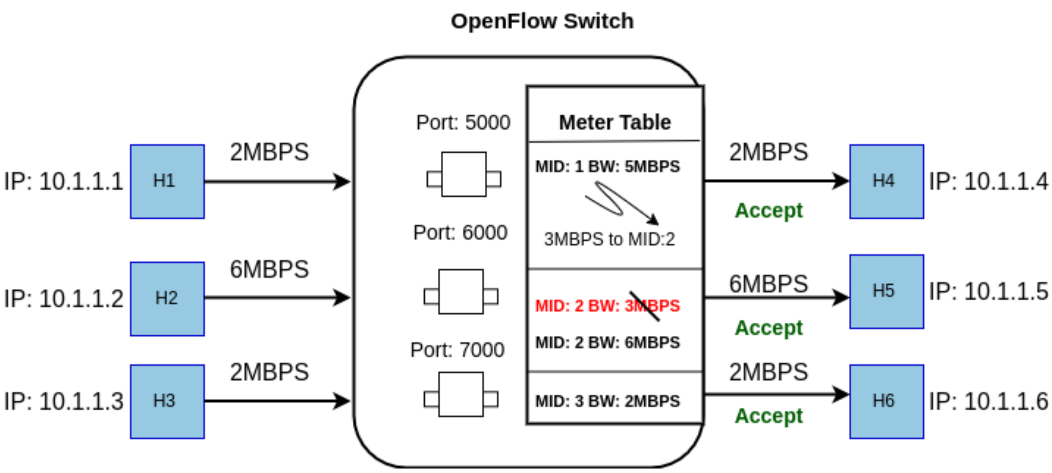


Figure 6.3: Bandwidth Allocation in an OpenFlow Switch with Adaptive Meter Table.

Problem Scenario: Figure 6.2 illustrates how an OpenFlow switch employs a meter table for static bandwidth allocation among hosts (H1, H2, H3). It highlights bandwidth assignments and the resulting traffic classifications, i.e., accepted, dropped, or considered waste, for hosts (H4, H5, H6). Due to fixed meter assignments, excess traffic from some hosts is dropped even when unused bandwidth exists elsewhere, leading to poor utilization and degraded QoS for critical IoT applications. In contrast, Figure 6.3, the meter table dynamically adjusts bandwidth among hosts (H1, H2, H3) based on real-time traffic demands. It shows how bandwidth is reassigned from MID 1 to MID 2, allowing all traffic to be accepted for hosts (H4, H5, H6), improving bandwidth utilization. Together, these figures underscore the transition from static to dynamic bandwidth management in OpenFlow switches, illustrating the critical improvements in performance and resource allocation efficiency achievable through the FALCON architecture.

6.2 FALCON: The Proposed Energy Efficient Bandwidth Orchestration Framework

In this section, we propose a heuristic-based model for efficient bandwidth management in a softwarized 6G network, utilizing two algorithms — S-FALCON for static bandwidth allocation and D-FALCON for dynamic bandwidth allocation. The model employs a heuristic approach to address the complexities of bandwidth allocation in heterogeneous traffic environments by integrating SDN with network slicing and meter tables.

6.2.1 Justification for using Heuristic Approach

The problem of dynamic bandwidth allocation in softwarized 6G IoT networks is inherently complex due to the combinatorial nature of flow-switch-edge mapping and multiple constraints on bandwidth and energy consumption. Hence, this problem can be modeled as a three-layered *bipartite graph* [93, 102], where IoT traffic flows, SDN switches, and edge nodes form distinct layers. The mapping of flows to switches and subsequently to edge nodes constitutes a *multistage assignment problem* [93, 102], which is an NP-hard problem. Furthermore, modeling our problem as a three-layered *bipartite graph* inherently ensures the absence of odd-length cycles. This is because IoT traffic flows, SDN switches, and edge nodes constitute distinct, non-overlapping vertex sets, as required by the definition of a *bipartite graph*. Traditional optimization and learning-based methods often result in high computational overhead and latency; hence, these solutions are impractical for real-time, resource-constrained IoT environments. In contrast, heuristic approaches provide a lightweight and scalable solution, with minimal computational effort. The FALCON framework employs a heuristic strategy to dynamically reallocate bandwidth using programmable SDN meter tables, while ensuring a trade-off between performance and computational efficiency.

6.2.2 S-FALCON: Static Bandwidth Allocation

S-FALCON is presented in Algorithm 6.1 and provides a novel integration of static bandwidth allocation with OpenFlow meter tables tailored for heterogeneous IoT traffic in softwarized 6G networks. It offers analytical insight into the limitations of fixed provisioning that the existing works have not addressed, and also contributes by modeling energy consumption explicitly under static constraints. Moreover, S-FALCON serves as a necessary baseline to demonstrate the performance and energy-efficiency gains of the dynamic D-FALCON algorithm.

6.2.3 D-FALCON: Dynamic Bandwidth Allocation

We propose D-FALCON, i.e., Algorithm 6.2, to allocate bandwidth dynamically across IoT devices, switches, and edge nodes. This approach ensures adaptive resource allocation and improved energy efficiency while minimizing packet loss. It also leverages real-time traffic monitoring to reallocate bandwidth resources based on traffic type and demand dynamically. It ensures that high-priority flows, such as VoIP and video traffic, receive sufficient bandwidth while maintaining network stability. We aim to *maximize* the objective function $f(y_s)$ while ensuring that an optimal amount of the bandwidth is allocated by meter table M_{xm} of each switch $s \in \mathcal{S}$. Hence, we define $f(y_s)$ as follows:

$$f(y_s) = \alpha \sum_{s \in \mathcal{S}} \sum_{n \in \mathcal{N}} \sum_{i \in [0, D_n]} y_{t_i,n,s} T_i - \beta \sum_{s \in \mathcal{S}} P_{drop}(s) \tag{6.5}$$

where $P_{drop}(s)$ refers to the packet drop at each switch $s \in \mathcal{S}$. Here, the parameters α and β serve as weights in the objective function, balancing the trade-off between maximizing throughput and minimizing packet drops. The parameters α and β are positive real values in the range $(0, 1)$, where α emphasizes throughput maximization and β prioritizes packet drop minimization. These parameters depend heavily on the specific operational context, network requirements, and priority constraints of individual IoT applications. Finally, the binary decision variable $y_{t_i,n,s}$ is defined as follows:

$$y_{t_i,n,s} = \begin{cases} 1, & \text{if traffic } t_i \text{ of IoT devices } n \in \mathcal{N} \text{ is} \\ & \text{transmitted through switch } s \\ 0, & \text{otherwise.} \end{cases} \tag{6.6}$$

For packet drop consideration, there is additional retransmission energy E_{drop} represented in Equation (6.7), where E_{Tx} is the energy cost of retransmitting a dropped packet.

$$E_{drop} = E_{\text{Tx}} \sum_{s \in \mathcal{S}} P_{drop}(s) \tag{6.7}$$

The total energy consumption E^e_{total} at an edge node $e \in \mathcal{E}$ is expressed as:

$$E^e_{total} = E^e_{use} + E_{drop} \tag{6.8}$$

In FALCON, we aim to minimize the total energy consumption E^e_{total} of an edge node $e \in \mathcal{E}$ with respect to the binary decision variable $x_{t_i,n,e}$. This variable indicates that traffic t_i from IoT device $n \in \mathcal{N}$ is associated with an edge node e. Mathematically,

$$\arg_{x_{t_i,n,e}} \min E^e_{total} \tag{6.9}$$

while satisfying the following constraints.

Bandwidth Allocation Constraint: The total allocated bandwidth for each switch s cannot exceed the available bandwidth B_s. Mathematically,

$$\sum_{t_i} y_{t_i,n,s} b_i \leq B_s, \quad \forall s \in \mathcal{S} \tag{6.10}$$

Meter Table Fairness Constraint: Each meter M_{xm} prioritizes high-traffic flows, such as video and VoIP, while ensuring fairness for low-priority flows. Hence, we get —

$$y_{t_i,n,s} M_{xm} \leq \frac{B_s}{F_s}, \quad \forall M_{xm} \in \mathcal{S} \tag{6.11}$$

Algorithm 6.1 S-FALCON: Static Bandwidth Allocation Algorithm

INPUTS:
$\mathcal{N}$: IoT Devices, $\mathcal{S}$: SDN Switches, $\mathcal{E}$: Edge Nodes, M_{xm}: Meter Table
OUTPUT:
T_i: Traffic Throughput, P_{drop}: Packet Drop, E^e_{total}: Total Energy Consumption
PARAMETERS:
b_i: Bandwidth requirement for traffic t_i, B_s: Bandwidth of switch s
PROCEDURE:
for Each switch $s \in \mathcal{S}$ and edge node $e \in \mathcal{E}$ **do**
  for Each meter $m \in M_{xm}$ **do**
    if $b_i \leq b_j$ **then**
      Calculate output traffic using Equation (6.1).
      Compute energy consumption E^e_{use} using Equation (6.3).
      Return T_i, E^e_{use}
    else
      Calculate output traffic using Equation (6.1) and drop the overflow traffic.
      Compute energy consumption E^e_{use} and energy for packet drop E_{drop} using Equations (6.3) and (6.7), respectively.
      Return T_i, P_{drop}, E_{drop}, E^e_{use}
    end if
  end for
  Compute total energy consumption E^e_{total} of an edge node e using Equation (6.8).
end for
return $T_i, P_{drop}, E^e_{total}$

Edge Node Energy Constraint: The total energy consumption at an edge node $e \in \mathcal{E}$ does not exceed its maximum energy capacity E^e to prevent energy exhaustion of edge nodes while optimizing network performance. Mathematically,

$$E^e_{total} \leq E^e, \quad \forall e \in \mathcal{E} \tag{6.12}$$

On the other hand, using FALCON, the packet drop $P_{drop}(s)$ for traffic flow t_i is minimized by dynamically reallocating unused bandwidth from underutilized meters, where

$$P_{drop}(s) = \frac{\sum_{\text{OM}}(b_i - B_s)}{T_i} \times 100 \tag{6.13}$$

OM denotes the overloaded meters of meter table M_{xm}. Equation (6.13) defines the packet drop percentage at each switch $s \in \mathcal{S}$ based on the amount of traffic exceeding the bandwidth allocation by overloaded meters. Specifically, it calculates the overflow traffic $(b_i - B_s)$ at

Algorithm 6.2 D-FALCON: Dynamic Bandwidth Allocation Algorithm

INPUTS:
$\mathcal{N}$: IoT Devices, $\mathcal{S}$: SDN Switches, $\mathcal{E}$: Edge Nodes, M_{xm}: Meter Table
OUTPUT:
T_i: Optimized Traffic Throughput, P_{drop}: Packet Drop, E^e_{total}: Total Energy Consumption
PARAMETERS:
b_i: Bandwidth requirement for traffic t_i, B_s: Bandwidth of switch s
PROCEDURE:
$cbw \leftarrow 0$
Initialize $new\_meters = [\,]$
for Each switch $s \in \mathcal{S}$ **do**
    for Each meter $m \in M_{xm}$ **do**
        if $(b_i > b_{config})$ **then**
            $new\_meters \leftarrow m$
        end if
        Calculate total consumed bandwidth as: $cbw \leftarrow cbw + b_i$
    end for
    Compute $available\_bw = B_s - cbw$
end for
if new_meters is empty **then**
    Return 0
end if
for Each meter m in new_meters **do**
    Gather list of meters with free bandwidth.
    Adjust bandwidth allocations for meters with free capacity.
end for
for Each switch $s \in \mathcal{S}$ and edge node $e \in \mathcal{E}$ **do**
    Reallocate bandwidth and adjust the meter table dynamically to optimize throughput
    and reduce packet drop.
    Calculate output traffic using Equation (6.1).
    Compute total energy consumption E^e_{total} using Equation (6.8).
    Return T_i, P_{drop}, E^e_{total}
end for
Return optimized traffic throughput, reduced packet drop, and total energy consumption based on the updated meter table.
return $T_i, P_{drop}, E^e_{total}$

each overloaded meter and normalizes this by the total throughput, converting it into a percentage. Therefore, Equation (6.13) reflects the performance for bandwidth allocation using meter tables in SDN-based IoT networks. Moreover, bandwidth reallocation is dynamically adjusted based on real-time traffic demands, as represented below.

$$y_s(t) = \arg\max_{y_s} \left(\frac{\text{Available bandwidth}}{\text{Traffic type demand}} \right) \tag{6.14}$$

6.2.4 Complexity Analysis

Let $|\mathcal{S}|$, $|\mathcal{E}|$, and $|\mathcal{M}|$ denote the number of SDN switches, the number of edge nodes, and the average number of meter entries per switch, respectively. These notations are uniformly used for the complexity analysis of both algorithms — S-FALCON and D-FALCON for consistency. The time complexity of Algorithm 6.1 (S-FALCON) is $\mathcal{O}(|\mathcal{S}||\mathcal{E}||\mathcal{M}|)$, which results from the nested iterations over switches, edge nodes, and their associated meter table entries during static bandwidth allocation. The time complexity of Algorithm 6.2 (D-FALCON) is $\mathcal{O}(|\mathcal{S}||\mathcal{M}| + |\mathcal{M}|^2 + |\mathcal{S}||\mathcal{E}|)$. This complexity arises because the algorithm first iterates over all switches and their respective meter entries to monitor bandwidth utilization, followed by bandwidth adjustments among overloaded and underutilized meters, which requires pairwise comparisons between meter entries. The number of IoT devices $|\mathcal{N}|$ influences the volume of incoming traffic processed by SDN switches and edge nodes. Although this impact is implicitly captured through traffic-related computations and has a limited effect on computational complexity compared to the structural parameters $|\mathcal{S}|$, $|\mathcal{E}|$, and $|\mathcal{M}|$, which drive bandwidth allocation and meter table adjustments. It is to be noted that increasing $|\mathcal{N}|$ affects runtime in practical deployments.

6.3 Performance Analysis

6.3.1 Experimental Setup

This section details the experimental setup to evaluate the FALCON architecture for efficient bandwidth management in a softwarized 6G network. The setup leverages Mininet[1] for network simulation, Ryu[2] as the SDN controller, and Open vSwitches[3] are served as the virtual switches to facilitate communication between hosts and manage flow entries, as mentioned

[1]https://mininet.org/
[2]https://ryu-sdn.org/
[3]https://www.openvswitch.org/

Table 6.1: Experimental Setup

Hardware	Intel® Core™ i7-9700 CPU @3.00GHz × 8
Operating System	Ubuntu 20.04.6 LTS
RAM	24 GB DDR4
Disk Space	1.0 TB
Network Emulator	Mininet (Version 2.31b1)
SDN Controller	Ryu Controller (Version ryu 4.34)
SDN Switch	Open vSwitch (Version ovs-vsctl 2.13.8)
Network Traffic Generator	iPerf tool (Version 2.0.13)
Network Interface Standard	Ethernet
Programming Language	Python3 (Version 3.8.10)
Benchmarks	T-RESIN, S-FALCON

Table 6.2: Simulation Parameters

Parameter	Value
Number of IoT Devices	50
Number of Open vSwitches	5
Number of Edge Nodes	10
Maximum Link Capacity	10 Mbps
Bandwidth for Meter-Table Entry	5 Mbps, 2.5 Mbps, 2.5 Mbps
Initial Energy of each Edge node	20 Joule [98]
Network Energy Consumption	50 nJ/bit [101]
Ethernet frame Size	1518 Byte [97]
Simulation Duration	120 Seconds

in Table 6.1. We evaluated the performance of FALCON architecture for topological configuration, including 50 IoT devices, 5 Open vSwitches, and 10 edge nodes as depicted in Table 6.2. Each host is connected to every switch using TCLink, configured with a maximum link capacity of 10 Mbps to simulate realistic bandwidth constraints. Traffic generation is performed using the iPerf[4] tool to create UDP traffic, including data, VoIP, and video, simulating different traffic types and loads. Each test case runs for 120 seconds to observe how the system handles traffic flows under realistic network constraints. We define the QoS parameter as the allocated bandwidth for each meter-table entry. For simulation, we consider three-meter entries of the meter table — M1: 5 Mbps, M2: 2.5 Mbps, and M3: 2.5 Mbps as allocated bandwidth for each switch. Table 6.3 outlines the five test cases (T1-T5) with varying incoming traffic distributions assigned to each SDN switch's meter table at different bandwidth levels.

[4]https://software.es.net/iperf/

Table 6.3: Test cases for Incoming Traffic Associated with SDN Switch's Meter Table

Testcases	Incoming Traffic (Mbps)		
	M1	M2	M3
T1	5	5	5
T2	5	1	4
T3	1	7	2
T4	8	1	1
T4	1	1	8

6.3.2 Benchmarks

We assess the performance of D-FALCON, i.e., Algorithm 6.2, and S-FALCON, i.e., Algorithm 6.1, with the existing scheme — T-RESIN. For S-FALCON scheme, SDN switch's meter tables are static and preinstalled. Hence, S-FALCON employs a static bandwidth allocation model where resources are pre-allocated without real-time adaptation. On the other hand, we consider T-RESIN as a benchmark proposed by Agrawal *et al.* [98] in comparison with D-FALCON scheme. The authors optimally allocated the resources to achieve high throughput and data flows, supporting a sustainable SDEN network. However, T-RESIN did not use the concept of a meter table for dynamic bandwidth allocation.

6.3.3 Performance Metrics

- *Packet Drop:* It calculates the percentage of packets that are dropped during transmission in the network. This metric helps to assess the reliability of the network and the effectiveness of bandwidth management under varying load conditions.

- *Switch Traffic:* It is calculated as the total amount of traffic handled by each Open vSwitch per unit of time. It provides insights into how traffic is distributed and processed within the SDN switch.

- *Network Throughput:* It examines the overall throughput of the network, focusing on how efficiently the network can handle heterogeneous traffic types. It helps evaluate network performance and resource allocation using dynamic bandwidth allocation.

- *Energy Consumption at Edge Node:* The total energy consumed at the edge node includes energy required for transmission and retransmission of data. It is evaluated based on the amount of data processed and allocated bandwidth. FALCON framework minimizes the retransmission energy by reducing packet loss and efficient bandwidth management to maintain network sustainability.

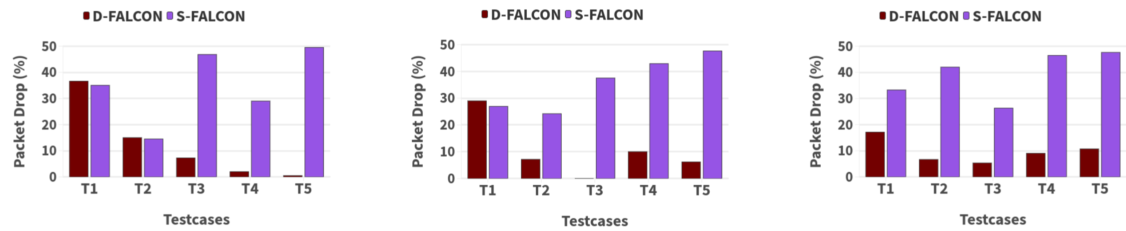
6.3.4 Result and Discussion

This section presents the performance evaluation of the proposed FALCON framework based on extensive simulation. We evaluate key network parameters such as packet drop rate, switch traffic, overall network throughput, and energy consumption at the edge nodes to demonstrate the effectiveness of dynamic bandwidth allocation. The proposed D-FALCON scheme is attributed to its real-time adaptive capability and dynamic bandwidth management. D-FALCON continuously monitors bandwidth usage through programmable SDN meter tables and dynamically reallocates resources from underutilized traffic flows to those experiencing congestion. This adaptive reallocation directly reduces packet drops, maximizes throughput, and minimizes retransmission-related energy consumption. Thus, the flexibility offered by real-time, heuristic-driven adjustments in D-FALCON inherently ensures better network performance and energy efficiency, particularly under fluctuating and heterogeneous IoT traffic conditions.

Figure 6.4 depicts that S-FALCON experiences significant variability, with packet drop percentages fluctuating significantly across test cases. In contrast, D-FALCON demonstrates a more stable performance with lower packet drop rates. For data traffic, S-FALCON shows an average packet drop of 35.98%, while D-FALCON improves this with a substantially lower average of 11.78%, as shown in Figure 6.4(a). Similarly, for VoIP traffic, S-FALCON records an average packet drop of 35.84%, whereas D-FALCON outperforms with a reduction to 10.47%, as illustrated in Figure 6.4(b). Figure 6.4(c) depicts packet drop percentages for video traffic, where S-FALCON has a fluctuating average packet drop of 43.18% in contrast to D-FALCON with an average packet drop of 13.78%. The results suggest that D-FALCON outperforms S-FALCON while reducing the overall packet drop by 26.32% over data, VoIP, and video traffic.

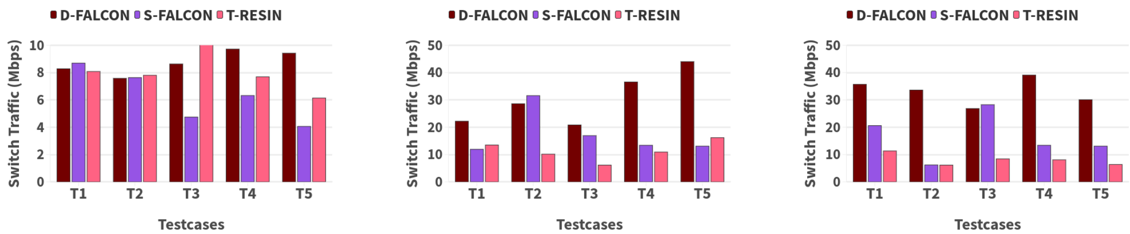
Switch traffic, as illustrated in Figure 6.5, demonstrates the efficiency of D-FALCON over S-FALCON and T-RESIN for all traffic scenarios. As depicted in Figures 6.5(a)–6.5(c), using D-FALCON, the associated switch traffic improves by 2.44%, 15.06%, and 13.73%

compared to S-FALCON for different traffic — data, VoIP, and video, respectively. For data traffic, D-FALCON and T-RESIN handle almost the same switch traffic as shown in Figure 6.5(a). Figures 6.5(b) and 6.5(c) demonstrate that D-FALCON outperforms T-RESIN in handling switch traffic with 21.07%, and 25% for VoIP and video traffic scenarios, respectively. In summary, D-FALCON tends to handle switch traffic across all scenarios, particularly in resource-intensive cases like video and VoIP, while S-FALCON and T-RESIN maintain a low switch traffic.



(a) Data Traffic (b) VoIP Traffic (c) Video Traffic

Figure 6.4: Percentage of Packet Drop for Heterogeneous Network Traffic — Data, VoIP, Video.

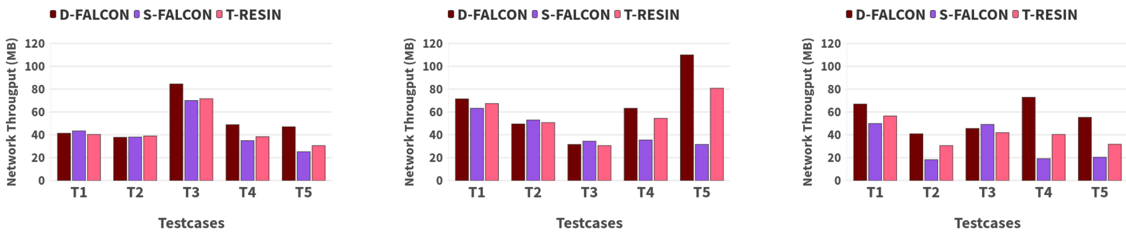


(a) Data Traffic (b) VoIP Traffic (c) Video Traffic

Figure 6.5: Analysis of Switch Traffic for Heterogeneous Network Traffic — Data, VoIP, Video.

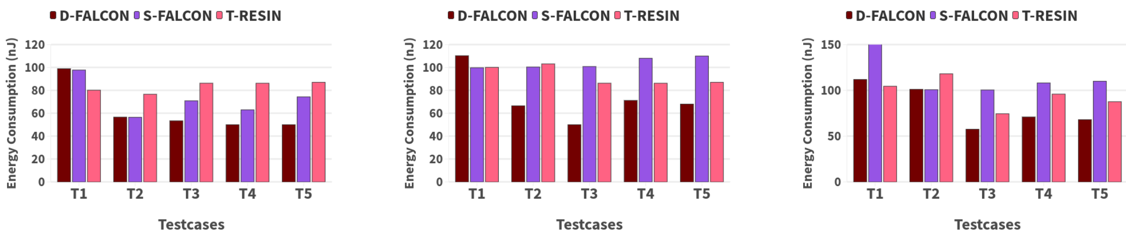
Figure 6.6(a) analyzes the network throughput for D-FALCON, S-FALCON, and T-RESIN for data traffic. D-FALCON increases network throughput by 9.59% and 7.94% in comparison to S-FALCON and T-RESIN, respectively. Similarly, for VoIP traffic, D-FALCON improves network throughput by 21.61% and 8.36% in comparison to S-FALCON and T-RESIN, respectively, as depicted in Figure 6.6(b). Figure 6.6(c) also analyzes that D-FALCON achieves higher throughput in high resource-intensive test cases, and T-RESIN and S-FALCON perform similarly in most test cases. D-FALCON improves network throughput by 24.95% and 16.09% in comparison to S-FALCON and T-RESIN, respectively, for video traffic.

As illustrated in Figure 6.7, D-FALCON demonstrates a remarkable reduction in energy consumption at edge nodes across all test cases and traffic types. D-FALCON reduces energy consumption by 10.71% and 21.5% compared to S-FALCON and T-RESIN, respectively, for data traffic, as shown in Figure 6.7(a). The reduction is even more significant for VoIP traffic, with 30.62% and 19.37% reductions compared to S-FALCON and T-RESIN, respectively, as observed in Figure 6.7(b). Figure 6.7(c) depicts the energy consumption reduction of D-FALCON by 39.82% and 12.24% with respect to S-FALCON and T-RESIN, respectively.



(a) Data Traffic (b) VoIP Traffic (c) Video Traffic

Figure 6.6: Overall Network Throughput for Heterogeneous Network Traffic — Data, VoIP, Video.



(a) Data Traffic (b) VoIP Traffic (c) Video Traffic

Figure 6.7: Energy Consumption at Edge Nodes for Heterogeneous Network Traffic — Data, VoIP, Video.

These findings highlight D-FALCON's efficacy in dynamically managing network resources within IoT environments, particularly in reducing packet drops, efficiently handling higher traffic loads, and significantly lowering energy consumption at edge nodes. This supports the sustainability goals of modern telecommunication networks, aligning with the advancements and high reliability required by next-generation 6G infrastructures.

6.4 Conclusion

In this chapter, we presented FALCON, a network architecture, that incorporates SDN and network slicing for enhanced bandwidth management in softwarized 6G networks, particularly tailored for IoT applications. FALCON dynamically optimizes bandwidth allocation to efficiently handle diverse traffic types, including data, VoIP, and video, that are critical in IoT environments. We observed that FALCON significantly reduces packet loss, increases throughput, and optimizes energy consumption effectively. Furthermore, the dynamic allocation strategies of D-FALCON outperform the static methods employed by S-FALCON. The adaptive FALCON, i.e., D-FALCON, enhances adaptability for fluctuating network conditions and sustainability objectives of advanced 6G developments in the IoT ecosystems.

Chapter 7

Conclusions and Future Research Directions

The objective of this thesis is to design adaptive and efficient resource orchestration mechanisms suitable for SDENs. Throughout the thesis, we addressed several key challenges associated with next-generation IoT ecosystems including heterogeneous traffic behaviors, stringent delay requirements, limited edge tier computational resources, dynamic bandwidth constraints, and energy-efficiency considerations in softwarized 5G/6G environments. Chapters 3 and 4 focused on the analytical modeling and formulation of throughput-aware and delay-aware orchestration schemes for SDENs. Chapters 5 and 6 presented the design of adaptive traffic-management and energy-efficient bandwidth-orchestration mechanisms for SDN-enabled 5G/6G networks supporting heterogeneous data, VoIP, and video traffic. Finally, we summarized the overall findings of this Thesis and outlined the potential avenues for future research.

In this final chapter, we take a holistic view of our work discussed so far. We discuss the summary of this thesis in Section 9.1. We list out the major contributions of the thesis in Section 9.2. In Section 9.3, we discuss the limitations of our work. Finally, we conclude the thesis while citing future directions in Section 9.4.

7.1 Summary of the Thesis

This thesis is presented in seven chapters. Chapter 1 presented a brief introduction to SDEN, next-generation wireless Networks, and IIoT. We discussed the motivation of the work while citing the main objectives of this Thesis.

In Chapter 2, we reviewed the existing literature on resource allocation for SDN-enabled IoT networks, delay-sensitive IoT applications, multi-tier edge computing, and 5G/6G environments. We highlighted the research gaps and summarized the limitations of the existing schemes, thereby identifying the problem area addressed in this Thesis.

In Chapter 3, we addressed the problem of resource allocation in IoE-enabled SDENs. In the existing literature, the researchers considered optimizing the performance of the SDN platform using a single-tier architecture, where the IoT devices are in the same tier. However, with the advent of edge computing, we can explore the two-tier architecture of edge networks - local and edge tiers - in the presence of SDN, which has not been explored. Hence, we propose a novel scheme, *named T-RESIN*, an evolutionary game-based resource allocation scheme for software-defined edge networks. Additionally, we aim to optimize the QoS of the edge-based IoE services while optimizing the throughput of the system. The IoT devices use the proposed scheme in the local tier to identify the optimized mapping to the SDN switches. On the other hand, in the edge tier, the proposed scheme aims to optimize the throughput while allocating the IoE service to the optimal subset of edge devices. We evaluated the performance of the T-RESIN scheme using the Python3-based Mininet platform in the presence of Ryu Controller and Open vSwitches. Compared to existing schemes such as RandomFlow and FlowMan [63], T-RESIN improves network throughput by 27.98–31.84%, reduces energy consumption by 4.59–14.57%, and lowers computational overhead at edge nodes by 9.5–22.29%, demonstrating its efficiency in managing heterogeneous IoE workloads in SDENs.

Chapter 4 presented a novel framework, *named D-RESIN*, designed to dynamically orchestrate resources within IoT-enabled SDN at the edge, explicitly focusing on minimizing delays. The proposed framework employs evolutionary game-theoretic approach to manage and optimize resource allocation across IoT devices, Open vSwitches, and Edge nodes. We implemented the proposed D-RESIN schemes using the Mininet network emulator with Ryu

SDN controller and Open vSwitches. Ryu is an element-based software defined networking framework, which offers software elements with well described API that enables developers to produce new network control functions. We observed that D-RESIN reduces average processing delay at the access tier by 52.43-88.82% and 32.71-87.91% compared to the existing schemes — T-RESIN [98] and FlowMan [63], respectively. At the edge tier, D-RESIN decreases the average processing delay by 35.44-85.10% compared to T-RESIN [98]. These simulation results highlight the effectiveness of D-RESIN in enhancing scalability and efficiency for delay-sensitive IoT applications.

Chapter 5 presented TRON, an adaptive traffic management and resource allocation framework, for SDN-enabled 5G/6G networks. The proposed framework leverages OpenFlow's Group Tables to dynamically optimize link utilization and traffic distribution across network paths. To effectively handle heterogeneous traffic patterns, TRON introduces dynamic traffic redirection using bucket weight adjustments for VoIP and data flows originating from IoT devices. We integrate the proposed TRON mechanism with the Ryu SDN controller to enable real-time flow redirection and validate it's effectiveness through a practical testbed using Mininet and Open vSwitch. Experimental results demonstrate that TRON improves link utilization by 22.5–43.25% and network throughput by 31.64–56.55% compared to baseline approaches such as D-RESIN [99] and RandomFlow. The simulation results highlight that TRON provides a lightweight and real-time traffic management framework for SDN-enabled 5G/6G networks.

In Chapter 6, we addressed the problem of heterogeneous bandwidth management in SDN-enabled 6G networks, which remains insufficiently explored for Industrial IoT applications. Hence, we propose a novel framework, *named FALCON*, to dynamically manage bandwidth in softwarized 6G networks while focusing on the sustainability of IIoT applications. FALCON integrates SDN, network slicing, and a heuristic-based bandwidth allocation strategy to address the challenges of heterogeneous traffic demands and sustainability. We introduce two variants: S-FALCON, which performs static bandwidth allocation using preconfigured meter table entries, and D-FALCON, which adaptively reallocates bandwidth in real time based on traffic fluctuations. The proposed D-RESIN framework utilizes programmable meter tables to perform dynamic bandwidth distribution at runtime, thereby reducing packet loss and enhancing network throughput for IIoT applications. The FALCON framework employs network bandwidth slicing to manage different traffic types, such as video, VoIP, and IIoT data, within distinct slices. We simulate the effectiveness of FALCON framework using the Mininet simulator. In FALCON, the Ryu SDN controller orchestrates bandwidth dynamically across SDN-capable Open vSwitches. The proposed dynamic FALCON scheme improves the average packet drop by 26.32% over data, VoIP, and video traffic than the existing schemes. D-FALCON improves overall network throughput by 7.94–24.95% and reduces edge node energy consumption by 10.71–39.82% across heterogeneous IIoT traffic, including data, VoIP, and video, compared to the benchmark schemes — T-RESIN [98] and S-FALCON. Through extensive simulation, we observe that FALCON reduces packet loss, enhances throughput, and reduces energy consumption at edge nodes for heterogeneous data traffic in comparison to the existing schemes for the IIoT ecosystem.

7.2 Contributions

This section presents the salient features of the contributions made in the Thesis. A synthesis of these contributions and their outcomes is discussed in the following sections.

Throughput-Aware Resource Orchestration Scheme for SDEN: We presented a throughput-aware dynamic resource-orchestration framework for IoE-enabled SDENs. The proposed scheme modelled the interactions among IoT devices, SDN switches, and edge nodes using an evolutionary game-theoretic approach to facilitate adaptive flow and process allocation under bandwidth, computational, and memory constraints. Analytical insights and experimental evaluation collectively demonstrated that T-RESIN achieves significant improvements in network throughput and overall resource utilization.

Delay-Aware Resource Orchestration Scheme for SDEN: We developed a delay-aware orchestration framework D-RESIN to meet stringent QoS requirements for latency-critical IoT applications. The scheme establishes evolutionary equilibrium across access and edge tiers and substantially reduces processing delay while enhancing QoS.

Traffic Management and Resource Allocation for SDN-Enabled 5G/6G Networks: We presented TRON, an adaptive traffic management and resource allocation mechanism designed for heterogeneous traffic in SDN-enabled 5G/6G networks. Leveraging OpenFlow

Group Tables, TRON dynamically adjusts bucket weights to distribute traffic efficiently across multiple paths. Testbed results using Mininet and Open vSwitch have shown improved link utilization, reduced network congestion, and higher overall network throughput.

Energy-Efficient Bandwidth Orchestration for SDN-Enabled 6G Networks: We introduced an energy-efficient bandwidth-orchestration framework FALCON for softwarized 6G-IIoT environments. The framework employed programmable meter tables to allocate bandwidth for heterogeneous IoT traffic through both static (S-FALCON) and dynamic (D-FALCON) strategies. Experimental analysis demonstrates reduced packet drops, improved throughput, and lower energy consumption at edge nodes, highlighting the effectiveness of the proposed scheme in sustainable 6G network environments.

7.3 Limitations

We made few assumptions while designing the proposed schemes.

- Each edge link between two nodes, i.e., IoT device and switch, has limited bandwidth.
- Full coverage of the network is ensured.
- The IoT devices are assumed to be stationary in nature.
- Ethernet is considered as the underlying networking standard.
- The SDEN architecture is comprised of IoT devices, SDN switches, and edge nodes with predefined bandwidth, computational, memory, and energy constraints.
- We considered that there is no network link failures and faulty IoT devices.
- The D-RESIN framework does not incorporate channel delay (transmission and propagation delay).
- The OpenFlow's Meter Table entries and their associated bandwidth allocations are considered to be fixed.

7.4 Future Research Directions

The work presented in this Thesis can be extended in several promising directions, as outlined below.

- As a future enhancement of T-RESIN, the management system can be improved to handle high device mobility and intermittent connectivity by incorporating intelligent handover mechanisms and predictive mobility models. We also plan to integrate edge–cloud computing platforms to leverage the computational power and storage capabilities of cloud resources, particularly in the presence of mobile nodes.

- The D-RESIN framework can be further extended by incorporating computation-related energy consumption and designing energy-efficient resource allocation mechanisms for SDENs. Additionally, D-RESIN may be enhanced to support priority-based IoT application management based on urgency and resource requirements.

- AI-driven traffic forecasting and adaptive resource allocation mechanisms can be incorporated to ensure seamless connectivity and minimize service disruptions in SDENs. The SDN controller can proactively allocate bandwidth, adjust flow priorities, and mitigate potential congestion by predicting traffic variations. These advancements will improve the overall network's resilience and efficiency in dynamic IoT environments.

- Future extension of TRON includes incorporating deep learning algorithms for predictive traffic management, applying game-theoretic models for strategic resource allocation, and exploring advanced optimization techniques to improve system efficiency and scalability in complex 5G/6G networks. The TRON framework also has the potential for integration with edge-cloud architectures, enabling context-aware flow control in latency-sensitive industrial and smart city applications.

- Future extension of FALCON includes focusing on bandwidth orchestration in IoT-enabled 6G network environments in the presence of network link failures and faulty IoT devices. The problem addressed in FALCON can also be revisited by incorporating machine learning technologies to dynamically refine resource allocation based on the availability of corresponding network configuration datasets.

Bibliography

[1] E. Siow, T. Tiropanis, and W. Hall, “Analytics for the Internet of Things: A Survey,” *ACM Computing Survey*, vol. 51, no. 4, pp. 1–36, 2019.

[2] M. M. Mogadem, Y. Li, and D. L. Meheretie, “A Survey on Internet of Energy Security: Related Fields, Challenges, Threats and Emerging Technologies,” *Cluster Computing*, vol. 25, pp. 2449–2485, 2022.

[3] M. B. M. Noor and W. H. Hassan, “Current Research on Internet of Things (IoT) Security: A Survey,” *Computer Networks*, vol. 148, pp. 283–294, 2019.

[4] L. Agrawal and N. Tiwari, “A Review on IoT Security Architecture: Attacks, Protocols, Trust Management Issues, and Elliptic Curve Cryptography,” in *Social Networking and Computational Intelligence*. Springer, Singapore, 2020, pp. 457–465.

[5] D. Evans, “The Internet of Everything: How More Relevant and Valuable Connections Will Change the World,” *Cisco Internet Business Solutions Group (IBSG), Cisco Systems, Inc., San Jose, CA, USA*, 2012.

[6] N. Ding, X. Ouyang, L. Gao, J. Huang, and G. Xing, “An Overview on Economic Analysis of Internet of Everything,” *IEEE Communications Surveys & Tutorials*, 2025.

[7] R. Dhaya and R. Kanthavel, “IoE Based Private Multi-Data Center Cloud Architecture Framework,” *Computers and Electrical Engineering*, vol. 100, 2022.

[8] I. Laroussi, L. Huan, and Z. Xiusheng, “How Will the Internet of Energy (IoE) Revolutionize the Electricity Sector? A Techno-Economic Review,” *Materials Today: Proceedings*, vol. 72, pp. 3297–3311, 2023.

[9] V. Tyagi, S. Singh, H. Wu, and S. S. Gill, “Load Balancing in SDN-Enabled WSNs Toward 6G IoE: Partial Cluster Migration Approach,” *IEEE Internet of Things Journal*, vol. 11, no. 18, pp. 29 557–29 568, 2024.

[10] W. Rafique, A. S. Hafid, and S. Cherkaoui, “Complementing IoT Services Using Software-Defined Information Centric Networks: A Comprehensive Survey,” *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 23 545–23 569, 2022.

[11] I. Alam, K. Sharif, F. Li, Z. Latif, M. M. K. arim, S. Biswas, B. Nour, and Y. Wang, “A Survey of Network Virtualization Techniques for Internet of Things Using SDN and NFV,” *ACM Computing Survey*, vol. 53, no. 2, pp. 1–40, 2020.

[12] M. Amadeo, C. Campolo, G. Ruggeri, A. Molinaro, and A. Iera, “SDN-Managed Provisioning of Named Computing Services in Edge Infrastructures,” *IEEE Transactions on Network and Service Management*, vol. 16, no. 4, pp. 1464–1478, 2019.

[13] R. Bruschi, F. Davoli, P. Lago, A. Lombardo, C. Lombardo, C. Rametta, and G. Schembra, “An SDN/NFV Platform for Personal Cloud Services,” *IEEE Transactions on Network and Service Management*, vol. 14, no. 4, pp. 1143–1156, 2017.

[14] A. Botta, W. de Donato, V. Persico, and A. Pescapé, “Integration of Cloud Computing and Internet of Things: A Survey,” *Future Generation Computer Systems*, vol. 56, pp. 684–700, 2016.

[15] C. Stergiou, K. E. Psannis, B.-G. Kim, and B. Gupta, “Secure Integration of IoT and Cloud Computing,” *Future Generation Computer Systems*, vol. 78, pp. 964–975, 2018.

[16] J. Dizdarevic, F. Carpio, A. Jukan, and X. Masip-Bruin, “A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration,” *ACM Computing Survey*, vol. 51, no. 6, pp. 1–29, 2019.

[17] M. Laroui, B. Nour, H. Moungla, M. A. Cherif, H. Afifi, and M. Guizani, “Edge and Fog Computing for IoT: A Survey on Current Research Activities & Future Directions,” *Computer Communications*, vol. 180, pp. 210–231, 2021.

[18] Q. Wu, S. Wang, H. Ge, P. Fan, Q. Fan, and K. B. Letaief, “Delay-Sensitive Task Offloading in Vehicular Fog Computing-Assisted Platoons,” *IEEE Transactions on Network and Service Management*, vol. 21, no. 2, pp. 2012–2026, 2023.

[19] G. Tefera, K. She, M. Shelke, and A. Ahmed, “Decentralized Adaptive Resource-Aware Computation Offloading & Caching for Multi-Access Edge Computing Networks,” *Sustainable Computing: Informatics and Systems*, vol. 30, 2021.

[20] S. S. Jazaeri, S. Jabbehdari, P. Asghari, and H. H. S. Javadi, “Edge Computing in SDN-IoT Networks: A Systematic Review of Issues, Challenges and Solutions,” *Cluster Computing*, vol. 24, pp. 3187–3228, 2021.

[21] A. C. Baktir, A. Ozgovde, and C. Ersoy, “How Can Edge Computing Benefit From Software-Defined Networking: A Survey, Use Cases, and Future Directions,” *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2359–2391, 2017.

[22] S. D. A. Shah, M. A. Gregory, S. Li, and R. D. R. Fontes, “SDN Enhanced Multi-Access Edge Computing (MEC) for E2E Mobility and QoS Management,” *IEEE Access*, vol. 8, pp. 77 459–77 469, 2020.

[23] M. Kim, D. Hyeon, J. Paek, and A. Kalla, “eTAS: Enhanced Time-Aware Shaper for Supporting Nonisochronous Emergency Traffic in Time-Sensitive Networks,” *IEEE Internet of Things Journal*, vol. 9, no. 13, pp. 10 480–10 491, 2022.

[24] Z. Liao, X. Pang, M. Jingyu Zhang, B. Xiong, and J. Wang, "Blockchain on Security and Forensics Management in Edge Computing for IoT: A Comprehensive Survey," *IEEE Transactions on Network and Service Management*, vol. 19, no. 2, pp. 1159–1175, 2021.

[25] W. Rafique, L. Qi, I. Yaqoob, M. Imran, R. U. Rasool, and W. Dou, "Complementing IoT Services Through Software Defined Networking and Edge Computing: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1761–1804, 2020.

[26] R. Trivisonno, R. Guerzoni, I. Vaishnavi, and A. Frimpong, "Network Resource Management and QoS in SDN-enabled 5G Systems," in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, San Diego, CA, USA, December 2015, pp. 1–7.

[27] S. Naithani, C. Sreenan, and A. Zahran, "SDN-Enabled Distributed Access Architecture Cable Networks," in *Proc. of IEEE 29th International Symposium on Local and Metropolitan Area Networks (LANMAN)*, London, United Kingdom, July 2023, pp. 1–6.

[28] P. P. Liborio, C. T. Lam, B. Ng, D. L. Guidoni, M. Curado, and L. A. Villas, "Airtime Aware Dynamic Network Slicing for Heterogeneous IoT Services in IEEE 802.11ah," in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, Nanjing, China, March 2021, pp. 1–6.

[29] S. Wijethilaka and M. Liyanage, "Survey on Network Slicing for Internet of Things Realization in 5G Networks," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 957–994, 2021.

[30] S. Ebrahimi, F. Bouali, and O. C. L. Haas, "Resource Management From Single-Domain 5G to End-to-End 6G Network Slicing: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 26, no. 4, pp. 2836–2866, 2024.

[31] T. Kumar, J. Partala, T. Nguyen, L. Agrawal, A. Mondal, A. Kumar, I. Ahmad, E. Peltonen, S. Pirttikangas, and E. Harjula, "Secure Edge Intelligence in the 6G Era," in *Security and Privacy for 6G Massive IoT*. John Wiley & Sons, Ltd, 2025, ch. 2, pp. 35–52.

[32] S. Bera and N. B. Mehta, "Network Slicing in 5G Edge Networks with Controlled Slice Redistributions," in *Proc. of the 17th International Conference on Network and Service Management (CNSM)*, Izmir, Turkey, December 2021, pp. 118–124.

[33] X. Zhang and Q. Zhu, "Scalable Virtualization and Offloading-Based Software-Defined Architecture for Heterogeneous Statistical QoS Provisioning Over 5G Multimedia Mobile Wireless Networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 12, pp. 2787–2804, 2018.

[34] W. Mao, Z. Zhao, Z. Chang, G. Min, and W. Gao, "Energy-Efficient Industrial Internet of Things: Overview and Open Issues," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7225–7237, 2021.

[35] I. Behnke and H. Austad, "Real-Time Performance of Industrial IoT Communication Technologies: A Review," *IEEE Internet of Things Journal*, vol. 11, no. 5, pp. 7399–7410, 2024.

[36] A. Sahbafard, R. Schmidt, F. Kaltenberger, A. Springer, and H.-P. Bernhard, "On the Performance of an Indoor Open-Source 5G Standalone Deployment," in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, Glasgow, United Kingdom, March 2023, pp. 1–6.

[37] B. A. Salau, A. Rawal, and D. B. Rawat, "Recent Advances in Artificial Intelligence for Wireless Internet of Things and Cyber–Physical Systems: A Comprehensive Survey," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 12 916–12 930, 2022.

[38] Y. Wu, H.-N. Dai, H. Wang, Z. Xiong, and S. Guo, "A Survey of Intelligent Network Slicing Management for Industrial IoT: Integrated Approaches for Smart Transportation, Smart Energy, and Smart Factory," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 2, pp. 1175–1211, 2022.

[39] H. Cao, H. Zhao, A. Jindal, G. S. Aujla, and L. Yang, "Energy-Efficient Virtual Resource Allocation of Slices in Vehicles-Assisted B5G Networks," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 3, pp. 1408–1417, 2022.

[40] Z. Sasan, M. Shokrnezhad, S. Khorsandi, and T. Taleb, "Joint Network Slicing, Routing, and In-Network Computing for Energy-Efficient 6G," in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, Dubai, UAE, April 2024, pp. 1–6.

[41] E. Li, L. Zeng, Z. Zhou, and X. Chen, "Edge AI: On-Demand Accelerating Deep Neural Network Inference via Edge Computing," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 447–457, 2020.

[42] J. Pan and J. McElhannon, "Future Edge Cloud and Edge Computing for Internet of Things Applications," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 439–449, 2018.

[43] G. Nain, K. Pattanaik, and G. Sharma, "Towards Edge Computing in Intelligent Manufacturing: Past, Present, and Future," *Journal of Manufacturing Systems*, vol. 62, pp. 588–611, 2022.

[44] D. J. Langleya, J. V. Doorn, I. C. Ng, S. Stieglitz, A. Lazovik, and A. Boonstra, "The Internet of Everything: Smart Things and Their Impact on Business Models," *Journal of Business Research*, vol. 122, pp. 853–863, 2021.

[45] Y. Liu, H.-N. Dai, Q. Wang, M. K. Shukla, and M. Imran, "Unmanned Aerial Vehicle for Internet of Everything: Opportunities and Challenges," *Computer Communications*, vol. 155, pp. 66–88, 2020.

[46] R. Singh and S. S. Gill, "Edge AI: A Survey," *Internet of Things and Cyber-Physical Systems*, vol. 3, pp. 71–92, 2023.

[47] J. Iannacci, "Internet of Things (IoT); Internet of Everything (IoE); Tactile Internet; 5G – A (not so Evanescent) Unifying Vision Empowered by EH-MEMS (Energy Harvesting MEMS) and RF-MEMS (Radio Frequency MEMS)," *Sensors and Actuators A: Physical*, vol. 272, pp. 187–198, 2018.

[48] G. Manogaran, T. N. Nguyen, J. Gao, and P. M. Kumar, "Deep Learning-Based Service Distribution Model for Wireless Network Assisted Internet of Everything," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 5, pp. 3004–3014, 2022.

[49] G. Wang, J. Yang, Y. Wei, C. Wang, K. Li, and C. Feng, "Deep-Reinforcement-Learning-Driven Patient State Analysis and Resource Management in Near-Field IoE Healthcare Networks," *IEEE Internet of Things Journal*, vol. 12, no. 13, pp. 22 647–22 657, 2025.

[50] S. H. Chae, S.-W. Jeon, and C. Jeong, "Efficient Resource Allocation for IoT Cellular Networks in the Presence of Inter-Band Interference," *IEEE Transactions on Communications*, vol. 67, no. 6, pp. 4299–4308, 2019.

[51] T. Kim, H. Park, Y. Jin, S.-S. Lee, and S. Lee, "Partition Placement and Resource Allocation for Multiple DNN-Based Applications in Heterogeneous IoT Environments," *IEEE Internet of Things Journal*, vol. 10, no. 11, pp. 9836–9848, 2023.

[52] Z. Guo, X. Zhu, Z. Wei, J. Cao, Y. Jiang, V. K. N. Lau, and S. Sun, "QoS-Aware Joint Massive Random Access Control and Resource Allocation With Colliding Preamble Reuse for Prioritized IoT," *IEEE Transactions on Vehicular Technology*, vol. 74, no. 7, pp. 11 143–11 160, 2025.

[53] G. Yang and Y. Yao, "Resource Allocation Control of UAV-Assisted IoT Communication Device," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 11, pp. 13 341–13 349, 2023.

[54] J. Son and R. Buyya, "Priority-Aware VM Allocation and Network Bandwidth Provisioning in Software-Defined Networking (SDN)-Enabled Clouds," *IEEE Transactions on Sustainable Computing*, vol. 4, no. 1, pp. 17–28, 2019.

[55] A. Bentaleb, A. C. Begen, and R. Zimmermann, "QoE-Aware Bandwidth Broker for HTTP Adaptive Streaming Flows in an SDN-Enabled HFC Network," *IEEE Transactions on Broadcasting*, vol. 64, no. 2, pp. 575–589, 2018.

[56] Y.-C. Wang and S.-Y. You, "An Efficient Route Management Framework for Load Balance and Overhead Reduction in SDN-Based Data Center Networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 4, pp. 1422–1434, 2018.

[57] S. Misra, A. Mondal, and S. Khajjayam, "Dynamic Big-Data Broadcast in Fat-Tree Data Center Networks With Mobile IoT Devices," *IEEE Systems Journal*, vol. 13, no. 3, pp. 2898–2905, 2019.

[58] S. Yuan, M. Peng, Y. Sun, and X. Liu, "Software-Defined Intelligent Satellite–Terrestrial Integrated Networks: Insights and Challenges," *Digital Communications and Networks*, vol. 9, no. 6, pp. 1331–1339, 2023.

[59] R. H. Jhaveri, S. V. Ramani, G. Srivastava, T. R. Gadekallu, and V. Aggarwal, "Fault-Resilience for Bandwidth Management in Industrial Software-Defined Networks," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 4, pp. 3129–3139, 2021.

[60] W. Ren, Y. Sun, H. Luo, and M. S. Obaidat, "A New Scheme for IoT Service Function Chains Orchestration in SDN-IoT Network Systems," *IEEE Systems Journal*, vol. 13, no. 4, pp. 4081–4092, 2019.

[61] M. Bagaa, T. Taleb, J. B. Bernabe, and A. Skarmeta, "QoS and Resource-Aware Security Orchestration and Life Cycle Management," *IEEE Transactions on Mobile Computing*, vol. 21, no. 8, pp. 2978–2993, 2022.

[62] S. R. Pokhrel, "Software Defined Internet of Vehicles for Automation and Orchestration," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3890–3899, 2021.

[63] A. Mondal and S. Misra, "FlowMan: QoS-Aware Dynamic Data Flow Management in Software-Defined Networks," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 7, pp. 1366–1373, 2020.

[64] V. Balasubramanian, M. Aloqaily, and M. Reisslein, "An SDN Architecture for Time-Sensitive Industrial IoT," *Computer Networks*, vol. 186, 2021.

[65] A. Mondal, S. Misra, and I. Maity, "Buffer Size Evaluation of OpenFlow Systems in Software-Defined Networks," *IEEE Systems Journal*, vol. 13, no. 2, pp. 1359–1366, 2019.

[66] N. Saha, S. Bera, and S. Misra, "Sway: Traffic-Aware QoS Routing in Software-Defined IoT," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 1, pp. 390–401, 2021.

[67] Q. Fan, J. Bai, H. Zhang, Y. Yi, and L. Liu, "Delay-Aware Resource Allocation in Fog-Assisted IoT Networks Through Reinforcement Learning," *IEEE Internet of Things Journal*, vol. 9, no. 7, pp. 5189–5199, 2022.

[68] D. S. Lakew, V. D. Tuong, N.-N. Dao, and S. Cho, "Adaptive Partial Offloading and Resource Harmonization in Wireless Edge Computing-Assisted IoE Networks," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 5, pp. 3028–3044, 2022.

[69] B. Sellami, A. Hakiri, S. B. Yahia, and P. Berthou, "Energy-Aware Task Scheduling and Offloading Using Deep Reinforcement Learning in SDN-Enabled IoT Network," *Computer Networks*, vol. 210, 2022.

[70] T. C. S. Xavier, F. C. Delicato, P. F. Pires, C. L. Amorim, W. Li, and A. Zomaya, "Managing Heterogeneous and Time-Sensitive IoT Applications Through Collaborative and Energy-Aware Resource Allocation," *ACM Transactions on Internet of Things*, vol. 3, no. 2, pp. 1–28, 2022.

[71] G. Manogaran and B. S. Rawal, "An Efficient Resource Allocation Scheme With Optimal Node Placement in IoT-Fog-Cloud Architecture," *IEEE Sensors Journal*, vol. 21, no. 22, pp. 25 106–25 113, 2021.

[72] S. C. Misra and A. Mondal, "FogPrime: Dynamic Pricing-Based Strategic Resource Management in Fog Networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 8, pp. 8227–8236, 2021.

[73] P. K. Deb, A. Mondal, and S. Misra, "AuGrid: Edge-Enabled Distributed Load Management for Smart Grid Service Providers," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 1, pp. 437–446, 2022.

[74] S. Deng, H. Zhao, W. Fang, J. Yin, S. Dustdar, and A. Y. Zomaya, "Edge Intelligence: The Confluence of Edge Computing and Artificial Intelligence," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7457–7469, 2020.

[75] X. Zhang, Y. Wang, S. Lu, L. Liu, L. Xu, and W. Shi, "OpenEI: An Open Framework for Edge Intelligence," in *Proc. of IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, Dallas, TX, USA, July 2019, pp. 1840–1851.

[76] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Towards an Intelligent Edge: Wireless Communication Meets Machine Learning," *IEEE Communications Magazine*, vol. 58, no. 1, pp. 19–25, 2020.

[77] J. Singh, P. Singh, M. Hedabou, and N. Kumar, "An Efficient Machine Learning-Based Resource Allocation Scheme for SDN-Enabled Fog Computing Environment," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 6, pp. 8004–8017, 2023.

[78] Y. He, Y. Wang, C. Qiu, Q. Lin, J. Li, and Z. Ming, "Blockchain-Based Edge Computing Resource Allocation in IoT: A Deep Reinforcement Learning Approach," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2226–2237, 2021.

[79] S. Ranjbaran, M. Amadeo, C. Marche, G. Ruggeri, A. Sinha, and M. Nitti, "Cloud-Edge Resource Management and Migration: Leveraging Online Learning for Digital Twin Re-placement," in *Proc. of IEEE 10th World Forum on Internet of Things (WF-IoT)*, Ottawa, ON, Canada, December 2024, pp. 1–6.

[80] H. Byeon, M. Alsaadi, A. Quraishi, A. AlGhamdi, T. A. Ahanger, I. Keshta, P. A. Xalikovich, M. Soni, and M. W. Bhatt, "Consumer Technology in Task Offloading and Edge Resource Allocation: AIoT and Edge Computing for Next-Generation Communication," *IEEE Transactions on Consumer Electronics*, vol. 71, no. 2, pp. 5356–5365, 2025.

[81] J. Huang, F. Yang, C. Chakraborty, Z. Guo, H. Zhang, L. Zhen, and K. Yu, "Opportunistic Capacity-Based Resource Allocation for 6G Wireless Systems With Network Slicing," *Future Generation Computer Systems*, vol. 140, pp. 390–401, 2023.

[82] L. Lyu, C. Chen, S. Zhu, and X. Guan, "5G Enabled Codesign of Energy-Efficient Transmission and Estimation for Industrial IoT Systems," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 6, pp. 2690–2704, 2018.

[83] M. Liyanagea, P. Porambageb, A. Y. Dingc, and A. Kalla, "Driving forces for Multi-Access Edge Computing (MEC) IoT integration in 5G," *ICT Express*, vol. 7, pp. 127–137, 2021.

[84] J. Li, C. Li, W. Yue, N. Cheng, Z. Sha, and M. Tian, "A Unified Framework for 6G Cross-Scenario Resource Representation and Scheduling," in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, Glasgow, United Kingdom, March 2023, pp. 1–6.

[85] A. Shukla, N. Ahmed, A. Roy, and S. C. Misra, "Softwarized Management of 6G Network for Green Internet of Things," *Computer Communications*, vol. 187, pp. 103–114, 2022.

[86] G. Manogaran, J. Ngangmeni, J. Stewart, D. B. Rawat, and T. N. Nguyen, "Deep-Learning-Based Concurrent Resource Allocation Method for Improving the Service Response of 6G Network-in-Box Users in UAV," *IEEE Internet of Things Journal*, vol. 10, no. 4, pp. 3130–3137, 2023.

[87] A. Thantharate and C. Beard, "ADAPTIVE6G: Adaptive Resource Management for Network Slicing Architectures in Current 5G and Future 6G Systems," *Journal of Network and Systems Management*, vol. 31, no. 9, 2022.

[88] J. Mei, X. Wang, K. Zheng, G. Boudreau, A. B. Sediq, and H. Abou-Zeid, "Intelligent Radio Access Network Slicing for Service Provisioning in 6G: A Hierarchical Deep Reinforcement Learning Approach," *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 6063–6078, 2021.

[89] A. Jean-Baptiste, P. Owezarski, P. Berthou, and I. Silvain, "Assessing the Energetical Cost of 5G Softwarization," in *Proc. of IEEE 30th International Symposium on Local and Metropolitan Area Networks (LANMAN)*, Boston, MA, USA, August 2024, pp. 33–38.

[90] H. Sami, H. Otrok, J. Bentahar, and A. Mourad, "AI-Based Resource Provisioning of IoE Services in 6G: A Deep Reinforcement Learning Approach," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3527–3540, 2021.

[91] D. K. Jain, S. K. S. Tyagi, S. Neelakandan, M. Prakash, and L. Natrayan, "Metaheuristic Optimization-Based Resource Allocation Technique for Cybertwin-Driven 6G on IoE Environment," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 7, pp. 4884–4892, 2022.

[92] Z. Han, D. Niyato, W. Saad, T. Basar, and A. Hjørungnes, *Game Theory in Wireless and Communication Networks*. Cambridge University Press, New York, NY, USA, 2012.

[93] H. P. Williams, *Logic and Integer Programming*, ser. International Series in Operations Research and Management Science. Springer, Boston, MA, US, March 2009.

[94] A. Drexl, "A Simulated Annealing Approach to the Multiconstraint Zero-One Knapsack Problem," *Computing*, vol. 40, no. 1, pp. 1–8, March 1988.

[95] S. Misra, G. Mali, and A. Mondal, "Distributed Topology Management for Wireless Multimedia Sensor Networks: Exploiting Connectivity and Cooperation," *International Journal of Communication Systems, Wiley*, vol. 28, no. 7, pp. 1367–1386, 2015.

[96] P. Bhavathankar, A. Mondal, and S. Misra, "Topology Control in the Presence of Jammers for Wireless Sensor Networks," *International Journal of Communication Systems, Wiley*, vol. 30, no. 13, January 2017.

[97] Z. Zhao, Z. Qiu, W. Pan, H. Li, L. Zheng, and Y. Gao, "Design and Implementation of a Frame Preemption Model Without Guard Bands for Time-Sensitive Networking," *Computer Networks*, vol. 243, 2024.

[98] L. Agrawal, A. Mondal, and M. S. Obaidat, "T-RESIN: Throughput-Aware Dynamic Resource Orchestration for IoE-Enabled Software-Defined Edge Networks," *International Journal of Communication Systems, Wiley*, vol. 37, no. 12, p. e5802, April 2024.

[99] L. Agrawal, A. Mondal, M. S. Obaidat, and E. Harjula, "Delay-Aware Dynamic Resource Orchestration for IoT-Enabled Software-Defined Edge Networks," *International Journal of Communication Systems, Wiley*, vol. 38, no. 7, p. e70072, March 2025.

[100] I. Seremet and S. Causevic, "Advances of configuring Quality of Service (QoS) in Software Defined Networks (SDN) by using meter table," in *Proc. of IEEE 19th International Symposium INFOTEH-JAHORINA (INFOTEH)*, East Sarajevo, Bosnia and Herzegovina, April 2020, pp. 1–5.

[101] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," in *Proc. of the 33rd Hawaii International Conference on System Sciences*, Hawaii, USA, 2000, pp. 1–10.

[102] W. Attaoui, E. Sabir, H. Elbiaze, and M. Guizani, "VNF and CNF Placement in 5G: Recent Advances and Future Trends," *IEEE Transactions on Network and Service Management*, vol. 20, no. 4, pp. 4698–4733, 2023.