UNIVERSITY OF LEEDS

Energy Efficient Processing in Opportunistic Vehicular Edge Clouds

Amal Aodah Alahmadi

Submitted in accordance with the requirements for the degree of

Doctor of Philosophy

The University of Leeds

School of Electronic and Electrical Engineering

March 2020

Intellectual Property Statement

The candidate confirms that the work submitted is his/her own, except where work which has formed part of jointly-authored publications has been included. The contribution of the candidate and the other authors to this work has been explicitly indicated below. The candidate confirms that appropriate credit has been given within the thesis where reference has been made to the work of others.

The work in Chapter 4 has appeared or will be partially appear in publications as follows:

A. A. Alahmadi, A. Q. Lawey, T. E. H. El-Gorashi and J. M. H. Elmirghani, "Distributed processing in vehicular cloud networks," *2017 8th International Conference on the Network of the Future (NOF)*, London, 2017, pp. 22-26.

The candidate developed the energy efficient distributed processing model in vehicular cloud, using Mixed Integer Linear Programming, generate the results and wrote the paper. Dr. Lawey, reviewed the model the generated results, and revised the paper. Dr. El-Gorashi revised the final draft of the paper. Prof. Elmirghani suggested the developed architecture and reviewed the paper.

A. A. Alahmadi, M. O. I. Musa, T. E. H. El-Gorashi and J. M. H. Elmirghani, "Energy Efficient Resource Allocation in Vehicular Cloud Based Architecture," *2019 21st International Conference on Transparent Optical Networks (ICTON)*, Angers, France, 2019, pp. 1-6.

The candidate extended the previous published model and considered a complete end-to-end architecture, generate the results and wrote the paper. Dr. Musa, reviewed the findings, and revised the paper. Dr. El-Gorashi validated the findings and revised the paper. Prof. Elmirghani reviewed the paper and suggested the review scope.

The work in Chapter 5 has appeared or will be partially appear in publications as follows:

R. Ma, A. A. Alahmadi, T. E. H. El-Gorashi and J. M. H. Elmirghani, "Energy Efficient Software Matching in Vehicular Fog," *2019 21st International Conference on Transparent Optical Networks (ICTON)*, Angers, France, 2019, pp. 1-4

The candidate developed the initial model, verified the generated results, and revised the paper. R.Ma produced the results and wrote the initial draft of the paper. Dr. El-Gorashi reviewed the paper. Prof. Elmirghani proposed the study and reviewed the paper.

Acknowledgements

First of all, I would like to express my sincere gratitude to my supervisor, Prof. Elmirghani for his insightful guidance, patience, and support throughout my PhD journey. Without his continuous encouragement and optimism, this achievement would not have been possible.

I would like to thank my co-supervisor, Dr. El-Gorashi, for her immense support, kindness, and encouragement in the past years.

I would also like to thank all past and current members of Institute of Communication and Power Networks (ICaPNet). It has been a great pleasure to share this journey with all of them. I appreciate their constructive discussions, support and friendship.

I would like to express my appreciation to my beloved family back home, my father, Aodah and my mother, Safia for their endless support, love and prayers. Big thanks to my sisters and brothers for their support and for being the joy of my life.

Most importantly, I would like to express my deepest gratitude and love to my husband, Nayef, for being my biggest supporter in anything and everything that I do in my life. I am blessed by sharing this journey with him, and blessed with his continuous love, care, patience, and encouragement.

Abstract

The Cisco Visual Networking Index of 2019 reports that more than six billion Machine-to-Machine (M2M) connections were added in 2017 and the number of connections is expected to grow by more than 50% by 2022. The proliferation of connected devices is accompanied by rapid growth in the generated traffic between the edge layer and data centres, and therefore is expected to lead to significant increase in the power consumption of the network infrastructure. This calls for new architectural designs capable of reducing the traffic congestion and power consumption in the network. At the same time, vehicles are going through a huge revolution in term of their on-board units and processing capabilities producing a new promising framework concept. This concept is a consequence of the integration of vehicles and cloud computing, referred to as Vehicular Edge Cloud (VEC).

This thesis investigates distributed processing in VECs, where a group of vehicles in a car park, at a charging station or at a road traffic intersection, cluster and form a temporary vehicular cloud by combining their computational resources in the cluster. We investigated the problem of energy efficient processing task allocation in VEC by developing a Mixed Integer Linear Programming (MILP) model to minimise power consumption by optimising the allocation of different processing tasks to the available network resources, cloud resources, fog resources and vehicular processing nodes resources. Three dimensions of processing allocation were investigated. The first dimension compared centralised processing (in the central cloud) to distributed processing (in the multi-layer fog nodes). The second dimension

introduced opportunistic processing in the vehicular nodes with low and high vehicular node density. The third dimension considered non-splittable tasks (single allocation) versus splittable tasks (distributed allocation), representing real-time versus non real-time applications respectively. The results revealed that a power savings up to 70% can be achieved by allocating processing to the vehicles. However, many factors have an impact on the power saving percentage such the vehicle capacities, vehicles density, workload size, and the number of generated tasks. It was observed that the power saving is improved by exploiting the flexibility of task splitting among the available vehicles. In addition to the processing allocation problem, this thesis investigated the software matching problem in VEC. The vehicles involved may not be equipped with the full set of software needed to process the tasks requested. Moreover, as vehicles in VEC represent processing at the edge layer, we studied the impact of edge processing on the propagation and queuing delay in a joint optimisation modelling intended to minimise both power consumption and delay. Our investigation showed a significant impact on the processing allocation decision and therefore, the power consumption, attributed to the location of the processing node and the service rate of the network controller.

Table of Contents

List of Tables

List of Figures

Abbreviation

AP	Access Point
BIP	Binary Integer Programming
CC	Central Cloud
CCA	Central Cloud Allocation
CFA	Cloud-Fog Allocation
CFVA	Cloud-Fog-VEC Allocation
CompaaS	Computing as a Service
DA	Distributed Allocation
DRR	Data Rate Ratio
DSRC	Dedicated Short-Range Communication
DVC	Dynamic Vehicular Cloud
EDR	Event Data Recorder
Gb/s	Giga bit per second
HVC	Hybrid Vehicular Cloud
IaaS	Infrastructure as a Service

ICT	Information and Communication Technology
IoT	Internet of Things
IoV	Internet of Vehicles
IP/WDM	Internet Protocol over Wavelength Division Multiplexing
ITS	Intelligent Transportation Systems
km	Kilometre
LAN	Local Area Network
LF	OLT Fog
M2M	Machine to Machine
Mb/s	Mega bit per second
MCC	Mobile Cloud Computing
MF	Metro Fog
MILP	Mixed Integer Linear Programming
MINLP	Mixed Integer Non-Linear Programming
MIPS	Million Instructions Per Second
NaaS	Networking as a Service
NF	ONU Fog

OBU	On-Board Unit
OLT	Optical Line Terminal
ONU	Optical Network Unit
P2P	Peer to Peer
PaaS	Platform as a Service
PN	Processing Node
PON	Passive Optical Network
PUE	Power Usage Effectiveness
RSU	Road-side Unit
SA	Single Allocation
SaaS	Software as a Service
SEaaS	Sensing as a Service
SN	Source Node
STaaS	Storage as a Service
SVC	Static Vehicular Cloud
V2I	Vehicle to Infrastructure
V2V	Vehicle to Vehicle

VANET	Vehicular Ad hoc Network
VEC	Vehicular Edge Cloud
VC	Vehicular Cloud
VM	Virtual Machine
VN	Vehicular Node
VNI	Visual Network Index
VoD	Video on Demand
VuC	Vehicles using Cloud
WAVE	Wireless Access in Vehicular Environment

Chapter 1
Introduction

The Cisco Visual Networking Index of 2019 reports that more than six billion M2M (Machine-to-Machine) connections were added in 2017, 28% of these connections are connected vehicles. This number is expected to increase by more than 50% by 2022. This expansion reveals exponential growth in global traffic estimated to exceeds 25 exabyte per month [1]. This growth is accompanied by a remarkable increase in energy consumption in the Information and Communication Technologies (ICT) sector. It is estimated that ICT technologies will be responsible for up to 12% of global emissions by 2030 [2]. With the increase of IoT applications and computational hungry cloud-based applications, emerging technologies such as distributed computing (or decentralised data centres) were developed. This framework is considered as a new extension of cloud computing, where requests are processed in a distributed fashion based on user location. Distributed cloud platforms can be composed of any available user-owned resources that allow processing, storage, networking and sensing. Following this concept, vehicular clouds can be formed of several vehicles with on-board processing, storage, and sensing devices. These vehicles are clustered together and act as servers in a mobile micro data centre at the edge of the network. Furthermore, with moving vehicles, the sensing capabilities of vehicles can be used as a form of mobile IoT platform with sensors that may include cameras, pollution sensors, traffic flow sensors and road surface sensors among others.

In this thesis we introduce a vehicular edge cloud framework (VEC) that can help transform edge processing, storage and sensing through the use of the capabilities of distributed vehicles to form intelligent vehicular based processing clusters at the edge of the network. These capabilities are expected to grow significantly with the introduction of autonomous vehicles in the near future. Currently, the enterprise parking lot may contain hundreds to thousands of vehicles that remain in the park for typically 7–8 hours per day [3]. If vehicles are connected in such a car park, using wireless connections or a fiber cable integrated with the charging cable and its plug, their processors (typically 2–10 processors per vehicle) can be networked, thus transforming the car park into a significant edge processing micro data centre. The vehicles may alternatively be equipped with a “processing box” that has processing, storage and wireless communication (WiFi for example) capabilities. Such a processing box can reduce security risks and eliminate the need to connect to the processors in the vehicle or can supplement the vehicle on-board processing capabilities. A set of VECs made up of the parking rows and floors in a car park can thus be formed. Similarly, cars in airports may be parked for one to two weeks, making the capabilities of such vehicles available to transform such car parks to processing units at the edge of the network on a semi-permanent basis as departing cars are replaced. On the shortest time scales, clusters of vehicles may be formed at traffic intersection points where the traffic light may own a computational problem and may assign chunks of such a computational problem to vehicle clusters at the intersection. The clusters report results before departing the intersection. At busy intersections in cities, typically at least one traffic stream is stationary, thus providing

opportunities to distribute computational tasks to nearby processors. These vehicles thus have the potential to form efficient short-term distributed computational resources at the edge of the network, much closer to the requesting entity.

With vehicles availability that can range from minutes to weeks, the networking and computational resources are highly dynamic. Therefore, appropriate network architectures and network algorithms are needed to better utilise these new forms of dynamic distributed computational resources. It is also essential to consider key design features in such networks including energy efficiency, latency, reliability and availability. The work reported in this thesis focuses on energy efficiency and latency and their joint trade-offs. This is a vast new field and hence there is significant potential to consider other metrics in the future that include reliability, availability, security and more broadly, resilience. The envisaged new form of VEC edge processors can thus reduce the cost of providing the computational services needed by making use of underutilised resources in vehicles and can enable new services. In order to focus on energy efficiency and latency evaluations of the processing allocation in VEC, some restrictions were assumed in the considered architecture. Therefore, vehicles are considered as homogeneous, in static (and opportunistic) mode. Moreover, all the available vehicles are assumed to be a part of the VEC (i.e. vehicles' owners agreed to participate as part of the VEC). The main contribution of this work can be highlighted in the processing allocation proposed model for the VEC based architecture to minimise the power consumption and latency considering different aspects such as, end-to-end network, vehicles densities, application variety, and source location.

Main work objectives and contribution details are explained next in Sections 1.1 and 1.2.

1.1 Research Objectives

The primary research objectives of the work reported in this thesis are:

1. To design an end-to-end network architecture for distributed VEC supplemented by processing at cloud, fog, and edge layers.
2. To introduce opportunistic vehicular edge clouds at the edge layer and investigate their feasibility using vehicles as distributed processing nodes.
3. To study and evaluate the processing allocation problem in the network considering the minimisation of the processing power consumption and the networking power consumption.
4. To study and evaluate the software matching problem in vehicular edge clouds where each vehicle may not have the full set of software needed to process the tasks requested.
5. To study and evaluate processing allocation problems that consider delay minimisation and also the joint minimisation of power consumption, propagation delay and queueing delay for the considered architecture.

1.2 Thesis Contribution

The main contributions of this thesis are summarised as follows:

1. A Mixed Integer Linear Programming (MILP) model has been constructed for minimising the processing power consumption and the networking power consumption by optimising the allocation of processing tasks in the architecture. The model incorporates central clouds, distributed fog processing nodes, and opportunistic vehicular edge clouds (VECs).
2. The impact of the density of vehicles and workload volume on the processing allocation and power consumption was investigated through the developed MILP model. Three density cases were evaluated which are zero vehicle availability, low and high vehicle densities.
3. The restriction of a task allocation to a single processor and the flexibility of splitting the task into multiple processing location were also investigated through an extension to the developed MILP model.
4. The impact of the processing workload volume was studied through the MILP model. This is done by evaluating multiple approaches with low and high processing workloads and communication traffic that is related to the amount of processing, with the relationship between processing needs and communication needs expressed as predefined ratios which cover a wide range of applications.
5. A MILP model was developed to investigate the problem of software matching in opportunistic VECs and to minimise the total power consumption, considering the popularity of the software packages. Four scenarios were introduced where the software availability in the

vehicles follow multiple known distributions or is optimised jointly with the processing allocation.

6. A MILP model was constructed to jointly minimise the total power consumption, the propagation delay and the queueing delay. The average propagation delay and average queuing delay considerd the distances between the network entities; and the traffic and service rates of each network device respectively.

1.3 Related Publications

The following list includes publications that resulted from the work presented in this thesis.

A. A. Alahmadi, A. Q. Lawey, T. E. H. El-Gorashi and J. M. H. Elmirghani, "Distributed processing in vehicular cloud networks," 2017 8th International Conference on the Network of the Future (NOF), London, 2017, pp. 22-26.

A. A. Alahmadi, M. O. I. Musa, T. E. H. El-Gorashi and J. M. H. Elmirghani, "Energy Efficient Resource Allocation in Vehicular Cloud Based Architecture," 2019 21st International Conference on Transparent Optical Networks (ICTON), Angers, France, 2019, pp. 1-6.

R. Ma, A. A. Alahmadi, T. E. H. El-Gorashi and J. M. H. Elmirghani, "Energy Efficient Software Matching in Vehicular Fog," 2019 21st International Conference on Transparent Optical Networks (ICTON), Angers, France, 2019, pp. 1-4

A. A. Alahmadi, T. E. H. El-Gorashi and J. M. H. Elmirghani, "Energy Efficient and Delay Aware Vehicular Edge Cloud", 2020 22nd International Conference on Transparent Optical Networks (ICTON), Bari, Italy, 2020.

1.4 Thesis Structure

Following this chapter, this thesis is organised as follows:

Chapter 2 Provides an overview of the concept and background of vehicular cloud computing (VCC), explores different network designs, opportunistic scenarios and potential services and applications that can be provided in such a framework. It also overviews the reference architecture of VCC and the entities included in each layer, and highlights the differences between processing in each layer of the described architecture. Finally, the chapter is concluded by listing several challenges facing VCC.

Chapter 3 provides an overview of the importance of energy efficiency in edge computing and how vehicles can be part of energy efficient edge processing. It also summarises some of the optimisation studies in vehicular clouds that focused on energy efficiency and delay.

In Chapter 4, the problem of energy efficient processing allocation in vehicular edge clouds is tackled over the considered end-to-end architecture, including central cloud, distributed edge nodes and vehicular nodes.

In Chapter 5, the problem of energy efficient processing allocation in vehicular edge clouds is extended given a realistic scenario by including the software matching allocation problem.

Chapter 6 considers processing allocation optimisation to minimise power consumption, propagation delay and queuing delay.

Finally, this thesis is concluded in Chapter 7 which summarises the main contributions of this thesis and discusses some future research directions.

Chapter 2
Overview of Opportunistic Vehicular Cloud

2.1 Introduction

Cloud computing has redefined the computation and communication environment by utilising multiple resources such as servers, storage devices, and other network hardware to provide on-demand services for end users. It has changed the way the Internet resources are used as thousands of computers communicate together as a cluster in the real world. This cluster aims to provide Internet users with high reliability and scalability, on-demand pay-as-you-go services at a cheaper cost. The cloud model can be broadly classified into three main layers: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). SaaS is the most common service for end users, where consumers can access on-demand services. PaaS provides the development tools to build a cloud platform to host web applications. IaaS is considered the basic service in any cloud-based network. It gives all the infrastructure details of any cloud architecture starting from physical components to virtual machine, including data partitioning, provisioning, and scheduling. Basically, IaaS refers to the capability (data centres) provided to the end user and the cloud resources in terms of processing and storage, depending on the running software.

Usually, conventional cloud data centres are centralised and can be accessed through the Internet. This centralised cloud structure faces several challenges such as single point failure, reachability, and transmission latency. Moreover, the huge growth of the cloud-based applications and the significant increase

in the services traffic, calls for new architectures and solutions to distribute the processing workload at high data rates to the end user while reducing the computational burden in the centralised data centres, and therefore, achieve better application performance. As a sequence, distributed computing (or decentralised data centres) has become a popular solution that shifts the workload from the centralised cloud to the fog layer and even to the edge layer, closer to end users. This framework provides the same concept of on-demand services to the end user. Thus, it is considered a new extension of cloud computing, where requests are processed in a distributed fashion based on the user location.

Considered architectures that accommodate the concept and features of distributed data centres are referred to as 'Cloudlets' [4], 'Fog' [5] or 'edge computing' [6]. The term Cloudlet is used to describe any decentralised architecture of data centres. Such framework can provide services to end users independently from the centralised cloud. Fog and edge are considered a cloudlet framework, and both offer similar functionality in term of the services provided and also in term of being closer to the end user. However, the fog layer is usually built of fixed nano data centres with better capability compared to edge entities which usually comprise micro servers or opportunistic processors. As both clusters (fog and edge) provide cloud-based services, many researches refer to these two clusters as "cloud". Figure 2.1 shows a simple illustration of the fog and edge layers, compared to the conventional centralised cloud which is considered the core layer and provides reliable support for the fog and edge layers.

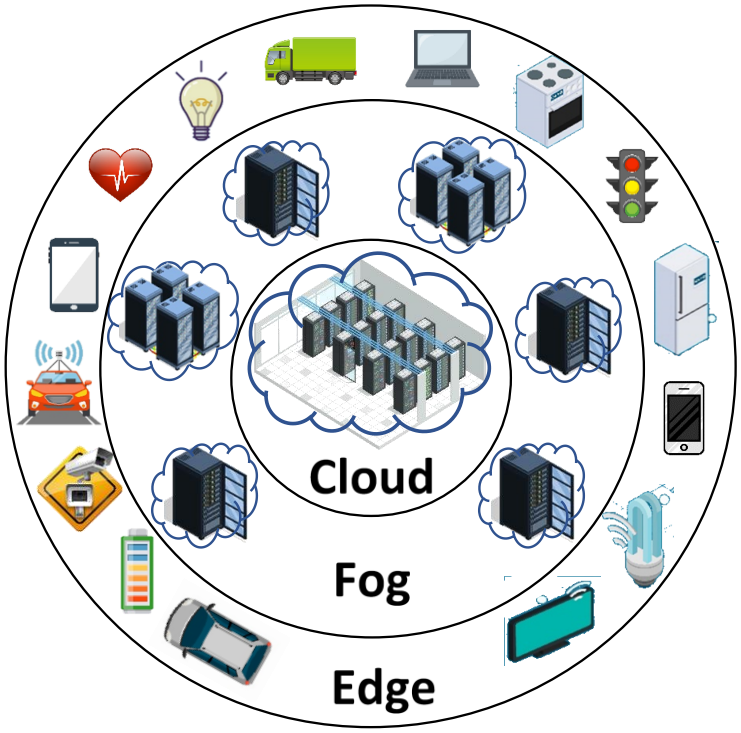

Figure 2.1 Three-layer illustration of cloud, fog, and edge

2.2 Edge Cloud Framework

Edge Clouds were introduced as an alternative solution to the centralised datacentres. It is based on decentralised data processing servers to reduce the distance between the end-user and the provider. Hence edge cloud computing relies on processing entities located at the edge of the network. With the rapid increase in the number of connected devices, resources located at the edge layer become promising opportunistic processing resources candidates. An examples of a platforms built out of underutilised edge entities can be found in mobile-based edge cloud [7]. In this form, mobile phone can be used to store or process other users requests in the absence of enough storage or connection in the original user device [8]. Another form of edge cloud includes distributed IoT devices to work as processing units for mobile devices or other resource-limited IoT nodes [9]. Vehicular-based cloud is a

new form of edge cloud [10]. As many motor companies have announced commercial autonomous vehicles in 2020 [11], it is expected that the capability of the vehicles and their computational resources will increase tremendously in the near future. Vehicles continue to be used typically for two to four hours per day [12]. Accordingly, a significant portion of the processing capability of these vehicles remain unused during the day. This makes the underutilised connected vehicles very promising candidates as edge processing nodes located at the edge layer, and referred to as a vehicular cloud (VC).

This chapter explains the vehicular cloud concept covering architectures, scenarios, services, potential applications, enabling technologies, and the challenges facing such framework.

2.3 The Transformation From Vehicular Ad hoc Network (VANET) to Vehicular Cloud

Vehicular Ad hoc Network (VANET) is one of the Intelligent Transportation Systems (ITS) key pillars created from the concept of establishing a network of vehicles for a specific need or situation [13]. These vehicles can communicate with each other and with infrastructure using wireless connections. It is different than any other ad hoc network because of its hybrid architecture and node movement. The main objective of VANET is to use information such as driving condition, vehicle specification, and street and traffic status to support driving safety applications. The importance of VANET increased recently because vehicles are now equipped with on-board smart embedded devices. The on-board unit (OBU) in vehicles includes computation processors, sensors, GPS-devices, communication devices, camera, and

event data recorders (EDR) [14]. All these components enable VANET communication and can support the ITS sector. As a result, a huge transformation is underway in VANET and in the ITS industry by providing services for cooperative driving, traffic congestion, collision avoidance, alternative route education, and road monitoring.

The huge improvement in the embedded resources in vehicles made these vehicles a very good potential candidate that can act as a building block for cloud data centres. The idea of importing the cloud computing into VANET infrastructure started by Olariu and his colleagues in [15]. From their point of view, although the vehicles' on-board capabilities are improved, these resources are wasted and underutilised. Vehicles are augmented with powerful computation, communication, positioning, storage and sensing resources. The computational capabilities of these vehicles can be combined to serve as a huge farm of mini OBUs. These resources can be a good infrastructure for the implementation of vehicular clouds over VANET. This collaboration between these two concepts resulted in three possible architectural frameworks, known as Vehicles using Cloud (VuC), Vehicular Edge Cloud (VEC), and Hybrid Vehicular Cloud (HVC). Each is described briefly below.

2.2.1 Vehicles using Cloud (VuC)

In this framework, vehicles (as end users) use the conventional cloud storage services to obtain traffic information. Many VANET characteristics should be considered such as connection reliability and vehicle velocity. In VuC, any of the roadside units (RSU), access points (AP), or LTE base stations can serve

as a gateway for vehicles to the centralised cloud. The role of vehicles in this architecture is to retrieve traffic information from the centralised cloud. Applications of VuC include vehicle system configuration, performance checking, traffic big data analysis, smart location-based advertisements, and vehicles witnesses [16], [17].

2.2.2 Vehicular Edge Cloud (VEC)

The focus in this framework is the vehicle itself, where collaborative vehicles act as the main computational resources replacing or complementing the conventional cloud. In VEC, any vehicle with a resource to share, can be part of this cluster after executing some initial steps based on the chosen protocol. The protocol facilitates how these vehicles communicate with the architecture controller and sets the boundaries of the cloud resource provisioning. From here on, these vehicles can join or leave the cloud based on predefined criteria. However, as the VEC is the main architecture framework in this thesis, more details about the VEC architecture, services, and potential applications will be provided further in the remaining part of this chapter. As the term VC is widely used in the literature to imply VEC, both terms will be used interchangeably in this work.

2.2.3 Hybrid Vehicular Cloud (HVC)

HVC is the combination of VEC and VuC where vehicles can be service providers and consumers in the same architecture. The applications used in VuC are usually Peer-to-peer (P2P) applications where the consumer vehicle

can communicate directly with a provider vehicle through vehicle-to-vehicle (V2V) protocol.

2.4 Vehicular Cloud Architecture

VC can be described as a layered architecture, as illustrated in Figure 2.2. This architecture relies on four layers, each described below.

- **Perception layer (Objects Layer):** this layer consists of physical devices located at the edge (i.e. vehicles). The data are collected from objects in this layer and are aggregated to the next upper layer to be transferred and allocated to the available resources. Note that these vehicles can act as resource providers receiving requests from other vehicle or any other connected devices.
- **Network layer:** the collected data from the object layer are sent through this layer to the available resources to be served. This network layer is responsible for the communication between the object layer and the service provider. It is comprised of different communication technologies enabling heterogeneous devices to communicate and transmit data. More details about the necessary standards that can be enabled in VC are explained in Section 2.7.3
- **Service layer**: It is also called the middleware or the cloud layer. This layer is responsible for providing the required services for the aggregated data from the lower layers. It also manages the access to the different available resources and controls the service allocation.

- **Application layer**: This layer provides end users with services and applications that are processed by the lower layer. The enabled services are mainly based on the provided resources (IaaS), including, but not limited, Computing as a Service (CompaaS), Storage as a Service (STaaS), Sensing as a Service (SEaaS), and Networking as a Service (NaaS). These services are described in Section 2.6. Depending on the services provided, VC can offer many applications classified as traffic-based or cloud-based applications (explained in Section 2.7).

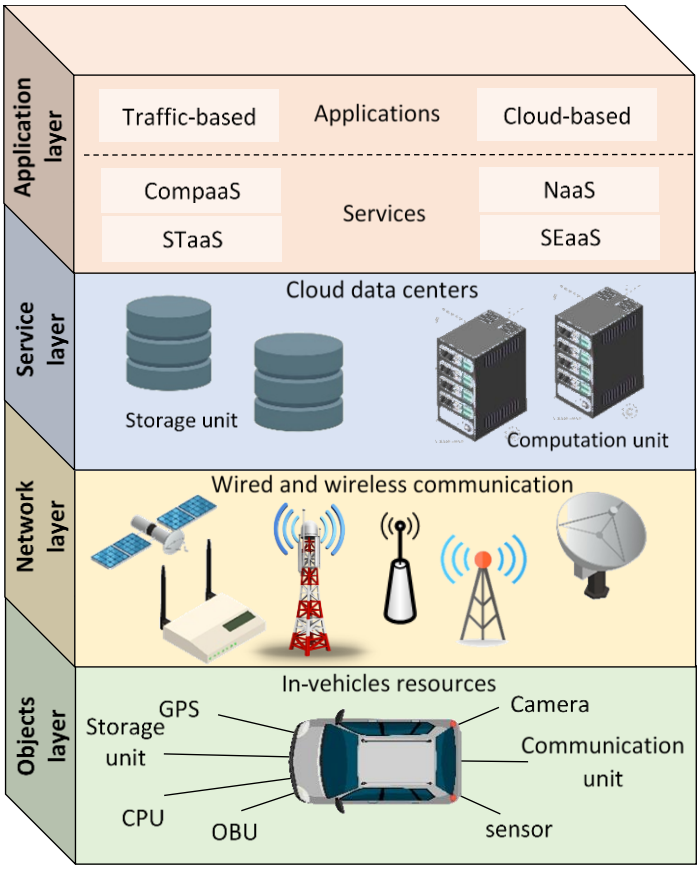

Figure 2.2 Vehicular cloud layered architecture

2.5 Vehicular Cloud Scenarios

The vehicular cloud architecture consists of a group of vehicles. These vehicles are clustered in different scenarios based on their locations and mobility status. Clusters can be made up of parking rows and floors in a car park of an urban area, shopping mall, companies, or airports. Other scenarios include vehicles clustered for a shorter time period compared to parking lots, such as at road intersections, petrol stations, and electrical charging points. The most challenging scenarios can be found in on-the-move vehicles where vehicles are treated as individual nodes or in a platooning form. This form of VC is challenged by vehicular mobility where its stability is dependent on the vehicle velocity and available connections. All VC scenarios consider these vehicles to be connected to the network infrastructure through a controller or by a connection to nearby vehicles.

The resources of vehicles involved can be merged in a fixable and dynamic way to offer services to other users based on the VC scenario. Figure 2.3 shows the different potential scenarios for VC which can be divided into two main categories: static VC and dynamic VC based on its location, availability duration, and its mobility status. These scenarios, and some related works, are explained and outlined next in Sections, 2.4.1 and 2.4.2.

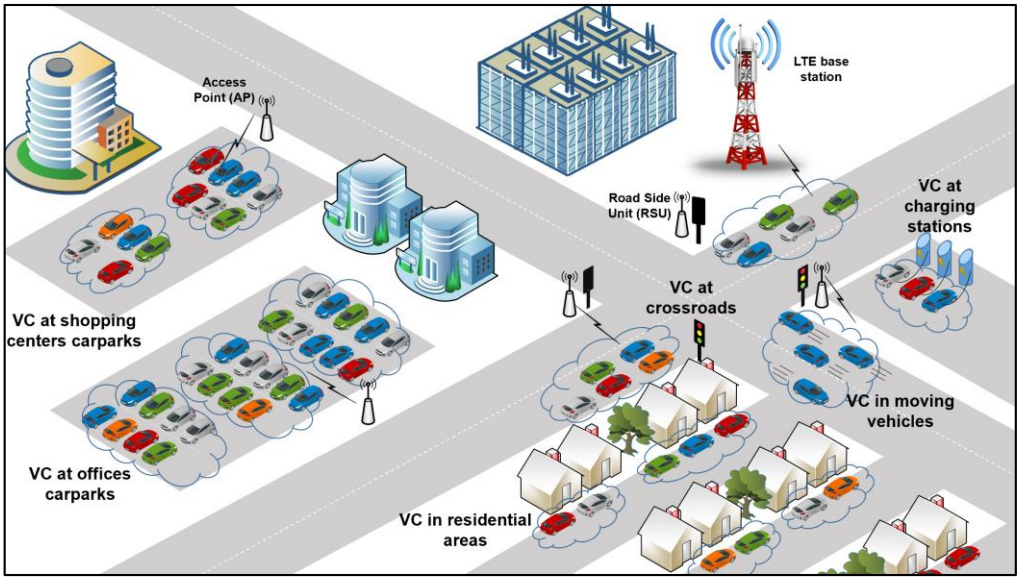

Figure 2.3 Vehicular Cloud (VC) Architecture, with vehicles formed in static or dynamic clusters.

2.4.1 Static Vehicular Cloud (SVC)

SVC is represented by stationary vehicles that are formed and remained in a static status for a period of time. The static scenario started with the idea of parked vehicle clusters [15]. The proposed concept is based on using the underutilised vehicle resources in parking lots as a data centre farm. This scenario gives more sustained services in terms of time and space. For instance, in airports, vehicles may be parked for several days, making the capabilities of such vehicles available to transform such car parks into mini servers' units at the edge of the network on a semi-permanent basis as departing cars are usually replaced. In such a proposal, vehicles form the data centre, connected to a centralised server in the airport responsible for assigning computation and storage tasks to these vehicles [12]. Similarly, short-term parked vehicles in a shopping mall [15], a company [18], or on-street parking spaces [19] can be seen as a small scale VC, with its available

resources and availability duration affected by the peak / off-peak periods of the day. This short-term VC can offer instant services that require less time and can be fulfilled with limited resources. Moreover, the vehicles in this type of parking lot can be good candidates to work as relay nodes to provide Internet connection services for nearby users [20]. As the vehicles density and residency duration is one of the main factors affecting the size of the resources and the service availability, some previous efforts were directed to analysing the density of the vehicles in different size parking lots and investigating its feasibility as a data centre [3]. Other efforts focused on the stability of the connection, to guarantee a reliable service provided [19]. Although the parking lot scenario is the main focus in SVC, other potential forms of vehicle clusters can be considered as opportunistic SVC. This includes vehicles at road intersections, vehicles stopping at petrol stations or electrical charging points (similar to the scenario proposed in this thesis). Vehicles in this scenario are available in a static mode for a short time compared to vehicles in parking lot. Moreover, these vehicles become a part of the edge cloud by stopping in one of the mentioned stops (intersection, petrol station,.. etc). This theme of SVC is different from the parking lot theme in its frequent arrival / departure rate and the varying density of vehicles over time.

2.4.2 Dynamic Vehicular Cloud (DVC)

DVC is formed of mobile vehicles that can be either clustered based on vehicle velocity or treated as individual nodes. Due to the vehicle's dynamic behaviour, DVC is more challenging with regards to resource availability and connection stability than SVC. When vehicles are in mobile status, they need

to rely on multiple continuous road controllers or roadside units (RSUs) to keep them connected and active in the DVC. Therefore, studying the mobility pattern of the DVC can help with the resource management and cloud feasibility [21]. However, this movement is a key factor in many VANET-based applications which offer updated collected information for the road and the traffic. An example for such proposal can be found in Car4ICT [10], where a DVC system is introduced to allow driving vehicles to work as mobile base stations to enhance connection stability and to pass traffic data to road controllers and even to other vehicles. Using these vehicles as sensors, information generators, and relay nodes can play an important role in road safety applications [22], [23].

Recently, with the development of autonomous vehicles and their advanced on-board systems, a new form of DVC was introduced as Internet of Vehicles (IoV) [24], [25]. In IoV, each vehicle is considered as an intelligent object with full decision-making capability. Thus, it can provide the basis for a promising form of individual data centre [26]. Another example of IoV scenario is introduced in [27], [28], where moving smart vehicles are clustered as a platooning cloud to create a very good size moving data centre providing long-term availability and service provision.

The next section will discuss the most important services that can be introduced using vehicular clouds, proposed for both SVC and DVC scenarios.

2.6 Vehicular Cloud Services

The key point of VC is to complement the centralised cloud by utilising resources at the edge layer to provide different services to the end user or any

other edge devices. Accordingly, VC can offer the main three services of centralised cloud: IaaS, PaaS, and SaaS (which were explained earlier in Section 2.1). However, With the collaboration of the vehicular clustered resources, IaaS becomes the main service provided by VC. Thus, based on the contributed resources of the vehicles, different services can be introduced. The most common services are grouped into four main categories as follows:

2.5.1 Computation as a Service (CompaaS)

In this service, the computational resources (OBU processors) of a VC cluster, when efficiently utilised, become powerful enough to be equivalent to a desktop computer, or even a large processing data centre. These computational resources can execute certain application tasks in a distributed fashion instead of using the centralised conventional cloud. There is limited work on computation services in VC due to the challenges with evaluating processing allocation and completion which can affect the VC processing performance. However, some researchers have considered CompaaS in their proposed VC by evaluating the job scheduling for the processing delay [29] or the utilisation of parking lot space [30].

2.5.2 Network as a Service (NaaS)

Vehicles with connection capability can serve as an access point providing access to the Internet to other vehicles or end users. Therefore, connected VC can play an important role in providing a network as a service and in improving connections in urban areas. This service was introduced with VANET technology where the focus was to enable communication between vehicles and the infrastructure access points. Thus, recent studies that

considered NaaS have focused on the evaluation of communication protocols and on setting up parked vehicles as connection relay nodes [19], [20]. This service can also be referred to as Relaying as a Services**,** where vehicles act as access points between end-users and the main service provider.

2.5.3 Storage as a Service (STaaS)

Vehicle OBUs are expected to have a good and persistent storage [12], [15]. Although these vehicles storage resources are increasing in a tremendous way, they are mostly underutilised, especially in parked vehicles. These available resources can be used in many applications that require temporary storage. For example, they can be used in long-term parking lots as a temporary backup data centre [15]. The data centre in the airport, proposed in [12], can be a very promising architecture for STaaS. However, more studies are needed to confirm the feasibility of the storage services and to investigate the security and reliability with providing such service.

2.5.4 Sensing as a Service (SEaaS)

According to [31], a 2013 model vehicle has on average of 70 built-in sensors and the number is expected to be 100 sensors in current models. The vehicles built-in sensors and the mobility behaviours of such vehicles can serve as a set of wireless sensors that collect environment and road safety information for many applications [32]. In addition, the mobility of the vehicle can help widen the geographical sensing coverage and therefore can lead to a better application gain. Like NaaS, sensing as a service was introduced with the VANET-based applications, as the involved vehicles can share the collected sensory data with other vehicles or the infrastructure controller [33].

The next section will highlight some of the potential applications that provide one or more of the described services.

2.7 Vehicular Cloud Application

Many potential VC applications were proposed and designed to deliver some of the previously explained services. Some of these applications were introduced using connected vehicles in VANET architecture, while other applications were introduced under the VC architecture umbrella. Currently, VC applications can be classified into two main categories: traffic-based and cloud-based applications.

2.6.1 Traffic-based Applications

These applications were introduced initially with VANET technology and gained continuous attention in the vehicular cloud context. VANET has received significant attention for sensing and networking based applications due to its communication ability in the vehicular environment. This is reflected also on the vehicular cloud enabled applications by employing vehicle resources to provide more information that can be shared with other vehicles or any other smart objects. Traffic signal management is one of the proposed application that can use vehicles to sense the congestion and send related information to traffic signals to optimise and re-time the signal duration based on the traffic density [34]. This can play an important role in releasing traffic congestion and can inform other vehicles about a congestion occurrence [35]. In addition, vehicles equipped with on-board cameras can offer on-demand life pictures of the road and the nearby traffic for other vehicles, which can be

used in road monitoring and infotainment applications [36]. More promising traffic-based applications are expected to arise with a major role in managing, predicting and processing traffic and road data.

2.6.2 Cloud-based Applications

Many aspects of centralised cloud-based applications have changed perspective since the introduction of fog distributed data centres. Likewise, VC are able now to utilise more vehicle resources beyond the communication and sensing based applications. These applications can be embedded in a VC architecture complemented by the centralised cloud or can work independent of the cloud. One of the promising scenarios in VC is the parked lot clustered vehicles. As introduced in Section 2.4.1, the proposed long-term parking VC in [12], guarantees a huge farm of vehicle resources which makes VC stable enough for processing and storage services. Similar scenarios with different parking locations and residency duration were also introduced in [19], [20]. With the popularity of the smart cities concept and the tremendous increase in the number of connected smart objects, VC can offer additional resources for the mobile and IoT applications. Moreover, it becomes attractive to provide more advanced processing jobs for the surrounding generated cloud-based requests [37] such as object recognition, environmental data analysis, health applications, and interactive and video streaming.

2.8 Vehicular Cloud Technologies and Key Factors

This section highlights some of the technologies that are key in any research in vehicular clouds.

2.7.1 Smart Vehicles

With the rapid evolution of vehicles, their OBU have become rich with a collection of sensors and advanced processors. The basic idea of “smart vehicles” started with the development of vehicles equipped with advanced communication and computation systems to address safety and traffic based applications [14]. With such advanced OBUs, vehicles can support functions beyond the traffic application, serving both real time and cloud-based applications. Moreover, with the new innovative functionalities in electrical and autonomous vehicles, one vehicle can work as standalone processor. This brings more promising potential in terms of implementing VC using such advance vehicular entities.

2.7.2 Vehicles formation and clustering

Detecting and clustering the available participating vehicles is a major part of establishing a vehicular cloud. Providing a stable connection and reliable communication is a key factor in building a stable cluster and therefore a reliable framework for VC applications. Such a process usually starts with identifying a cluster head so vehicles can communicate with such a cluster head in order to join or leave the cloud. This cluster head works as a coordinator and maintains the opportunistic behavior of the vehicles. The cluster head can be a fixed controller (RSU or AP) [38], or any other vehicle [39]. The technique that can be used to choose the head of the cluster to coordinate the vehicles communication has a great impact on the stability of the clustered vehicles [40]. Moreover, engaging the clustering techniques to form groups of nodes can greatly improve the performance of vehicular

networks [39]. Many efforts have contributed to the vehicular cloud cluster formation, surveyed in [41]. Regardless the executed technique, the setup of clustering consists of four main steps [53], as follow:

1- Resource discovery, this is the initial step where a cluster head starts to identify nearby potential vehicles that can become part of the cluster. This includes the communication process and establishing the connection between the head and other vehicles.

2- Cloud formation, when the cluster head confirms the accepted vehicles. it will store their information and manage allocating tasks to the participant vehicles.

3- Cloud maintenance, the cluster head keep maintaining the process of joining and leaving of the vehicles, and reforming the cloud at different times when needed.

4- Resource release, this is when the cluster head indicates the end of the cluster or the exit of a vehicle from this cluster.

2.7.3 Wireless Communication

Communication quality in edge layer entities is considered a crucial factor of enabling vehicular cloud. Three major design features should be considered in any vehicle network. First, vehicles need to integrate components of different ad hoc technologies such as WiFi, WiMax, Bluetooth, 5G, and LTE to achieve a stable and efficient communication [42]. Second, the vehicle network employs different positioning technologies to acquire real-time locations of the vehicles [43]. Lastly, data forwarding, and routing techniques are used in vehicular network [44].

Figure 2.4 shows a communication setup that can support the vehicular based environment. Vehicles communication can begin for example by employing Dedicated Short-Range Communication (DSRC) technology, developed by the IEEE wireless access in vehicular environments (WAVE) working group as IEEE 802.11p standard. Two communications modes play part in DSRC communication. The first mode, vehicle-to-vehicle (V2V), operates in an autonomous way with other vehicles, where there is no need for a communication infrastructure. The other communication mode: vehicle-to-infrastructure (V2I), where the vehicles need a static infrastructure equipped with a powerful device (access point, base station, or roadside unit) to help vehicles communicate with the infrastructure.

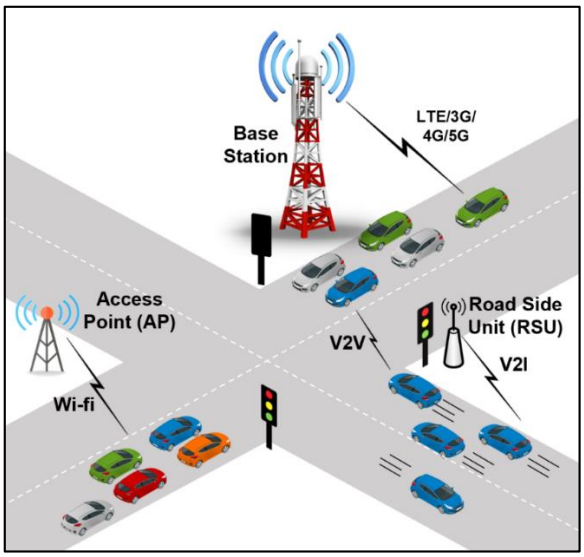

Figure 2.4 Vehicular cloud communication modes

However, as the IEEE 802.11p standard (WAVE) has a limited data rate, the increase in the number of connected vehicles or generated traffic becomes an issue [45]. Thus, it becomes a challenge to operates these vehicles as cloud resource entities with a limited data rate and unstable connection. Conversely,

conventional Wi-Fi (IEEE 802.11b/g/n) connection can provide the vehicle with the required high bandwidth at the expense of more energy. Despite this, conventional Wi-Fi can cope with the increased traffic and can guarantee a high throughput. The low power extended version of the Wi-Fi (IEEE 802.11 ah) and low power wide area network (LPWAN) can be energy efficient replacements to the conventional Wi-Fi, but with lower data rates. This may be detrimental for such protocol, particularly with high demand in multimedia applications at the edge network. Currently, with the rapid improvement in cellular technologies and the resultant high reliability and faster data rate, cellular networks become a very promising medium for such framework. However, cellular networks have a higher power consumption influenced by the LTE and 5G base station [46].

2.7.4 Virtualisation

Virtualisation, in essence, is the abstraction of the physical resources such as processors and storage in order to map different logical partitions (virtual machines) into a single physical resource. IaaS is considered a typical example of virtualisation where a service provider offers virtualised physical resources (e.g. processors) to its customers. Virtualisation allow vehicles to migrate to cloud environment by sharing their OBU resources between many customers including the vehicle owner in an isolated manner [47]. This helps in providing a privacy level in the allocated requests to the vehicle processer and privacy to the vehicle owner data. As one of the main reasons of using vehicles is to utilise their unused resources, by creating multiple virtual machines (VMs) in the vehicle processor, this helps in increasing the

utilisation of the vehicle processors and can reduce the number of physical resources used. However, as a single OBU has a limited resource capability compared to conventional cloud servers, one VM can group multiple OBU resources in one virtual pool to be used as one resource entity [34]. Moreover, due to the dynamics of vehicles and therefore the instability of the VM resources, virtual nodes migration was proposed to migrate the VM from one vehicle to another and therefore avoid processing interruptions of the allocated workload [48], [49].

2.9 Vehicular Cloud Challenges

In this section, we highlight some of the main challenges that need to be addressed in order to successfully implement vehicular clouds.

1- Transport information: Current studies on vehicular clouds assume statistics for the arrival and departure of vehicles in certain road intersections and car parks. Detailed and accurate transport information is needed to provide realistic evaluation of multiple VC scenarios. These transport studies can model the flow of vehicles in cities using real city layouts and calibrated transport models used in vehicular flow studies and in city planning. These studies can be carried out at fine time scales down to milliseconds and for time durations in the order of hours, days and weeks. This can lead to the establishment of accurate statistics of the opportunities for VCs to form in streets, at intersections, at car parks together with the expected durations and capabilities of such VCs.

2. Dynamic nature of vehicles, VC can be formed in static or mobile scenarios as explained in Section 2.5. The opportunistic and dynamic behaviour of the vehicles brings a real challenge when establishing the formation of VC clusters and maintaining the stability of their communication. Models predicting the number of vehicles [50], their arrival and departure [51], and their accurate location [52], can help in estimating the amount of available resources and maintaining reliability of the requests scheduling.

3. Incentive and rental cost: A major requirement when implementing VCs in practice is to identify mechanisms that can be used to encourage vehicle owners to participate and offer their vehicle's resources. These owners need to be rewarded properly for the rented resources. Building a solid incentive model can help ensure the participation and commitment of vehicles owners. Moreover, setting the model to know who will pay and who will be charged for what can be a key factor to achieve success in establishing VCs.

4. Security and privacy: VC services rely on using the in-vehicles resources in a shared mode. This includes accessing the vehicle processors and communication system, which may cause critical issues regarding the security of the processed data, privacy of owner information, and safety of vehicle operation. Security and privacy are very crucial and important in any communication system. Due to the vehicle mobility and the VC heterogeneous network, providing security and privacy through all these communication technologies is a major challenge. Examples of security and privacy challenges in VC include vehicles identity management, owner personal information, and vehicles authentication [53]. Applying virtualisation

techniques in vehicles will divide the physical resources into many small virtual resources (i.e. processors). These virtual processors will run independently of each other, and isolated from the main vehicle processor. Therefore, virtualisation may help deliver the required isolation in sharing resources to guarantee a level of security for both vehicle owner and request provider [54].

4. Reliability and dependability: Reliability is essential when VC is to be implemented in practice. The dynamic nature of vehicular flow implies that VC availability will be variable and will be dictated by factors beyond the network itself. This presents a new challenge as the usual communication setting assumes that the network resources, including computational, sensing and storage resources are deterministic, known and available. The availability of these resources is stochastic in nature in VC posing new challenges. Moreover, providing independent VC service provisioning is a real challenge. Current solutions to guarantee reliability include using fixed fog to ensure a minimum level of service and to provide dependant support for the VC.

7. Standardisation: As discussed previously in Section 2.7.3, VC rely on heterogeneous communication infrastructure. Moreover, clustering a group of vehicles may include heterogeneous processors and technologies. Therefore, standardisation is a key challenge for the uptake of the VC concept in terms of network architecture, routing protocols and algorithms for the establishment, management and disintegration of VC. In addition, standardisation of usage, measurement and monitoring is essential for rewards and reliability.

2.10 Summary

Vehicular cloud is a very promising cloud-based framework. This chapter has provided a detailed overview of VC as it is the main part of the architecture proposed in this thesis. A brief description of the different processing layers has been given to understand how resources differ between these layers. Also, the architecture of VC has been discussed with illustration of the different frameworks and scenarios that can shape the VC. To understand how the idea of VC can be effective in the edge network, a summary of potential services and applications that can be provided has been given. In addition, a brief description of the technologies that need to be embedded in VC are highlighted. Finally, the chapter was concluded by listing several challenges facing VC.

The focus of this thesis is to investigate the processing allocation in vehicular-based edge cloud. Thus, the network in this work is assumed to include vehicles in a static form, in order to highlight the feasibility of vehicles as processing nodes and study the energy efficiency and delay of the proposed processing allocation model. Mobility of vehicles is a major challenge in such framework. Therefore, studying potential processing migration techniques for mobile can be the next step to the current study. As an alternative approach, the opportunism characteristic can be studied by assuming varying densities of available vehicles to capture different periods over the day. Other challenges that can be tackled in the future for the current allocation model, including security, privacy, mobility, and incentive award modelling.

Next chapter will discuss the processing allocation and the advantage of using edge processing to improve network energy efficiency. Moreover, some of the VC optimisation related work will be also reviewed.

Chapter 3
Related Work in Processing Allocation and Energy Efficiency

3.1 Introduction

Resource management is the process of dealing with provisioning, discovering, scheduling, allocating, and monitoring data centres' resources. This area is considered an important part of any cloud framework, and equally important, in vehicular cloud (VC). Resources in vehicles (OBU storage, processors, sensors, communication devices) are clustered and managed in the local environment of the vehicles at the edge layer. Providing resource management at the edge can increase the speed of decision making and can minimise communication costs. Moreover, one of the advantages of processing tasks at the edge (i.e., in VC) is that it makes it possible to deliver reliable and good service for the end user. Thus, it is important for research to focus on how to schedule tasks for potential resources and allocate these tasks to the resources that are available. Many previous studies have focused on resource management and the scheduling of processing in distributed clouds at the edge, which are summarised in [55], [56]. Yet, there are few recent efforts that have tackled this challenge in relation to the VC.

In this chapter, we explore the research efforts that have been developed to study, investigate, and optimise the processing allocation in VC architecture. Moreover, we summarise the studies that have considered energy efficient allocation and delay-aware considerations. In Section 3.2, we highlight the importance of energy efficiency in the ICT sector and edge computing, which motivates us to consider energy efficiency in this thesis. In Section 0, we

summarise some of the optimisation efforts that have been proposed for the vehicular cloud, with a focus on the efforts that consider energy efficiency in VC.

3.2 Energy Efficiency and Edge Computing

The main clouds in the Internet use thousands of servers and hardware from several data centres to process all the application requests that come from users. Cloud applications that are hosted by these data centres consume a large amount of energy for processing and for cooling the hardware [57]. Power consumption has been escalating along with the rapidly increasing use of cloud infrastructure. It is estimated that ICT technologies will be responsible for up to 12% of global emissions by 2030 [2]. Accordingly, issues relating to greening the ICT sector have received more attention in recent years. There is a growing recognition of the need for more research to develop green cloud computing infrastructure in order to save energy and reduce the negative impact of this power expenditure on the environment. Providing energy efficient and reliable network infrastructure has been the main focus of attention in recent years, which has highlighted the need for more research to develop new architecture and solutions that will provide robust and energy efficient infrastructure. Previous research efforts have contributed solutions for reducing the power consumption of cloud data centres and core networks [58], and have considered many research approaches, including virtualisation [59], network design and optimisation [60], content distribution [61], and renewable energy [62].

Recent proposed solutions include building decentralised architectures that integrate distributed edge servers in order to mitigate the traffic burden on central data centres, and thus save more energy [63]. Moreover, as these distributed data centres provide access to the computational resources in distributed servers at the edge of the network, they provide cloud-based services that are in close proximity to the end user. Hence, edge computing offers a good solution for the conventional cloud to offload its processing workload to these distributed servers and therefore, save power. Thus, it is very important to make sure that as these mini data centres are built, it is taken into consideration that they should not significantly increase the power overhead in the ICT sector. Based on this idea, many research efforts have been focused on investigating different architectural and network designs. Moreover, efforts have been made to investigate and solve optimisation problems in resource management and workload allocation in order to achieve energy efficient data centres that are at the edge of the network. Most of the studies already conducted have focused on fixed distributed servers. For example, in the study presented in [64], the authors studied and analysed the energy efficiency of processing applications in nano data centres compared to those in the central cloud. Their study shows that there are many factors that affect the efficiency of the nano data centre, such as the server's location, the equipment of the access network, and the number of user requests. Other studies, however, have focused on utilising available ICT resources, such as nano datacentres, considering the power consumed for such a framework. This research area mainly focuses on utilising distributed resources, such as IoT nodes [65], smartphones [66], and other mobile devices, including

vehicles. Many researchers have also focused on the area of mobile cloud computing to investigate energy consumption. The main concerns regarding this framework involve the limited power resources of the mobile devices and the ways in which they can be used as distributed processing units, given their limited power supply [66]. Other efforts relating to energy consumption in mobile cloud computing (MCC) are summarised in [67]. Fewer studies have focused on investigating energy efficient network proposals, including those using vehicles as distributed servers. In the next section, we summarise some of the optimisation efforts that have been proposed for the vehicular cloud, with a focus on those that investigate the energy efficiency relating to building cloud infrastructure using vehicle-based processing.

3.3 Mixed Integer Linear Programming (MILP) modeling

Optimisation is the concept of analyzing many complex decisions in order to solve a mathematical problem. In other word, it will select inputs that will result in the best possible outputs (i.e. deciding on the most effective allocation of available resources to achieve a design with the least power consumption). To find the answers to most optimisation problems, we need to use a special type of program called an optimisation algorithm (or techniques). Optimisation problems often involve the words maximise or minimise. They also consider limits or constraints on the resources involved or boundaries restricting the possible solutions. In all optimisation problems, two parts are needed to solve a mathematical problem. The first part is the objective function, which is the value that the optimisation program need to optimise (minimise or maximise) to get the optimal value. This objective includes a decision variable (also

referred to as design variable) and it involves one or more value that the optimisation algorithm is allowed to choose or change during the optimisation process. The second part of the optimisation problem is the list of constraints that the program needs to satisfy in order to find the optimum solution for the objective function.

Mixed Integer Linear programming (MILP) is the most common optimisation problems. A Linear programming problem is represented as linear objective function and linear constraints. It is commonly used because of its simple formulation syntax and the easiness to implement and upgrade [68]. It can increase problem complexity with the need of linearisation for some complex non-linear formulas. However, It is faster to run and get an optimal solution using linear solvers compared to using non-linear solvers. For all these reasons, linear programming is chosen in this thesis to implement the processing allocation problem and evaluate the energy efficiency in the considered architecture.

3.3.1 General format of a linear program

As any optimisation problem, A linear program consists of a linear objective function that is minimised or maximised, a set of decision variables, and a set of linear constraints. Each constraint is to be within a certain boundary. An optimum solution (feasible point) is constructed from the objective function satisfying all constraints. The standard form of a linear problem can be written as:

Minimise (or Maximise)

$$F = c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

Subject to

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1$$

$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2$$

$$\ldots$$

$$a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n \leq b_n$$

$$x_1 \geq 0, x_2 \geq 0, \ldots, x_n \geq 0$$

where F is the objective function (to be minimise or maximise, and x represents one or more decision variables. Looking to the objective value is one of the most common ways to tell how well an optimisation has worked. In the case where there are multiple objectives, they are usually summed, multiplied, or otherwise combined to form a single value. Normally, the decision variables are continuous. If some or all of them are constrained to integers, the problem, then, is called mixed integer linear programming (MILP). The more decision variables there are, the more difficult an optimisation problem becomes to solve. Constraints define limitations on the set of feasible solutions. Some of the essential constraints to every network design problem include link and node capacity constraints, and flow conservation constraints.

The MILP optimisation problem uses certain software packages to implement and solve such problems. The most popular software is the AMPL (A Mathematical Programming Language) [69]. The AMPL/CPLEX solver was used to solve the proposed MILP models on a Intel(R) Core i5 CPU at 3.2GHz with 16GB RAM, or in a High Performance Computing cluster with 16 cores and 256 GB RAM.

3.3.2 Example of network modelling problems

For more details about the MILP modelling and notations, this section will discuss two examples of common network design problems that were solved using linear programming and considered in the model proposed in this thesis.

3.3.2.1 Link-Path Formulation

In this example, we consider a simple network of three nodes where each node is connected to the other two nodes, as shown in Figure 3.1(a). A path between any two nodes can be defined as the sequence of links that network demand can travel from the source node to the destination node. Assuming that both demands and links are bi-directional, different paths can carry out demands flow in the given network, as illustrated in Figure 3.1(b).

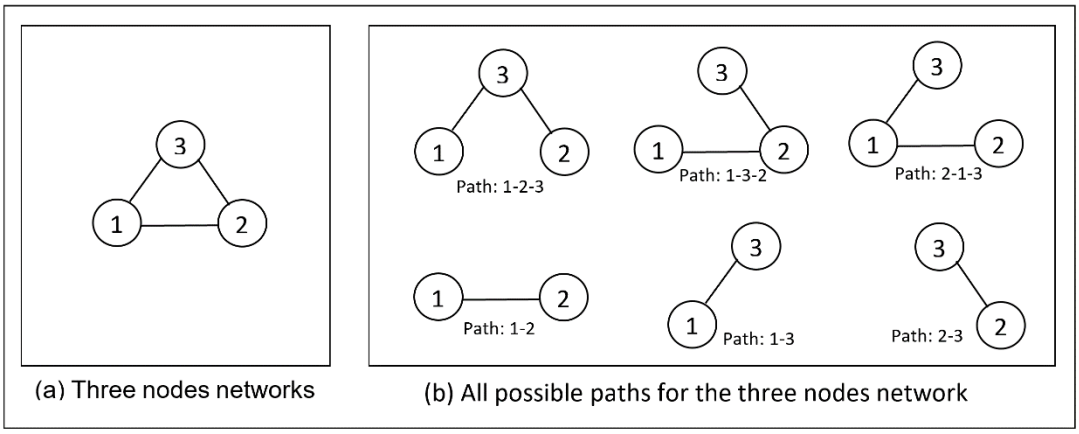

Figure 3.1 Link-path formulation example for three nodes network

Suppose that the demand volume between any two nodes ij is assumed to be bi-directional, and denoted as h_{ij}. Thus, we consider the following three demand volumes:

$$h_{12} = 5, \qquad h_{13} = 7, \qquad h_{23} = 8$$

The demand volume for the demand pair (1,2) can be routed over two paths. The first path is the direct link route 1-2 and the second path 1-3-2 via node 3, as shown in Figure 3.1b. Similarly, the demand volume for the other two pairs (1,3) and (2,3) each can be routed via two different paths. Using x to denote the amount of flow traffic on each path, we can formulate the demand volume as :

$$x_{12} + x_{132} = h_{12} = 5$$
$$x_{13} + x_{123} = h_{13} = 7$$
$$x_{23} + x_{213} = h_{23} = 8$$

Considering the fact that the path-flows are non-negative for all paths (i.e. $x \geq 0$), the demand volume should not exceed the capacity for each link. We will use c_{ij} to denote the capacity of each link connecting any two nodes ij. Accordingly, the capacity formulation can be written as:

$$x_{12} + x_{132} + x_{213} \leq c_{12}$$
$$x_{132} + x_{13} + x_{213} \leq c_{13}$$
$$x_{132} + x_{123} + x_{23} \leq c_{23}$$

Lets assume that the capacity of the first two links is 10 ($c_{12} = c_{13} = 10$) and the third link is 15 ($c_{23} = 15$)

Suppose our objective function in this problem is to minimise the total routing cost, and the cost for each individual link is set to 1. Thus the total routing cost for all flow variables can be written as:

$$F = x_{12} + 2x_{132} + x_{13} + 2x_{123} + x_{23} + 2x_{213}$$

Finally, the minimisation problem of the path-link formulation is:

Minimise

$$F = x_{12} + 2x_{132} + x_{13} + 2x_{123} + x_{23} + 2x_{213}$$

Subject to:

$$\begin{aligned}
x_{12} + x_{132} &= 5 \\
x_{13} + x_{123} &= 7 \\
x_{23} + x_{213} &= 8 \\
x_{12} + x_{132} + x_{213} &\leq 10 \\
x_{132} + x_{13} + x_{213} &\leq 10 \\
x_{132} + x_{123} + x_{23} &\leq 15 \\
x_{12}, x_{132}, x_{13}, x_{123}, x_{23}, x_{213} &\geq 0
\end{aligned}$$

According to the giving input values, the optimal solution and optimal cost resulted from the above optimisation model is:

$$x_{12} = 5, \quad x_{13} = 7, \; x_{23} = 8, \; x_{123} = x_{132} = x_{213} = 0, \quad F = 20$$

3.3.2.2 Node-Link Formulation

In this problem, the described formulation is valid when both demands and links are directed. Considering the demand flow in each link from a node perspective, the traffic flow entering a node must be equal to the traffic flow leaving the same node. In other word, the total outgoing traffic flow minus the incoming traffic, for any intermediate node, must be zero. In case of a source node, the total outgoing flow minus the incoming flow must be equal to the demand volume. For the destination node, the total incoming flow minus the outgoing flow must equal to the demand volume. This is referred to as flow conservation. For the source node, excluding the source and destination nodes). Figure 3.2 illustrates the flow conservation at each node in the three nodes network for a demand sent from node 1 to node 2. In this figure h_{12} represents the demand volume sent from node 1 and destined to node 2. The

traffic flow of the demand (1,2) directed through the link ij is represented as $x_{ij,12}$. For the given three nodes network, Figure 3.2 (a) shows that the originating demand at source node 1 h_{12} has two outgoing arcs: the first arc directed to node 2 ($x_{12,12}$), and the second arc directed to node 3 ($x_{13,12}$). The intermediate node 3, in Figure 3.2 (b), has one incoming directed arc from node 1 ($x_{13,12}$) and one outgoing directed arc to node 2 ($x_{32,12}$). Finally, the destination node 2 in Figure 3.2 (c), has two possible incoming arc from node 1 and 3 represented as ($x_{12,12}$) and ($x_{32,12}$), respectively.

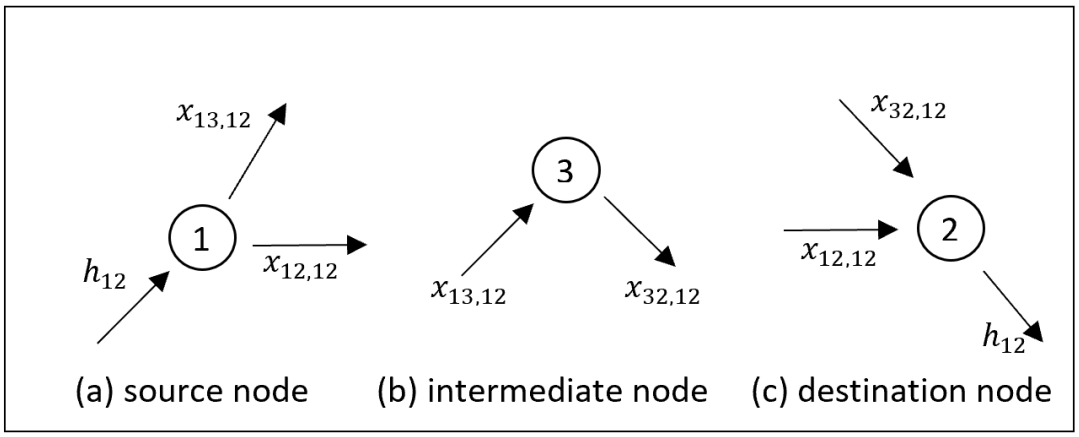

Figure 3.2 Flow conservation illustration for the three nodes network

Applying the flow conservation definition described earlier, we can write the node-link formulation for each node as the following:

For node 1 (source node): $-h_{12} - x_{21,12} - x_{31,12} + x_{12,12} + x_{13,12} = 0.$

For node 3 (intermediate node): $-x_{13,12} - x_{23,12} + x_{31,12} + x_{32,12} = 0.$

For node 2 (destination node): $-x_{12,12} - x_{32,12} + h_{12} + x_{21,12} + x_{23,12} = 0.$

Taking into consideration the assumption of directed link, the incoming traffic at source node 1 and outgoing traffic from destination node 2 are equal to 0.

$$x_{21,12} = 0, \qquad x_{31,12} = 0, \qquad x_{23,12} = 0$$

Using the above formulations, we can write the flow conservation equations for the demand (1,2) as:

$$\begin{aligned} x_{12,12} + x_{13,12} &= h_{12} \\ -x_{13,12} + x_{32,12} &= 0 \\ -x_{12,12} - x_{32,12} &= -h_{12} \end{aligned}$$

Considering available demands from node 1 to node 3 (1,3), and from node 2 to node 3 (2,3), we can also write the flow conservation for both demands as:

$$\begin{aligned} x_{12,13} + x_{13,13} &= h_{13} \\ -x_{12,13} + x_{23,13} &= 0 \\ -x_{13,13} - x_{23,13} &= -h_{13} \end{aligned}$$

$$\begin{aligned} x_{21,23} + x_{23,23} &= h_{23} \\ -x_{21,23} + x_{13,23} &= 0 \\ -x_{13,23} - x_{23,23} &= -h_{23} \end{aligned}$$

Considering the capacity of each link, the total traffic flow between any two node traversing a link should not exceeds the capacity of the link. This constraints can be written for all three links as:

$$\begin{aligned} x_{21,12} + x_{12,13} &\leq c_{12} \\ x_{13,12} + x_{13,13} + x_{13,23} &\leq c_{13} \\ x_{13,12} + x_{13,13} + x_{13,23} &\leq c_{32} \end{aligned}$$

Putting together all the above explained equations, we can formulate the node-link problem as an optimisation problem to minimise the cost, as the following:

Minimise

$$F = x_{12,12} + x_{13,12} + x_{32,12} + x_{12,13} + x_{13,13} + x_{23,13} + x_{21,23} + x_{13,23} + x_{23,23}$$

Subject to

$$
\begin{aligned}
x_{12,12} + x_{13,12} &= h_{12} \\
-x_{13,12} + x_{32,12} &= 0 \\
-x_{12,12} - x_{32,12} &= -h_{12} \\
x_{12,13} + x_{13,13} &= h_{13} \\
-x_{12,13} + x_{23,13} &= 0 \\
-x_{13,13} - x_{23,13} &= -h_{13} \\
x_{21,23} + x_{23,23} &= h_{23} \\
-x_{21,23} + x_{13,23} &= 0 \\
-x_{13,23} - x_{23,23} &= -h_{23} \\
x_{12,12} + x_{12,13} &\leq c_{12} \\
x_{21,23} &\leq c_{21} \\
x_{13,12} + x_{13,13} + x_{13,23} &\leq c_{13} \\
x_{23,13} + x_{23,23} &\leq c_{23} \\
x_{32,12} &\leq c_{32}
\end{aligned}
$$

Recall that all $x \geq 0$

3.4 Optimisation Modelling in Vehicular Clouds

In this section, we will summarise previous research on optimisation work relating to the vehicular cloud. Moreover, we will highlight the efforts that have been made to minimise energy consumption, considering aspects such as service delay, vehicle density, workload volume, and application variety. These research studies are summarised in Table 3.1 based on the processing layers, the objective, and other considerations that are related to the work proposed in this thesis.

Table 3.1 Summary of the surveyed literature of VC optimisation modelling considering the aspects investigated in this thesis.

Study	Processing location			Modelling consideration				
	Central cloud	Distributed fog	Vehicular cloud	Energy minimisation	Delay minimisation	Splittable allocation	Software allocation	Vehicle density
[70]	√	-	√	- *	-	-	√	-
[71]	-	-	√	- *	-	-	√	-
[72]	-	-	√	- *	-	-	-	-
[73]	√	√	√	√	√	-	-	-
[74]	-	√	√	√	√	-	-	-
[75]	√	√	√	√	√	-	-	-
[76]	-	√	√	-	√	-	-	√
[77]	-	-	√	-	√	√	-	√
[78]	-	-	√	-	√	√	-	-

* Generic cost minimisation

3.3.3 Cost-based optimisation

In any cloud system, the costs include a measurement of the amount of resources used to provide a service for a user. In the case of distributed processing that uses edge servers (i.e. vehicles) that are owned by other users, the resource provisioning comes with an operational cost of the network infrastructure and a server's owner rental (reward) cost.

Thus, most of the optimisation efforts in the vehicular cloud focus on providing a distributed system with efficient, cost-accurate models by either minimising the operational cost or maximising the reward cost for the servers' owners. As reward cost is beyond this thesis scope, we will summarise some of the optimisation models that involve a proposed cost-minimisation model. In addition, as energy is considered one of the cost measurements, we will discuss the work focused in minimising network in VC. Moreover, we will highlight the few studies that investigate energy as an optimisation measure

in VC. Then, we will give a brief summary of the work that focuses on maximising user rental costs.

The work in [70] proposed a generic optimisation model for storage allocation in parked vehicles to minimise the total communication cost. The allocation problem considered forwarding video-on-demand requests to vehicles that had the required video available. The minimised communication costs included request service costs and management costs. The video request was allocated to a vehicle based on the probability of that vehicle having the requested video cache available. The authors' proposed optimisation model was mathematically analysed and developed through multiple management policies based on replicating (or not replicating) the video-cached copies among the available vehicles. Their results conclude that the communication costs are reduced by increasing the number of available vehicles, thereby reducing the number of downloaded cache replicas. This work only considered the costs of downloading the video cache into the vehicle. Moreover, no further details were given about the optimisation model, the developed algorithm, or the system's infrastructure.

In a similar work, the authors of [71] optimise job allocation based on the requested service type. The developed model minimises the job completion cost, which is a function of the communication and computing costs of the allocated task. The total cost was formulated using mixed integer non-linear programming (MINLP). The proposed allocation relies on choosing the optimum vehicle to offload the processing workload from the road-side unit (RSU), considering pre-defined predicted vehicle mobility algorithms. The proposed optimisation model was proven to have better total cost compared

to a conventional case, where each processing job was allocated to all available vehicles surrounding the RSU. This work found that in-vehicle processing reduces the burden of the RSU. However, the proposed system cannot guarantee reliable service delivery by considering processing only from opportunistic vehicles or the limited-resourced RSU.

The authors of [72] also proposed a generic cost minimisation based on a time-scheduling optimisation model in order to satisfy task completion. They executed the model using binary integer programming (BIP) and included both processing and networking costs, taking into account two available communication mediums (WAVE and 3G / 4G). They consider tasks that are executed in independent and parallel rounds in order to capture different time slots and to study dynamic resource variability (i.e. vehicles' resources). The model works under the assumption that each task is processed only in one location, which restricts the type of application that can be accommodated in such a system. The findings of the proposed work show that vehicles with a WAVE connection can offer low-cost processing. However, connection stability and limited capacity can be an issue in terms of service reliability.

Few studies have investigated energy efficiency in VC-based optimisation models. However, in [73], the authors tackled the problem of offloading from smartphones to another processing node in order to reduce the computational overhead of the limited smartphone processors. They developed a flexible offloading algorithm to optimise the task-computing placement between the smartphone (locally), central cloud, cloudlet server, or VC. In each allocation placement, they assessed the energy consumed and response time with multiple capacities of processing locations and multiple input sizes. Their

proposed algorithm gives priority to processing tasks in the central cloud and in cloudlet locations. It offloads the task to VC only in cases where both other locations do not satisfy the capacity or time demands of the task. Their findings show that the allocation of tasks to VC creates the lowest delay in most input sets (except for those with very high-demand traffic) and achieves the lowest response time compared to the other processing locations.

The authors in [74] presented another attempt to minimise energy consumption by optimising the task-offloading decision in the VC framework. Their proposed framework consists of distributed servers in RSU and collocated vehicles, with each RSU also working as a processing resource. The vehicles are assumed to be a user that generates requests to be processed locally or offloaded to an RSU or another vehicle if the request exceeds its limited processor capacity. The model optimises this decision by minimising the total energy consumed, which includes the local processing energy, the offload transmission energy, and the processing energy in either the RSU or the vehicle's processor. In addition, the model considers the processing time constraints of some of the generated tasks. An extensive mathematical analysis was given of the offloading problem, and energy consumption was analysed based on the portion of the offloaded workload and the assumed transmission power of the RSU and vehicles' access point.

Using a different approach, some researchers developed a joint optimisation model that has more than one objective. For example, the authors in [75] proposed a fitness estimation model to allocate tasks to a network of three layers, which include the central cloud, cloudlet nodes, and VC. This model was designed using three objectives to minimise network delay, minimise

power consumption, and maximise the availability of the processing resources (as virtual machines). The total network delay is calculated from the propagation, queuing, transmission, and processing delay. The queuing delay considered is based on the waiting time of the processing tasks in each processor and is calculated from the total execution time. This is the same for the power consumption, which is calculated only based on the power needed to execute each allocated task. The study mainly focused on the proposed fitness estimation model, which studied the capability of the available resources and their availability, a problem that is considered a challenge due to the vehicles' mobility.

3.3.4 Delay-based optimisation

In this section, we explore some optimisation efforts that have been carried out in order to minimise service delay. These efforts exist in the context of time or delay for workload processing, response, transmission, propagation, and queuing.

The work in [76] proposed a joint optimisation model for task allocation, to minimise service delay and quality loss. The optimisation model considered the presence and absence of a fog node and its capacity within the studied area. They minimised the maximum service delay which is composed of the time needed for transmission and processing, considering the maximum tolerable delay for video streaming applications. The authors investigated the trade-off between the service delay and the computing quality loss ratio and achieved a balanced optimisation between the two objectives. Due to the model complexity in testing real time traffic, they based their evaluation on an

optimisation-based heuristic algorithm considering two periods with different vehicle densities. Their results found that the developed algorithm reduced the service delay by up to 27% and enhanced the quality loss ratio by 56%.

The study in [77] proposed a reliable task scheduling model in the vehicular cloud in order to minimise the execution time and satisfy job deadline using mixed integer linear programming (MILP). They considered the MapReduce execution model to distribute big data processing jobs among the available vehicles without considering any fixed node. This allocation was made by the VC cluster head, which is a selective vehicle located around the participant vehicles. As these vehicles are in a dynamic mode, the quality of the provided service depends on how reliable the connection is with the cluster head, and how much time is needed to process and send the output back to this cluster head. These two factors were investigated by minimising the response delay and maximising the success probability. The optimised model was also developed using a simulation-based algorithm to test the solution, one that considered vehicle density and an increase in the level of collision in order to study the system's reliability. It was found that the increased number of vehicles reduced the job execution time and increased the percentage of successful job execution. In conclusion, the proposed scheduling solution achieved a faster time and satisfied the required level of reliability.

In another study in [78], the authors proposed an efficient task-scheduling model in the vehicular cloud to minimise the average completion time. In the proposed VC scenario, a fixed fog node offloaded a job (divided into many tasks) to the available surrounded VC cluster. The tasks are offloaded simultaneously and processed in parallel, where each vehicle executes a task,

sends the generated output to the vehicle executing the next task, and so on. The minimised delay in the proposed model consists of processing time, communication time and queuing time. The latter time is calculated based on the generation time of the first task, the processing time, and the completion time of the last task. The evaluated results are based on a comparison between the optimisation-based developed algorithm and the previous algorithm from their literature (greedy algorithm). It is found that their developed algorithm can reduce the average response time by 12% compared to the greedy algorithm.

3.5 Summary

This chapter has provided a brief overview of resource management and processing allocation in distributed data centres. Moreover, it highlighted the importance of energy efficiency in the ICT sector and how efforts have evolved in edge computing architecture generally, and in the vehicular cloud, specifically. The chapter also presented a summary of some of the previous work that has been proposed in the area of allocation optimisation models in the vehicular cloud. Two aspects of optimisation problems were highlighted that offer a minimisation of power consumption or service delay. With this background, Chapter 4 considers our vehicular edge cloud architecture and the optimum allocation of processing tasks to minimise power consumption.

Chapter 4
Energy Efficient Processing Allocation in Opportunistic Vehicular Edge Clouds

4.1 Introduction

The huge growth in increasingly connected edge devices and resource hungry applications calls for more proposals in distributed processing systems to offload the computational burden in the centralized data centres, and to improve the performance of applications. Distributed cloud platforms can be composed from any available user-owned resources that allow processing, storage, networking and sensing. Following this concept, vehicular clouds can be formed if the vehicle on-board processing, storage, and sensing devices are clustered together to form short-term cloud units composed of many vehicles (each vehicle effectively acting as a server in a mobile micro data centre) at the edge of the network [3]. Furthermore, considering these edge entities as a Computing as a Service (CompaaS) providers can help in turning these vehicles from service consumer to cloud-based provider for many applications that are generated from the surrounding connected entities.

In this chapter, we consider a Vehicular Edge Cloud (VEC) based architecture, where a group of vehicles in a car park, at a charging station or at a traffic signals intersection, cluster and form a temporal vehicular cloud by combining their computational resources in the cluster, as illustrated in Figure 4.1. The aim of the VEC is to exploit the underutilised computational resources in vehicles' on-board units (OBUs) and to increase the distributed processing resources to serve the demands required by a smart city environment. This architecture is integrated into a multi-layer fog mini data centre and supported

by a central cloud data centre. All the considered cloud-fog-edge resources, referred to as processing nodes (PNs), act as a service provider for smart cities' demands. These processing demands are assumed to be generated from IoT sensor nodes distributed across the street and in the city near car parks, charging stations or road intersections. The main contribution in this work is to find the optimum placement to allocate and process the generated tasks in order to minimise the total power consumption of the end-to-end architecture, taking into account realistic network devices parameters. This work studies the joint energy efficiency of network and processing along three dimensions. The first dimension compares the centralised processing (in the central cloud) to distributed processing (in the multi-layer fog nodes). The second dimension introduces opportunistic processing in the vehicular nodes. The third dimension considers non-splittable tasks (single allocation) versus splittable tasks (distributed allocation), representing real-time versus non real-time applications.

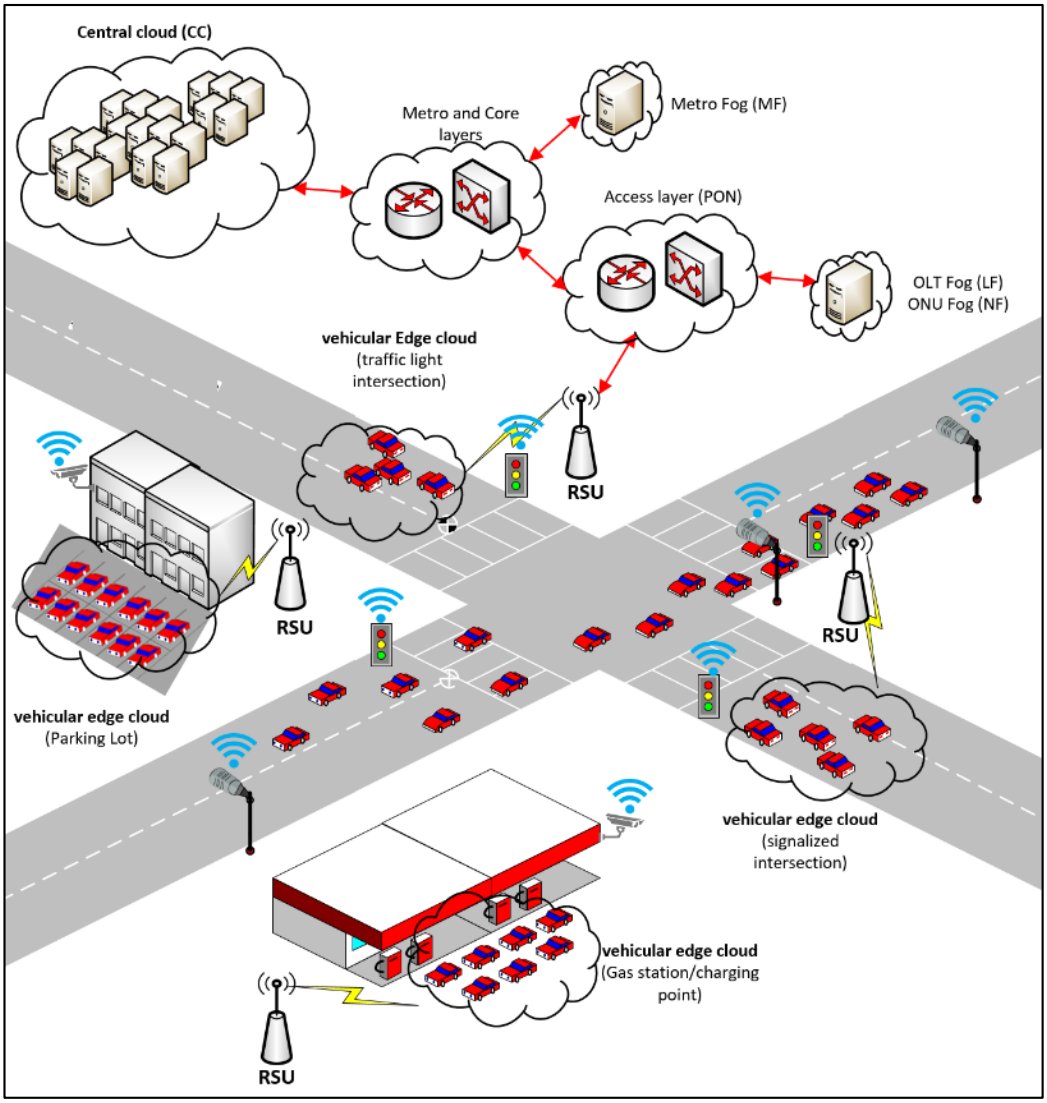

Figure 4.1 Cloud-Fog-Vehicular Edge Cloud Architecture

4.2 Energy Efficient Cloud-Fog-Vehicular edge cloud Architecture

The considered integrated cloud-fog-VEC end-to-end architecture is shown in Figure 4.2. It is composed of four distinct layers with four processing locations at core, metro, access, and edge layers.

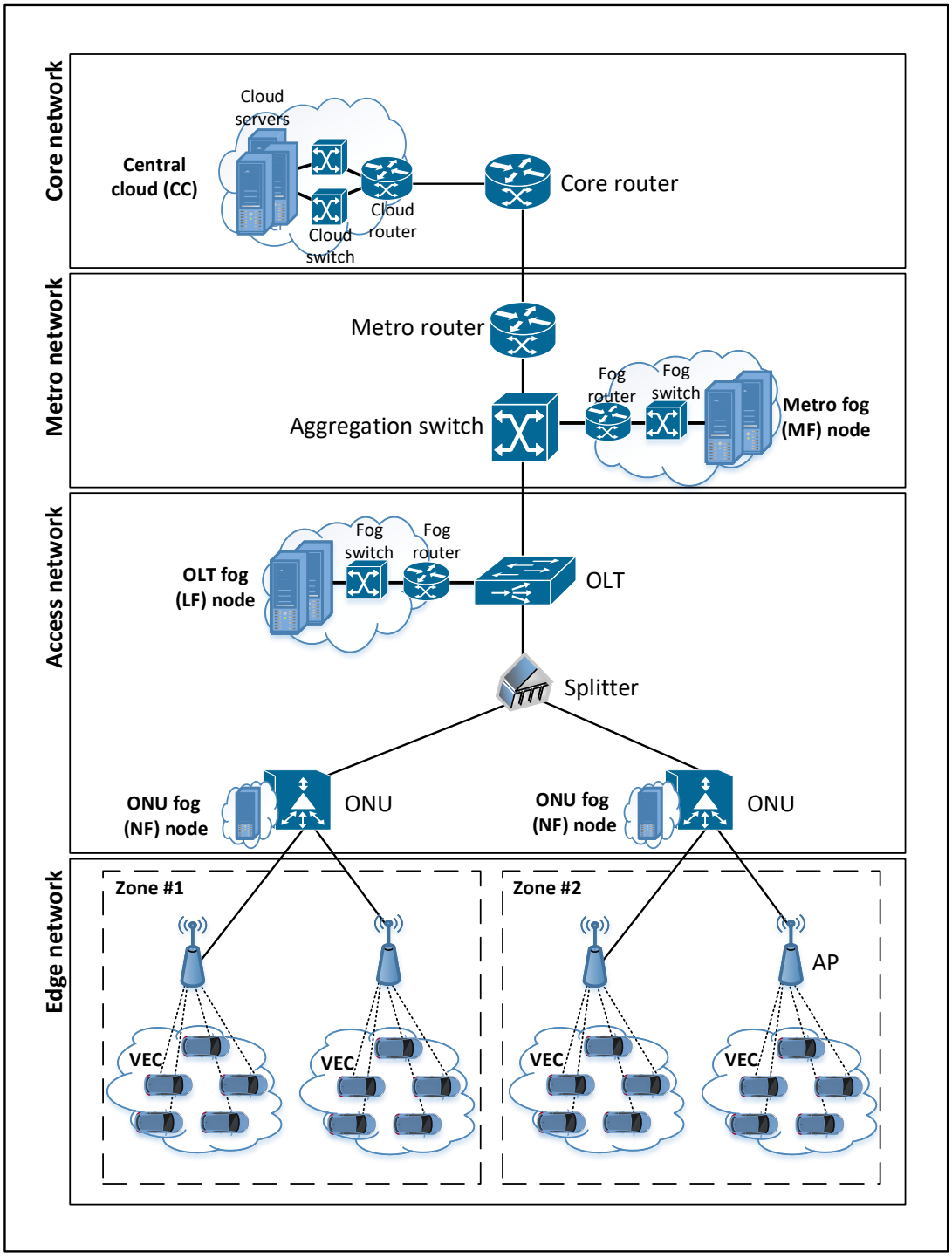

Figure 4.2 End-to-End Cloud-Fog-VEC Architecture

Edge network

This network forms the bottom-most layer and represents a snapshot of a smart city layout. Two types of edge entities are defined in this layer, namely

source nodes (SNs) and vehicular nodes (VNs). SNs are distributed IoT nodes (for example wireless sensors) which are usually responsible for collecting multimedia and environmental data to generate information used by security and environmental monitoring applications. The edge network also includes one or more temporal VNs clustered in car parks, charging stations or road intersections forming a VEC. These vehicles are equipped with OBUs and can work as a processing node to process and analyse the collected data. Both IoT SNs and VNs are connected to the wired infrastructure through an Access Point (AP). The AP acts as a controller of the network. It starts by collecting the generated tasks from nearby IoT nodes. Then, AP executes the optimisation model to choose the optimum location to process the task. Finally, AP executes the actual allocation by forwarding the task to the optimum location of the available PNs in cloud, fog, and edge layers. The communication medium between the AP and edge nodes (IoTs and VNs) is selected according to the controller unit (AP) and the communication protocol in the edge nodes (i.e. wireless connection). This AP is assumed to have full knowledge of the available resources. It is also assumed that the AP is attached to a computational unit with enough capability to execute the processing allocation algorithm and fulfil its coordination and allocation roles.

As seen in Figure 4.2, the design of the edge layer is based on multiple zones, where each zone represents one geographical area. Each zone may also include one or more VEC. Every VEC is represented by VNs clustered in a car park, at a charging station or at an intersection and an AP. VNs within the same VEC can communicate only with one local AP. As the AP has the role of collecting and allocating tasks, it can communicate with other VEC clusters

through the access network via an Optical Network Unit (ONU). Moreover, tasks generated from one zone can also be allocated to other zones, through a Passive Optical Network (PON) and via an Optical Line Terminal (OLT). The PON design, including access layer entities, will be explained next.

Access network

This layer consists of a PON with several ONU devices, each connected to the AP devices distributed in the same zone. These ONUs are connected to an OLT via a fibre link using a passive optical splitter. Fixed fog processing nodes can be deployed at both ONUs and the OLT, named as ONU fog (NF) and OLT fog (LF), respectively. Processing nodes located at the ONU are small and limited in their processing capability, but provide a closer processing opportunity to the edge source nodes. The former nodes also provide more reliable processing nodes that serve processing demands which cannot be satisfied by the VEC. On the other hand, a processing node located at the OLT has more processing capability compared to the NF and VEC, and is considered as a supportive processing layer for the generated demands to guarantee a reliable service provision.

Metro network

The metro layer is an intermediate network between the access layer and the core layer. It consists of a switch, which aggregates collected data from the edge and access layers, and a metro router, while simultaneously serving as a gateway between the access layer and core layer. Another fixed fog is included in this layer and connected to the aggregation switch. This metro fog

(MF) node is equipped with a server that has higher computational capabilities compared to the previously-explained processing nodes.

Core network

This layer includes core routers integrated into it is the central cloud which has its own cloud routers, cloud switches, and cloud servers. The central cloud supports the architecture with sustainable high-processing servers to execute tasks that cannot be executed in the lower processing nodes.

It is worth mentioning that all four layers are scalable. For model complexity runtime, we assumed one PN in each cloud and fog layer, except NF (in some scenarios) and VEC (as we are assessing the opportunistic behaviour of the VNs through VNs density).

4.3 Power Profile of Processing and Networking Equipment for the Considered Architecture

The main objective of the proposed model is to minimise the total power consumption of all devices involved in processing the demands or networking the associated traffic. Therefore, it is necessary to describe the power profile of these devices and how it is related to the generated processing/traffic demands. The power profile considered in this work is based on a linear power profile calculated for both servers and network devices based in equation (4.1) [59], [79]. Hence, the power consumption of all network equipment, including processing nodes, consists of a linear proportional part (dynamic power) and an idle part (static power), as shown in Figure 4.3. It is very important to consider the idle power consumption (P_{idle}), as it can represent a large

percentage (up to 95%) of the maximum power consumption (P_{max}) [80], [81]. Hence, the total power consumption calculated in the model considers these values to calculate the power per processor MIPS and the power per bit/sec, using:

$$P_L = P_{idle} + L\left(\frac{P_{max} - P_{idle}}{C_{max}}\right) \quad (4.1)$$

where C_{max} is the maximum workload the device can handle, L is the workload allocated to the processing or networking device, P_{idle} is the idle power of the device when, $L = 0$, and P_{max} is the power consumption when the workload $L = C_{max}$.

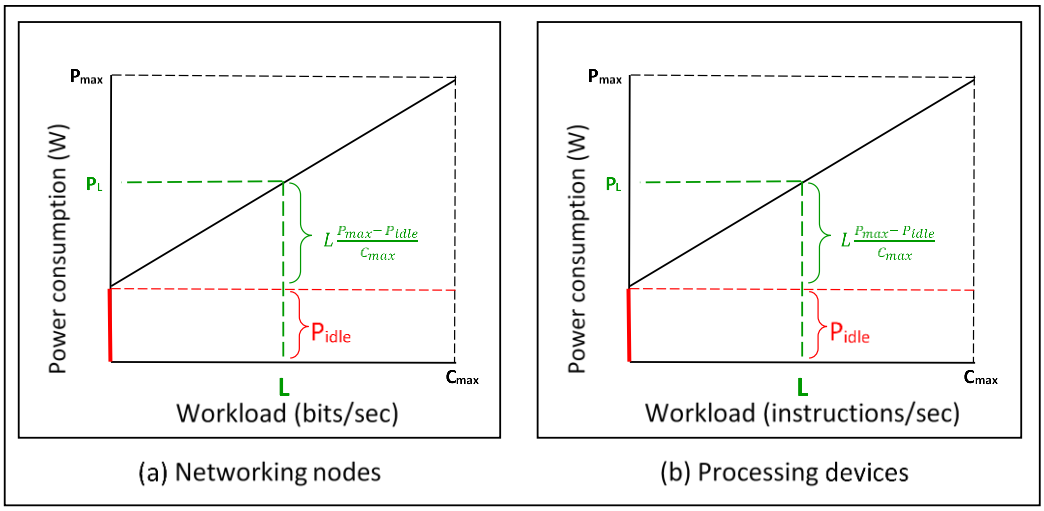

Figure 4.3 Power consumption profile for (a) Networking nodes, and (b) Processing devices

4.4 MILP Model for Energy Efficient Processing Allocation in Cloud-Fog-Vehicular edge cloud.

This section introduces the Mixed Integer Linear Programming (MILP) model that has been developed to minimise the total power consumption by optimising the processing allocation of different demands into the available

processing locations in the integrated cloud-fog-VEC architecture. We will start with declaring the sets, parameters and variables that will be used in the MILP model equations. To simplify the reading of these notation, below are some clarification points regarding the naming conventions:

- Sets are in capital letters.
- Variable of any power calculation (The right hand side of equation) is in three capital letters with a capital superscript.
- Uppercase superscripts, in the parameters, are a part of the name and not an index.
- Lowercase superscripts and subscripts are parts of the parameters and variables values (indices)

The sets, parameters, and variables are declared as follows:

Sets:

N	Set of all nodes.
NN_i	Set of neighbours of node i, $\forall\ i \in N$.
PN	Set of processing nodes, where $PN \subset N$.
SN	Set of source nodes, where $SN \subset N$.
RR	Set of core router ports, where $RR \subset N$.
MR	Set of metro router ports, where $MR \subset N$.
MS	Set of metro switches, where $MS \subset N$.
O	Set of OLT nodes, where $O \subset N$.
U	Set of ONU nodes, where $U \subset N$.
A	Set of AP nodes, where $A \subset N$.

CC	Set of central cloud servers, where $CC \subset PN$.
CR	Set of CC router ports, where $CR \subset N$.
CS	Set of CC switches, where $CS \subset N$.
MF	Set of Metro fog servers, where $MF \subset PN$.
MFR	Set of MF router ports, where $MFR \subset N$.
MFS	Set of MF switches, where $MFS \subset N$.
LF	Set of OLT fog servers, where $LF \subset PN$.
LFR	Set of LF router ports, where $LFR \subset N$.
LFS	Set of LF switches, where $LFS \subset N$.
NF	Set of ONU fog processors, where $NF \subset PN$.
VN	Set of vehicular nodes processors, where $VN \subset PN$.

Parameters:

ω_s	Processing requirement of the task generated from source node $s \in SN$, in Million Instructions per Seconds (MIPS).
$\mathcal{F}_s$	Traffic flow demand generated from source node $s \in SN$ (in Mb/s).
F_s	Traffic flow to processing demand ratio for source node $s \in SN$, where $F_s = \frac{\mathcal{F}_s}{\omega_s}$.
C_d	Maximum capacity of processing node $d \in PN$ (in MIPS).
L_{ij}	Maximum capacity of the link between nodes (i, j) (in Mb/s), where $i \in N$, $j \in NN_i$.
P_{max}^n	Maximum power consumption (W) of each node $n \in N$.
P_{idle}^n	Idle power consumption (W) of each node $n \in N$.
Ω_{max}^n	Maximum capacity (Mb/s or MIPS) of each node $n \in N$.
ρ^{CC}	Central cloud PUE.
ρ^{MF}	Metro fog node PUE.

ρ^{LF} OLT fog node PUE.

ρ^{NET} Network devices PUE.

τ Portion of the network devices idle power attributed to the application.

υ Number of splits a task can be divided into.

Variables:

X_{sd} Processing workload, in MIPS, generated from source node $s \in SN$ and allocated to processing node $d \in PN$.

δ_{sd} Binary variable, $\delta_{sd} = 1$ if workload generated from source node $s \in SN$, is allocated to processing node $d \in PN$, 0 otherwise.

δ_d Binary variable, $\delta_d = 1$ if any workload is allocated to processing node $d \in PN$, 0 otherwise.

λ_{sd} Traffic flow sent from source node $s \in SN$ to processing node $d \in PN$.

λ_j Total traffic in node $j \in N$

λ_{ij}^{sd} Traffic flow sent from source node $s \in SN$ to processing node $d \in PN$ through physical link nodes (i,j) (in Mb/s), where $s \in SN$, $d \in PN$, $i \in N$, $j \in NN_i$.

Ψ_i Binary variable, $\beta_i = 1$ if any traffic traverses network node $i \in N$, 0 otherwise.

$M1$ Large enough number with unit of MIPS.

$M2$ Large enough number with unit of MIPS.

$M3$ Large enough number with unit of Mb/s.

TPC^{CC} Total power consumption of CC.

PPC^{CC} Processing power consumption of CC.

NPC^{CR} Networking power consumption of CR.

NPC^{CS} Networking power consumption of CS.

TPC^{MF} Total power consumption of MF.

PPC^{MF} Processing power consumption of MF.

NPC^{MFR} Networking power consumption of MFR.

NPC^{MFS} Networking power consumption of MFS.

TPC^{LF} Total power consumption of LF.

PPC^{LF} Processing power consumption of LF.

NPC^{LFR} Networking power consumption of LFR.

NPC^{LFS} Networking power consumption of LFS.

TPC^{NF} Total power consumption of NF.

TPC^{VN} Total power consumption of VN.

PPC^{VN} Processing power consumption of VN processor.

NPC^{VN} Networking power consumption of VN wireless adapter.

TPC^{NET} Total power consumption of the infrastructure network.

TPC^{RR} Total power consumption of core router.

TPC^{MR} Total power consumption of metro router.

TPC^{MS} Total power consumption of metro switch.

TPC^{O} Total power consumption of OLT.

TPC^{U} Total power consumption of ONU.

TPC^{A} Total power consumption of AP.

The total power consumption is composed of the following:

1) The total power consumption of CC (TPC^{CC}), which is composed of the processing power consumption (PPC^{CC}) and the networking power consumption of the CC router and switch (NPC^{CR}) and (NPC^{CS}), given as:

$$\mathrm{TPC}^{CC} = (PPC^{CC} + NPC^{CR} + NPC^{CS})\ \rho^{CC} \quad (4.2)$$

where ρ^{CC} is the PUE of the central cloud data centre.

$$PPC^{CC} = \left[P_{idle}^{n} \sum_{d \in CC} \delta_d + \frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}} \left(\sum_{s \in SN} \sum_{d \in CC} X_{sd}\right)\right], n \in CC \quad (4.3)$$

$$NPC^{CR} = \left[\tau\ P_{idle}^{n} \sum_{i \in CR} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}}\right) \left(\sum_{i \in CR} \lambda_i\right)\right], n \in CR \quad (4.4)$$

$$NPC^{CS} = \left[\tau\ P_{idle}^{n} \sum_{i \in CS} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}}\right) \left(\sum_{i \in CS} \lambda_i\right)\right], n \in CS \quad (4.5)$$

Equation (4.3) shows the calculation of the processing power consumption of CC servers. Equations (4.4) and (4.5) depict the calculations of the networking power consumption which is composed of the power consumption of the routers (CR) and switches (CS) of the CC network. Note that τ here represents the portion of the idle power attributed to the considered application traffic, which is equal to 6% [82]. More details will be given later in Section 4.5.1.

2) The total power consumption of the MF (TPC^{MF}), which is composed of the processing power consumption (PPC^{MF}) and the networking power consumption of the metro fog router and switch (NPC^{MFR}) and (NPC^{MFS}), given as:

$$\mathrm{TPC}^{\boldsymbol{MF}} = (PPC^{MF} + NPC^{MFR} + NPC^{MFS})\ \rho^{MF} \quad (4.6)$$

where ρ^{MF} is the PUE of the Metro Fog node.

$$PPC^{MF} = \left[P_{idle}^{n} \sum_{d \in MF} \delta_d + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}}\right) \left(\sum_{s \in SN} \sum_{d \in MF} X_{sd}\right)\right], n \in MF \quad (4.7)$$

$$NPC^{MFR} = \left[\tau\, P_{idle}^{n} \sum_{i \in MFR} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}}\right)\left(\sum_{i \in MFR} \lambda_i\right)\right], n \in MFR \quad (4.8)$$

$$NPC^{MFS} = \left[\tau\, P_{idle}^{n} \sum_{i \in MFS} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}}\right)\left(\sum_{i \in MFS} \lambda_i\right)\right], n \in MFS \quad (4.9)$$

Equation (4.7) shows the calculation of the processing power consumption of the MF server. Equations (4.8) and (4.9) depict the calculations of the networking power consumption which is composed of the power consumption of the routers (MFR) and switches (MFS) of the MF network.

3) The total power consumption of the LF (TPC^{LF})**,** which is composed of the processing power consumption (PPC^{LF}) and the networking power consumption of the router and switch of the OLT fog (NPC^{LFR}) and (NPC^{LFS}), given as:

$$\mathrm{TPC}^{LF} = (PPC^{LF} + NPC^{LFR} + NPC^{LFS})\ \rho^{LF} \quad (4.10)$$

$$PPC^{LF} = \left[P_{idle}^{n} \sum_{d \in LF} \delta_d + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}}\right)\left(\sum_{s \in SN} \sum_{d \in LF} X_{sd}\right)\right], n \in LF \quad (4.11)$$

$$NPC^{LFR} = \left[\tau\, P_{idle}^{n} \sum_{i \in LFR} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}}\right)\left(\sum_{i \in LFR} \lambda_i\right)\right], n \in LFR \quad (4.12)$$

$$NPC^{LFS} = \left[\tau\, P_{idle}^{n} \sum_{i \in LFS} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}}\right)\left(\sum_{i \in LFS} \lambda_i\right)\right], n \in LFS \quad (4.13)$$

Equation (4.11) shows the calculation of the processing power consumption of the LF server. Equations (4.12) and (4.13) depict the calculation of the networking power consumption which is composed of the power consumption of the routers (LFR) and switches (LFS) of the LF network.

4) The total power consumption of the NF (TPC^{NF}), which consists of the processing power consumption of NF processor and given as:

$$\mathrm{TPC}^{NF} = \left[P_{idle}^{n} \sum_{d \in NF} \delta_d + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}} \right) \left(\sum_{s \in SN} \sum_{d \in NF} X_{sd} \right) \right], n \in NF \quad (4.14)$$

It is worth mentioning that for NF, neither PUE nor networking power consumption is considered. This is attributed to the architecture of the NF as we assume that the processor is a Raspberry Pi board attached to an outdoor ONU. Hence, networking the incoming traffic to NF processor will be handled by the ONU and the networking power consumption of ONU will be calculated using Equation (4.23). Note that the ONU is assumed to be dedicated to the considered application. Therefore, the 6% fraction of ONU idle power was not included in Equation (4.14).

5) The total power consumption of the VN (TPC^{VN}), which is composed of the processing power consumption (PPC^{VN}) and the networking power consumption (NPC^{VN}), and given as:

$$\mathrm{TPC}^{\boldsymbol{VN}} = (PPC^{VN} + NPC^{VN}) \quad (4.15)$$

where

$$PPC^{VN} = \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}} \right) \left(\sum_{s \in SN} \sum_{d \in VN} X_{sd} \right), n \in VN \quad (4.16)$$

$$NPC^{VN} = P_{idle}^{n} \sum_{d \in VN} \delta_d + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}} \right) \left(\sum_{s \in SN} \sum_{d \in VN} \lambda_{sd} \right), n \in VW \quad (4.17)$$

Equation (4.16) shows the calculation of the processing power consumption of the VN processor. Note that the idle power of the VN power profile is not represented in this equation. This is because the vehicles may spend short

time periods in a car park, at a charging station or at road intersections. Therefore, processors have to be ON to utilise their processing capability immediately as soon as a vehicle has stopped (at the intersection or in other car parks, more generally). Hence, allocating workload to the VN processor will not consume extra power as a result of activating the processor. Equation (4.17) describes the networking power consumption of the VN wireless adapter (VW). This adapter is assumed to be dedicated for the traffic required by the IoT source nodes. Hence, allocating traffic to this adapter will consume extra power (P_{idle}^{VW}) as a result of activating the adapter. Similar to the ONU, the VN WiFi adapter is assumed to be installed in the VN and dedicated for the generated task traffic. Hence, activating the wireless adapter will consume the full idle power.

6) The total power consumption of the infrastructure network (TPC^{NET})**,** which is composed of the power consumption of core routers (TPC^{RR}), metro router (TPC^{MR}), metro aggregation switch (TPC^{MS}), OLT (TPC^{O}), ONU (TPC^{U}), and AP (TPC^{A}), and given as:

$$\mathrm{TPC}^{NET} = (TPC^{RR} + TPC^{MR} + TPC^{MS} + TPC^{O} + TPC^{U} + TPC^{A}) \quad (4.18)$$

where

$$TPC^{RR} = \rho^{NET}\left[\tau P_{idle}^{n} \sum_{i \in RR} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}}\right)\left(\sum_{i \in RR} \lambda_i\right)\right], n \in RR \quad (4.19)$$

$$TPC^{MR} = \rho^{NET}\left[\tau P_{idle}^{n} \sum_{i \in MR} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}}\right)\left(\sum_{i \in MR} \lambda_i\right)\right], n \in MR \quad (4.20)$$

$$TPC^{MS} = \rho^{NET}\left[\tau P_{idle}^{n} \sum_{i \in MS} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}}\right)\left(\sum_{i \in MS} \lambda_i\right)\right], n \in MS \quad (4.21)$$

$$TPC^{O} = \rho^{NET} \left[\tau P_{idle}^{n} \sum_{i \in O} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}} \right) \left(\sum_{i \in O} \lambda_i \right) \right], n \in O \quad (4.22)$$

$$TPC^{U} = \left[P_{idle}^{n} \sum_{i \in U} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}} \right) \left(\sum_{i \in U} \lambda_i \right) \right], n \in U \quad (4.23)$$

$$TPC^{A} = \left[P_{idle}^{n} \sum_{i \in A} \Psi_i + \left(\frac{P_{max}^{n} - P_{idle}^{n}}{\Omega_{max}^{n}} \right) \left(\sum_{i \in A} \lambda_i \right) \right], n \in A \quad (4.24)$$

The PUE of the network devices, ρ^{NET}, in the Equations (4.19)–(4.22), defines the added power consumption of network devices such as core routers, metro routers, metro switches, and OLTs attributed to cooling and lighting typically. A PUE=1 is considered for ONU and AP as both are small outdoor devices and no additional cooling installation is required for both of them. In addition, the 6% fraction of the idle power is not calculated for the AP as it is assumed to be dedicated for the considered architecture and the application considered.

The objective of the model is defined as follows:

Objective:

Minimise the total power consumption of all processing nodes and their interconnecting networks and the infrastructure network devices, given as:

$$TPC^{CC} + TPC^{MF} + TPC^{LF} + TPC^{NF} + TPC^{VN} + TPC^{NET}. \quad (4.25)$$

Subject to the following constraints:

1) Flow conservation of the network:

$$\sum_{\substack{j \in NN_i \\ i \neq j}} \lambda_{ij}^{sd} - \sum_{\substack{j \in NN_i \\ i \neq j}} \lambda_{ji}^{sd} = \begin{cases} \lambda_{sd} & if\ i = s \\ -\lambda_{sd} & if\ i = d \\ 0 & otherwise \end{cases} \tag{4.26}$$

$$\forall\ s \in SN, d \in PN, i, j \in N.$$

Constraint (4.26), ensures that the total incoming traffic is equal to the total outgoing traffic for all nodes in the network, excluding the source and destination nodes.

2) Processing allocation:

$$\sum_{d\ \epsilon\ PN} X_{sd} = \omega_s \qquad \forall\ s \in SN \tag{4.27}$$

$$X_{sd} \geq \delta_{sd} \qquad \forall\ s \in SN, d \in PN \tag{4.28}$$

$$X_{sd} \leq M1\ \delta_{sd} \qquad \forall\ s \in SN, d \in PN \tag{4.29}$$

$$\sum_{s\ \epsilon\ SN} \delta_{sd} \geq \delta_d \qquad \forall\ d \in PN \tag{4.30}$$

$$\sum_{s\ \epsilon\ SN} \delta_{sd} \leq M2\ \delta_d \qquad \forall\ d \in PN. \tag{4.31}$$

Constraint (4.27) ensures that the total processing workload sent from source node s, allocated to a processing node d is equal to the workload demand ω_s generated from source node s. Constraints (4.28) and (4.29) are used in the conversion of X_{sd} to its equivalent binary variable. When $\delta_{sd} = 1$, the task generated from source node s is allocated to processing node d. Constraints (4.30) and (4.31) are used to ensure that the binary variable $\delta_d = 1$ if processing node d is allocated any processing workload.

3) Total aggregated traffic traversing a node:

$$\sum_{s \in SN} \sum_{d \in PN} \sum_{i \in NN_j} \lambda_{ij}^{sd} = \lambda_j \qquad \forall\ j \in N \tag{4.32}$$

$$\lambda_i \geq \Psi_i \qquad \forall\ i \in N \tag{4.33}$$

$$\lambda_i \leq M3\ \Psi_i \qquad \forall\ i \in N \tag{4.34}$$

Constraint (4.32) calculates the total aggregated traffic traversing node $j \in N$. Constraints (4.33) and (4.34) are used in the conversion of λ_i into its equivalent binary variable. When $\beta_i = 1$, the node i is activated and traffic is travels through this node.

4) Flow allocation:

$$\lambda_{sd} = F_s\ X_{sd} \qquad \forall\ s \in SN, d \in PN. \tag{4.35}$$

Constraint (4.35), ensures that the traffic from source node s to processing node d is equal to the data rate of the workload generated from source s, where F_s is the ratio of the traffic to processing workload of the demand generated from source node s.

5) Processing node capacity:

$$\sum_{s \in SN} X_{sd} \leq C_d \qquad \forall\ d \in PN \tag{4.36}$$

Constraint (4.36) ensures that each demand generated from source node s allocated to a processing node d does not exceed the processing capacity of this processing node d.

6) Link capacity:

$$\sum_{s \in SN} \sum_{d \in PN} \lambda_{ij}^{sd} \leq L_{ij} \qquad \forall\ i \in N, j \in NN_i, i \neq j \tag{4.37}$$

Constraint (4.37), ensures that the traffic generated from source s to processing node d does not exceed the capacity of the link between any two nodes (i, j).

7) AP aggregated capacity limit:

$$\sum_{s \in SN} \sum_{d \in PN} \sum_{j \in NN_i \cap VN} \lambda_{ij}^{sd} \leq \Omega_{max}^{A} \qquad \forall\ i \in A \tag{4.38}$$

Constraint (4.38) ensures that the total traffic traversing AP does not exceeds the capacity of the AP.

8) Single assignment (no splitting):

$$\sum_{d \in PN} \delta_{sd} \leq \upsilon \qquad \forall\ s \in SN \tag{4.39}$$

Constraint (4.39) ensures that the processing task is not split. This may be essential in real time applications where there is no time to assemble partial results from partial processing locations. Removing this equation (instead of setting the right-hand side of the equation to υ splits) allows the optimisation to select the best number of splits to minimise the total power consumption. This is the other extreme compared to no splitting. Future work can consider partial splitting (different values of υ) and hence inter processors communication where parts of the tasks are processed.

It is worth mentioning that the objective function and the all the constraints in this model are originally in a linear form. Therefore, no further linerisation was

needed to execute the developed model and obtain the optimum location. However, some linerisation is introduced later, in Chapter 6, in an extended version of this model.

The processing allocation problem considers different evaluations with the availability of some or all processing nodes. For the VEC, multiple cases are considered (as well) to capture different vehicle densities. As mentioned previously, these VNs are clustered in a car park, at a charging station or by a road intersection within the coverage of an AP. Each vehicle is equipped with an OBU which defines the processing capability of the vehicle, and a wireless communication adapter to communicate with the AP. All VNs need to communicate with the AP, as no direct communication is allowed between VNs. All VNs are assumed to be homogeneous with the same processor capabilities. The vehicles thus work as a service provider for some of the collected data from applications related to IoT source nodes (SNs). These application tasks range from small-scale applications (with low demands), which do not require much processing capacity, to large-scale applications (high demands), which require a powerful processing node and sufficient communication link capacities to send and process the tasks. Additionally, various cases were evaluated with different ratios between processing demand and data rate demand to cover a wider range of applications. However, we assume that the tasks generated in each instance have the same processing and data rate requirements.

The allocation flow process of any generated task follows six phases:

1- A task is generated by a nearby IoT SN and is sent to an AP located in the same geographical zone. AP has full knowledge of the available resources. This task includes the required processing and data rate per task.

2- The AP sends a positive acknowledgment to the task SN.

These two phases are not evaluated in the model, as they are considered control signals and, therefore, generate negligible traffic that consumes little power.

3- The data to be processed are sent from the SN to the AP through the wireless channel. This phase is also not considered in the model, as it is a common phase for all tasks and will not affect the task allocation decision.

4- The data to be processed are offloaded from the AP to one or more of the available PNs (VN, NF, LF, MF, CC). As this phase carries the main data, it will affect the power consumption and the task allocation decision. Therefore, it is treated as the main component of the model.

5- The extracted knowledge resulting from the processed data is sent back from the processing PNs to the AP.

6- The extracted knowledge is reassembled and sent from the AP to the source node that requested the service.

As we assume that the extracted knowledge has a small volume compared to the main data, the last two phases are not included in the optimisation model.

4.5 Input data for MILP Model

The proposed MILP model was developed using the network in Figure 4.2. All devices parameters are obtained from data sheets to reflect a realistic values for the capacity and power values. Also, the parameters each layer servers were chosen based on the fact that the central cloud (CC) has the largest processing capacity and best processing power efficiency and PUE, followed by the other bottom layers. This is to reflect the fact that the larger the data centre is, the most efficient its servers are [82]. The network devices also are chosen to ensure that its capacity can serves all the generated traffic, specially those located at the bottleneck of the network (i.e. OLT). Finally, the generated requests vary from low to high parameters to ensure the robustness of the architecture with high processing and networking demands.

This section will explain in details the input data considered for the model parameters, including the processing node capacities and efficiencies, the capacities of network devices and the power efficiencies, PUE, link capacities, and generated workload.

4.5.1 Idle power consumption

Accurate values for the idle power consumption of each device are not easy to obtain in data sheets all the time. Accordingly, we based our idle power consumption values on multiple frameworks driven from the literature. First, according to [80], [81], up to 95% of the maximum power consumption for network devices (routers and switches) is attributed to idle power (90% is assumed in this thesis). Therefore, this figure will be quoted in describing the network device's power consumption in the present work. Second, based on

[83], a processing node consumes 60% of the maximum power. Thirdly, in high-power-capacity network equipment, IoT applications account for a small portion of the idle power consumption. Therefore, it would be unjust to ascribe the total power consumption to a specific application. For this reason, we elected, instead, to use the 2017–2022 Cisco Visual Networking Index (VNI) [1] to express the traffic of IoT applications as a fraction of the total traffic in smart cities, such as the smart city scenario in this work. In particular, the work in [1] reported that, by 2022, IoT traffic will constitute around 6% of all global IP traffic on the Internet. Therefore, we used this number (6%) to attribute part of the idle power to these types of applications.

4.5.2 PUE

The power usage effectiveness (PUE) is a factor used to measure the power efficiency of any network or data centre. It estimates how much power is used for the actual computing and communication, in relation to the total power resulting from computing and communication equipment plus non-IT equipment such as cooling, lighting, ventilation, etc. The PUE values of the network infrastructure and each processing node are listed in Table 4.1.

According to Google's report in [82], PUE values have an inverse relationship to the "Space Type" of the data centre. PUE decreases with increase in the data centre size and geographical location. We assumed that PUE, at any layer, is indicative of both structure (network equipment) and function (processing). As the central cloud (CC) is a large data centre that uses sophisticated liquid and air cooling, it typically has lower PUE values compared to other processing nodes.

The telecom infrastructure is owned typically by a carrier (e.g. BT Openreach). This gives the telecom infrastructure one PUE. On the other hand, the cloud infrastructure is typically owned and operated by a cloud services provider (e.g. Amazon), which may or may not share the same building with the telecom carrier (BT Openreach), depending typically on the size of the cloud service provider and the number of racks and servers they own. This may result in a different PUE for the network devices and fog processing nodes (for example, metro switch and metro fog).

Table 4.1 PUE values for the network devices and processing nodes

Processing node	**PUE value**
Central cloud PUE ($\boldsymbol{\rho}^{CC}$).	1.1 [82]
Metro fog PUE ($\boldsymbol{\rho}^{MF}$).	1.4 [82]
OLT fog PUE ($\boldsymbol{\rho}^{LF}$).	1.5 [82]
Network devices PUE ($\boldsymbol{\rho}^{NET}$).	1.5 [61]

4.5.3 Capacity and power consumption of network devices

In our evaluation, we have chosen our processing devices based on the fact that the top-most layer PN (CC) has the largest processing capacity and best processing power efficiency. Conversely, the bottom-most PN (VN) has the lowest capacity and power efficiency. Moreover, all the vehicles are assumed to have homogenous processors. These processors are chosen with advanced OBU that are capable to be shared as edge nodes and to highlight these vehicles, as smart vehicles, from ordinary vehicles with old processors. The capacities and efficiencies of the processors of the other fog nodes vary

between the CC and VN. In this model, the top-layer processor is the best in terms of capacity and efficiency.

It is worth mentioning that only one CC is considered as a local data centre. This CC is assumed to have servers that are sufficient to process all of the generated tasks. The other three fog layers (MF, LF, and NF), each has only one processor (one server in each of MF/LF, and one Raspberry Pi chip in the NF). The number of vehicles in the VEC varies based on the scenario considered. However, all VNs are assumed to be homogeneous, with the same processing capability. For security purposes, we assumed that the VN processor and its wireless communication transmitter and receiver are combined in a separate "box" which is not linked to the vehicle CAN bus. In time, the entertainment or other less critical processors in the vehicle may participate, and security must be considered if these processors of the vehicle main processors (e.g. controlling engine, windows, wipers, etc.) are used. The development of security and trust frameworks for the opportunistic vehicular clouds is outside the scope of the current work, but can build on existing cloud security and vehicle security frameworks.

The capacity of each PN is defined in terms of the Instructions per Second (IPS) it can provide. Based on device datasheets, IPS is not considered when defining CPU capability. Hence, according to [84], this value is estimated using:

$$IPS = COR \times CPS \times IPC \tag{4.40}$$

where COR is the number of cores in a processor, CPS is the processor clock rate in GHz. Both values can be extracted from the server datasheets. IPC is

defined as the number of instructions per cycle, and is estimated for each processing node based on the fact that a high performance processor can execute four IPCs [85].

Table 4.2 lists all values used in Equation (4.40) with the resultant capacity, in million instructions per second (MIPS), for each PN. In addition, the same table shows the power efficiency, in W/MIPS, calculated using the linear power profile method explained earlier in Section 4.3.

Table 4.2 Processing node power, capacity, and efficiency parameters.

PN	Model	P_{max} (W)	P_{idle} (W)	Cores	GHz (cycle/sec)	IPC (Ins/cycle)	Capacity (MIPS)	Efficiency (W/MIPS)
CC	Intel Xeon E5-2680 [86]	115	69	10	3.6	4	144k	0.00032
MF	Intel Xeon E5-2630 [87]	85	51	10	2.2	4	88k	0.00039
LF	Intel Xeon E5-2609 [88]	85	51	8	1.7	4	54.4k	0.00063
NF	RPI 4 Model B [89]	15	9	4	1.5	1	6k	0.001
VN	MobiWAVE iMX6 [90]	10	6	2	0.8	2	3.2k	0.00125

For the other network devices, Table 4.3 shows the power consumption and capacity values. Comparing the network devices' power values in this table, we can see that the core router port consumes a maximum of 638 W at 40 Gb/s. On the other hand, the metro aggregation router consumes 25 W, at the same port data rate. This gap between the power consumption values of the two infrastructure routers is due to the devices' different operating functions. The core router has many more functions, including traffic engineering, layer lookup tables, etc. On the other hand, the metro router has to do much less

than the core router in terms of traffic engineering and number of other nodes to interface with. Therefore, in general, metro routers consume less power at a given data rate compared to the more sophisticated core router operating at the same data rate. Moreover, the metro layer relies more on layer 2 metro switches and, therefore, the metro switch has a higher power consumption, as shown in Table 4.3. As such, at 40 Gb/s, in our scenario, the metro layer consumes 500 W + 25 W peak power versus 638 W at the core layer.

Table 4.3 Network devices power and capacity parameters

Network device parameter		Value
Core layer	Core router port maximum power consumption	638 W [91]
	Core router port idle power consumption	574.2 W [91]
	Core router port maximum capacity	40 Gb/s [91]
Metro layer	Metro router port maximum power consumption	25 W [92]
	Metro router port idle power consumption	20 W [92]
	Metro router port maximum capacity	40 Gb/s [92]
	Metro switch maximum power consumption	500 W [93]
	Metro switch idle power consumption	450 W [93]
	Metro switch maximum capacity	1800 Gb/s [93]
Access layer	OLT maximum power consumption	50 W [94]
	OLT idle power consumption	45 W [94]
	OLT maximum capacity	1920 Gb/s [94]
	ONU maximum power consumption	15 W [95]

	ONU idle power consumption	13.5 W [95]
	ONU maximum capacity	10 Gb/s [95]
Edge layer	AP maximum power consumption	11 W [96]
	AP idle power consumption	4.8 W [96]
	AP maximum capacity	1.167 Gb/s [96]
CC network	CC router port maximum power consumption	25 W [92]
	CC router port idle power consumption	20 W [92]
	CC router port maximum capacity	40 Gb/s [92]
	CC switch maximum power consumption	460 W [97]
	CC switch idle power consumption	368 W [97]
	CC switch maximum capacity	600 Gb/s [97]
MF & LF Network	MF/LF router port maximum power consumption	13 W [98]
	MF/LF router port idle power consumption	11.7 W [98]
	MF/LF router port maximum capacity	40 Gb/s [98]
	MF/LF switch maximum power consumption	245 W [99]
	MF/LF switch idle power consumption	220.5 W [99]
	MF/LF switch maximum capacity	200 Gb/s [99]
VEC Network Adapter	Vehicle wireless adapter maximum power consumption	2.5 W [100]
	Vehicle wireless adapter idle power consumption	1.5 W [100]
	Vehicle wireless unit maximum capacity	72.2 Mb/s [100]

As seen, Table 4.3 shows the power consumption and capacity values as required and represented by Equations (4.2)–(4.24). In contrast, Table 4.4 provides direct comparison between the power consumption in the different

layers, taking into account the percentage of idle power attributable to cloud and fog traffic. In addition, Table 4.4 shows energy efficiency values for network devices. For example, the core router idle power consumption is 34.45 W at 40 Gb/s per port and with 6% utilisation attributed to cloud and fog IoT traffic. The corresponding idle power consumption in the metro layer is 27+1.35=28.35 W. Moreover, the corresponding energy efficiency figures (i.e. the slope of the linear part of Figure 4.3) are 1.595 W/Gb/s and 0.091 W/Gb/s.

Table 4.4 Network devices power, capacity and efficiency parameters.

Network Layer	Device	P_{max} (W)	P_{idle} (W) (90%)	P_{idle} (W) (6%)	C_{max} (Gb/s)	Efficiency ($W/_{Gb/s}$)
Core layer	Core router port [91]	638	574.2	34.452	40	1.595
Metro layer	Metro router [92]	25	22.5	1.35	40	0.063
	Metro switch [93]	500	450	27	1800	0.028
Access layer	OLT [94]	50	45	2.7	1920	0.003
	ONU [95]	15	13.5	- *	10	0.150
Edge layer	AP [96]	11	4.8	- *	1.167	5.313
CC network	CC Router port [92]	25	22.5	1.35	40	0.063
	CC Switch [97]	460	414	24.84	600	0.077
MF network	MF Router port [98]	13	11.7	0.702	40	0.033
	MF Switch [99]	245	220.5	13.23	200	0.123
LF network	LF Router port [98]	13	11.7	0.702	40	0.033
	LF Switch [99]	245	220.5	13.23	200	0.123
VN network	VN Wi-Fi adapter [100]	2.5	1.5	- *	0.0722	13.850

* the device is assumed to be fully dedicated to the application considered

4.5.4 Processing and data rate requirements

In our evaluation, as explained in Section 4.5.3, we define the CPU capacity in MIPS. Taking inspiration from [101], one of the analysed IoT sensors was

a smart city-based sensor designed to detect environmental events using an earthquake prediction algorithm. It is reported that a job with a size of 11.72 kB (0.09 Mb/s data rate) needs 78 MIPS to be processed. Through simple calculations, we derived the processing requirement. We assumed a minimum required data rate equal to 1 Mb/s, and calculated the required MIPS to execute 1 Mb of data as follows:

$$\Phi = \frac{78MIPS}{0.09Mb/s} = 866 \cong \frac{1000\ MIPS}{\frac{Mb}{s}} \tag{4.38}$$

Based on the above calculation, and as we assumed a futuristic increase of the traffic demands, we selected the 1,000 MIPS as the minimum required processing demand per task and examined a range of settings where we increased this demand up to 10,000 MIPS per task.

We assumed that the data rate of any task rises with increase in the processing workload. Therefore, in Equation (4.41), we introduce a relationship between the processing workload demand and the data rate demand for each requested task as a ratio, termed as 'data rate ratio' (DRR). Fixed DRR, equal to 0.001, is considered for the evaluations in Sections 0 and 4.6.3. Therefore, the required data rate ranged from 1 Mb/s to 10 Mb/s. On the other hand, different DRR values were assessed in Section 4.6.4.

$$DRR = \frac{traffic\ (Mb/s)}{processing\ (MIPS)} \tag{4.41}$$

It should be noted that small DRR value represents an (IoT) application where small data volumes are generated, for example by measuring a physical quantity, followed by extensive processing of the data. Large DRR values may represent situations where large data volumes are generated followed by

limited processing, for example as in video streaming or some forms of gaming.

4.6 Power Consumption and Processing Allocation Results

4.6.1 Scenarios considered

This section describes the scenarios and the architecture considered in the proposed processing allocation MILP model. As mentioned earlier in this chapter, three main dimensions were studied, summarised as follows:

1) Centralised versus distributed processing (in CC, MF, LF, and NF)
2) Opportunistic processing (in VEC)
3) Single versus distributed task allocation (non-splittable versus splittable demands)

Two different architectural designs, inherited from Figure 4.2, were assessed based on the design of the edge network. The first architecture considers one zone, with multiple VEC clusters. This design captures an urban area where many car parks, charging stations or road intersections are available. It also depicts VEC clusters, each consisting of VNs parked in a car park, at a charging station or stopping at a road intersection and connected to an AP located in the same cluster. With different potential locations for the source generation and with different APs in each cluster, we can assess the effect of different realistic scenarios. In the second architecture, we expanded the PON network to include multiple zones, with numerous ONUs, each with one or more AP (VEC cluster). Considering different zones in this architecture allowed us to mimic an expanded urban area (a city, for example), where the

infrastructure can connect multiple VECs located in different zones within the city.

In each architecture, four cases were evaluated to capture the main task allocation dimensions, as follows:

- ***Central Cloud Allocation (CCA)***. In this case, centralised processing was evaluated by allocating all tasks to the central cloud servers. This represents a baseline approach to which other cases (considering the VEC) are compared.
- ***Cloud-Fog Allocation* (*CFA*):** In this case, we considered a cloud-fog architecture, with available cloud and fixed fog PNs, but with no available VNs. This case introduces processing over distributed locations. It also represents off-peak periods over the day where no vehicles are in the city car parks and charging stations (and at intersections) or situations where vehicles are not participating in the resource provisioning service.
- ***Cloud-Fog-VEC Allocation with low vehicular nodes density* (*CFVA-L*):** In this case, VNs are introduced at a low density. Each VEC within a cluster includes two VNs, with the total number of available VNs equal to eight. This choice limits the size of the problem to a size that can be handled efficiently by the MILP given that the allocation problem is known to be NP hard. This case represents low peak periods with a limited number of vehicles.
- ***Cloud-Fog-VEC Allocation with high vehicular nodes density* (*CFVA-H*):** In this case, the number of VNs is increased to 15 VNs per VEC, with a total of 60 available VNs over four VEC clusters/zones. This

case represents a high density of VNs, as experienced in the peak periods during the day.

The last two cases, CFVA-L and CFVA-H are evaluated to highlight the dynamic of vehicles in the model, specifically as reflected in the number of vehicles available to take part in the opportunistic task allocation at any point in time. The different vehicle densities represent different periods of time that the car park, road intersection, .. etc. is affected by the mobility of the vehicles and therefore the location of interest has a varied number of vehicles over the day. Also these two cases, CFVA-L and CFVA-H, are assessed with single and distributed (non-splittable and splittable) allocation to capture different real-time demands.

With each architecture, four scenarios are evaluated to capture the impact of the number of generated tasks and the location of these tasks, as follows:

- Scenario 1: one task generated from one cluster/zone.
- Scenario 2: one task generated from each cluster/zone.
- Scenario 3: Five tasks generated from one cluster/zone.
- Scenario 4: Five tasks generated from each cluster/zone.

Scenario 1 and scenario 4 capture the extreme ends of the generated tasks. On the other hand, scenario 2 and scenario 3 capture the in-between cases where the number of tasks varies between one task, in scenario 1, and 20 tasks in scenario 4. Moreover, in scenarios 1 and 3, the tasks are generated from one end of the architecture. This aims to capture the need of allocating takes to other network clusters (non-neighbours VEC), and therefore, the benefit of the expanded architecture. Scenarios 2 and 4 aimed to evaluate the

processing allocation in case of congested network, where all clusters/zones are congested with the generated tasks.

The two architectures and their results are explained next in Sections 4.6.2 and 4.6.3 As the evaluations in these two sections considered a fixed DRR value, with DRR=0.001, Section 4.6.4 presents a comparison study at different DRR values, to evaluate the impact of the different processing and traffic volume on the processing allocation and the total power consumption.

4.6.2 Processing Allocation in Cloud-Fog-VEC Architecture with One Zone and Multiple VEC Clusters

The end-to-end architecture considered in this evaluation is similar to the architecture presented in Figure 4.2. However, the edge network evaluated in this section includes only one zone (with one ONU), as shown in Figure 4.4. This zone consists of one ONU (as mentioned above) and four clusters. Each cluster represents a car park, a charging station or road intersection with an AP and connected VNs. Tasks are generated from nearby IoT SNs located in the same cluster. These SNs are assumed to have connection only with their local AP and cannot communicate with any AP located in a different cluster. Therefore, different tasks can be allocated by each AP. In this section, four scenarios are defined to capture different situations based on the number of generated tasks and the location of the source node. In all evaluated scenarios, the tasks processing requirements ranged from 1,000 MIPS to 10,000 MIPS. The required data rate for the tasks increased, based on DRR=0.001, from 1 Mb/s to 10 Mb/s.

Furthermore, four cases were assessed, as explained in Section 4.6.1, referred to as Central Cloud Allocation (CCA), Cloud-Fog Allocation (CFA), Cloud-Fog-VEC Allocation with Low VN (CFVA-L) and Cloud-Fog-VEC Allocation with high VN (CFVA-H), both with single (SA) and distributed (DA) allocation strategies.

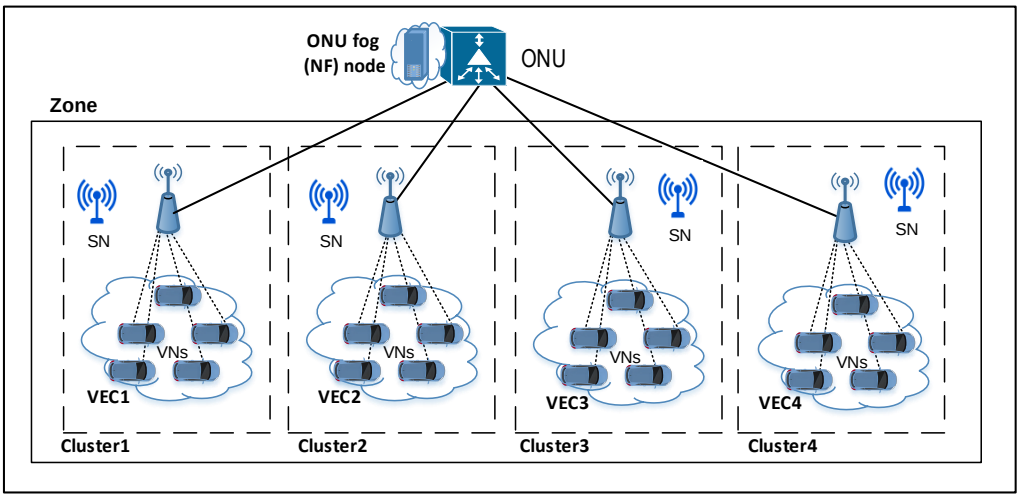

Figure 4.4 Edge network design with one zone

4.6.2.1 Scenario 1: One task generated from one cluster

In this scenario, we assumed that, in any time instance, one task is generated from a source node located in one of the four clusters. Figure 4.5 shows the total power consumption of the different cases considered. As the power consumed is a function of the networking and processing power consumption, the processing placement becomes a result of a trade-off between networking and processing power values to achieve the optimum processing allocation. This optimised allocation at each processing node (PN) of the CC, Metro Fog (MF), OLT Fog (LF), ONU Fog (NF), and Vehicular Edge Cloud (VEC) is summarised in Figure 4.6.

Figure 4.5 shows that allocating all tasks to CC consumes the highest power with a linear increase in relation to the size of the generated tasks. Given the absence of VNs in CFA, tasks are allocated to the most efficient fixed PN, until it is fully exhausted or becomes too thin to accommodate the task. For example, with low workload demands, NF is the most efficient PN, as seen in Figure 4.6. This is because it is closer to the edge and therefore consumes less networking power, achieving up to 87% power saving, compared to CAA. However, at 7000 MIPS, NF becomes too thin to allocate tasks with high demands. Hence, tasks are allocated to the next most efficient fixed fog node (LF). This explains the jump in the power consumption where the power saving is reduced to 44%, (also compared to CCA).

When vehicular nodes become available, both single and distributed allocation strategies were assessed with low and high VN density. In the case of single allocation (CFVA-L and CFVA-H), and as shown in Figure 4.5, the model achieves an early power saving of, on average, 70%, compared to CFA. This saving is caused by allocating the tasks to the available VN (as in Figure 4.6) thus, avoiding the activation of the ONU and its fog server. However, this saving is limited to cases where demands are within the VN processor's capacity. This explains the increase in power consumption, at 4000 MIPS, where all tasks are allocated to NF, and therefore in this case the power consumption is the same as that of the CFA. It is also observed that, in single allocation, the VN density has no effect on the allocation decision, and therefore, has no influence on the power consumption. This is because, with the no-splitting constraint, only one VN is needed to serve the generated task.

In distributed allocation, Figure 4.5 shows that, with low demands, the same power saving as that seen in the single allocation is achieved. This is because one VN was enough to serve the generated task, and therefore the splitting flexibility was not needed. However, a continuous power saving is observed regardless of the VN capacity limitation (i.e. beyond the 3000 MIPS), achieving up to 71% power saving, in comparison to single allocation. This is attributed to the splitting ability of the model and bin-backing the split task into the available VNs, therefore achieving better power saving. Moreover, it was observed that when the available VNs are not sufficient to serve the full task, the model initially allocates a major part of the task to the most efficient fixed PN until it is fully utilised, and then allocates the remaining part of the task to the available VN. For instance, in Figure 4.6 (in CFVA-L and at 7000 MIPS), it can be seen that the available VNs (with a total of 6400 MIPS capacity) are not able to serve the whole task. Hence, the model activates NF with 6000 MIPS allocation, and then allocates the remaining sub-task to the local VEC. This causes the majority of the task to be allocated to NF, as it has a more efficient processor than the VN. In contrast with single allocation, VN density shows an impact on the task allocation and the power consumption. The high density VN increased the capacity of the VEC and, therefore, more tasks were allocated to the available VNs and ONU activation was avoided. Thus, further power saving of up to 50% were achieved, compared to the low VN density (CFVA-L (DA)).

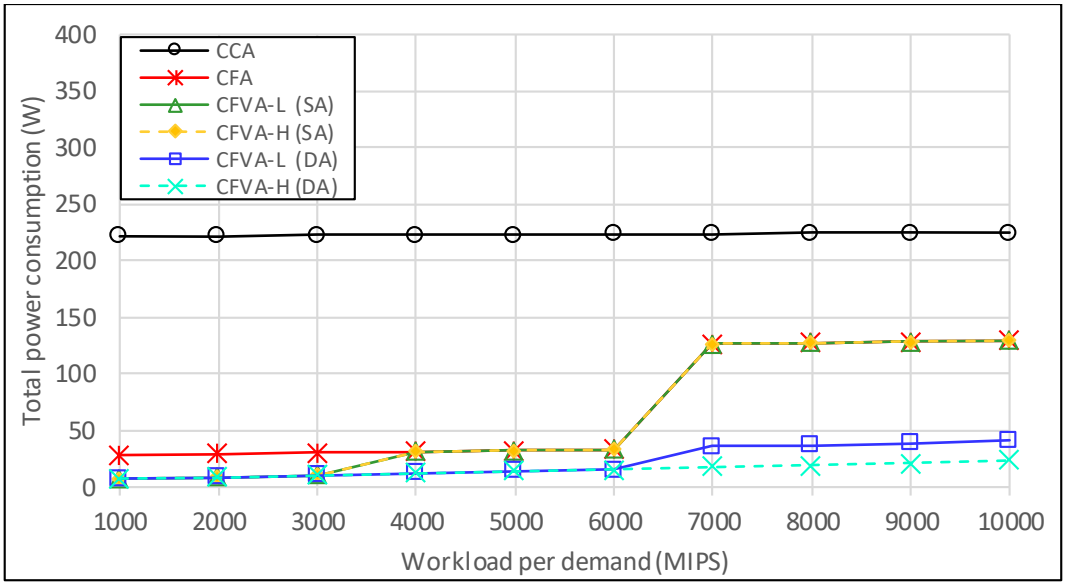

Figure 4.5 Total power consumption, in Scenario 1 with one zone.

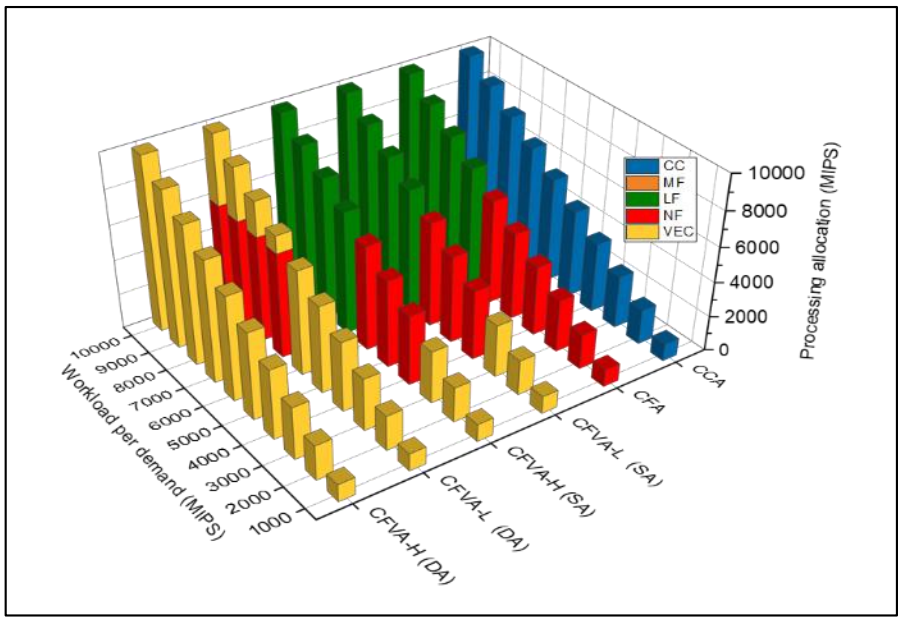

Figure 4.6 Processing allocation in each PN, in Scenario 1 with one zone.

Part of the aim of this work is to design a VEC architecture that is able to expand and connect many VEC clusters together in a cloud-supported architecture. Accordingly, it is very important to study the allocation behaviour among the VEC clusters considered. Figure 4.7 summarises the processing

allocation in each individual VEC, taking into account the fact that the task was generated from VEC1 (this will be referred to as local VEC).

The single allocation results show that, regardless of the VNs' density, the tasks allocated to VECs were limited to the 1000–3000 MIPS range. This is due to the VN limited capacity. Hence, all tasks were allocated to one VN in the local VEC, as the model was constrained by the "no splitting" condition. Similarly, in distributed allocation, a single task with high demand was split and allocated to VNs located in the local VEC. No splittable tasks were allocated to the other VEC clusters, even when the VNs in the local VEC were exhausted. This is because allocating sub-tasks to a non-local VEC will activate another ONU. Hence, it is more efficient to utilise the NF, with its efficient processor, rather than allocating the sub-task to a non-local VEC. On the other hand, the same figure shows that, as the VNs' density increased to 15 VNs per VEC, the local VEC became enough to allocate the whole task, even with the increasing processing demand.

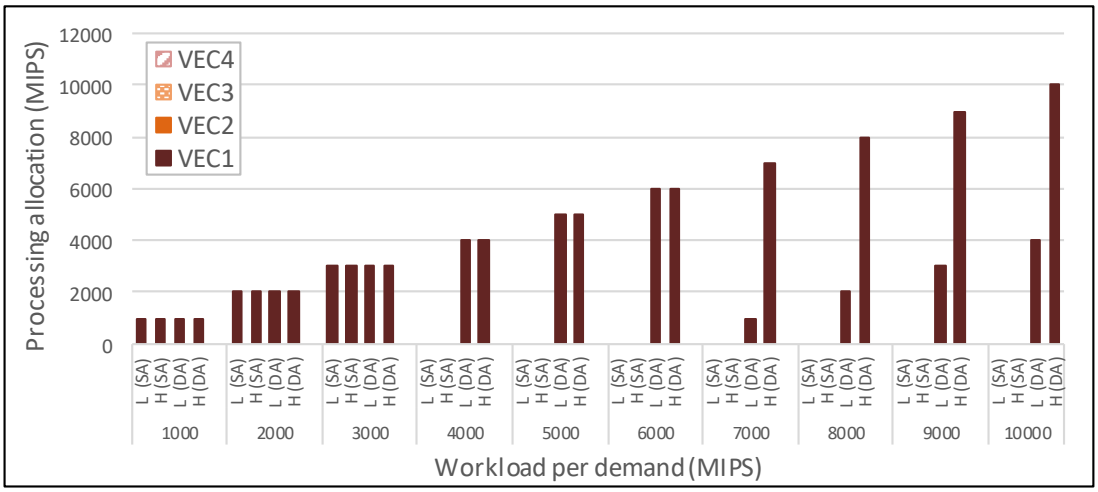

Figure 4.7 Processing allocation in each VEC in CFVA (SA) and CFVA (DA), in Scenario 1 with one zone.

4.6.2.2 Scenario 2: One task generated from each cluster

In this scenario, we aimed to investigate the increase of the tasks and the activation of multiple APs, by generating one task from each cluster with a total of four tasks. Figures 4.8 and 4.9 show the total power consumption and the processing allocation in each PN, respectively.

In this scenario, we observed less power saving, as shown in Figure 4.8, compared to Scenario 1. This is attributed to the increase in the generated tasks. In addition, Figure 4.9 shows an allocation behaviour which is relatively comparable to scenario 1. The bottom-most processing nodes have the most efficient total power consumption due to their associated low networking power consumption. Hence, tasks are allocated first to these nodes if they can satisfy the processing workload. Moreover, when the most efficient PN cannot accommodate all of the generated tasks, an upper PN becomes the most efficient location, and is fully utilised first. For example, in CFA, with 2000 MIPS or more, NF cannot allocate all of the four tasks (with a total of 8000 MIPS), as this exceeds the NF capacity. Although NF can accommodate three out of the four tasks, all tasks were allocated to the LF, as seen in Figure 4.9. This behaviour is attributed to two factors. First, activating one PN is more efficient than activating both NF and LF. This is due to the power overhead and idle power resulting from activating two PNs. Second, once the OLT and its fog server are activated, it is more efficient to allocate all tasks to this PN, as its processor has better efficiency compared to the NF processor. Despite the NF limitation, allocating tasks to LF yields up to 41% power saving, compared to CCA.

Similar to Scenario 1, VN density has no effect on the single allocation cases (CFVA-L and CFVA-H), as the number of generated tasks is small enough to be satisfied by the available VNs when the demands of the (four tasks) are within the VN processor's capacity. However, in CFVA-L(SA), we observed an allocation in LF; specifically, when the VNs were exhausted. The tasks were allocated to LF rather than NF. This is, again, due to the same reason of avoiding the overhead resulting from activating two PNs. This explains the jump in the power resulting at 4000 MIPS, as seen in Figure 4.8. Moreover, CFVA-L and CFVA-H, with distributed allocation, have comparable allocation results to Scenario 1, except that with increasing demands beyond 7000 MIPS. Here, all tasks were allocated to the LF, as both NF and VEC combined cannot satisfy the required demands. Despite the small power saving achieved in this scenario, CFVA (in both allocation strategies and with both VN densities) can still save power by up to 87% and 61%, compared to CCA and CFA respectively. Moreover, splitting tasks in distributed allocation guarantees a continuous power saving, with high demands, of, on average, 59%, compared to single allocation.

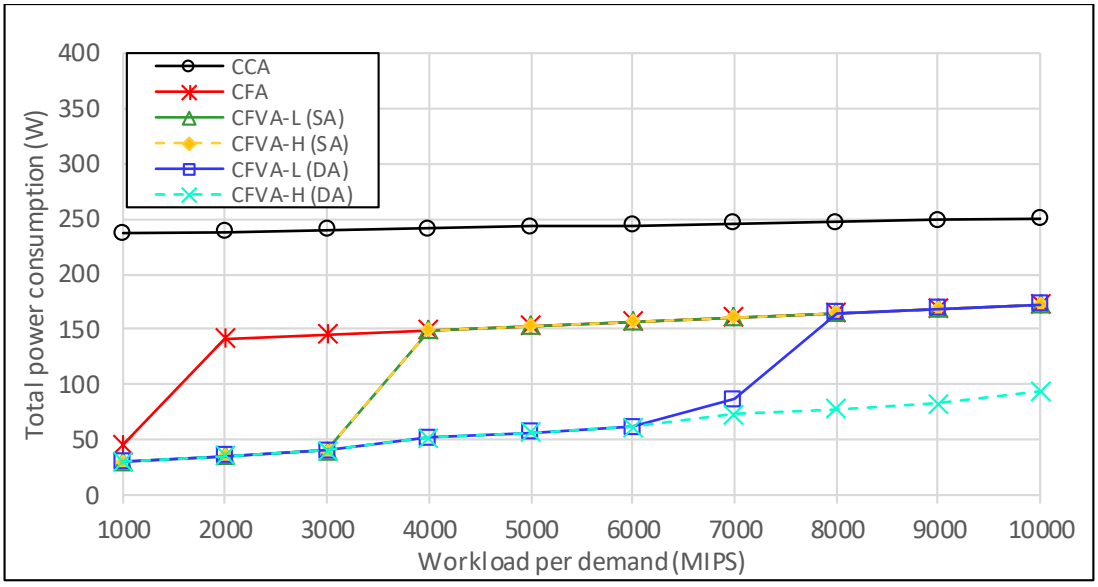

Figure 4.8 Total power consumption, in Scenario 2 with one zone.

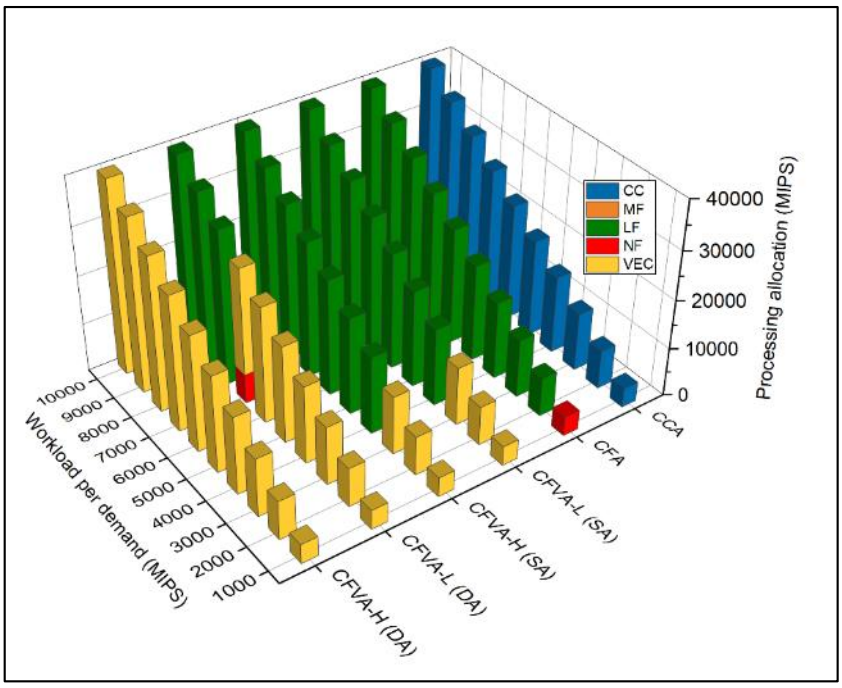

Figure 4.9 Processing allocation in each PN, in Scenario 2 with one zone.

Figure 4.10 summarises the processing allocated in each VEC in this scenario. Recalling that each task was generated from a different cluster, an equal utilisation is observed, for all VEC clusters, as each VEC is considered a local processing node for one task. Despite this, in DA, and by 7000 MIPS,

the VEC was unequally utilised, as the insufficient capacity of the available VNs caused the activation of the ONU and NF processor. Thus, it was more efficient to fully utilise the NF and then allocate the remaining processing from each task to its local VEC.

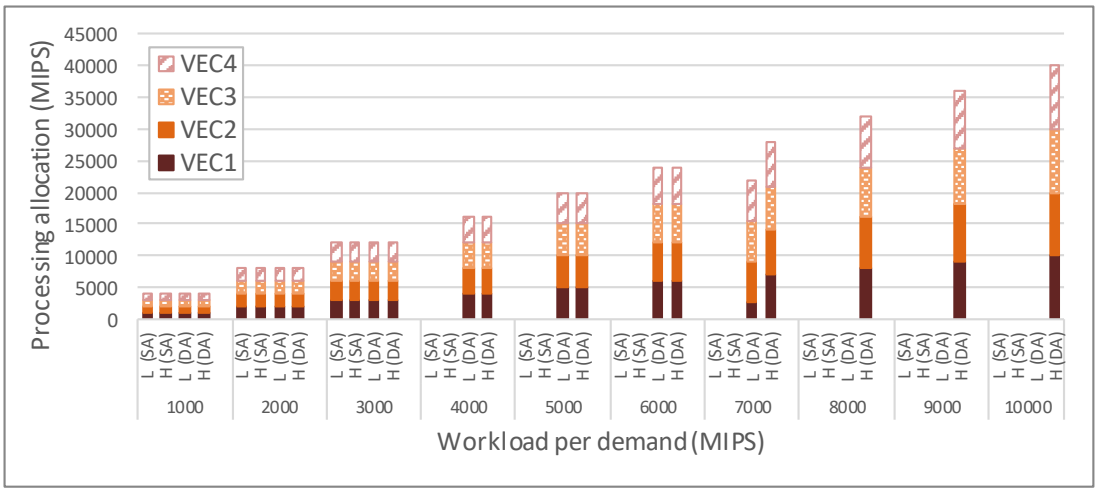

Figure 4.10 Processing allocation in each VEC in CFVA (SA) and CFVA (DA), in Scenario 2 with one zone.

4.6.2.3 Scenario 3: Five tasks generated from one cluster

In this scenario, the number of generated tasks has increased to five tasks originating from one cluster. The allocation trend in this scenario follows the same trend observed in Scenarios 1 and 2, highlighted through the following two points. First, the PN nearest to the SN is the most efficient in total power consumption. Second, when the most efficient PN cannot satisfy all the required demands, an upper layer PN is activated and is fully utilised before utilising other PNs. Following this, the optimisation begins to allocate the remaining tasks (or splittable tasks), if there are any, to the nearest PN, which has enough capacity to accept the allocation. The results supported by these trends are shown in Figures 4.11 and 4.12.

The results illustrate that, despite the increase in the number of generated tasks, the VEC is still attractive, and efficient, when it comes to serving the generated tasks. Moreover, unlike the previous two scenarios, increasing the VNs' density affected the single allocation (SA) results in CFVA-H, with up to 42% power saving compared to CFVA-L. This is because, with low density, the local VEC was not enough to serve the five tasks generated (with 2000 and 3000 MIPS). Thus, allocating tasks to NF was more efficient than sending these tasks to a non-local VEC, as observed in Scenario 1. Moreover, increasing the number of VNs in each cluster, in CFVA-H (SA), enables the optimisation to allocate all generated tasks to the local VEC, whenever this is sufficient.

In distributed allocation (DA), CFVA-L shows an early increase in the power consumption, as seen in Figure 4.11. This is attributed to a new allocation behaviour where the optimisation allocates tasks to a non-local VEC, as seen in Figure 4.13 (at 3000–6000 MIPS). The non-local VEC in this case became an efficient location combined with the NF and local VEC. Moreover, utilising these two locations (NF and VEC) is more efficient than activating the OLT and its fog server (LF). However, this allocation causes an increase in the power consumption, which explains the early and continuous power saving in CFVA-H(DA) with 34%–48% power saving compared CFVA-L (DA).

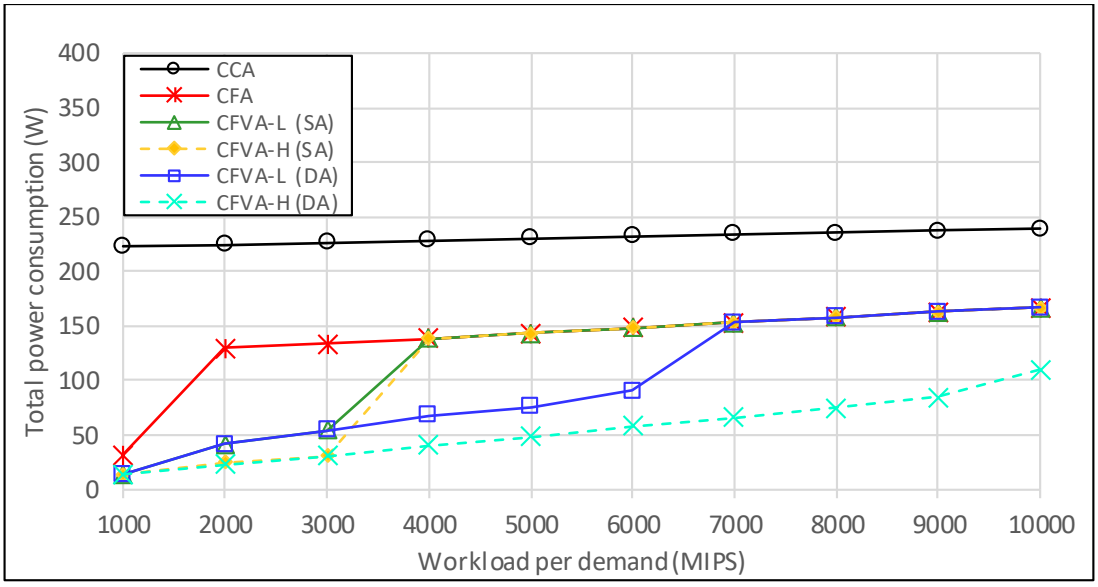

Figure 4.11 Total power consumption, in Scenario 3 with one zone.

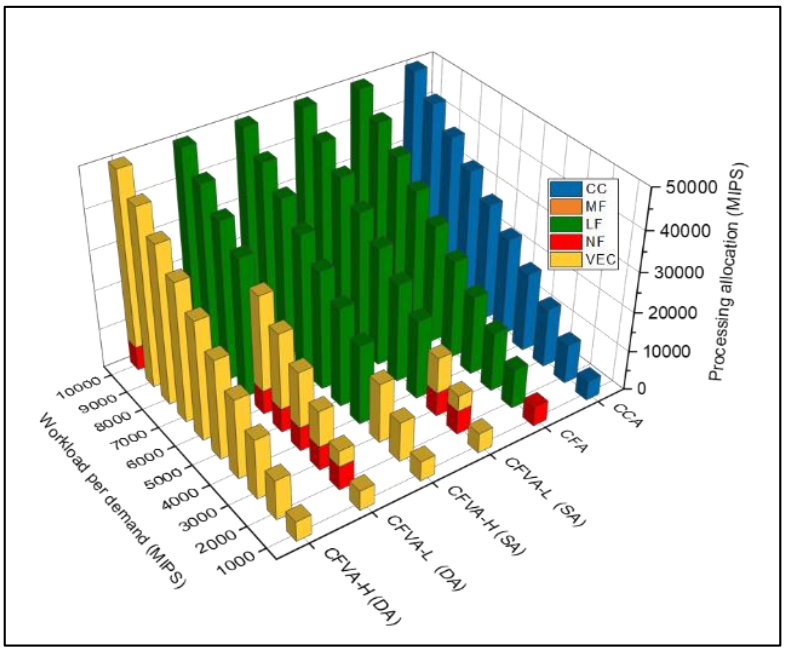

Figure 4.12 Processing allocation in each PN, in Scenario 3 with one zone.

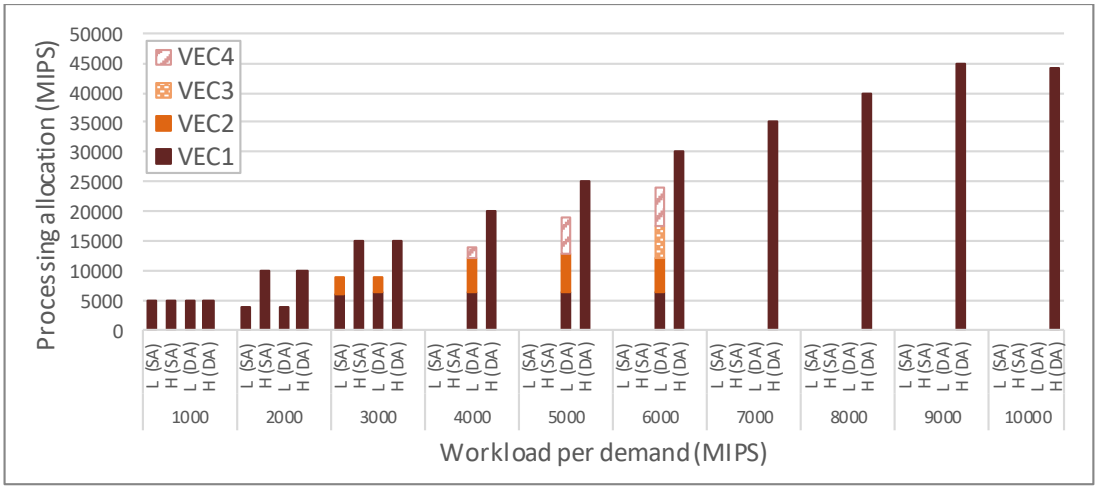

Figure 4.13 Processing allocation in each VEC in CFVA (SA) and CFVA (DA), in Scenario 3 with one zone.

4.6.2.4 Scenario 4: Five tasks generated from each cluster

In this scenario, we further increased the number of generated tasks and assumed that each cluster received five generated tasks from a local source node.

As seen in Figure 4.14, CCA experiences a nonlinear increase in the power consumption. This is due to the extra server(s) activated to accommodate the increased demands. It is also observed that the power savings in CFA and CFVA experience a dramatic reduction, compared to the previous scenarios. This is also attributed to the increased number of generated tasks and the elevated number of activated PNs, as shown in Figure 4.15.

In CFA, the model still yields a power saving of 20%–37% with the low demands, compared to CCA. This occurs when tasks are allocated to NF, LF, and MF (Figure 4.15). However, this saving stopped by 5000 MIPS when all tasks were allocated to the highest processing location, i.e. CC. The reason for this allocation is attributed to two factors. First, the high-demand (in terms

of number of MIPS needed) tasks cannot be satisfied by the access layer PNs, which are the most efficient location (i.e. LF and NF). Second, although MF and LF servers, combined, can serve the required generated workload, it is more efficient to activate one server in the CC instead of activating two MF and LF servers. For example, activating MF and LF consumes a combined idle power of 102 W. On the contrary, activating one server in the CC consumes only 69 W. Moreover, the CC has a very low PUE of 1.1, compared to 1.4 and 1.5 PUE in MF and LF, respectively. As a result, CC becomes the optimum placement to process the generated tasks.

In CFVA-L (SA), similar behaviour is observed with less VEC utilisation and early activation for the CC server. In CFVA-H (SA), the increase of VNs density (with low demands) has a clearer effect on the power consumption, compared to the low density case. As seen in Figure 4.14, a 36%–42% power saving is achieved by CFVA-H (SA) compared to CFVA-L (SA). Although this power saving is limited to two cases with low demands, this can show that the total capacity of the available VNs processor is crucial in the allocation decision. The impact is confirmed in the distributed allocation (CFVA-L and CFVA-D), where more tasks are allocated to the VEC, as seen in Figure 4.15. Moreover, the splitting and bin-packing flexibility of the model increase the VEC utilisation in, and show the impact on, the power saving.

Despite the common behaviour of allocating the high-demand tasks to CC, one feature was noticed, namely that in some cases the optimisation allocated a portion of processing to VEC. For instance, in CFVA-L (DA) at 7000 MIPS, the MILP solution allocated all tasks to CC, but at 8000 MIPS, a portion was

allocated to the VEC. This can be justified through two observations. The first is the common behaviour of selecting the PN with the smallest spare capacity that is enough to allocate the tasks, and then allocating the remaining portion to the bottom most sufficient PN (VEC in this case). The second reason is that activating each server in CC consumes a power overhead resulting from the idle and PUE values. Thus, it is more efficient to allocate the remaining processing (16000 MIPS) to VEC rather than activating a new CC server.

Figure 4.16, shows that both VNs' density and the distributed allocation strategy have a substantial impact on the processing allocation in VEC. Moreover, the source of the generated demands is another factor which helps to maintain an equal utilisation for all VEC clusters, despite the increase in the processing demands.

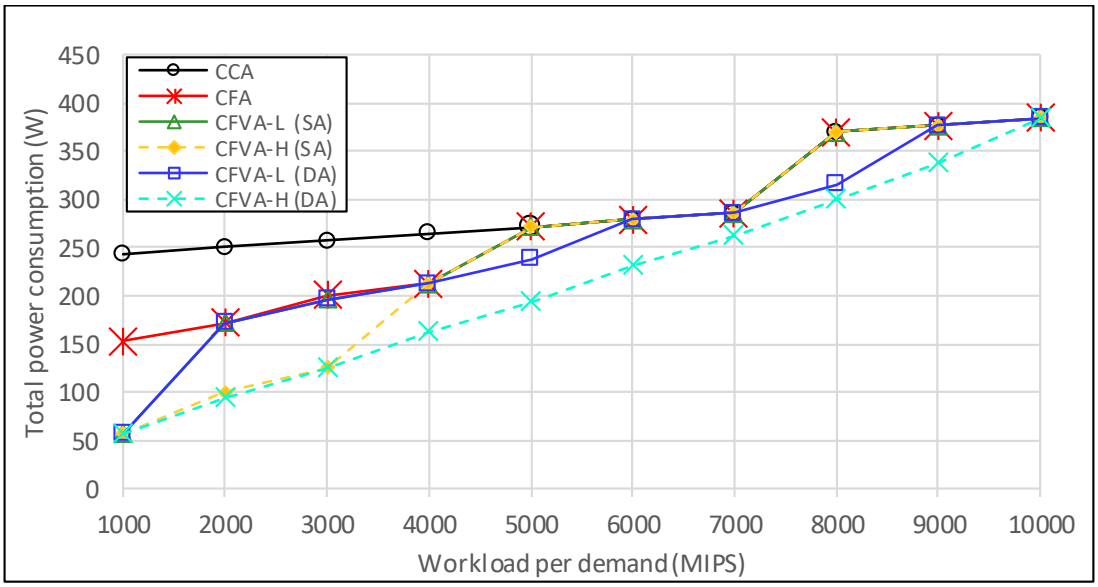

Figure 4.14 Total power consumption in Scenario 4, with one zone.

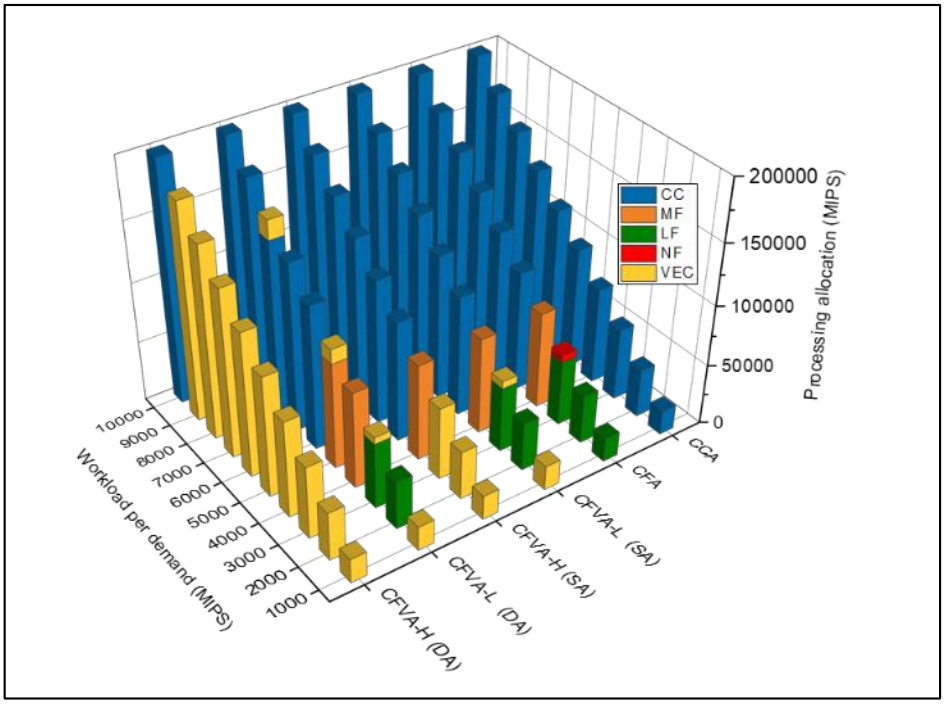

Figure 4.15 Processing allocation in each PN, in Scenario 4, with one zone.

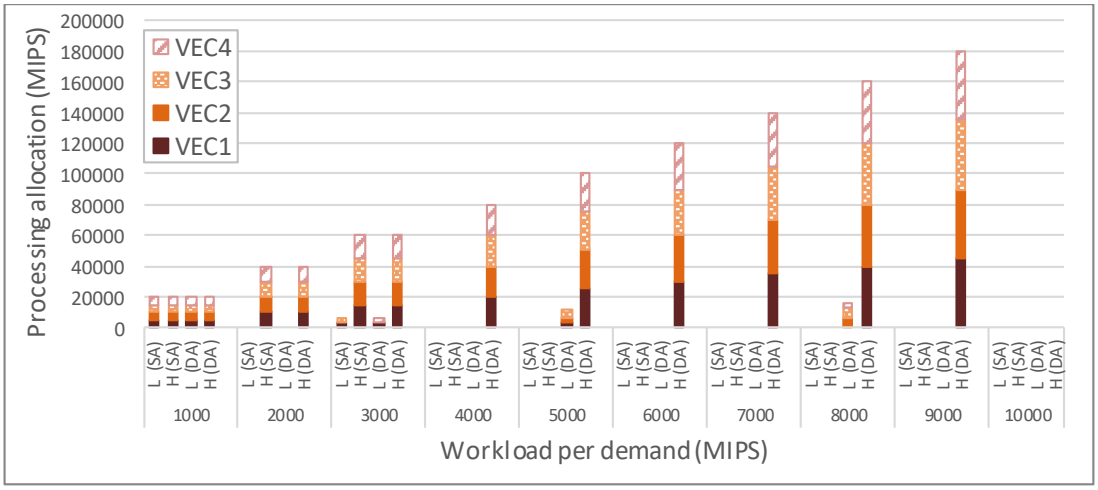

Figure 4.16 Processing allocation in each VEC in CFVA (SA) and CFVA (DA), in Scenario 4 with one zone.

4.6.3 Processing Allocation in Cloud-Fog-VEC Architecture with Multiple Zones

In this section, similar to the previous section, we based our evaluation on the architecture shown in Figure 4.2. However, instead of having one zone in the

edge network, we considered four zones representing different geographical areas, illustrated in Figure 4.17. The access network has also expanded, with four ONUs, each allocated a fog unit (NF) with a total of four NF nodes (instead of one NF node in Section 4.6.2). To reduce the model's complexity, i.e. its runtime, and to be able to examine scenarios with high VNs density, we considered a setting where each zone includes only one cluster (with one VEC and one AP). The aim of this expanded design is to evaluate the processing allocation with multiple VEC clusters located in different zones. Similar to the evaluation scenarios described in Section 4.6.1, four scenarios were considered, with the same highlighted cases and tasks requirements.

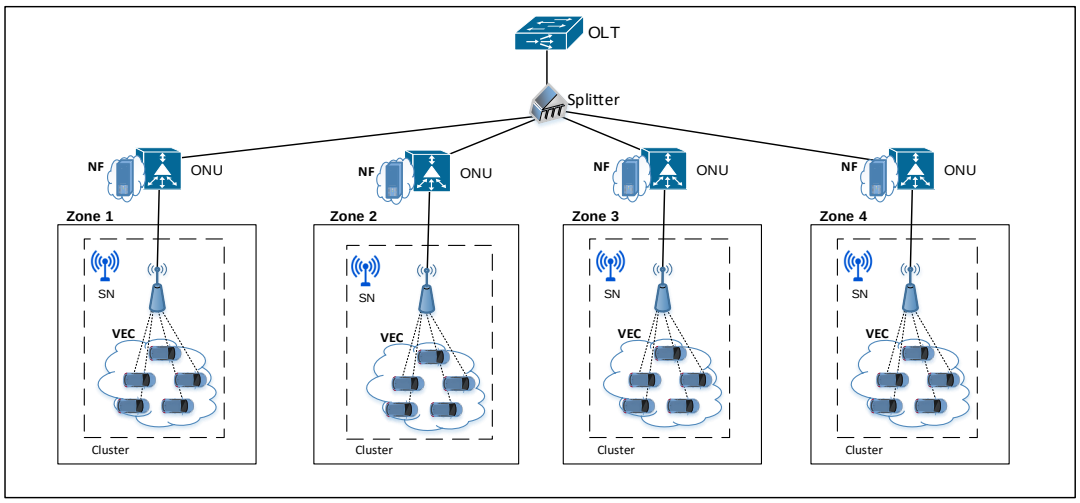

Figure 4.17 Edge network design with four zones

The results of the power consumption, allocation decision, and VEC utilisations are provided in Figures 1 to 12 in Appendix A.

The fours scenarios confirm a minimum effect of expanding the architecture on the power consumption and the processing allocation. Scenario 1 produced identical results to the same scenario with one zone (in Section

4.6.2.1). the other three scenarios produced a comparable results compared to the same results with one zone. The effect of the expanded architecture can be highlighted with increase of the number of active ONUs (this is because each zone is centralised by an ONU). Increasing the number of ONUs (and NF nodes) increases the power consumption whenever tasks are generated from each zone. This is due to the need of activating the ONUs to process or forward the tasks to upper PNs. On the other hand, increasing the ONUs can save more power whenever they can fulfil the generated tasks and therefore, avoid activating upper layer NFs (i.e. LF). Other than this effect, the power consumption results follow the same trends observed in the one zone architecture.

4.6.4 The effect of demands variety on the processing allocation in Cloud-Fog-VEC Architecture with One Zone

The evaluations in the previous two sections (4.6.2 and 4.6.3) were conducted with fixed demands requirements for the processing (MIPS) and data rate (Mb/s). As explained in Section 4.5.4, we considered a generated demand with minimum of 1000 MIPS requirement per task. Moreover, the relation between the required processing and data rate is based on a defined value referred to as Data Rate Ratio (DRR). Thus, the increased processing demands in the previous evaluations were accompanied by a fixed increase in the required data rate based on a DRR value equal to 0.001. In this section, we aim to explore different ranges of DRR values that create varied ranges of generated demands which can represent multiple applications. The goal of

exploring these ranges is to examine a vast array of input sets to assess the considered architecture and the developed model with different types of potential applications.

We considered two sets of processing demands (high and low) to capture tasks needs in terms of processing workload. The high demands input set varied from 1000 MIPS to 10000 MIPS requirement per task. On the other hand, the low demand set varied between 100 and 1000 MIPS per task. Moreover, we considered increasing the required data rate per task, for both processing demand sets. Different DRR values were defined, varying between 0.001 and 0.8 to consider different scenarios with low and high data rates. These DRR values were chosen from a wide range of possible values where different behaviours of the processing allocation were observed. It is worth mentioning that results with a DRR value below 0.001 were not presented, as they proved to have the same model behaviour and allocation decisions as the results of 0.001. The same was true for values higher than 0.8, where comparable results were observed to the DRR=0.8 results. The former (DRR<0.001) indicates that traffic is very low and processing power consumption dominates the allocation decisions leading to the use of the most energy efficient processor. The converse is true at DRR>0.8, where traffic power consumption dominates, leading to the use of the nearest processor.

The following are the values considered for DRR: 0.001, 0.04, 0.08, 0.1, 0.2, 0.4, and 0.8. As examples of the types of applications represented by these DRR values, a DRR value of 0.001 represents a task that is intensive in processing and light in communication, for example sensing simple data and

then processing it intensively. At the other extreme, the DRR of 0.8 may represent processing video streams, which is intensive in communication and processing. Other applications could include video streaming, images or large sensor files.

In this evaluation, we considered end-to-end architecture, where the edge network consists of one zone and four VEC clusters, as illustrated earlier in Figure 4.4. We evaluated the scenario where one task is generated from one cluster. It is worth mentioning that, as we challenge the network with high DRR values, and therefore high traffic, the source node generating the task is assumed to have a wired connection to the AP to accommodate the large amount of generated traffic. For example, with a processing demand of 1000 MIPS at DRR of 0.8, the generated traffic is equal to 800 Mb/s, which can be sent through the wireless 1 Gb/s connection of the AP. However, with a 10000 MIPS generated task, the traffic is equal to 8 Gb/s. This traffic can be accommodated by the wired connection while the AP, in this case, consumes power to act as a coordinator.

The next section will explain the results at different DRR values with high and low demands for one generated task. Both data sets are tested with single and distributed allocation strategies. A case was considered where all the processing nodes (CC, MF, LF, NF, and VN) are available, and the vehicular nodes exist with low density (CFVA-L).

4.6.4.1 Scenario 1: One generated task with high processing demand

This section describes the results (for both single and distributed allocation strategies) when one generated task is under increasing demand from 1000 to 10000 MIPS.

First, Figures 4.18 and 4.19 show the total power consumption and processing allocation in each PN, respectively, with single allocation strategy.

Figure 4.18 illustrates that the total power consumption for single allocation strategy has relatively comparable behaviours for all DRR values. Moreover, Figure 4.19 displays that the processing allocation in all PNs remains the same for all DRR values between 0.8 and 0.08. This is due to the high data rate associated with these DRR values and with the high processing demands. As a result, no processing was allocated to the VEC despite the fact that a VN can process a single task up to 3200 MIPS. This is because the minimum processing demand for the single task (1000 MIPS) requires a minimum data rate of 80 Mb/s, which exceeds the connection capability of a VN. Consequently, tasks were accommodated by NF and LF for all these DRR values (0.8 – 0.08).

With low DRR values, 0.04 and 0.02, the required data rate associated with low processing demands fit within the capacity of the VN connection. Therefore, tasks with 1000 MIPS, in DRR=0.04, and with 1000–3000 MIPS, in DRR=0.02, were allocated to the VEC, as shown in Figure 4.19. This explains the drop in power consumption for both ratios, saving up to 75% power compared to DRR=0.8, as depicted in Figure 4.18. However, the VEC allocation was terminated afterwards and the model followed the same allocation behaviour observed with other DRR values. This could be due to

the VN connection limitation (in 0.04), or due to connection and processing capacity limitation (in 0.02). Yet, a saving of up to 42% is achieved, compared to DRR=0.8, as both ratios (0.02 and 0.04) result in a low data rate requirement and, therefore, less power consumption.

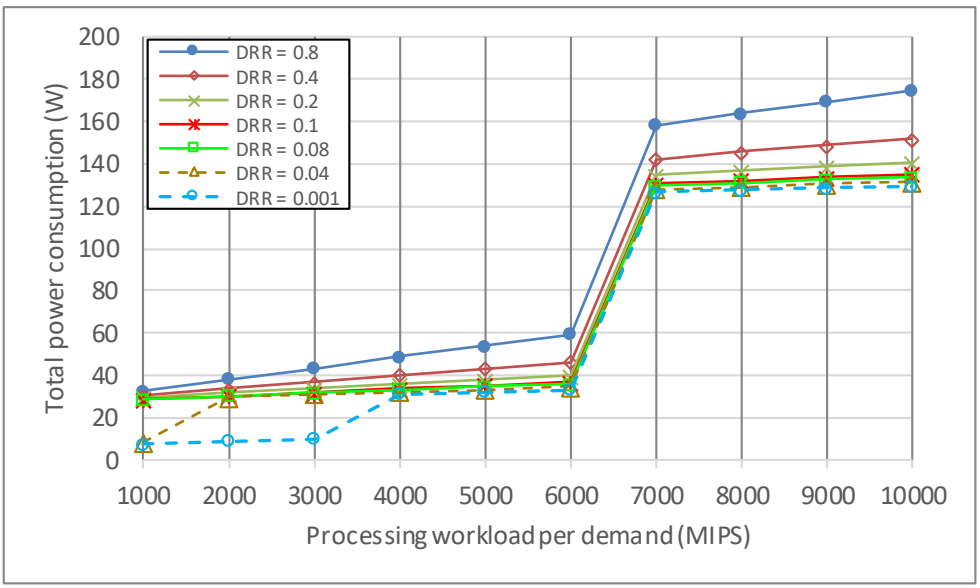

Figure 4.18 Total power consumption in single allocation with high processing requirements

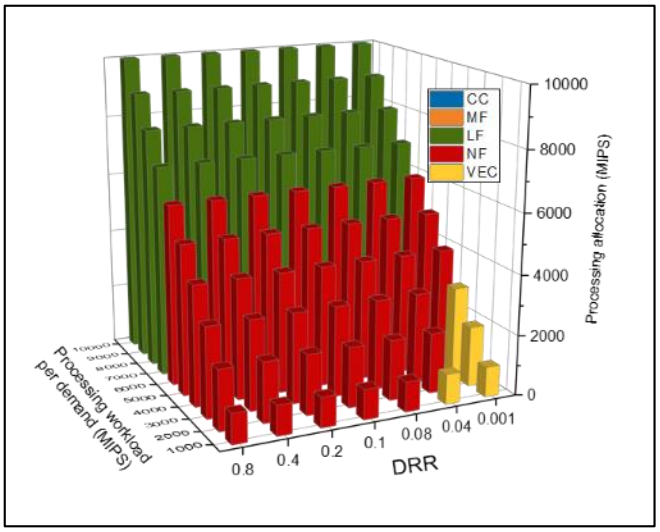

Figure 4.19 Processing allocation in each PN in single allocation with high processing requirements

Figures 4.20 and 4.21 show the total power consumption and processing allocation in each PN, respectively, for the distributed allocation strategy. We observed that splitting the task can solve the VN connection limitation, as the required flow will be split proportionally with the given workload. Therefore, the optimisation was able to achieve better utilisation of the VEC with medium DRR values (0.2 – 0.08) and the low DRR values (0.04 and 0.02), as seen in Figure 4.21. For instance, with a DRR value of 0.2, the 200 Mb/s required data rate, accompanied by the 1000 MIPS task, was split among three VNs. Similarly, tasks with DRR equal to 0.1 (up to 2000 MIPS) and 0.08 (up to 3000 MIPS) were allocated to VEC by splitting the processing allocation among the available VNs. According to these results, and as shown by Figure 4.20, distributed allocation was able to achieve up to 63% power saving with medium DRR values, compared to single allocation. This saving increased to 63% – 68% in some cases in the three medium DRR values, as the optimum solution resulted in the use of the VEC with NF full utilisation. This is a result of the higher power consumption which results from activating OLT and LF. The above matches the model behaviour with expanded architecture, as confirmed in Sections 4.6.2 and 4.6.3. With low DRR values (0.02 and 0.04), an improvement in VEC utilisation and power savings is achieved. This is attributed to the low-demanding data rate per task in relation to the required processing workload and the low DRR values. The low data rate resulted in the optimisation splitting the task among the available four VNs without exceeding the capacity of the VN connection. By this allocation, the model saved power of 34% – 81% (with DRR 0.04), compared to single allocation.

This saving percentage is based on the amount of extra workload allocated to VEC after utilising the NF. With DRR=0.02, the power consumption saving increased further to up to 84%, as all tasks were allocated to the VEC.

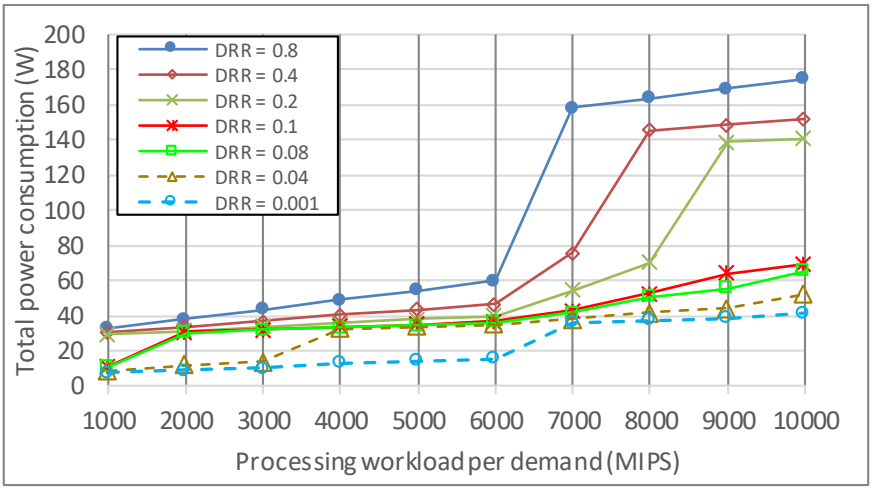

Figure 4.20 Total power consumption in distributed allocation with high processing requirements

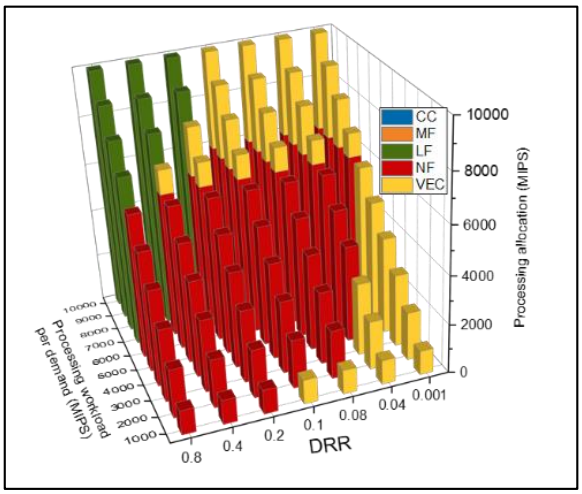

Figure 4.21 Processing allocation in each PN in distributed allocation with high processing requirements

4.6.4.2 Scenario 2: One generated task with low processing demand

Similar to Scenario 1, this scenario evaluates a single generated task but with low processing demands, ranging from 100 – 1000 MIPS. As seen in Figures

4.22 and 4.23, the results showed similar processing allocation behaviours to Scenario 1. However, as the low processing workload decreased the data rate requirement, a huge power saving is expected, and was confirmed, as seen in Figure 4.22. Figure 4.23 shows that DRR=0.8 caused a data rate bottleneck even with the lowest possible demand (i.e. 100 MIPS). However, with DRR=0.4, the VEC had a task allocation even with the lowest generated demand. In the same figure, we observe that the VEC utilisation increased for medium DRR values (0.2–0.04), compared to Scenario 1. This is due to the low processing workload that can be satisfied by one VN processor. This VEC utilisation stopped when the data rate exceeded the VN connection capacity, based on the DRR value. In both low DRR values (0.04 and 0.02), a full tasks allocation was achieved by the available VNs. This confirms that, regardless of the processing demand, VEC represents the most efficient PN, as long as its VNs and their connection can satisfy the tasks' requirements.

The results for the distributed allocation in Figure 4.24 and Figure 4.25 show comparable improvement values in the power consumption and allocation to that achieved by Scenario 1 under high demands.

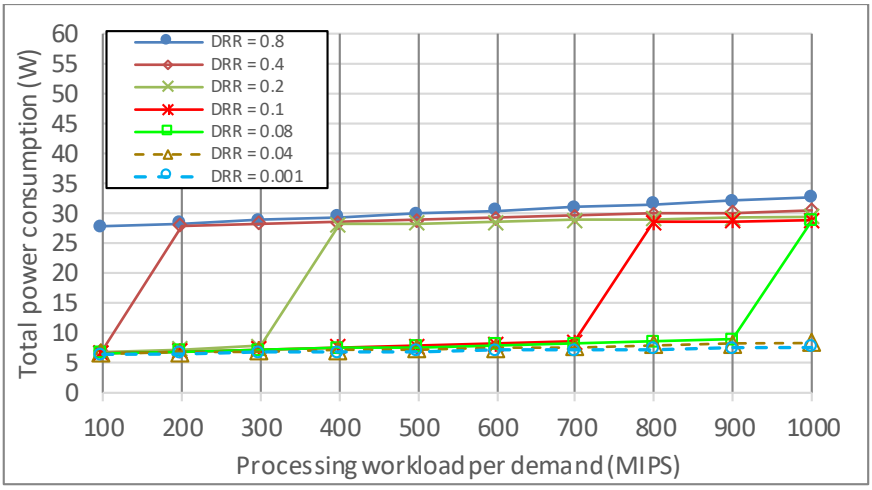

Figure 4.22 Total power consumption in single allocation with low processing requirements

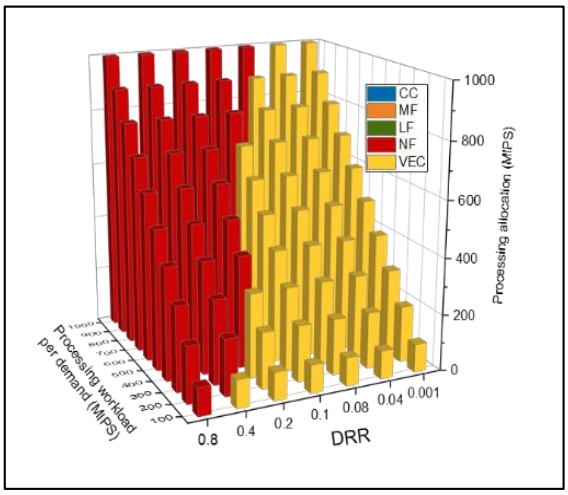

Figure 4.23 Processing allocation in each PN in single allocation with low processing requirements

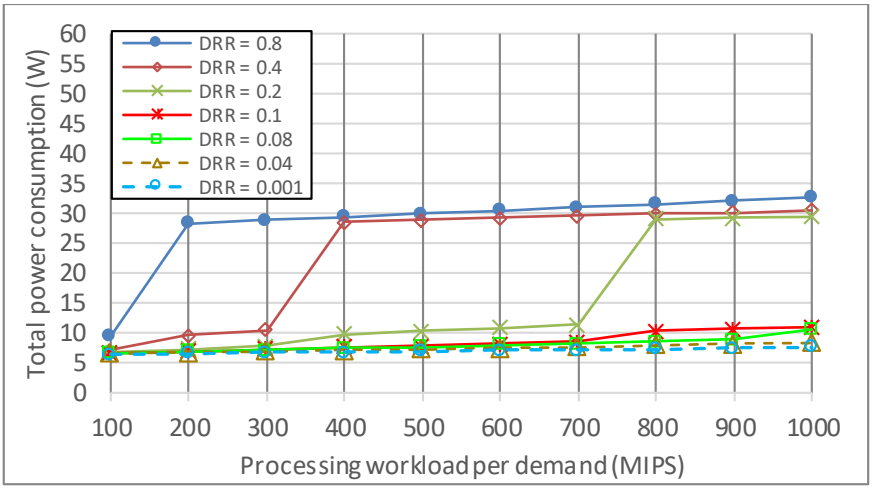

Figure 4.24 Total power consumption in distributed allocation with low processing requirements

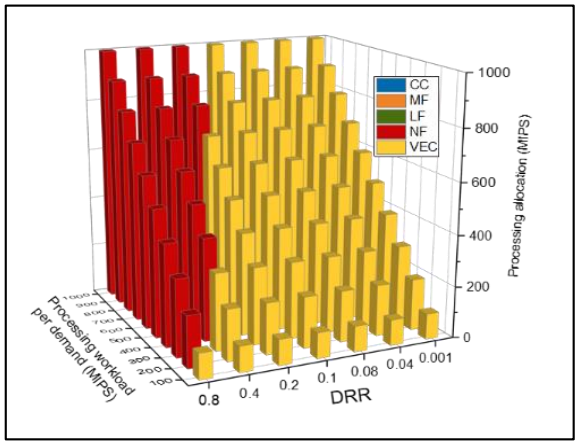

Figure 4.25 Processing allocation in each PN in distributed allocation with low processing requirements

4.7 MILP Model Validation

In this section, we provide an analysis of the results in order to verify the MILP model and provide confidence in the model results provided in this chapter. The validation is conducted through checkpoints with multiple cases obtained from the provided scenarios. Table 4.5 shows that if the allocation is correct, the MILP calculated power consumption is correct as well. Two potential solutions were tested with one solution matching the MILP allocation. On the other hand, Table 4.6 shows that the allocation and the calculated power are correct by examining all the possible allocation solutions and hence shows that the MILP resulted in the optimum allocation and the minimum total power consumption.

In each tables, the following attributes are added to achieved the model validation:

- A reference is added, at the top of the table, to the scenario that is evaluated.
- The columns titled as 1, 2, 3, .. etc, represent the different tested cases (potential solutions with the potential allocation decisions) where one of them should match the MILP model solution in order to validate its result.
- The rest of the table is divided into two sections to calculate and validate the power consumption of processing and networking. This is done using the same devices parameters used in the MILP model ,$W/_{MILPS}$, $W/_{Mb/s}$, idle power, and PUE, which summarised in Tables 4.1, 4.2, and 4.4.

- The first section (with blue border) calculates the processing power consumption of each processing node based on the allocated processing to this node, in all potential cases. Accordingly, the total processing power consumption is calculated and compared to the MILP to find the case that matches the model result (coloured in Red).
- The second section of the table (with green border) calculates the networking power consumption of each networking device based on the traffic traversing this node in potential case. Similarly to the processing power, the total networking power consumption is calculated and compared to the MILP to find the case that matches the model result (coloured in Red).
- Finally, the total power consumption is summed out of the processing and networking power consumption. The final result, in the tail of the table is compared to the MILP model result to validate it with a reference to the associated figure in Section 4.6.

Table 4.5 Analytic verification of the optimal choice in Scenario 2 with one zone network.

Checkpoint 1					
Edge network design: one zone with 4 clusters **(CFA)** **Scenario 2:** one generated task per cluster, each with 3000 MIPS and 3Mb/s					
Two potential Solutions →				**1**	**2**
Allocation decision →				All tasks in LF	2 tasks allocated to NF and 2 tasks allocated to LF
Processing Nodes				**Processing Power (W)**	
	W/$_{MIPS}$	**Idle(W)**	**PUE**		
CC	0.00032	96	1.1		
MF	0.00039	51	1.4		
LF	0.00063	51	1.5	[(12000 * 0.00063) + 51] * 1.5 = **87.84**	[(6000 * 0.00063) + 51] * 1.5 = **82.17**
NF	0.001	9	-		(6000 * 0.001) + 9 = **15**
VN	0.00125	6	-		
Total Processing Power (W) →				**87.8**	**97.1**
MILP Result →				**87.8 W**	
Network Devices				**Networking Power (W)**	
	W/$_{Mb/s}$	**Idle(W)**	**PUE**		
CC Switch	0.00008	24.8	1.5	-	-
CC Router	0.00006	1.35	1.5	-	-
Core Router	0.0016	34.5	1.5	-	-
Metro Router	0.00006	1.35	1.5	-	-
MF Switch	0.00012	13.23	1.4	-	-
MF Router	0.00003	0.7	1.4	-	-
Metro Switch	0.00003	27	1.5	-	-
LF Switch	0.00012	13.23	1.5	[(12 * 0.00012) + 13.23] *1.5= **19.847**	[(6 * 0.000003) + 2.7] * 1.5 = **19.846**
LF Router	0.00003	0.7	1.5	[(12 * 0.00003) + 0.7] * 1.5 = **1.0505**	[(6 * 0.000003) + 2.7] * 1.5 = **1.0503**
OLT	0.000003	2.7	1.5	[(12 * 0.000003) + 2.7] * 1.5 = **4.0501**	[(10 * 0.000003) + 2.7] * 1.5 = **4.05**
ONU	0.00015	13.5	-	(12 * 0.00015) + 13.5 = **13.502**	(16 * 0.00015) + 13.5 = **13.502**
VN WIFI	0.0139	1.5	-	-	-
AP	0.0053	4.8	-	(3 * 0.0053) + 4.8 = **4.837 * 4**† **= 9.6** †4 APs	(4 * 0.0053) + 4.8 = **4.837 * 4 = 9.6**
Total Networking Power (W) →				**57.7**	**57.7**
MILP Result →				**57.7W**	
Total Power Consumption (W)				**145.55**	**154.88**
MILP Result →				**145.55 W → refer to Figure *4.8***	

Table 4.6 Analytic verification of the optimal choice in Scenario 1 with one zone network.

Checkpoint 2								
Architecture#1: one zone with four clusters **[CFVA-L (SA)]** **Scenario 1:** one task generated from one cluster, with 3000 MIPS and 3Mb/s								
All possible solutions →				**1**	**2**	**3**	**4**	**5**
Allocation decision →				**CC**	**MF**	**LF**	**NF**	**VN**
Processing Nodes				**Processing Power (W)**				
PN	**W/$_{MIPS}$**	**Idle (W)**	**PUE**					
CC	0.00032	69	1.1	[(3000 * 0.00032) + 69] * 1.1 = **76.956**				
MF	0.00039	51	1.4		[(3000 * 0.00039) + 51] * 1.4 = **73.038**			
LF	0.00063	51	1.5			[(3000 * 0.00063) + 51] * 1.5 = **79.335**		
NF	0.001	9	-				(3000 * 0.001) + 9 = **12**	
VN	0.00125	-	-					3000 * 0.00125 = **3.75**
Total Processing Power (W) →				**76.956**	**73.038**	**79.335**	**12**	**3.75**
MILP Result →								**3.75 W**
Network Devices				**Networking Power (W)**				
	W/$_{Mb/s}$	**Idle (W)**	**PUE**					
CC Switch	0.00008	24.8	1.1	[(3 * 0.00008) + 24.8] *1.1= **27.28**				-
CC Router	0.00006	1.35	1.1	[(3 * 0.00006) + 1.35] *1.1= **1.485**				-
Core Router	0.0016	34.5	1.5	[(3 * 0.0016) + 34.5] *1.5= **51.757**				-
Metro Router	0.00006	1.35	1.5	[(3 * 0.00006) + 1.35] *1.5= **2.025**				-
MF Switch	0.00012	13.23	1.4	-	[(3 * 0.00012) + 13.23] *1.4= **18.52**			-
MF Router	0.00003	0.7	1.4	-	[(3 * 0.00003) + 0.7] *1.5= **1.05**			-
Metro Switch	0.00003	27	1.5	[(3 * 0.00003) + 27] *1.5= **40.5**	[(3 * 0.00003) + 27] *1.5= **40.5**			-
LF Switch	0.00012	13.23	1.5			[(3 * 0.00012) + 13.23] *1.5= **19.85**		
LF Router	0.00003	0.7	1.5			[(3 * 0.00003) + 0.7] * 1.5 = **1.501**		
OLT	0.000003	2.7	1.5	[(3 * 0.000003) + 2.7] * 1.5 = **4.05**	[(3 * 0.000003) + 2.7] * 1.5 = **4.05**	[(3 * 0.000003) + 2.7] * 1.5 = **4.05**		
ONU	0.00015	13.5	-	(3 * 0.00015) + 13.5 = **13.5005**	(3 * 0.00015) + 13.5 = **13.5005**	(3 * 0.00015) + 13.5 = **13.5005**	(3 * 0.00015) + 13.5 = **13.5005**	
VN WIFI	0.0139	1.5	-	-				(3 * 0.0139) + 1.5 = **1.5417**
AP	0.0053	4.8	-	(3 * 0.0053) + 4.8 = **4.8159**	(3 * 0.0053) + 4.8 = **4.8159**	(3 * 0.0053) + 4.8 = **4.8159**	(3 * 0.0053) + 4.8 = **4.8159**	(3 * 0.0053) + 4.8 = **4.8159**
Total Networking Power (W) →				**145.4**	**82.4**	**43.7**	**18.3**	**6.36**
MILP Result →								**6.36 W**
Total Power Consumption (W)				**145.55**	**155.4**	**123**	**30.3**	**10.11**
MILP Result →				**10.11 W → refer to Figure *4.5***				

4.8 Summary

In this chapter, we have investigated the processing allocation optimisation problem in vehicular edge cloud integrated with central cloud and distributed fog processors. This architecture was evaluated through a MILP optimisation model to minimise the total power consumption. The evaluation considered multiple cases to study the impact of workload volume, task generation density, vehicles density, and task allocation strategy (single and distributed). Two architectural designs were evaluated with a single or multiple VEC clusters. The results of the investigation showed that vehicles with enough capacity turn out to be a very attractive option for processing the generated workload and saving power. As a result, a power savings up to 70% is achieved by allocating processing to the vehicles. This percentage varied based on the assessed scenario. Moreover, splitting the tasks between multiple vehicles achieved potentially power saving compared to the scenario with single allocation. The overhead power (idle power and PUE) of each processing servers is a key factor affecting the allocation decision. Accordingly, with high generated tasks associated with high required demands, the central cloud becomes more efficient. It was also shown that expanding the access layer with multiple ONUs has minor effect on the allocation decisions, as the local VEC is always more efficient than the non-local VEC.

The evaluated scenarios in this chapter represented idealistic situations in which all vehicles are assumed to have all the required software packages to process the generated tasks. In the next chapter, we will investigate a realistic

situation where not all vehicles have all the required software. Moreover, this realistic evaluation will consider the assumption that tasks will be generated based on the popularity of each software package.

Chapter 5
Software Matching in Vehicular Edge Clouds

5.1 Introduction

Smart vehicles, with their underutilised resources, can be used as an Infrastructure as a Service (IaaS) platform to provide different services. Computation as a Service (CompaaS), presented in Chapter 4, is one of the main services needed to build a conventional cloud-like system out of the underutilised computational capabilities of vehicles. In the main model (developed in Chapter 4), an 'idealistic' assumption was made that all the available vehicles are already provided with all types of possible IoT data processing software packages. Therefore, these vehicles are assumed to process all types of generated tasks. However, this assumption is not a reality, as these vehicles may come with limited processing software packages capability compared to a conventional data centre. Moreover, the software packages may generally be factory-fitted and limited [102]. In this chapter, we take the model developed in Chapter 4 a step further, towards a realistic situation where the required software packages needed to serve certain tasks may not always be available in the clustered vehicles. The aim of this chapter is to emphasize the optimum processing allocation results from the original idealistic model, in Chapter 4, with a realistic aspect (vehicles heterogeneity). One level of heterogeneity is assumed where not all vehicles have the same processors that support same software packages. Moreover, this chapter evaluates different situations of packages availability among all participating vehicles (VEC) through considering different distribution of the software

availability. This is to highlight the effect of possible criteria of clustering vehicles based on the available software packages.

We assessed the impact of the availability of software packages in vehicular edge clouds (VEC) on the processing allocation, the number of tasks allocated to the VEC and the overall power consumption. The evaluated problem is summarised as follows:

1. Certain vehicles have their own particular software packages, and we will therefore need to optimise the processing allocation accordingly, by matching the software required by a task to the available software in the vehicular nodes (VN), as illustrated in Figure 5.1, in order to minimise the total power consumption.
2. With the limited availability of software in the available vehicles, we will optimise the allocation of the software packages in these vehicles and optimise the processing allocation, accordingly.
3. Demands are generated based on the popularity of the software packages. The popularity is assumed to follow a Zipf distribution, which is typical [103]. Based on this, we will study the effect of this popularity on the software and processing allocation in the vehicles.

The evaluation was carried out by examining two possible sets of scenarios. The first set (three scenarios) considered a situation where each VN came with one or more pre-allocated software packages in its processor attached memory. In the second set of scenarios, each VN is allocated the required software along with the task allocation. More details about the evaluation scenarios and the considered situations is given in detail in Section 5.4.

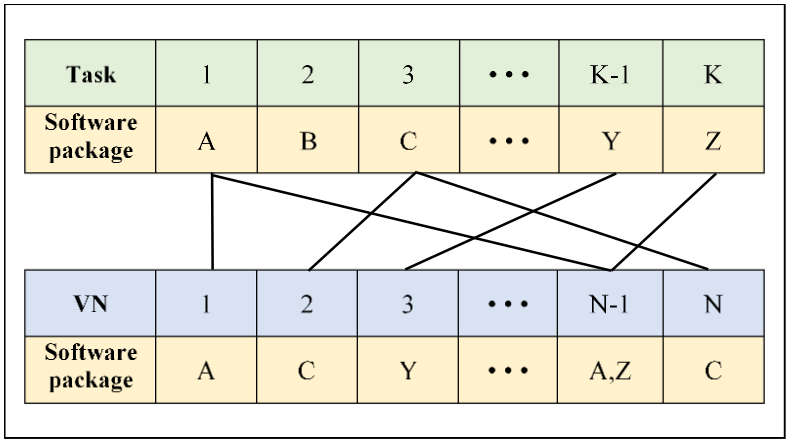

Figure 5.1 Software matching illustration

5.2 Energy Efficient Software Matching in vehicular edge clouds

In this model, the architecture used is shown in Figure 5.2. The architecture consists of one central cloud (CC) and one vehicular edge cloud (VEC). No fog processing nodes were considered in the metro or access layer, in order to reduce the time needed to run the MILP and to focus on the allocation between the central cloud and the distributed vehicles, which are the two ends where processing tasks can be allocated. As CC provides a reliable support for the considered architecture, it hosts all the required software packages. The VEC cluster has varying number of vehicular nodes (VN) which ranged between 8 and 20 VNs in the settings we considered. Ten different software packages are assumed, each with a popularity calculated using Zipf distribution, with popularities between 34% and 3%. As the tasks, in reality, might be generated from different source nodes, we assume that each SN works for a specific purpose and therefore, requires a certain software package. The downloaded version of this software is available in the SN,

which has the ability to send this downloaded package to any nearby VN to accommodate the required tasks of this SN.

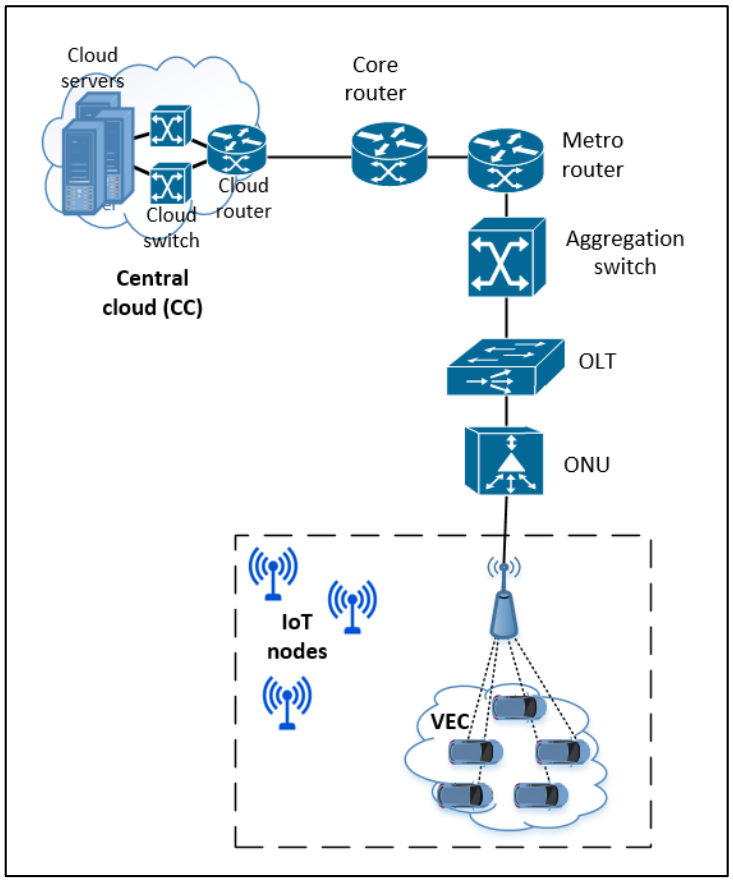

Figure 5.2 Cloud-VEC based architecture for software matching model

5.3 MILP Model

In order to evaluate the software matching problem, the MILP model introduced in Chapter 4 is considered. The model is modified to include the additional sets and parameters that need to be incorporated to describe the software matching problem.

The additional sets are as follows:

SW Set of software packages.

The following parameters are introduced as:

Ψ_{di}^{VN} Indicator of the availability of software package i, $i \in SW$, in vehicular node $d \in VN$. $\Psi_{di}^{VN} = 1$ if the software package i, $i \in SW$, is available in vehicular node $d \in VN$, it is 0 otherwise.

Ψ_{si}^{SN} Indicator of source node $s \in SN$, requiring software package i, $i \in SW$. $\Psi_{si}^{SN} = 1$ if source node $s \in SN$ requires software $i \in SW$, it is 0 otherwise.

k Maximum number of software packages allowed in each vehicular node.

r_i Maximum number of vehicular nodes hosting each software packages $i \in SW$ (Software replicas).

Φ_i Data rate needed to download software package $i \in SW$.

The total power consumption required to download the software package in the AP and the VN communication adapter is calculated as follows:

$$DPC^{A} = \left[\sum_{i \in SW} \sum_{j \in A} \sum_{d \in VN} \Psi_i . \Psi_{di}^{VN} . \Phi_i . \left(\frac{P_{max}^{A} - P_{idle}^{A}}{\Omega_{max}^{A}}\right)\right] \quad (5.1)$$

$$DPC^{VN} = \left[\sum_{i \in SW} \sum_{d \in VN} \Psi_{di}^{VN} . \Phi_i . \left(\frac{P_{max}^{VW} - P_{idle}^{VW}}{\Omega_{max}^{VW}}\right)\right] \quad (5.2)$$

Equation (5.1) calculates the power consumed by the AP to download and forward the required software from the SN to VN, where Φ_i is the required data rate to download software i. Equation (5.2) calculates the power consumed by the VN downloading the software package i.

Objective:

To minimise the total power consumption of all the processing nodes and their networks, and the infrastructure network devices, as explained in Chapter 4,

including the two power consumption components calculated in Equation (5.1) and (5.2), given as:

Minimise:

$$TPC^{CC} + TPC^{MF} + TPC^{LF} + TPC^{NF} + TPC^{VN} + TPC^{NET} + DPC^{A} + DPC^{VN} \quad (5.3)$$

where $TPC^{CC}, TPC^{MF}, TPC^{LF}, TPC^{NF}, TPC^{VN}$ and TPC^{NET} are the total power consumption of CC, MF, LF, NF, VN and the network infrastructure, respectively. DPC^{A} and DPC^{VN} are the power consumption attributed to downloading the software packages in AP and VN, respectively.

The objective function is subject to all of the constraints introduced in Chapter 4, and the following additional constraints:

1) Task allocation based on software matching:

$$\delta_{sd} \leq \sum_{i \in SW} \Psi_{si}^{SN} \cdot \Psi_{di}^{VN} \qquad \forall\ s \in SN, d \in VN. \quad (5.4)$$

Constraint (5.4) ensures that tasks generated from source node s, which require software package i, can be allocated at vehicular node d, if d hosts the required software i. The binary variable δ_{sd}, is set to zero, if the source node does not require the software package i, and/or the processing node does not have the software package for all required software packages. Moreover, δ_{sd} in Equation (4.25) and (4.26) in Chapter 4, is set to one, as appropriate.

2) The number of software packages allowed in each VN:

$$\sum_{i \in SW} \Psi_{di}^{VN} \leq k \qquad \forall\ d \in VN \quad (5.5)$$

Constraint (5.5) defines the different cases where the maximum number of allowed software packages per VN ranges from 0 to 10 software packages per VN, where k is the maximum number of hosted software.

3) The number of VNs hosting each software package (Scenario 3):

$$\sum_{d \in VN} \Psi_{di}^{VN} \leq r_i \qquad \forall \ i \in SW \tag{5.6}$$

Constraint (5.6) defines the maximum number of VNs that can host each software package. Note that this constraint is defined only in Scenario 3, where the value of r_i is given. Scenario 4 does not consider this constraint as the parameter Ψ_{di}^{VN} becomes an optimisation variable.

5.4 Power Consumption and Software Matching Scenarios

This section discusses the evaluated scenarios in the energy efficient software matching optimisation, using the architecture illustrated in Figure 5.2. Four scenarios were assessed based on the availability of the software replicas in the available VNs, as explained next in Section. The results are summarised in Section 5.4.2. All of the four scenarios were assessed with different densities of VNs (8, 10, and 20 VNs) and with both of the allocation strategies that were considered in Chapter 4 (single and distributed). Three scenarios were conducted where we assume that the participant vehicles have already one or more software packages installed in their processor, based on vehicle manufacture or user-based software preference. In these scenarios, installing any software packages prior allocating tasks is not possible. If the available vehicles don't have the required package to process the tasks, these tasks

will be allocated to the central cloud which fulfils all the required software types. In the fourth scenario, the vehicles are assumed to have no software packages and, hence, cannot process the generated tasks immediately. However, these vehicles are able to communicate with the source node to download any required packages and ,therefore, process the their associated tasks.

As explained in the previous section, the tasks generated are assumed to follow Zipf distribution, based on ten potential types of software packages. To ensure that all the software packages considered are required in the total of the generated tasks, we fixed the number of tasks to 25, each generated with 1000 MIPS, 1 Mb/s and a software requirement based on the software type popularity. Table 5.1 summarises the software package required for each generated task, based on Zipf distribution.

Table 5.1 Popularity percentage and generated tasks for each software package (for a total of 25 tasks).

Software package ID	A	B	C	D	E	F	G	H	I	J
Popularity percentage	34%	17%	11%	9%	7%	6%	5%	4%	4%	3%
Number of generated tasks	9	4	3	2	2	1	1	1	1	1

The optimisation results allocated the generated tasks to the available vehicular nodes based on three factors: a matching scheme between the task's required software and the software compatible VNs, the VNs processors' capacities and the communication medium capacity. To evaluate the energy efficient software matching model, four scenarios were evaluated,

each scenario had different cases where the number of allowed software packages in each VN ranged from 0 to 10 software per VN.

In Scenario 1, each VN was assumed to have a deterministic (pre-allocated) software package that served one or more of the considered software packages. This assumption was based on the situation that VNs come with an advanced processor that can handle many different IoT applications available in smart cities; therefore, no software downloading was considered for the available VNs. The number of allowed packages per VNs varied between zero and ten software packages per VN (SW/VN). The availability of the software replicas in the clustered VNs was assumed to have a uniform distribution; thus, with 1 SW/VN, one copy of a software was available in each VN, and with 2 SW/VN, two copies of the software were available in each VN, and so on. Accordingly, all the software packages considered were available in VEC, in each case. It is worth mentioning that uniform placement of software packages is not ideal given that the demands follow a heavy tail Zipf distribution.

In Scenario 2, the same details as Scenario 1 are applied, except that the availability of the software replicas in the clustered VNs was assumed to have a random and uniform distribution. We considered such a scenario as a more realistic situation of the available VNs distributed with specific available software. Therefore, this distribution was worse than the uniform distribution in Scenario 1, as some software packages may be missing in the available vehicles, and therefore, in the VEC. Ideally, this distribution should be an

average over many runs, however, the difference between this scenario and scenario 1 was small. Therefore, a single realisation is reported.

Scenario 3 was similar to the first scenario. The software available in each VN remained deterministic as an input to the model. However, the number of VNs hosting each software package was calculated based on Zipf distribution. The number of allocated software replicas was added as a constraint to the model. In this scenario, we wanted to evaluate an ideal case where these VNs with their pre-allocated software were clustered based on the software popularity to serve the requested tasks.

On the contrary to scenarios 1, 2, and 3, in Scenario 4, the number of VNs hosting each software package was optimised. Therefore, the model allocated each software in the available VNs and optimised the tasks allocation based on this decision. This scenario was applied to a case where the architecture controlled (AP) had no pre-knowledge of the software popularity, and therefore, an instant knowledge was created based on the collected tasks. In this case, the downloading overhead was introduced as the power consumed due to downloading the software package from the source node to the hosting VN, through the AP. This assumption accounts for situations where the vehicle participating in the edge cloud lacks the required package to serve certain tasks using the required software.

5.5 Power Consumption and Software Matching Results

This section discusses the results of the energy efficient software matching optimisation; the effect of VN density; software popularity; and the number of

software replicas in the available vehicular nodes, using the scenarios described in the previous section.

5.5.1 Scenario 1: Software Matching with Software Packages Based on Uniform Distribution

Figures 5.3 and 5.4 show the total power consumption and the percentage of allocated processing in VEC, for single and distributed allocation strategies, respectively. All the results are represented for cases with 8, 10, and 20 available VNs, and with different sets of software in each VN, based on uniform distribution. In addition, a certain number of available packages per VN was considered. This number varied between zero software per VN (0SW/NV) and ten software packages per VN (10SW/VN). The case of 0SW/VN represents a baseline case where all tasks are allocated to the central cloud (CC), as vehicular nodes have no pre-allocated software packages in their on-board unit (OBU), and therefore, cannot serve the demands presented by the tasks. On the other hand, 10SW/VN represents a case where smart vehicles with advanced OBU and many pre-allocated software packages are present and can meet the software requirements of the tasks .

It was observed that for the single allocation (Figure 5.3) with a low and medium number of vehicular nodes (8–10VNs) and a low number of available packages per VN (1–3SW/VNs), all the tasks were allocated to the CC. Although VEC capacity with 8VNs and 10VNs was enough to allocate 96% and 100% of the generated tasks, respectively, the limited software replicas in these VNs limited the allocation to the VNs with the required software packages. Based on the results in Chapter 4, it was noted that allocating all

tasks to one processing location was more efficient. This was observed in the results, as activating the CC consumes a high network power due to switching all the devices in the route to the CC. Therefore, avoiding activation of the VN connection for partial allocation saves power, and is the reason that all the tasks are allocated to the CC, despite the VEC being able to serve part of these tasks. For the case with 8VNs, the VEC was not efficient enough to allocate the generated tasks, regardless of the increased number of available software replicas in each VN. However, with 10VNs, increasing the available package replicas to 4SW/VN was enough to accommodate all generated tasks in order to satisfy the required software. When increasing the number of vehicles to 20 VNs, the VEC becomes rich with resources and variety of software. Therefore, VEC was large enough to satisfy all the generated tasks, with only 2SW/VN in each vehicle.

The same results were observed for the distributed allocation strategy (Figure 5.4), with 10 and 20 available VNs. For the 8 VNs, the splitting flexibility introduced in the distribution strategy achieved better power efficiency, as VEC was able to allocate all the generated tasks by having at least 6SW/VNs, due to the variety of available packages per vehicular node, and the ability to bin-pack the split tasks among the remaining processing capacity of VNs.

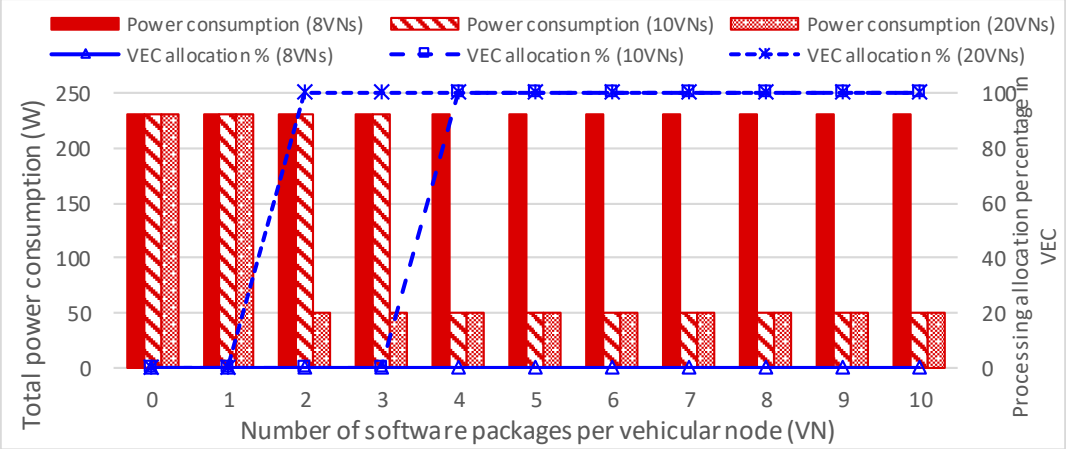

Figure 5.3 Total power consumption and VEC processing allocation percentage in single allocation strategy with uniformaly distributed software packages over the available VNs (Scenario 1)

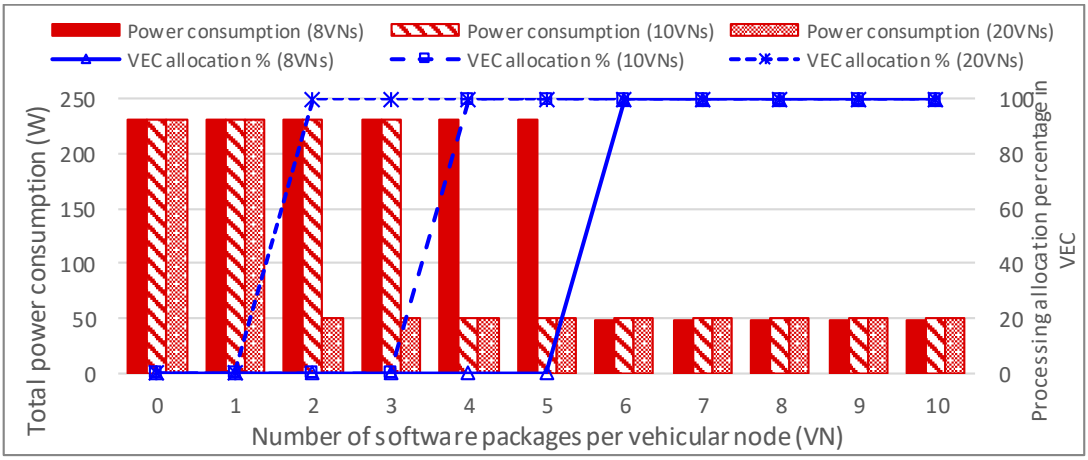

Figure 5.4 Total power consumption and VEC processing allocation percentage in distributed allocation strategy with uniformly distributed software packages over the available VNs (Scenario1)

As noted earlier (Table 5.1), the generated tasks followed a Zipf popularity distribution, however, in this scenario, the software packages available in the VNs followed a uniform distribution. Table 5.2 lists the number of VNs utilised and the number of individual replicas used in the VEC, for both single and distributed allocation strategies. This table helps to analyse how the number of available software packages in each VN can affect the allocation decision, and if the allocation strategy has an effect on the number of software packages

used in each VN, as well as the total number of replicas used among the VEC. As the available VNs were hosting pre-allocated software packages, there was no download overhead caused by using the available software replicas. This allowed the model to use many replicas to satisfy the generated tasks, as long as the minimum VNs were utilised and no extra VN was used by the extra software replicas. We can also observe that the number of replicas used increased in the distributed allocation strategy, but at the same time, the number of utilised VNs decreased, as the splitting allowed the model to use more replicas in each VN, by allocating less processing. It is worth mentioning that in the single allocation, the total number of used replicas cannot exceed 27 replicas in VEC, due to the limited processing capacity of each VN (3200 MIPS). As a consequence, and with the task requirement considered (1000 MIPS), no more than three tasks with different software requirements can be allocated to any VN. Therefore, no more than three software replicas can be used in each VN, regardless of the number of available packages. This explains the increase in the number of replicas used in the distributed allocation strategy, as some VNs were allocated tasks belonging to more than 3 software packages.

Table 5.2 Number of VNs utilised and software replicas used in single and distributed allocation strategies (Scenario 1).

Number of available VNs	Single Allocation		Distributed Allocation	
	Number of VNs utilised	Number of individual software replicas used in VEC	Number of VNs utilised	Number of individual software replicas used in VEC
8 VNs	0	0	8	15–23
10 VNs	9	14–20	8	18–24
20 VNs	9	14–17	8	15–30

As part of the evaluation, we wanted to test if the processing allocation, based on task software types and the available packages in VNs, would follow the same distribution. In other words, we wanted to test if the popularity of generated tasks was reflected in the number of software replicas used in the available VNs, after the optimisation. Figures 5.5 and 5.6 show how the processing allocation and the replicas used affected the available VNs. We observed that software popularity was not always reflected in the number of each software replicas used. The results circled in red show some cases where a more popular software was represented by a fewer number of replicas in the available VNs. For instance, in the single allocation strategy (Figure 5.5) with 10VNs and 5SW/VN, 6% of the software replicas used belonged to software D, while 12% belonged to software E, which was less popular than D. Most of these cases existed in the distributed allocation strategy (Figure 5.6), as the optimisation splitted more tasks and used more replicas, in order to use less VNs, regardless of the popularity of the software required by the tasks in order to save power consumption.

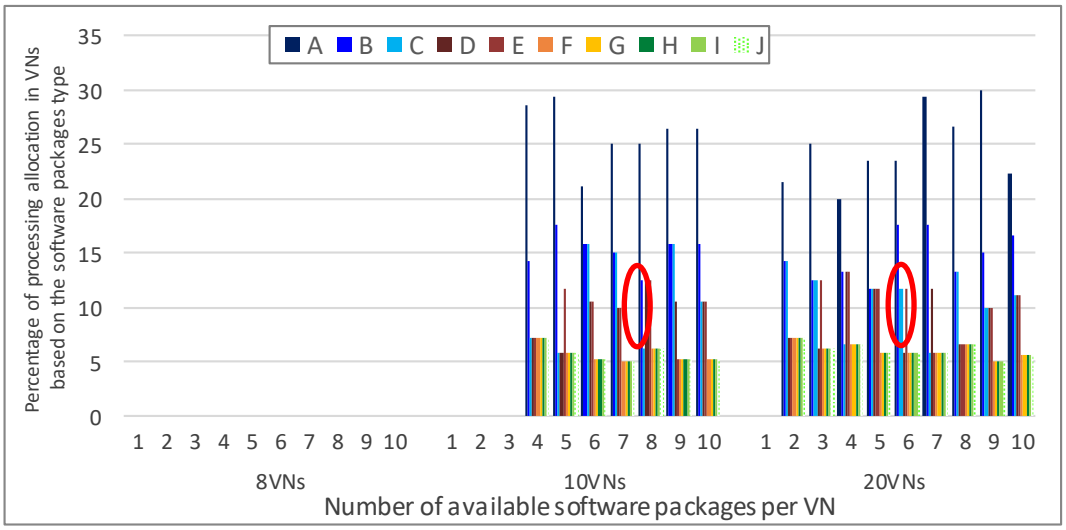

Figure 5.5 processing allocation based on software packages in single allocation strategy (Scenario 1).

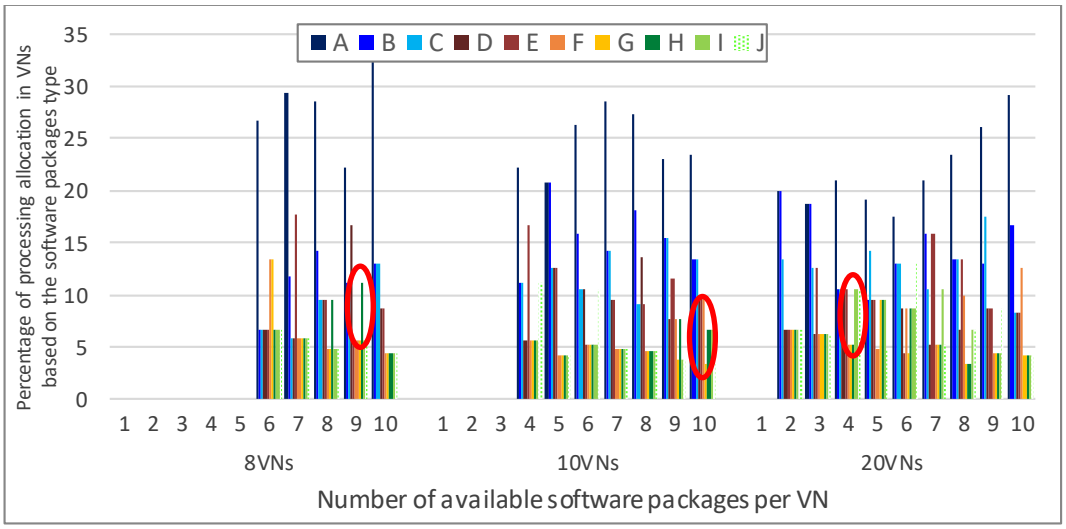

Figure 5.6 processing allocation based on software packages in distributed allocation strategy (Scenario 1).

5.5.2 Scenario 2: Software Matching with Pre-Allocated Software Packages Based on Random and Uniform Distribution

In this scenario, software packages were randomly distributed over the available VNs. The total power consumption and the VEC utilisation for the single allocation, shown in Figure 5.7, have the same results as in Scenario 1. This shows that regardless of how the software packages were distributed in the available VNs, 8 VNs did not have enough variety of software packages to absorb all of the tasks. Moreover, the 10–20 VNs case was not able to serve these with the low number of available software per VN (less than 4SW/VN and 2SW/VN with 10 and 20 VN, respectively). With a high number of SW/VN, the variety of software per VN enabled full task allocation in the VN, again, regardless of the software distribution. In the distributed allocation, Figure 5.8 shows a similar result to scenario 1, except for the case of 5SW/VN. With the uniform distribution, 5SW/VN was enough to allocate all of tasks to the VEC. On the other hand, with random and uniform distribution (this scenario),

5SW/VN was not enough, and all the tasks were allocated to the CC, because with random and uniform distribution, some replicas could be available more than others. If it happened that these replicas belonged to high popularity software, more tasks would be allocated to the VN and therefore, the VEC utilisation increased.

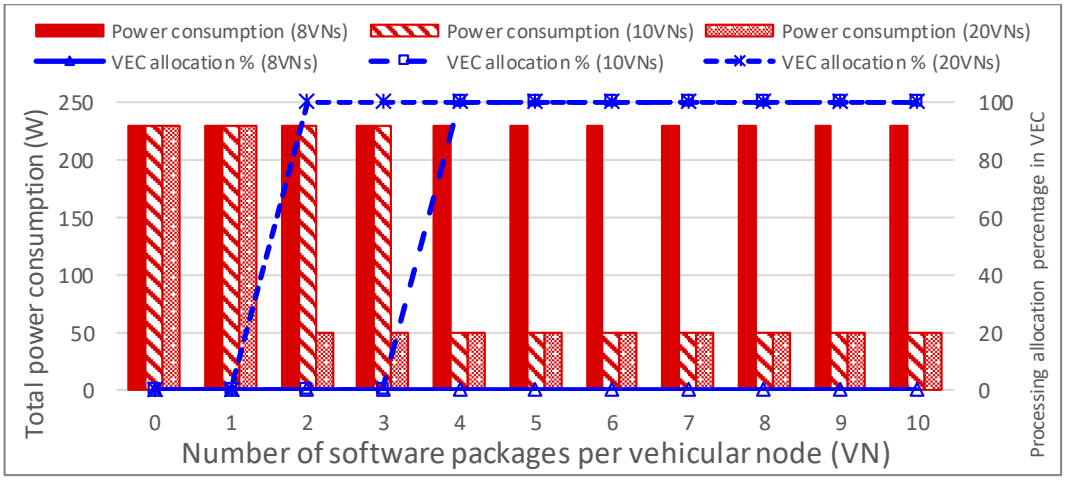

Figure 5.7 Total power consumption and VEC processing allocation percentage in single allocation strategy with randomly distributed software packages over the available VNs (Scenario 2).

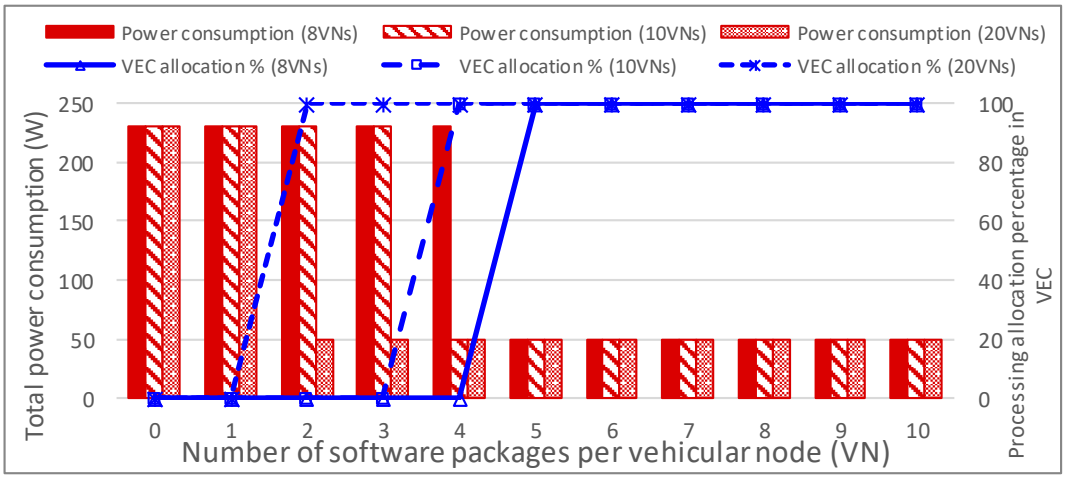

Figure 5.8 Total power consumption and VEC processing allocation percentage in distributed allocation strategy with randomly distributed software packages over the available VNs (Scenario 2).

Table 5.3 shows a different, but a comparable number of used replicas in this scenario, compared to Scenario 1. As mentioned previously, since there was

no extra power overhead to be consumed by using specific replicas in the VN, it did not matter how many replicas were used in each VN, as long as the number of activated VN communication units was minimised. This was confirmed by the similarity of the number of utilised VNs in both scenarios (1 and 2). In addition, Figure 5.9 and 5.10 confirm that the allocation did not necessarily follow the Zipf distribution of the generated software tasks.

Table 5.3 Number of utilisedVNs utilised and software replicas used in single and distributed allocation strategies (Scenario 2).

Number of available VNs	Single Allocation		Distributed Allocation	
	Number of VNs utilised	Number of individual software replicas used in VEC	Number of VNs utilised	Number of individual software replicas used in VEC
8 VNs	0	0	8	17–22
10 VNs	9	16–21	8	17–22
20 VNs	9	15–21	8	15–32

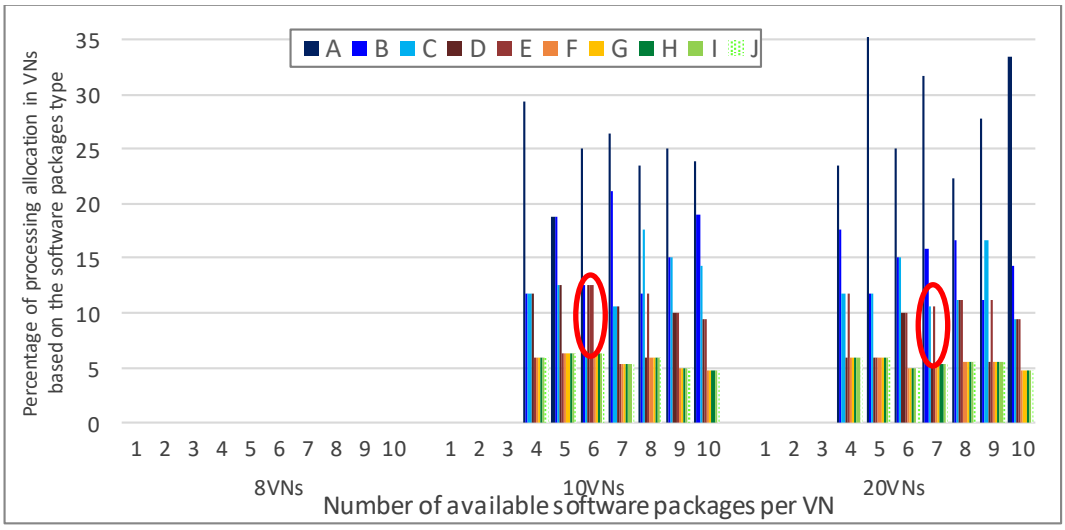

Figure 5.9 processing allocation based on software packages in single allocation strategy (Scenario 2).

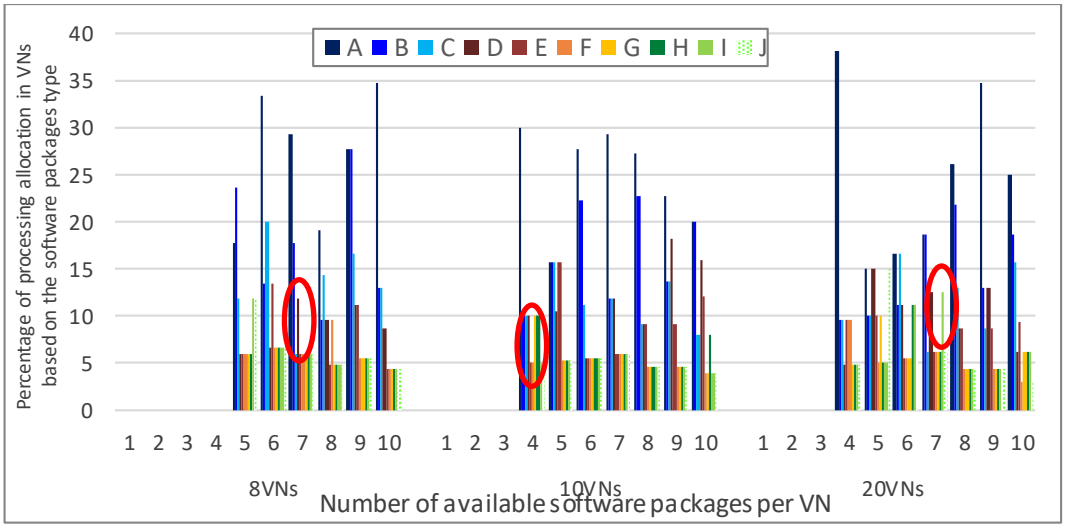

Figure 5.10 processing allocation based on software packages in distributed allocation strategy (Scenario 2).

5.5.3 Scenario 3: Software Matching with Pre-Allocated Software Packages Based on Zipf Distribution.

In this scenario, the software packages were assumed to be pre-allocated in the available VNs, based on Zipf distribution, following the software popularity. Table 5.4 lists the maximum number of VNs that were assumed to host each pre-allocated software package; this number is represented by the parameter r_i in (5.6), and was calculated for each case based on the software popularity, taking into account the following:

1) Each software must be hosted by at least one VN.

2) In the case of 1SE/VN, the total number of replicas cannot exceed the total number of available VNs. This meant that with 8VNs, two software packages would not be hosted by any VN.

3) With the increased number of allowed software per VN, the number of VNs hosting each package also increased, taking into account that no software could be hosted by more than the total number of available VNs. (i.e. the

highest popularity software would be hosted by a maximum of 8, 10, or 20 VNs, based on the total number of available VNs).

Table 5.4 maximum number of VNs hosting each pre-allocated software package.

Software details		Software package	A	B	C	D	E	F	G	H	I	J
		Software popularity	34%	17%	11%	9%	7%	6%	5%	4%	4%	3%
Maximum number of VNs that can host each software package	8VNs	**1** SW/VN	1	1	1	1	1	1	1	1	0	0
		2 SW/VN	4	3	2	1	1	1	1	1	1	1
		3-10 SW/VN	8	4	3	2	2	1	1	1	1	1
	10VNs	**1** SW/VN	1	1	1	1	1	1	1	1	1	1
		2 SW/VN	7	3	2	2	1	1	1	1	1	1
		3-10 SW/VN	10	5	3	3	2	2	2	1	1	1
	20VNs	**1** SW/VN	7	3	2	2	1	1	1	1	1	1
		2 SW/VN	14	7	4	4	3	2	2	2	2	1
		3-10 SW/VN	20	10	7	6	5	4	3	2	2	1

Figures 5.11 and 5.12 show the results of the total power consumption and the processing allocation in the single and distributed allocation strategies. Compared to Scenarios 1 and 2, Scenario 3 showed a general improvement with lower power consumption and improved VEC utilisation, in both allocation strategies. Similar to Scenario 1, Figure 5.11 shows that in the single allocation strategy, the VEC with 8VNs was incapable of accommodating any task, regardless of the increase in the number of software packages per VN. This is attributed to the limited capacity of the VEC and that each VN could serve only three (1000 MIPS) tasks, with a total of 24 tasks that could be served in the VEC. Therefore, all the tasks were sent to the CC as the optimum location. With the number of vehicles increasing to 10 and 20 VNs, all tasks were allocated to the VEC by at least 2SW/VN and 1SW/VN, respectively. Hence, taking into consideration the popularity of the software replicas hosted

by the available VNs can reduce the VEC resources needed in order to satisfy the software demand, and therefore, the generated tasks. Figure 5.12 shows that in the distributed allocation strategy, further improvement was observed with 8 VNs, where tasks were allocated to the VEC with at least two software packages available per VN.

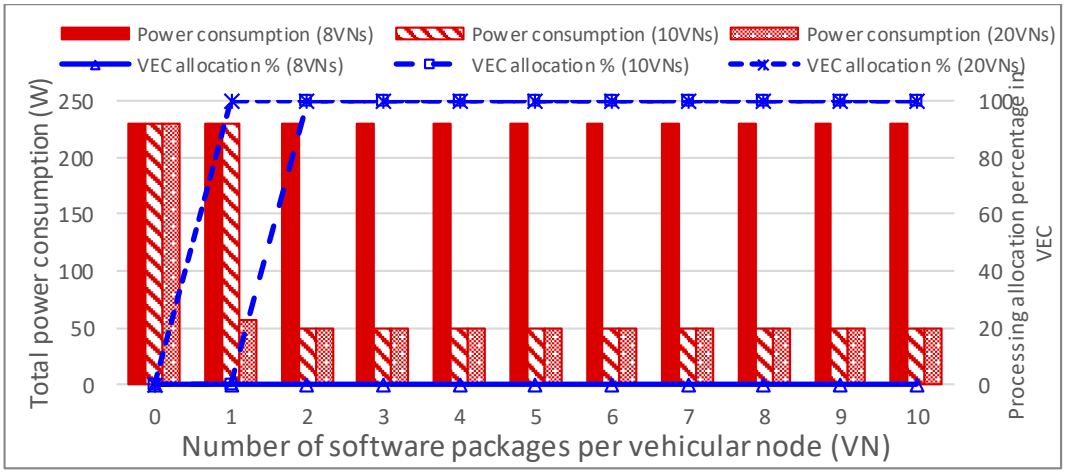

Figure 5.11 Total power consumption and VEC processing allocation percentage in single allocation strategy with Zipf distributed software packages over the available VNs (Scenario 3)

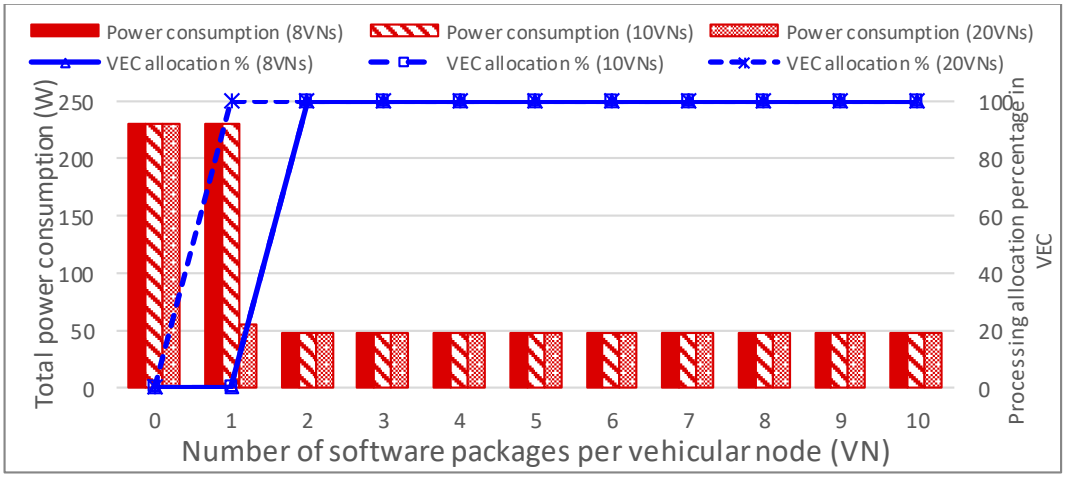

Figure 5.12 Total power consumption and VEC processing allocation percentage in distributed allocation strategy with Zipf distributed software packages over the available VNs (Scenario 3)

Table 5.5 shows that the optimisation results still utilised the same number of VNs in both allocation strategies. However, one of the highlighted improvements in this scenario was utilising the available VNs with the least number of available software in each VN (1SW/VN). Hence, in this case, 13 VNs were utilised (equal to the minimum number of required software replicas) in both the single and distributed allocation strategies. This explains the increase in power consumption with 1SW/VN, observed in Figures 5.13 and 5.14, compared to the other cases where VEC was fully utilised (2–10 SW/VN). As seen previously in Scenario 1, the number of total replicas in the single allocation could not exceed 27 replicas, and this number increased in the distribution allocation for the splitting case.

Similarly, it was observed that the actual allocation did not always follow the popularity. However, the allocation of tasks over 8VNs, followed the popularity for all cases and all software packages.

Table 5.5 Number of VNs utilised and software replicas used in single and distributed allocation strategies (Scenario 3).

Number of available VNs	Single Allocation		Distributed Allocation	
	Number of VNs utilised	Number of individual software replicas used in VEC	Number of VNs utilised	Number of individual software replicas used in VEC
8 VNs	0	0	8	15–24
10 VNs	9	15–21	8	16–24
20 VNs	13 (1SW/VN) 9 (other cases)	13–18	13 (1SW/VN) 8 (other cases)	13–27

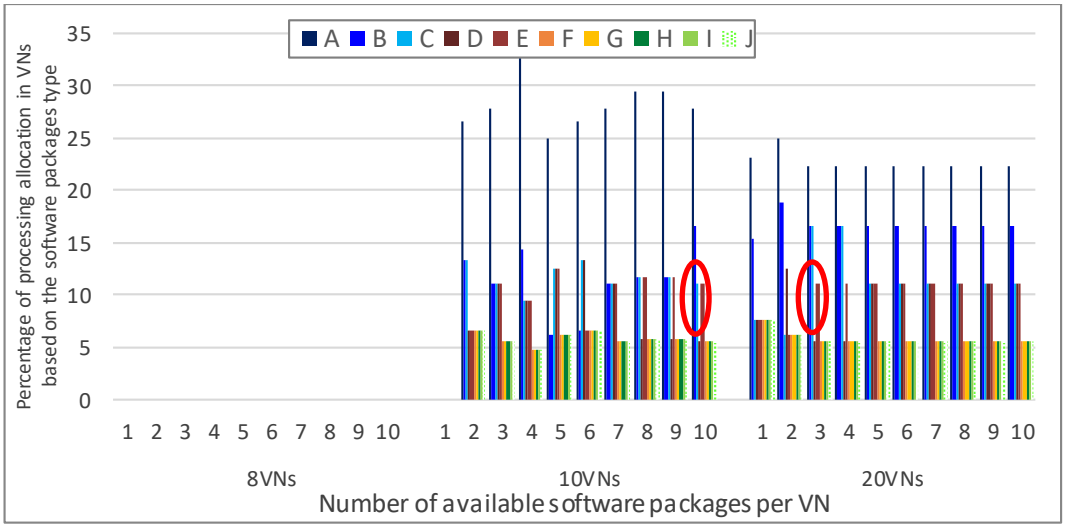

Figure 5.13 processing allocation based on software packages in single allocation strategy (Scenario 3).

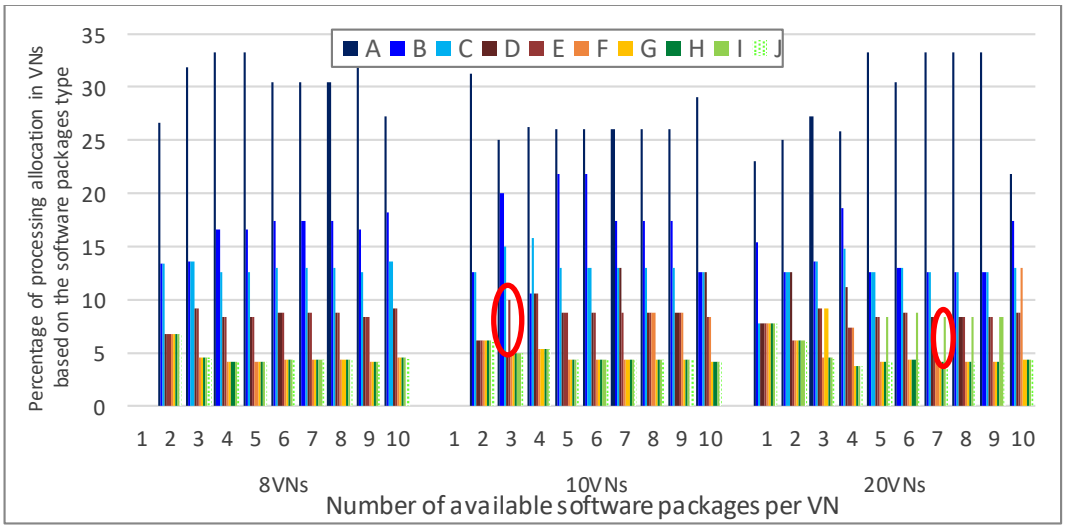

Figure 5.14 processing allocation based on software packages in distributed allocation strategy (Scenario 3).

5.5.4 Scenario 4: Optimised Resource Allocation with Download Overhead Consideration

In Scenario 4, the VNs were assumed to have no pre-allocated software, and the AP had no pre-knowledge of the software popularity. Therefore, an instant knowledge was created based on the collected tasks, and the optimum VN

needed to download the software packages to be able to process the requested demands. Based on these assumptions, instead of having $\boldsymbol{r_i}$ as a predefined number of software replicas in each VN, the parameter Ψ_{di}^{VN} became a decision variable used by the optimisation. Therefore, the optimisation decided how many VNs would download, host each software package and optimise the tasks allocation based on this decision. In this case, the downloading overhead was introduced as the power consumed due to downloading the software package from the source node, and therefore AP, to the utilised VN. All the software packages were assumed to have a size of 10Mb, which was equal to the maximum generated traffic per task in the data set used in Chapter 4. It is worth noting that other cases were tested with a different download size for each package (10–50 Mb). However, it was shown that the size of the download had very minor effect on the allocation decision and therefore, the power consumption. For this reason, we fixed the download size for all packages to 10 Mb.

Figures 5.15 and 5.16 show the results for the single and distributed allocation strategies. Both figures show comparable processing allocation behaviour, compared to Scenarios 1 and 2. However, there was a small increase in the power consumption resulting from the download overhead of the software package. The results suggest that the model followed the software popularity, with the software allocation among the available VNs, and caused the same VEC utilisation for the same cases, as seen in Scenarios 1 and 2.

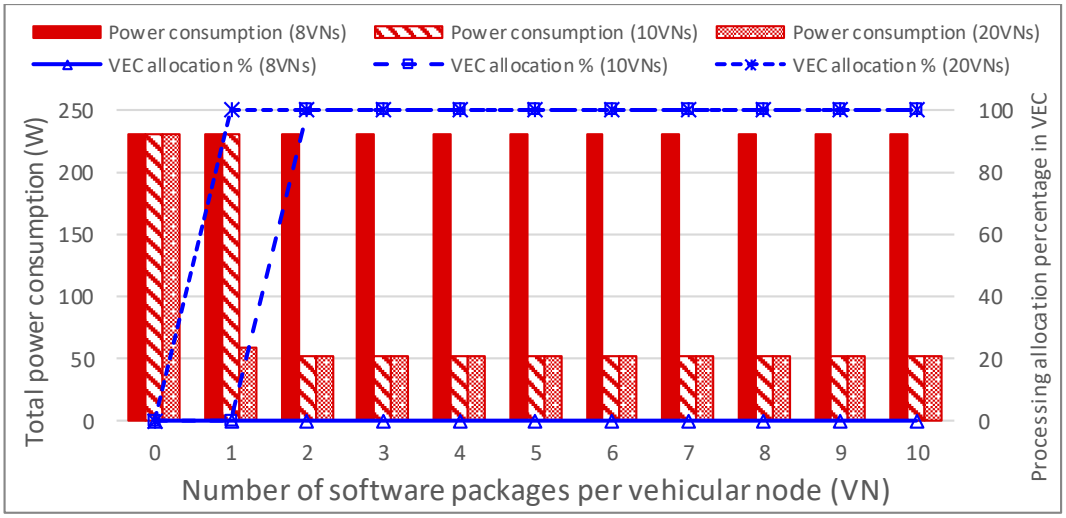

Figure 5.15 Total power consumption and VEC processing allocation percentage in single allocation strategy with optimised number of software packages over the available VNs (Scenario 4).

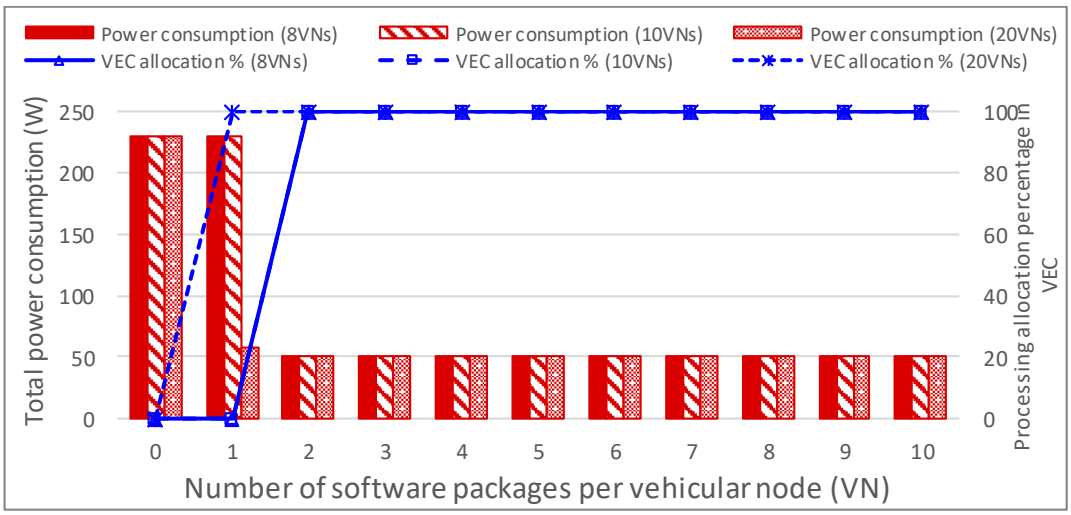

Figure 5.16 Total power consumption and VEC processing allocation percentage in distributed allocation strategy with optimised number of software packages over the available VNs (Scenario 4).

As shown in Table 5.6, the number of software replicas was fixed for all cases, except for 1SW/VN, with 13 and 14 replicas in single and distributed allocation strategies, respectively. This was attributed to the fact that each replica represented a software download to one of the available VNs, hence, the model optimised the number of downloads and the number of VNs utilised in

order to minimise the total power consumption. Therefore, the number of replicas was minimised and remained fixed for all cases.

Table 5.6 Number of VNs utilised and software replicas used in single and distributed allocation strategies (Scenario 4).

Number of available VNs	Single Allocation		Distributed Allocation	
	Number of VNs utilised	Number of individual software replicas used in VEC	Number of VNs utilised	Number of individual software replicas used in VEC
8 VNs	0	0	8	14
10 VNs	9	13	8	14
20 VNs	13 (1SW/VN) 9 (other cases)	13	13 (1SW/VN) 8 (other cases)	13 (1SW/VN) 14 (other cases)

The Software allocation results in Figures 5.17 and 5.18 summarise the software allocation among the available VN. It is shown that, in the single allocation, each software package was downloaded and allocated by the same number of VNs, regardless of the number of allowed software for each VN. It was also observed that all software allocations followed the software popularity of the generated tasks. However, as seen in Figure 5.18, in the distributed allocation strategy, not all software allocations followed the popularity of demands, although these cases were minor compared to the other scenarios. The reason that the model did not follow popularity is that downloading extra replicas of a particular software to more VNs, for the sake of decreasing the number of VNs utilised, causes more power consumption than the power saved by eliminating the use of extra VNs.

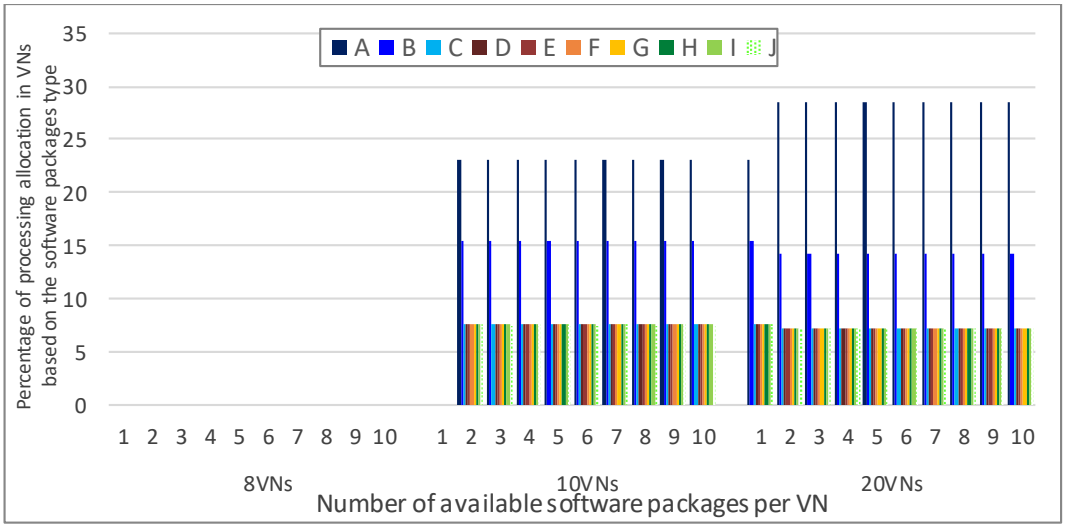

Figure 5.17 processing allocation based on software packages in single allocation strategy (Scenario 4).

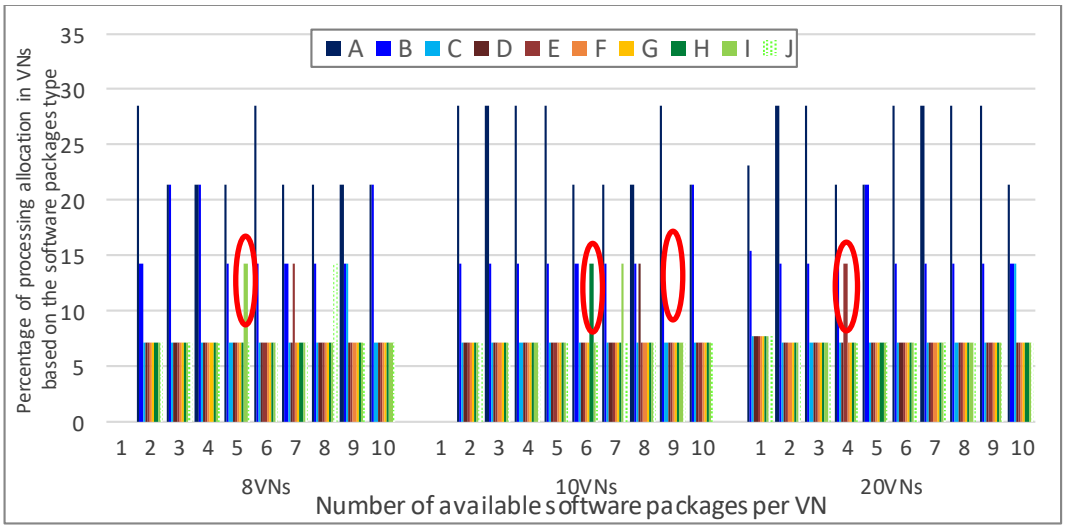

Figure 5.18 processing allocation based on software packages in distributed allocation strategy (Scenario 4).

5.6 Summary

In this chapter, we have investigated a realistic aspect of processing allocation in VEC, by assuming that vehicles do not have all the required software packages to satisfy the work demand. The developed MILP model, described in Chapter 4, was extended to study the software matching problem in VEC,

considering only two layers of processing (at the central cloud and at the vehicular edge cloud). Four scenarios were developed to evaluate three situations. Firstly, each vehicle has its own pre-allocated software packages based on different software package popularity probability distributions (uniform, random, and Zipf distributions). Secondly, vehicles have no pre-allocated software packages in their OBU, but the optimisation determines the best software to install in each node. Thirdly, each software type has a given request probability based on its popularity, where the popularities follow a Zipf Distribution. Accordingly, and depending on the software popularity, processing tasks were generated with the requested software type. The software matching problem was evaluated in terms of power consumption and popularity of the allocated tasks for multiple vehicle densities and with single and distributed strategies. This study has found that in general, increasing the number of vehicles increases the number of software replicas, and therefore this increases the allocation of tasks in VEC. It was also shown that with pre-allocated software packages, the number of replicas used does not affect the power or the allocation decisions. On the other hand, by optimising the software allocation, the optimisation minimises the number of installed replicas in each vehicle due to the download overhead.

The next chapter will investigate another realistic aspect of the proposed model to optimise the delay alongside the p

ower consumption for the cloud-fog-VEC architecture.

Chapter 6
Energy Efficient and Delay Aware Vehicular Edge Cloud

6.1 Introduction

One of the promising advantages of fog and edge computing is their ability to reduce the delay experienced by traffic and processing tasks in general, by placing tasks at processing units close to the end-user and thus, improving the quality of the delivered services. With the rapid increase in the number of connected devices in the edge layer and the exponential rise in real-time based traffic, research needs to focus on optimising processing and routing in cloud-fog architectures to minimise delay.

In the previous chapters, we have considered minimising the energy consumption of processing and networking by optimising the allocation of the processing resources in the cloud-fog-VEC architecture. In this chapter, we study the trade-off between power consumption and delay. We consider queuing delay at different network nodes as the traffic in the network nodes is routed via multi-hop routes and is affected by the service rate of each node. In addition to queuing delay at the different network nodes, propagation delay needs to be taken into account as the considered end-to-end architecture covers a large geographical area from the edge to the core. We optimise the allocation of the processing resources in a multi-objective MILP optimisation model to minimise the power consumption and delay equally.

6.2 Propagation Delay and Queuing Delay Calculation

We consider the architecture shown in Figure 6.1 with processing resources at central cloud (CC), metro fog (MF), OLT fog (LF), ONU fog (NF) and

vehicular edge cloud (VEC). The edge layer consists of only one zone, with one VEC cluster and multiple source nodes.

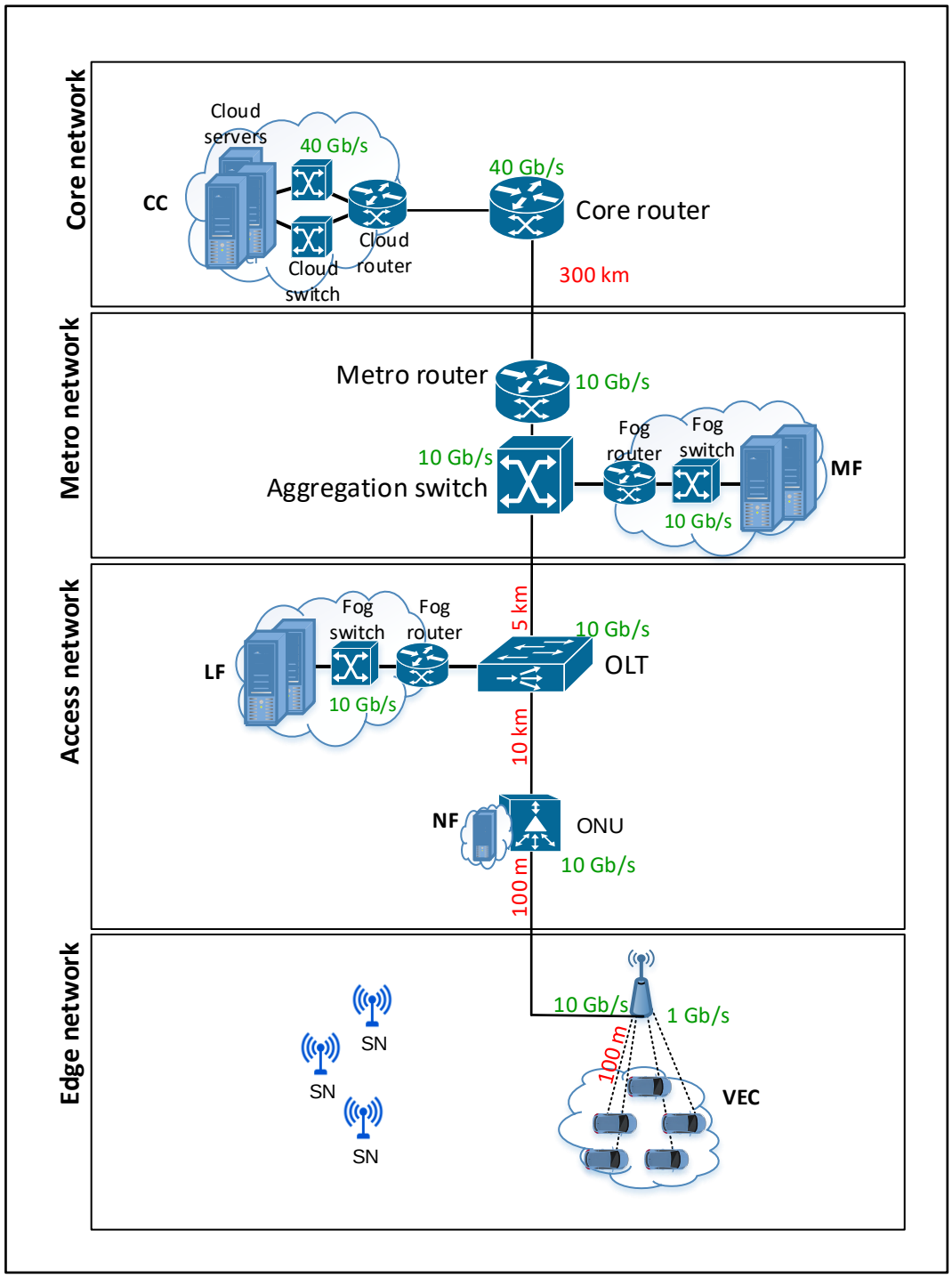

Figure 6.1 The Cloud-Fog-VEC architecture, with estimated distances (in red) and network devices service rate (in green).

We study the propagation and queuing delay. The distances illustrated in Figure 6.1 (coloured in red) are used to estimate the propagation delay. The distances are based on the following assumptions:

1. The distance between the AP and surrounding VNs is set to the standard coverage range of IEEE 802.11, namely 100 m [104].
2. The distance between the AP and ONU is estimated based on an assumption that one ONU can connect to multiple APs in a wired local area network (LAN) with a maximum of 100 m distance [105].
3. The distance between the ONU and OLT is based on typical PON designs in the field [106]. Here, we considered a design where the OLT is located in the telecom main office in the centre of the city. ONUs usually represent devices located at the end-users location (i.e., at home); usually such distance are around 5–20 km [106] so we assumed a distance equal to 10 km.
4. The distance between the OLT and metro node (router and switch) was estimated based on the metro network design. The metro network usually has a radius of 20–120 km [107]. The OLT can be either collocated with the metro node in the same telecom office or located somewhere else in the local area of the metro node (1–10 km away). We based our estimation on the latter scenario with an approximate distance equal to 5 km between the OLT and metro node. The MF node here is assumed to be located within few kilometres of the metro node. Same distance is assumed to exist between OLT and LF.
5. The distance between the metro node and the core node (including the associated CC), is given as the distance between two large cities,

assuming the current city does not have a large central cloud. An example of such a distance is taken as the distance between Leeds and a large data centre in London, a 300 km distance.

The queuing delay was modelled for each networking node as an M/M/1 queue with one server, where arrivals follow a Poisson process and the service rate is negative exponentially distributed, summarised in Figure 6.2.

The propagation delay was calculated for each network location. The queueing delay was calculated based on traffic and the capacity of the nodes. Here the aggregated traffic delivered and the node or device maximum service rate defined the arrival rate and service rate, respectively, and are used to determine the queueing delay as given below

$$Queuing\ Delay = \frac{1}{\mu - \lambda} \tag{6.1}$$

where μ is the service rate, and λ is the arrival rate (summation of the traffic delivered assuming a Poisson process) in the network device. We have considered in this work delay at the packet level. We used the Ethernet maximum packet size of 1500 bytes and therefore expressed the arrival data rates as packets per second and expressed the service rates (transmission rates) in packets per second.

It is worth noting that, based on our considered architecture, the AP should work as a network coordinator to connect the edge layer to the wired infrastructure, and therefore, deliver the processing tasks to the optimum PN. Accordingly, we assumed that the AP works at two different service rates based on the interface used; either the wired fiber infrastructure (with a 10

Gb/s service rate), or wireless medium (with 1 Gb/s service rate). Moreover, the core node, with the associated data centre, was assumed to work at 40 Gb/s, as they are part of the IP/WDM network. Other network devices were assumed to have a 10 Gb/s service rate based on GPON. The services rate values are defined in Figure 6.1, (coloured in green).

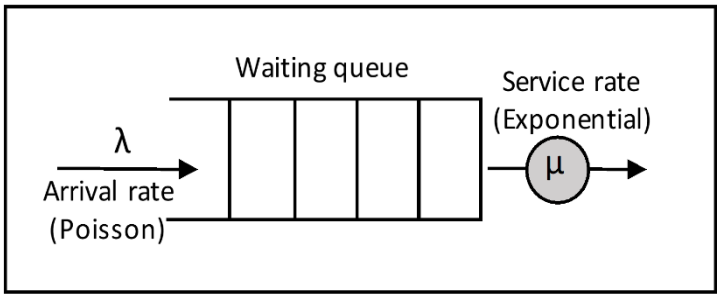

Figure 6.2 M/M/1 Queueing model.

6.3 Modification to the MILP Model

The MILP model introduced in Chapter 4 was extended to jointly minimise power consumption and delay. To continue to use linear programming, Equation (6.1) was converted to a linear form using lookup table [108]. The lookup table is predefined with all the possible generated traffic combinations (arrival rates indicator), indexed with the calculated queuing delay based on a fixed service rate. As we have three different service rate values in our designed network, a separate lookup table was defined for each. Based on this arrival rate indicator, the queuing delay for a node was given as the value corresponding to the indicator in the lookup table. Moreover, as we evaluated the model with increasing total generated traffic (100Mb/s-1000Mb/s), ten lookup tables were built as a part of the input file for each service rate. All the pre-defined lookup tables are provided in Appendix B. the increase of the

lookup table in the input file affect the complexity of the model and therefore, increase the running time due to increase the searching time for the associated delay to the generated traffic. However, this linerisation is essential to solve the processing allocation problem with the queuing delay minimisation. Without this process, solving such model in MILP is not possible.

The modified MILP defines the following additional sets, parameters, and variables:

Sets:

A	Set of AP nodes considering the wired interface, where $A \subset N$.
AL	Set of AP nodes considering the wireless interface, where $AL \subset N$.
AR	Set of arrival rates.
SR	Set of service rates.

Parameters:

H_{as}	Queuing delay at arrival rate $a \in AR$ and service rate $\in SR$, in the lookup table.
$G1$	Large enough number with units of Mb/s.
$G2$	Large enough number with units of ms.
D_{ij}	Distance between any two nodes (i,j), where $i \in N,\ j \in NN_i$.
$\mathbb{C}$	Speed of light, $\mathbb{C} = 299{,}792\ \frac{km}{s}$.
ΔRI	Refractive index of fibre, it defines the ratio of the speed of light in fibre to speed of light in free space; $\Delta RI = \frac{2}{3}$.

Variable	
ζ_{ij}^{sd}	Binary variable $\zeta_{ij}^{sd} = 1$ if traffic flow sent from source node s to processing node d traverses physical link (i,j), where $s \in SN$, $d \in PN$, and $i, j \in N$.
Q_{ij}^{sd}	Queuing delay at node j experienced by the traffic from source node s to processing node d traversing physical link (i,j), where $s \in SN$, $d \in PN$ and $i, j \in N$.
Q_i	Queuing delay experienced by traffic aggregated at node $i \in N$.
Q_{sd}	Queuing delay of the traffic sent from source node $s \in SN$ to processing node $d \in PN$.
Q	Total queuing delay of the network.
R_{sd}	Propagation delay of the traffic sent from source node $s \in SN$ to processing node $d \in PN$.
R	Total propagation delay of the network.
λ_{i}	Arrival rate (total traffic) at each node $i \in N$.
σ_{ij}	Arrival rate indicator for node $i \in N$, $\sigma_{ij} = 1$ if the arrival rate of node i matches rate $j \in AR$, it is 0 otherwise.

All the power consumption equations in Chapter 4, Equations (4.2) to (4.21), were considered in this model. The total power resulted from adding all these power consumption components is as follows;

$$P = TPC^{CC} + TPC^{MF} + TPC^{LF} + TPC^{NF} + TPC^{VN} + TPC^{NET} . \quad (6.2)$$

Additionally, the following equations are used to calculate the propagation and queuing delay for the network.

1) The total propagation delay (R), is calculated based on the propagation delay between all source node and processing node pairs and is given as

$$\mathrm{R} = \sum_{s \in SN} \sum_{d \in PN} \mathrm{R}_{sd} \qquad \forall \ s \in SN, d \in PN \qquad (6.3)$$

where R_{sd} is is the propagation delay of the path traversed by traffic sent from each source node $s \in SN$ to the processing node $d \in PN$, and is calculated as follows;

$$\mathrm{R}_{sd} = \sum_{\substack{i \in N \\ i \notin AL}} \sum_{\substack{j \in Nmi \\ j \notin VN}} \zeta_{ij}^{sd} \ \frac{D_{ij}}{\Delta RI \ \mathbb{C}} \qquad \forall \ s \in SN, d \in PN, d \notin VN \qquad (6.4)$$

$$\mathrm{R}_{sd} = \sum_{\substack{i \in N \\ i \in AL}} \sum_{\substack{j \in Nmi \\ j \in VN}} \zeta_{ij}^{sd} \ \frac{D_{ij}}{\mathbb{C}} \qquad \forall \ s \in SN, d \in VN. \qquad (6.5)$$

Equation (6.4) and (6.5) calculate the propagation delay for the traffic sent to the processing nodes via fibre or wireless links, respectively. A refractive index ΔRI with the value of $\frac{2}{3}$ is added to Equation (6.4) to define the ratio of the speed of light in fibre to the speed of light in free space.

2) The total queuing delay (Q), which is calculated based on the queuing delay experienced by traffic between all the source node and processing node pairs, and is given as

$$\mathrm{Q} = \sum_{s \in SN} \sum_{d \in PN} \mathrm{Q}_{sd} \qquad \forall \ s \in SN, d \in PN \qquad (6.6)$$

where Q_{sd} is the queuing delay of the path traversed by traffic sent from each source node $s \in SN$ to processing node $d \in PN$, and is calculated as

$$\mathrm{Q}_{sd} = \sum_{\substack{i \in N \\ i \notin SN \cup AL}} \sum_{j \in Nmi} \mathrm{Q}_{ij}^{sd} \qquad \forall \ s \in SN, d \in PN, d \notin VN \quad (6.7)$$

$$\mathrm{Q}_{sd} = \sum_{\substack{i \in N \\ i \notin SN \cup A}} \sum_{j \in Nmi} \mathrm{Q}_{ij}^{sd} \qquad \forall \ s \in SN, d \in VN. \quad (6.8)$$

Equations (6.7) and (6.8) calculate the queuing delay for a traffic demand by summing the queuing delay experinced by the demand at each node. As mentioned earlier, the traffic handled by the AP can be sent via one of two interfaces; wired interface at 10 Gb/s and wirless interface at 1 Gb/s; thus, Equation (6.7) considers the traffic sent via the wired interface of the AP while Equation (6.8) considers the traffic sent to the vehicular nodes via the 1G wireless interface.

The joint objective is defined as:

Minimise

$$\alpha P + \beta R + \gamma Q \quad (6.9)$$

where α, β, and γ are weight factors used for the following purposes: (i) to scale the terms so that they are comparable in magnitude; (ii) to emphasise and de-emphasise terms (power, queuing delay and propagation delay); and (iii) to accommodate the units in the objective function. Therefore, α is a unitless factor, and β & γ have units of $\frac{Watt}{sec}$.

In addition to the constraints in Chapter 4, the model is subject to the following additional constraints:

1) The traffic estimation at each node:

$$\sum_{s\in SN}\sum_{d\in PN}\sum_{j\in Nmi}\lambda_{ij}^{sd} = \lambda_i \qquad \forall\ i \in N, i \notin A \cup AL \quad (6.10)$$

$$\sum_{s\in SN}\sum_{d\in PN}\sum_{j\in Nmi\cap U}\lambda_{ij}^{sd} = \lambda_i \qquad \forall\ i \in A \quad (6.11)$$

$$\sum_{s\in SN}\sum_{d\in PN}\sum_{j\in Nmi\cap VN}\lambda_{ij}^{sd} = \lambda_i \qquad \forall\ i \in AL \quad (6.12)$$

Constraint (6.10) calculates the traffic arrival at each node in the network except APs. Constraint (6.11) and (6.12) estimate the arrival traffic wired and wireless AP interfaces, respectively.

2) The arrival rate indicator:

$$\sum_{j\in AR}\mathrm{H}_{ij}\ \ j = \lambda_i \qquad \forall\ i \in N, i \notin A \cup AL \quad (6.13)$$

$$\sum_{j\in AR}\mathrm{H}_{ij}\ j = \lambda_i \qquad \forall\ i \in A \quad (6.14)$$

$$\sum_{j\in AR}\mathrm{H}_{ij}\ j = \lambda_i \qquad \forall\ i \in AL \quad (6.15)$$

Constraints (6.13) to (6.15) create indicators of the arrival rate for each node. This is equal to 1 if the arrival rate is equal to j:

$$\sum_{j\in AR}\mathrm{H}_{ij} \le 1 \qquad \forall\ i \in N, i \notin A \cup AL \quad (6.16)$$

$$\sum_{j\in AR}\mathrm{H}_{ij} \le 1 \qquad \forall\ i \in A \quad (6.17)$$

$$\sum_{j\in AR}\mathrm{H}_{ij} \le 1 \qquad \forall\ i \in AL \quad (6.18)$$

Constraints (6.16) to (6.18) ensure that each node has no more than one arrival rate indicator for a given service rate.

3) Queuing delay estimation:

$$\sum_{j \in AR} \mathrm{H}_{ij} \,.\, \eta_{js} = \mathrm{Q}_i \qquad \forall\ i \in RR \cup CR \cup CS\,,\ s = \frac{40Gb}{s} \quad (6.19)$$

$$\sum_{j \in AR} \mathrm{H}_{ij} \,.\, \eta_{js} = \mathrm{Q}_i$$

$$\forall\ i \in N\,, i \notin RR \cup CR \cup CS \cup A \cup AL, \quad s = \frac{10Gb}{s} \quad (6.20)$$

$$\sum_{j \in AR} \mathrm{H}_{ij} \,.\, \eta_{js} = \mathrm{Q}_i \qquad \forall\ i \in A\,,\ s = \frac{10Gb}{s} \quad (6.21)$$

$$\sum_{j \in AR} \mathrm{H}_{ij} \,.\, \eta_{js} = \mathrm{Q}_i \qquad \forall\ i \in AL\,, \quad s = \frac{1Gb}{s} \quad (6.22)$$

Constraints (6.19) and (6.20) estimate the traffic delay for each node that operates at 40 Gb/s or 10 Gb/s, respectively. Constraints (6.21) and (6.22) estimate the delay for the AP wired and wireless interfaces, respectively.

$$\lambda_{ij}^{sd} \geq \zeta_{ij}^{sd} \qquad \forall\ s \in SN, n \in PN \quad (6.23)$$

$$\lambda_{ij}^{sd} \leq G1\ \zeta_{ij}^{sd} \qquad \forall\ s \in SN, n \in PN \quad (6.24)$$

Constraints (6.23) and (6.24) set $\zeta_{ij}^{sd} = 1$ if the traffic demand between the source node $s \in SN$ and the processing node $d \in PN$ is routed throughlink $(i, j) \in N$.

$$\mathrm{Q}_{ij}^{sd} = \mathrm{Q}_i\ \zeta_{ij}^{sd} \qquad \forall\ s \in SN, d \in PN \quad (6.25)$$

$$\mathrm{Q}_{ij}^{sd} \leq G2\ \zeta_{ij}^{sd} \qquad \forall\ s \in SN, d \in PN \quad (6.26)$$

$$\mathrm{Q}_{ij}^{sd} \leq \mathrm{Q}_i \qquad \forall \ s \in SN, d \in PN \qquad (6.27)$$

$$\mathrm{Q}_{ij}^{sd} \geq \mathrm{Q}_i - G2\left(1 - \zeta_{ij}^{sd}\right) \qquad \forall \ s \in SN, d \in PN \qquad (6.28)$$

Equation (6.25) calculates the queuing delay at node j for the traffic sent from source node s to processing node d, through node i. As Equation (6.25) involves the multiplication of two variables, Q_{ij}^{sd} and Q_i, it is cannot be included in the model with its original non-linear form, and thus, the problem cannot be solved in MILP. Therefore, Equation (6.25) is converted to constraints (6.26) to (6.28) to remove the nonlinearity and replace the relationship with an equivalent linear relationship.

6.4 Scenarios and Results

The model presented in the previous section is considered with the following variations cases of the objective function:

1) Minimising the total power consumption only, by setting α to 1; and β and γ to zero in Equation (6.9).

2) Minimising the traffic propagation delay only, by setting the values of β to 1; and α and γ to zero in Equation (6.9).

3) Minimising the power consumption and traffic propagation delay jointly, by setting the values of α to 1, γ to zero, and β to a value that ensures equal importance of the power consumption and propagation delay. This was done as follows:

a) Running the MILP model with the minimising power consumption objective.

b) Running the MILP model with the minimising propagation delay objective.

c) Calculating $\beta = \frac{P}{R}$ such that $\alpha P = \beta R$, where P is the power consumption and R is the propagation delay.

d) Runing the MILP model with the objective of minimising both the power consumption and propagation delay and comparing the αP and βR values.

e) Adjusting the value of β again, so that the joint objective function produced the required equality ie $\alpha P = \beta R$.

4) Minimising the traffic queuing delay, by setting the values of α and β in Equation (6.9) to zero, and the value of γ to 1.

5) Minimising the power consumption and traffic queuing delay jointly, by setting the values of α to 1 and β to zero, and γ to a value that ensures equal importance of the power consumption and queuing delay. This was done by repeating the same steps mentioned previously in 3), replacing β and R with γ and Q, respectively. This was carried out to ensure that the following equality $\alpha P = \gamma Q$ was achieved.

6) Minimising the power consumption, traffic propagation and queuing delay jointly, by setting the values of α to 1, and the values of β and γ to two ratios, to ensure that $\alpha P = \beta R = \gamma Q$, using, again, the same steps described in 3).

The previously described objective functions were combined into four scenarios that highlight the individual effects of the propagation and queuing delay, combined with the power consumption on the processing allocation

decision, and both power and delay values. These scenarios are summarised as the following:

- Scenario 1: where three objective functions were considered to minimise total power consumption (case 1), minimising propagation delay (case 2), and minimising both, jointly (case 3).
- Scenario 2: where three objective functions were considered to minimise total power consumption (case 1), minimising queuing delay (case 4), and minimising both, jointly (case 5).
- Scenario 3: with one objective function (minimising queuing delay only), considering different service rate for the AP.
- Scenario 4: where three objective functions were considered to minimise power and propagation delay (case 3), minimising power and queuing delay (case 5), and minimising power, propagation, and queuing delay (case 6).

All the above scenarios considered a cloud-fog-VEC allocation (CFVA) with low-density VNs (8VNs) and single allocation (no task splitting). We assessed the allocation problem with ten generated tasks having low processing demands (100–1000 MIPS) and a fixed data rate ratio (DRR) of 0.1. The reason for choosing a low demand with a high DRR was to generate intensive traffic (10–100 Mb/s) per task, in order to study the delay results on such a congested network. It worth mentioning that the queuing delay value is expressed in sec/packet as given in (6.1), where we used the Ethernet MTU packet size of 1500 Byte.

6.4.1 Scenario 1: Power and Propagation Delay Minimisation

In this scenario, we study the joint minimisation of the power consumption and the propagation delay (objective function case 3), and compare the results to the two cases where only the power (objective function case 1) or propagation delay (objective function case 2) are minimised. Figures 6.3 and 6.4 illustrate the total power consumption and the average propagation delay for the three cases considered in the objective function versus the total traffic generated from the ten tasks, each 100–1000 Mb/s.

Figure 6.3 shows that the power minimised case produced the lowest power consumption. The jumps in the curves in the three minimisation functions are due to moving the allocation at higher traffic to a less efficient PN that can support the traffic. For example, in the power minimised case, the small jumps at 400 Mb/s were caused by moving the allocation from VN to NF, as seen in Figure 6.5. The optimisation did this because of the limited VN connection data rate (72.2 Mb/s per VN) cannot serve more than one task with the 40 Mb/s required data rate. Activating the ONU and its processor (NF) caused all tasks to be allocated to the NF instead of activating two locations (as seen and confirmed from the results shown in Chapter 4). Another power rise occurred at 800 Mb/s, when all tasks were allocated to the LF. The delay minimised objective led to higher power consumption (by up to 35%) compared to the power minimised case (Figure 6.3). This is due to all (or the majority of) tasks being allocated in VNs, even with the NF being activated (Figure 6.5), which is not a power-efficient allocation decision. With the joint minimisation of the power and delay, MILP results (at 100–700 Mb/s) led to a

power consumption comparable to the power minimised optimisation, due to the fact that minimising delay required poor energy-efficiency choices, i.e., the placement of tasks in two PN (NF & VN), which was not allowed by a MILP that weighs power and delay equally. However, the power consumption results beyond 700 Mb/s became comparable with the delay minimised optimisation, as the propagation delay became a limiting factor which cause the model to allocate tasks to both locations (NF and LF) in order to achieve a lower average delay over all of the tasks. It is worth mentioning that the power consumption resulting from the joint optimisation (the green curve in Figure 6.3) is not exactly in the middle of the power minimised and delay minimised curves, as might be expected, because the number of placement options (PNs) was finite and discrete. Practical systems will typically have a similar number of processing locations (PNs) or even fewer as building fog processing nodes at a higher granularity (e.g. few hundred meters) is not practical.

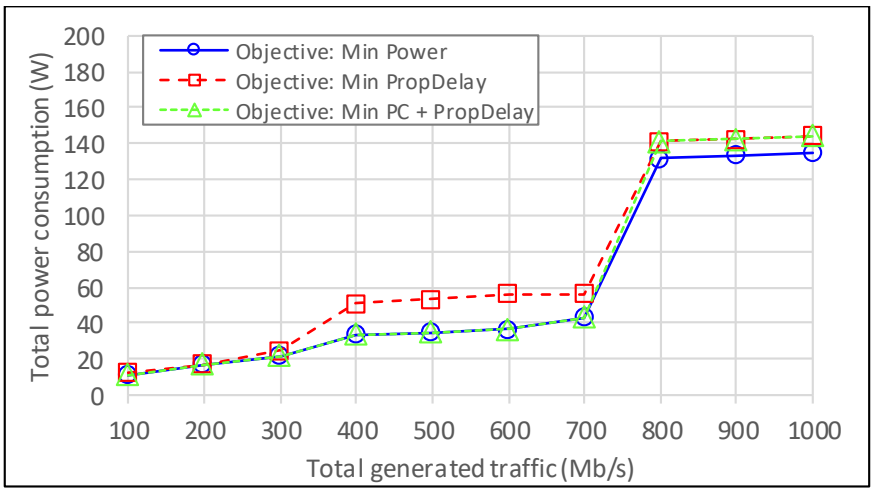

Figure 6.3 Total power consumption (Scenario 1)

Figure 6.4 displays the average propagation delay for the same three objective function cases. For the power minimised case, it can be seen that the low power consumption in Figure 6.3 has resulted in the high level of propagation delay in Figure 6.4. The delay starts low (similar to the other two optimisation cases), as the tasks are allocated in the three cases to VN, as VNs are the most efficient PN for both power and propagation delay minimisation. However, the delay increases with increase in the traffic as the tasks are allocated to further locations (i.e., NF and LF). It can also be seen from the zoomed-in figure (in Figure 6.4) that tasks allocated to NF (at 300–700 Mb/s) experienced more propagation delay compared to the VN allocation, despite the fact that both the PNs were 100 m away from the AP (as mentioned in Section 6.2). This disparity is attributed to the fact that the speed of light in fibre, connecting the AP to NF, is affected by a refractive index of $\frac{2}{3}$, as defined in Equation (6.4), which causes an increase in the propagation delay of the NF compared to VN. Thereafter, the delay jumps from 0.5 µs to 50 µs when the allocation changed from NF to LF (beyond 700 Mb/s), as the LF is located 10 km away from the ONU. This shows the effect of the PN location on the propagation delay and therefore, on the allocation decision. This effect was confirmed when minimising the propagation delay, as seen in Figure 6.4, when the model allocated more tasks to the available VNs even if other PNs nodes were activated (as seen in Figure 6.5). Similar to the power results shown in Figure 6.3, minimising both the power and delay showed an average delay relatively comparable to the case of minimised power at 100–700 Mb/s, and to the case of minimised delay beyond 700 Mb/s. This is due to the same reason mentioned previously, as the finite number of

available processing locations causes comparable results to either one of the other cases rather than having results in between the two other cases.

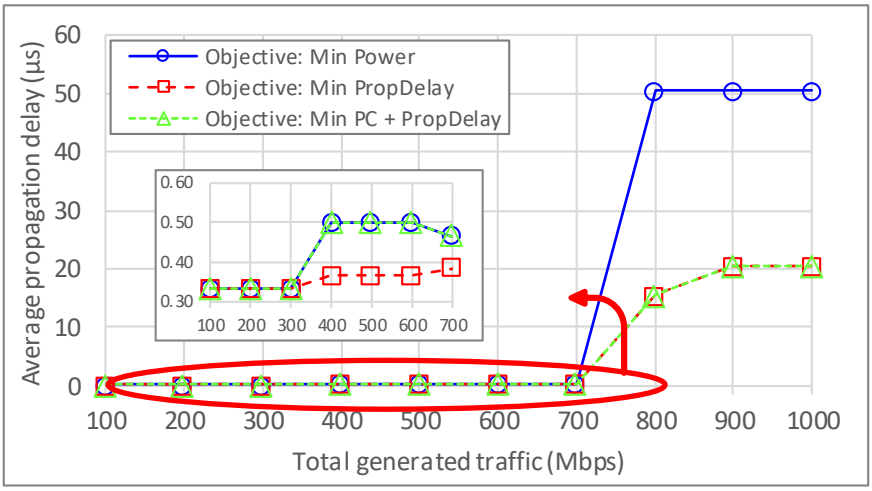

Figure 6.4 Average propagation delay (Scenario 1), enclosed zoom-in figure shows the result in 100–700 Mb/s range.

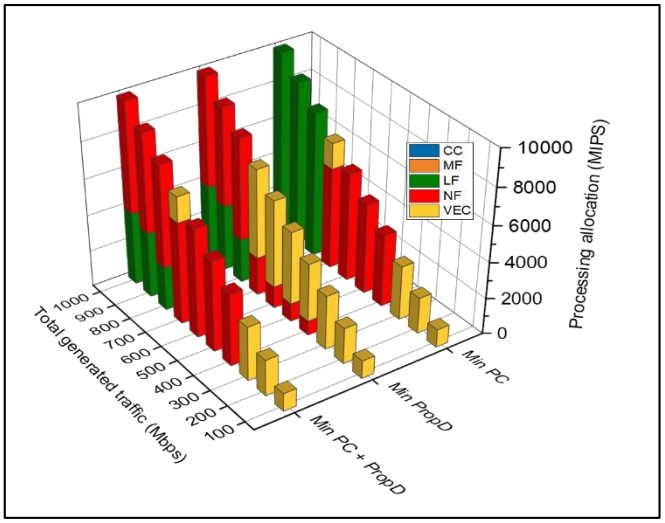

Figure 6.5 Processing allocation in each PN (Scenario 1)

6.4.2 Scenario 2: Power and Queuing Delay Minimisation

In this scenario, we study the joint minimisation of the power consumption and the queuing delay (objective function case 5), and compare it to the power minimised case (objective function case 1), and the queuing delay minimised

objective function (objective function case 4). Figure 6.6 shows the total power consumption for the three cases. Relatively comparable power consumption results can be observed for the three optimisation objectives. The power consumption minimised case demonstrated an increase in the power consumption whenever the allocation is changed to a less efficient location (similar to the previous result shown in Figure 6.3). When minimising the queuing delay, the results showed an early increase in the total power consumption as the VNs were avoided and the tasks were allocated to the NF, as seen in Figure 6.8. This allocation decision is attributed to the bad service rate of the AP wireless interface, which causes a very high queuing delay at this interface and therefore, the VN becomes an inefficient location in terms of the average queuing delay. This allocation explains the comparable power consumption, in this case, to the power minimised case (at 400–600 Mb/s), as the tasks in both cases were allocated to NF. However, as the generated traffic increases (beyond 600 Mb/s), the power consumption jumps to 140 W an increase by 70%, compared to the power minimised case. At this point, the NF became exhausted and was unable to accommodate all the generated tasks, and thus, the LF was activated to accommodate the remaining workload. This explains the continued and saturated increase in the power consumption (by 7%) compared to power minimised case (at 800–1000 Mb/s). In the latter case, activating one PN to serve all the generated tasks is more efficient than activating two or more PNs. Recalling the reason from Chapter 4, activating each fixed fog node consumes extra power overhead due to the idle power and PUE of the network devices and servers. In the case of the joint minimisation of both the power consumption and queuing delay,

the power consumption results were very comparable to the queuing delay minimised case. Including the queuing delay in the objective function avoids allocating any tasks to VN, as it causes a significant increase in the delay. One case resulted in the allocation of a portion of the processing workload to VNs, at 700 Mb/s, where the model achieved a balance between the power and the delay by allocating a small portion of the workload to VNs (140 Mb/s) with the majority of the workload allocated to NF (560 Mb/s). This occurred when the NF became exhausted (with 5600 MIPS allocation) and was not able to bin-pack any more workload to achieve full utilisation; hence, the optimisation allocated the remaining 1400 MIPS to VN, as activating the LF would cause a significant increase in the power consumption and therefore, the objective function balance might not be achieved.

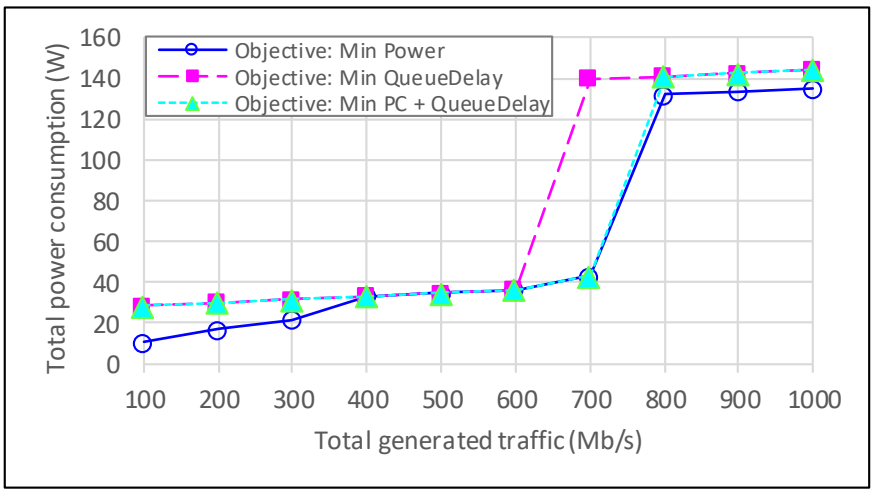

Figure 6.6 Total power consumption (Scenario 2)

Figure 6.7 shows the average queuing delay for the same three objective function cases. As described earlier in Section 6.1, the queuing delay of a node is influenced by the service rate of this node, hence, the PNs with a high service rate (central cloud and fixed fog nodes) had a lower delay, compared

to the wireless-based VNs, due to the low service rate of the wireless devices (i.e., AP). For example, in the power minimised case (in Figure 6.7), low demand tasks were allocated to VNs (as seen in Figure 6.8). This caused a high average queuing delay (13–17 µs) due to the low AP service rate (1 Gb/s). As soon as the traffic increased and the tasks were allocated to NF (as VN became insufficient), the average queuing delay dropped by 85% (from 17 µs to 2.5 µs) and saturates at this value as long as the NF remains the most power efficient placement. The queuing delay experienced a continued increase due to the increase in the number of hops to the optimum allocation at the LF. With the queuing delay minimised objective, the delay was reduced by an average of 84%, compared to the power minimised case at low generated traffic. This reduction was due to the allocation of tasks to the fixed fog (NF), with a better service rate (10 Gb/s). The model maintained a continuous low average queuing delay with increase in the total traffic, which became comparable with the power minimised case, and then increased slightly due to part of the task being allocated to an extra PN. However, the model yielded up to 43% lower average queuing delay with high traffic, compared to the power minimisation case. The case where both the power and delay were minimised produced a comparable average delay to the delay minimised case. As mentioned previously with the power consumption results, the queuing delay became a limiting factor that constrained the optimisation and resulted in the tasks being allocated to VNs, due to the bad service rate of the AP wireless interface, which caused a huge increase in the queuing delay. One exception occurred at 700 Mb/s, where the balance was achieved

by allocating the tasks to two PNs (i.e. NF and VN) due to NF insufficient capacity, as seen in Figure 6.8.

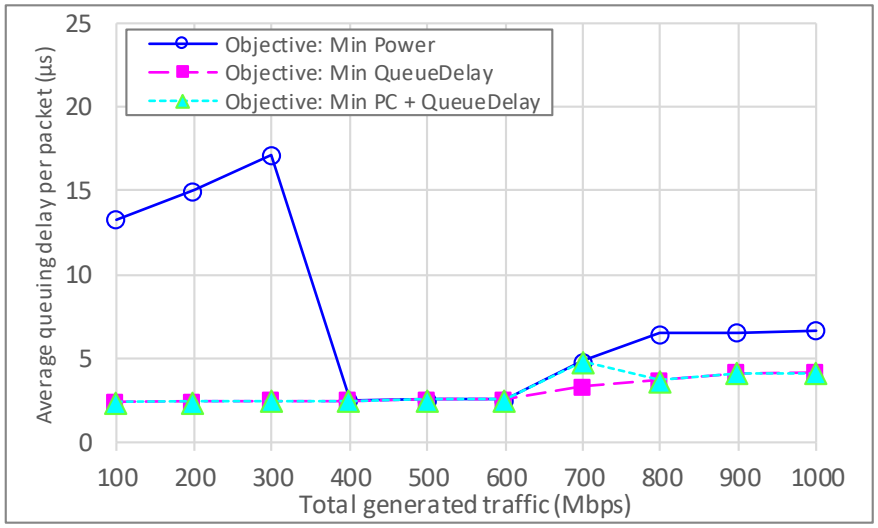

Figure 6.7 Average queuing delay (Scenario 2)

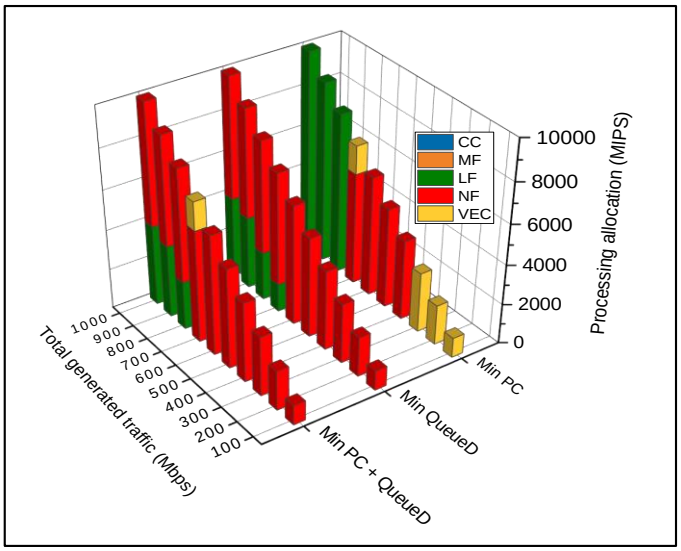

Figure 6.8 Processing allocation in each PN (Scenario 2)

6.4.3 Scenario 3: Queuing Delay Optimisation with Multiple AP Service Rate

As observed previously in Scenario 2, the queuing delay in the access layer became a limiting factor which made the allocation of tasks to the VNs unfeasible due to the wireless devices' insufficient service rate. The AP is the

control gate for VNs, it is thus important to investigate the effect of the multiple service rates on the VNs allocation and the average queuing delay. Two objective functions were examined: minimising the queuing delay (Figures 6.9–6.11) and minimising the power and queuing delay, jointly (Figures 6.12–6.14). In each optimisation function setting, three cases were investigated where the AP wireless interface was set to 1, 5 and 10 Gb/s service rate. The case where the AP operated at 1 Gb/s was the default service rate, and its related results were discussed earlier in Scenario 2 (Figures 6.6–6.7) and are provided in this scenario as a baseline case. The case where the AP operated at 10 Gb/s represented the other extreme where the AP has a high service rate equivalent to the service rate of the wired interface. The third case where the AP service rate was 5 Gb/s provided an average service rate between the other two cases. This helps in the determination of the impact of the service rate, and in particular if a higher service rate is not required to achieve a better queuing delay.

Figures 6.9 and 6.10 show the total power consumption and average queuing delay when the queuing delay was minimised at the three defined service rates for the AP wireless interface. The results show that with an increased service rate, the queuing delay is reduced and the VN allocation increased. This may increase or decrease the power consumption based on the number of activated PNs. For example, at 5 Gb/s service rate, the restriction in the VNs allocation was relaxed, and the tasks were allocated to the VN alongside the NF (as in Figure 6.11), which justifies the small increase in power consumption (an average of 14%), compared to the 1 Gb/s case. However, a power consumption saving of 67% was observed at 5 Gb/s service rate at a

700 Mb/s incoming traffic rate, as some tasks were allocated to VN instead of LF in the 1 Gb/s case, at the same 700 Mb/s traffic. On the other hand, the average queuing delay in the 5 Gb/s service rate case (in Figure 6.10) did not achieve a significant improvement up to an incoming traffic rate of 600 Mb/s despite the increase in the AP service rate. This is attributed to the accumulated queuing delay based on the number of hops in the route leading to the PN.

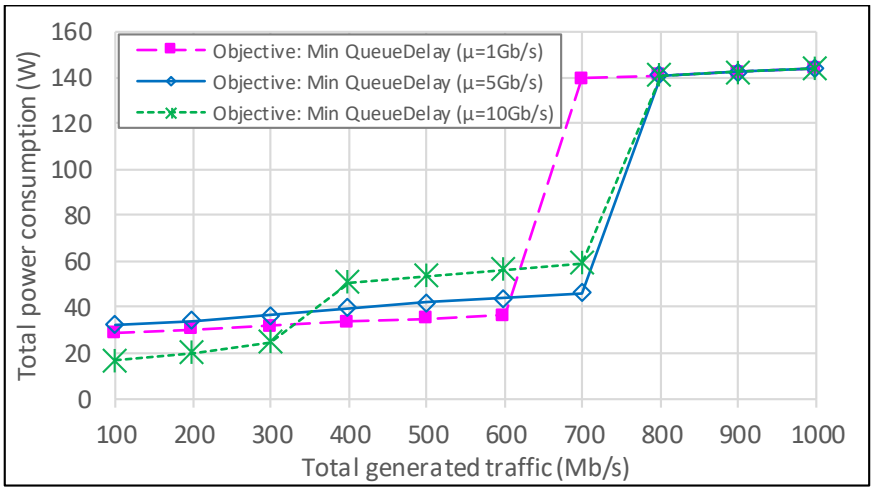

Figure 6.9 Total power consumption, under average queuing delay minimisation objective, with multiple AP service rates (Scenario 3).

For example, at 300 Mb/s traffic and with 1 Gb/s service rate, all ten tasks were allocated to the NF (as seen in Figure 6.11). The resulting average queuing delay was 2.5 μs, caused by the delay of the route leading to the NF (including the AP wired interface and the ONU). On the other hand, at the same 300 Mb/s traffic and with 5 Gb/s service rate, the tasks were allocated to the two PNs (LF and VN). The average queuing delay over all the tasks was equal to 2.4 μs, with only 0.1 μs improvement. This limited reduction in the queuing delay was due to a comparable queuing delay resulting from the two allocations because the decrease achieved by the better AP service rate

resulted in a comparable queuing delay of the two hop route to the NF (AP and ONU). By increasing the AP wireless service rate to 10 Gb/s, we saw an early reduction in the power consumption (Figure 6.9) as all the tasks were fully allocated to VN (as shown in Figure 6.11). However, the power consumption increased aftward because of two reasons: Firstly due to the activation of two PNs located at the NF and VN, and secondly due to the utilisation of more VNs, which caused extra power overhead resulting from activating the VN communication adapter. This resulted in the increase in power consumption by 34% and 21%, compared to the cases with 1 and 5 Gb/s service rates. However, the average queuing delay (Figure 6.10) shows significant reduction by an average of 48% compared to the 1 Gb/s case, because the service rate of both the AP interfaces became equivalent. Hence, the VN which is one hop away from the AP became more efficient than the PN that is two hops away (i.e., the NF).

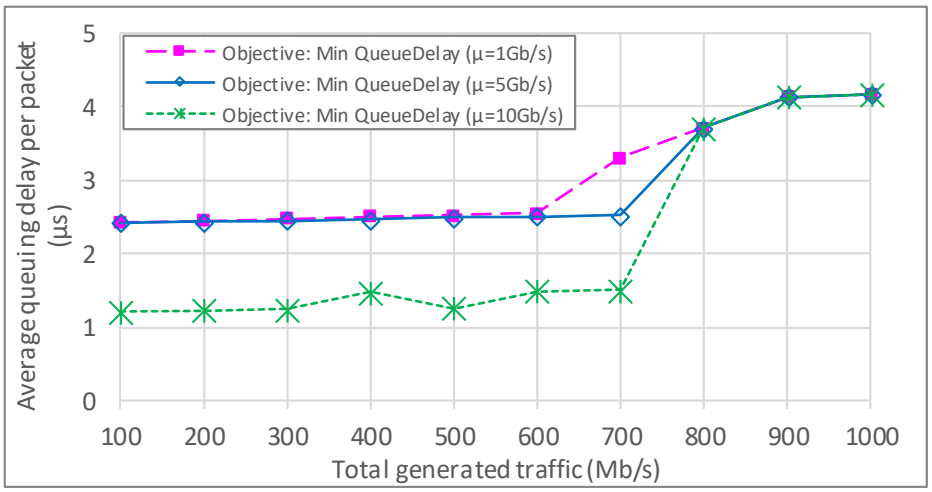

Figure 6.10 Average queuing delay, under average queuing delay minimisation objective, with multiple AP service rates (Scenario 3).

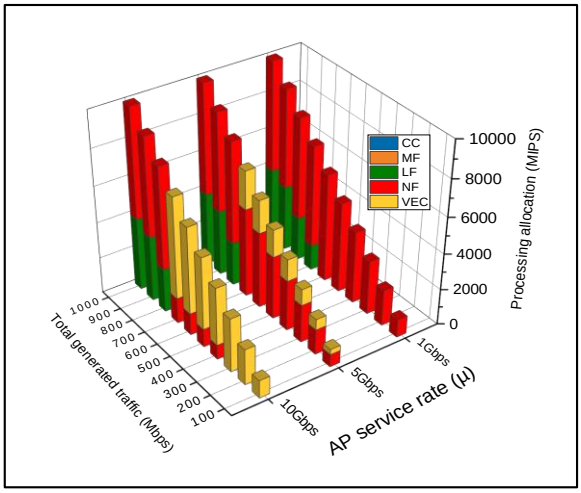

Figure 6.11 Processing allocation in each PN, under average queuing delay minimisation objective, with multiple AP service rates (Scenario 3).

Figures 6.12 and 6.13 show the total power consumption and the average queuing delay when the power and queuing delay were jointly minimised with the same three AP's service rates. The results show the small effect of the AP service rate on power consumption. This effect was only observed at low generated traffic, as the increase in the AP service rate relaxed the restriction of allocating tasks to VNs. Therefore, these VNs became an optimum location for both the queuing delay and power consumption. Moreover, a reduction in the average queuing delay (in Figure 6.13) was achieved whenever more tasks were allocated to the VNs with a high AP service rate. This reduction became significant when the tasks were allocated to the VN alongside the NS. This can be seen at 700 Mb/s traffic, where the average queuing delay was 4.8 µs when allocating tasks to VNs (as shown in Figure 6.14) due to the poor AP service rate. Although the optimisation resulted in the same allocation decision, with 5 Gb/s and 10 Gb/s service rates, the average queuing delay dropped by 47% and 52%, respectively, compared to the 1 Gb/s service rate

case. This confirmed that the increase in the AP service rate had a small effect on the allocation decision in the power and delay joint optimisation.

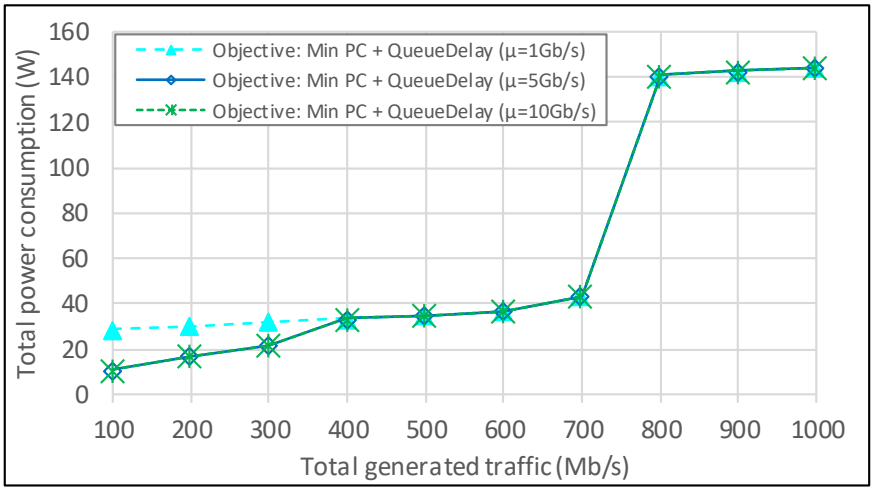

Figure 6.12 Total power consumption, under power consumption and average queueing delay minimisation objective, with multiple AP service rates (Scenario 3).

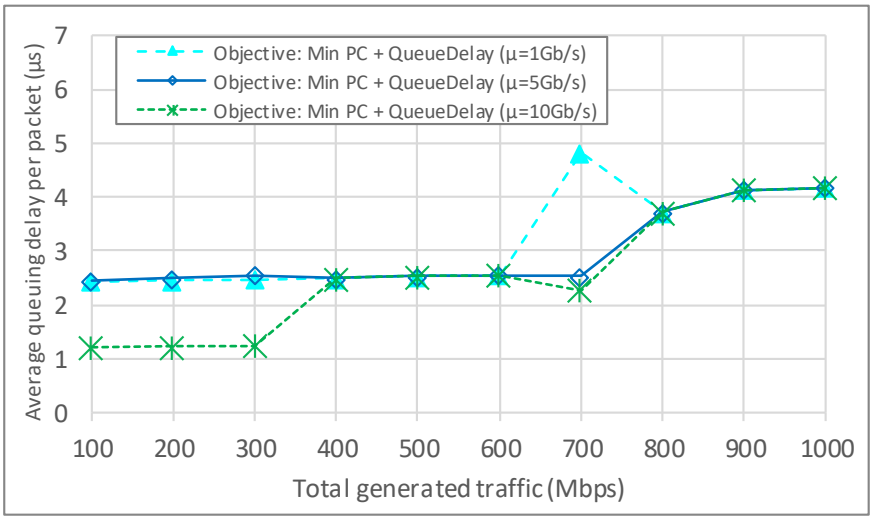

Figure 6.13 Average queuing delay, under power consumption and average queueing delay minimisation objective, with multiple AP service rates (Scenario 3).

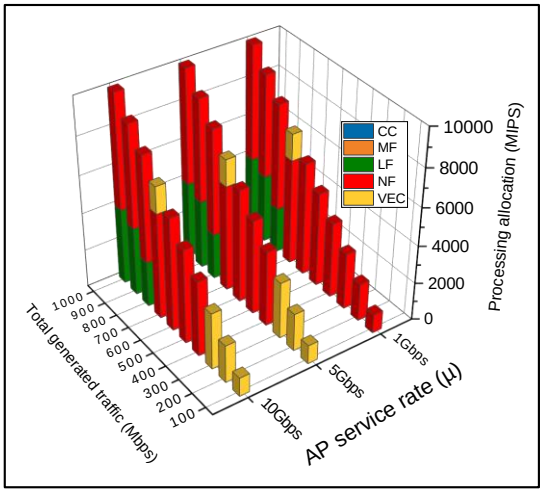

Figure 6.14 Processing allocation in each PN, under power consumption and average queueing delay minimisation objective, with multiple AP service rates (Scenario 3).

6.4.4 Scenario 4: Power, Propagation Delay and Queuing Delay Multi-Objective Minimisation

In this scenario, we study the joint optimisation of the power consumption, propagation delay and queuing delay, compared to the joint optimisation of the power consumption and propagation delay discussed in Scenario 1 and the joint optimisations of the power consumption and queuing delay discussed in Scenario 2

Figure 6.15 shows the total power consumption for the three objective function cases considered. The results show relatively comparable power results for the three objectives. We noticed, however, that whenever the optimisation objective included the queuing delay, an immediate restriction was applied on allocating any workload to the VNs, as shown in Figure 6.18. This was attributed to the fact that the allocation in VN caused a very high queuing delay at the AP wireless interfaces, which excluded the VNs from the allocation decision. Therefore, the multi-objective function followed the same outcomes

of the power minimised objective and the queuing delay objective when considering the average propagation delay and the average queuing delay results (Figure 6.16 and 6.17).

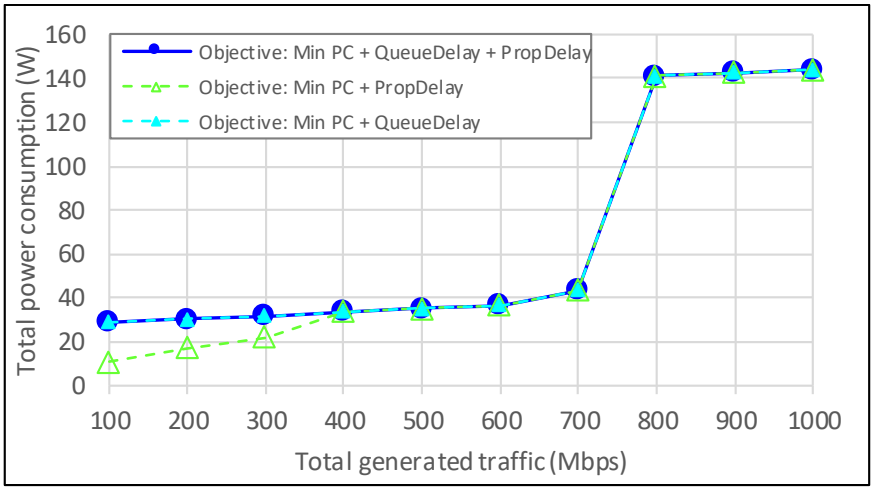

Figure 6.15 Total power consumption (Scenario 4)

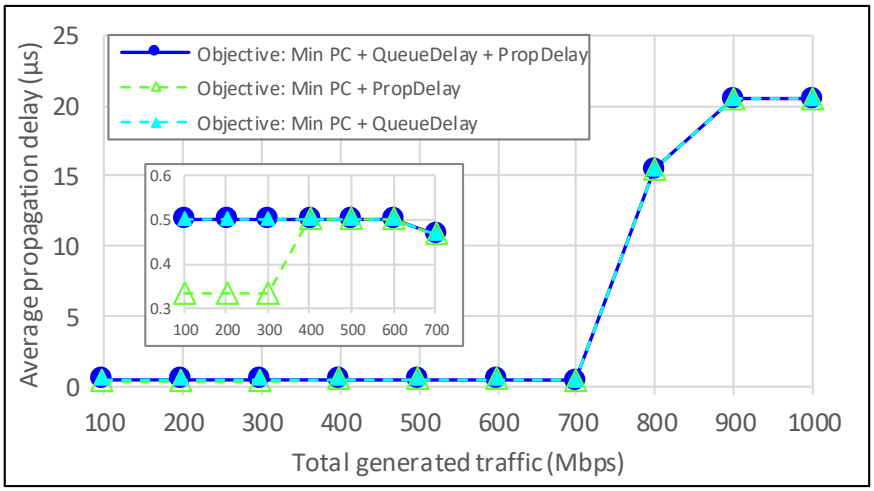

Figure 6.16 Average propagation delay (Scenario 4)

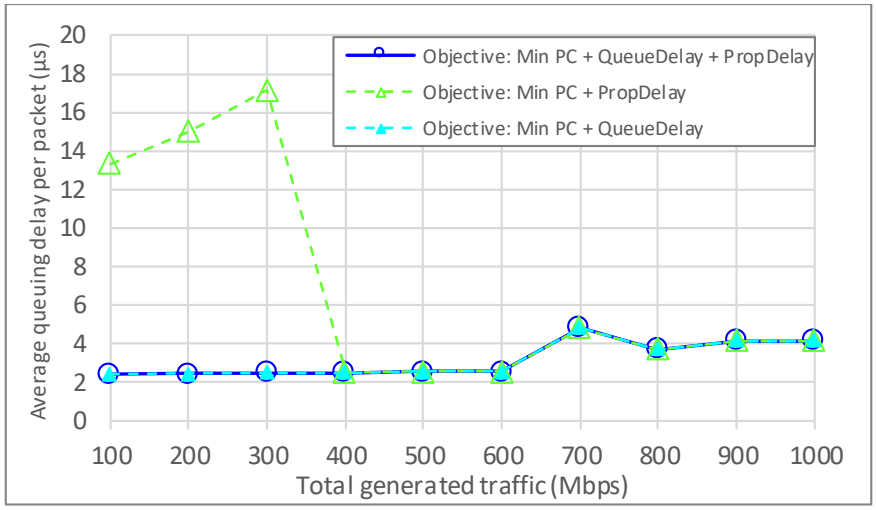

Figure 6.17 Average queuing delay (Scenario 4)

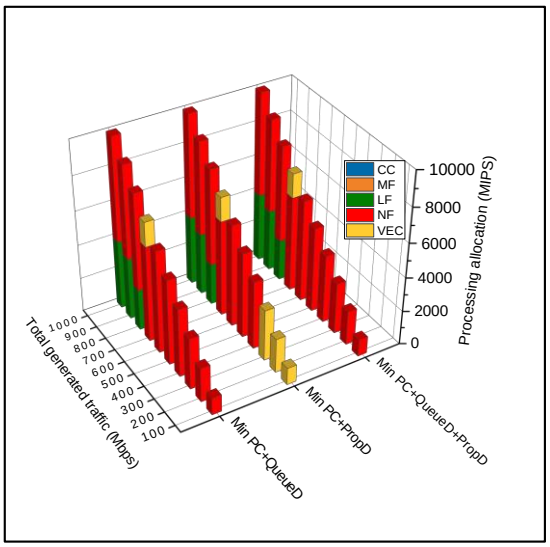

Figure 6.18 Processing allocation in each PN (Scenario 4)

6.5 Summary

In this chapter, we have invistegated the joint optimisation of power consumption, propagation delay and queueing delay when allocating processing tasks to cloud, fog and VEC processing nodes in an architecture which provides multiple processing locations. The study was carried out by extending the MILP introduced in Chapter 4, where the propagation delay and queuing delay are added. The evaluation considered different cases of the

objective function where power consumption, propagation delay and queueing delay are examined separately or together.

Our results show that the closer the PN is to the AP, the lower the power consumption and delay, as the distance and number of hops affect the propagation delay and queuing delay. However, the queuing delay at the AP becomes a limiting factor when it operates at a low service rate compared to the traffic arrival rate. Thus, processing task allocation at the VN was avoided whenever the objective function included queueing delay and the AP operated at a low service rate. Increase in the AP service rate result in a lower queuing delay and better VN utilisation.

Future work can introduce additional optimisation components to the delay such as processing and transmission delay alongside propagation and queuing delay. It will also consider more detailed AP queueing models such as models that have finite buffer size (M/M/1/N), (N is the buffer size), models that include multiple servers (c servers), M/M/c/N and potentially a finite population of sensor nodes, where S sensor nodes are considered, namely M/M/c/N/S models. This can modify the limiting factors and highlight different aspects that need to be improved to reduce the delay in the considered architechire and allocation mode.

Chapter 7
Conclusions and Future Directions

This chapter provides the conclusions of the work presented in this thesis. It also provides some potential directions for future research in vehicular edge cloud networks.

7.1 Conclusions

The purpose of this thesis was to address the energy efficiency of processing in opportunistic vehicular edge clouds. Vehicles with advanced OBU processors, parked in parking lots, charging points, gas stations, or stationary at intersections can be clustered and can form a processing pool or a mini data centre. With the support of the central cloud and multiple distributed fixed fog servers, the processing nodes can deliver reliable services to the edge based IoT applications. Multiple optimisation problems were investigated including processing allocation and software matching, in order to address the energy efficiency and delay challenges in the network architectures considered. Mixed Integer Linear Programming (MILP) models were developed to minimise the total power consumption of the considered architectures.

In Chapter 4, four possible processing layers were introduced: the core layer (central cloud), the metro layer (metro fog nodes), the access layer (OLT and ONU fog nodes), and the edge layer (vehicular nodes). The considered cloud-fog-VEC architecture was described in term of realistic values of the processing and networking devices capacities and efficiencies. This

architecture was evaluated through a MILP optimisation model whose objective is to minimise the total power consumption by optimising the processing allocation decision. The evaluation considered multiple cases to study the impact of workload volume, task generation density, vehicles density, and task allocation strategy (single and distribution). Two architectural designs were assessed where the edge layer consist of either a single VEC cluster (one ONU) or multiple VEC clusters (each with an individual ONU). Moreover, tasks were generated from different clusters (through different AP), and allocating these tasks to local and non-local VEC was evaluated. The results of this investigation show that the bottom-most processing nodes are the optimum allocation to process the generated tasks. Vehicles become a very attractive option to process the generated workload and save power, if the total capacity of the clustered vehicles are enough to serve the whole workload. Thus, the vehicle capacity and the number of available vehicles play an important role in the allocation decision. Another key factor of the allocation decision is the overhead power (idle power and PUE) of each processing server. Accordingly, with high generated tasks and high processing demands, the central cloud becomes more efficient than activating multiple vehicles and multiple fog nodes. When processing task splitting is allowed, the results show power saving between 40%-70% compared to no splitting (based on the scenario). Moreover, the results show that splitting tasks between processing layers is to be avoided unless this will avert activating less efficient processing nodes. It was also shown that expanding the access layer with multiple ONUs has minor effect on the allocation decisions, as the local VEC is always more efficient than the non-local VEC.

In Chapter 5, the MILP model was extended to study the software matching problem in VEC. To help focus on this problem, we considered only two layers of processing (at the central cloud and at the vehicular edge cloud). The study included three more considerations. Firstly, each vehicle has its own pre-installed software packages, and the model allocates processing accordingly. Based on this, different probability distributions (uniform, random, and Zipf distributions) were considered for the software availability. Secondly, consideration was given to a setting where vehicles have no pre-installed software packages in their OBU, but the optimisation determines the best software to install in each node. Thus, the model optimises the software allocation jointly with the processing allocation. Thirdly, consideration was given to a setting where each software type follows a certain popularity based on Zipf Distribution. Accordingly, processing tasks are generated with the requested software type, based on the software popularity. The software matching problem was evaluated in term of the power consumption and the popularity of the allocated tasks, for multiple vehicle densities and with single and distributed strategies. This study has found that in general, increasing the number of vehicles increases the number of software replicas, and therefore this increases the allocation of tasks in VEC. It is also shown that with pre-allocated software packages, the number of replicas used does not affect the power or the allocation decisions. On the other hand, by optimising the software allocation, the model minimises the number of installed replicas in each vehicle due to the download overhead.

In Chapter 6, the MILP model developed in Chapter 4 was extended to investigate the joint optimisation of power consumption, propagation delay

and queueing delay. All the processing layeres introduced in the architecture in Chapter 4 were considered in this chapter. More details were given regarding the estimated distances between network devices and the maximum service rate of each device. These formed inputs which were considered in the evaluation of the propagation and queuing delay. The evaluation was undertaken in different cases reflected in the objective function where power consumption, propagation delay and queueing delay were examined separately or together. The results revealed that the closer the PN is to the AP, the lower the power consumption and delay is, as the distance and number of hops affect the propagation delay and queuing delay. It was also found that the queuing delay at the AP becomes a limiting factor when it operates at a low service rate compared to the traffic arrival rate. Thus, the allocation of processing tasks at the VN was avoided whenever the objective function included queueing delay and the AP operated at a low service rate. Increase in the AP service rate resulted in a lower queuing delay and better VN utilisation.

7.2 Future Research Directions

In the following we list some of the general future research directions and their relation to the work presented in this thesis:

1- It is of interest to develop heuristic algorithms and simulation-based experiments to approximate the constructed MILP models and scale up the current architecture with increased number of processing nodes and vehicular nodes. This can also validate the developed MILP

models in a realistic scaled architecture and therefore present results that can relate directly to potential implementations in a smart city.

2- Modelling the vehicle mobility and using prediction mechanisms rather than considering deterministic varied number of vehicles is of interest. This can mimic the dynamic behaviour of vehicles and can help evaluate the effect of dynamic arrival and departure of vehicles in the car parks and their impact on the reliability of the processing at vehicular nodes and the task completion success. This can incorporate task migration to other available vehicles or to fixed nodes.

3- Extending the joint optimisation of delay minimisation to include processing delay, packet transmission delay and packet reception delay alongside propagation delay and queuing delay. Thus, the delay evaluation extends to the total service delivery not only to the propagation delay. This will probably give more accurate evaluation of the completion time and thus can assess the validity of applications that require intensive processing, for example. The optimisation can also consider other queuing delay models for the AP such as (M/M/1/N) with a finite buffer of size N, (M/M/c/N) with c servers and (M/M/c/N/S) with finite population of S sensor nodes. This can modify the limiting factors and highlight different aspects that need to be improved to reduce the delay in the considered architechire and can influence the processing allocation decisions.

4- Evaluating the link and node resilience of the vehicles and their connection to network controller. Since these vehicles base their

connection on wireless protocols, the stability of the connection and the opportunistic availability of vehicles can be intensively tested through resilience-based modelling.

5- A promising future direction is to investigate caches embedded in vehicles for Video-on-Demand applications by considering these vehicles as a mobile cache point to deliver video streaming to end users (for example, cache points in buses).

References

[1] Cisco White Paper, “Cisco Visual Networking Index: Forecast and Trends, 2017–2022,” 2019.

[2] S. Zeadally, S. U. Khan, and N. Chilamkurti, “Energy-efficient networking: Past, present, and future,” *J. Supercomput.*, vol. 62, no. 3, pp. 1093–1118, 2012.

[3] T. Kim, H. Min, J. Park, J. Lee, and J. Jung, “Analysis on characteristics of vehicle and parking lot as a datacenter,” *Proc. 2017 4th Int. Conf. Comput. Appl. Inf. Process. Technol. CAIPT 2017*, vol. 2018-Janua, pp. 1–4, 2018.

[4] U. Shaukat, E. Ahmed, Z. Anwar, and F. Xia, “Cloudlet deployment in local wireless networks: Motivation, architectures, applications, and open challenges,” *J. Netw. Comput. Appl.*, vol. 62, pp. 18–40, 2016.

[5] A. Yousefpour *et al.*, “All one needs to know about fog computing and related edge computing paradigms: A complete survey,” *Journal of Systems Architecture*. 2019.

[6] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge Computing: Vision and Challenges,” *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, 2016.

[7] M. R. Rahimi, J. Ren, C. H. Liu, A. V. Vasilakos, and N. Venkatasubramanian, “Mobile cloud computing: A survey, state of art and future directions,” *Mob. Networks Appl.*, vol. 19, no. 2, pp. 133–143, 2014.

[8] S. Pal and T. Henderson, “MobOCloud : Extending Cloud Computing with Mobile Opportunistic,” *Proc. 8th ACM MobiCom Work. Challenged networks - CHANTS ’13*, p. 57, 2013.

[9] R. Hasan, M. M. Hossain, and R. Khan, “Aura: An IoT based cloud infrastructure for localized mobile computation outsourcing,” *Proc. - 2015 3rd IEEE Int. Conf. Mob. Cloud Comput. Serv. Eng. MobileCloud*

2015, pp. 183–188, 2015.

[10] O. Altintas, F. Dressler, F. Hagenauer, M. Matsumoto, M. Sepulcre, and C. Sommery, “Making cars a main ICT resource in smart cities,” in *IEEE Conference on Computer Communications Workshops*, 2015, pp. 582–587.

[11] S. A. Bagloee, M. Tavana, M. Asadi, and T. Oliver, “Autonomous vehicles: challenges, opportunities, and future implications for transportation policies,” *J. Mod. Transp.*, vol. 24, no. 4, pp. 284–303, 2016.

[12] S. Arif, S. Olariu, J. Wang, G. Yan, W. Yang, and I. Khalil, “Datacenter at the airport: Reasoning about time-dependent parking lot occupancy,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 11, pp. 2067–2080, 2012.

[13] S. U. Rehman, M. A. Khan, T. A. Zia, and L. Zheng, “Vehicular Ad-Hoc Networks (VANETs) - An Overview and Challenges,” *J. Wirel. Netw. Commun.*, vol. 3, no. 3, pp. 29–38, 2013.

[14] A. M. Vegni, M. Biagi, and R. Cusani, “Smart Vehicles , Technologies and Main Applications in Vehicular Ad hoc Networks,” *InTech*, pp. 3–20, 2013.

[15] S. Olariu, I. Khalil, and M. Abuelela, “Taking VANET to the clouds,” *Int. J. Pervasive Comput. Commun.*, vol. 7, no. 1, pp. 7–21, 2011.

[16] R. Hussain, Z. Rezaeifar, and H. Oh, “A Paradigm Shift from Vehicular Ad Hoc Networks to VANET-Based Clouds,” *Wirel. Pers. Commun.*, vol. 83, no. 2, pp. 1131–1158, 2015.

[17] J. Joshi, K. Jain, Y. Agarwal, M. J. Deka, and P. Tuteja, “TMaaS : Traffic Management as a Service Using Cloud in VANETs,” in *Smart Instrumentation, Measurement and Applications (ICSIMA), 2015 IEEE 3rd International Conference on. IEEE*, 2015, pp. 1–6.

[18] H. S. Hassanein, S. Abdelhamid, and K. Elgazzar, “A framework for vehicular cloud computing,” in *2015 International Conference on*

Connected Vehicles and Expo, ICCVE 2015 - Proceedings, 2015, pp. 238–239.

[19] N. Liu, M. Liu, W. Lou, G. Chen, and J. Cao, “PVA in VANETs: Stopped cars are not silent,” in *Proceedings - IEEE INFOCOM*, 2011, pp. 431–435.

[20] F. Dressler, P. Handle, and C. Sommer, “Towards a vehicular cloud - Using parked vehicles as a temporary network and storage infrastructure,” in *Proceedings of the 2014 ACM international workshop on Wireless and mobile technologies for smart cities*, 2014, pp. 11–18.

[21] K. Ezirim and S. Jain, “Taxi-cab cloud architecture to offload data traffic from cellular networks,” in *Proceedings of the WoWMoM 2015: A World of Wireless Mobile and Multimedia Networks*, 2015, pp. 1–6.

[22] S. Abdelhamid, H. S. Hassanein, and G. Takahara, “Vehicle as a resource (VaaR),” *IEEE Netw.*, vol. 29, no. 1, pp. 12–17, 2015.

[23] E. Lee, E. K. Lee, M. Gerla, and S. Y. Oh, “Vehicular cloud networking: Architecture and design principles,” *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 148–155, 2014.

[24] F. Yang, S. Wang, J. Li, Z. Liu, and Q. Sun, “An overview of Internet of Vehicles,” *China Commun.*, vol. 11, no. 10, pp. 1–15, 2014.

[25] J. Contreras-Castillo, S. Zeadally, and J. A. Guerrero-Ibanez, “Internet of Vehicles: Architecture, Protocols, and Security,” *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3701–3709, 2018.

[26] S. K. Datta, “Vehicles as Connected Resources,” *Ieee Veh. Technol. Mag.*, no. June, pp. 26–35, 2017.

[27] H. J. Desirena Lopez, M. Siller, and I. Huerta, “Internet of vehicles: Cloud and fog computing approaches,” *2017 IEEE Int. Conf. Serv. Oper. Logist. Informatics*, pp. 211–216, 2017.

[28] E.-K. Lee, M. Gerla, G. Pau, U. Lee, and J.-H. Lim, “Internet of Vehicles: From intelligent grid to autonomous cars and vehicular fogs,”

Int. J. Distrib. Sens. Networks, vol. 12, no. 9, pp. 241–246, 2016.

[29] L. F. Bittencourt, J. Diaz-Montes, and R. Buyya, “Mobility-aware application scheduling in fog computing,” *Cloud Comput.*, 2017.

[30] P. Ghazizadeh, R. Mukkamala, and S. El-tawab, “Scheduling in Vehicular Cloud Using Mixed Integer Linear Programming,” *Proc. first Int. Work. Mob. sensing, Comput. Commun. ACM*, pp. 7–11, 2014.

[31] S. Abdelhamid, H. S. Hassanein, and G. Takahara, “Vehicle as a Mobile Sensor,” *Procedia - Procedia Comput. Sci.*, vol. 34, pp. 286–295, 2014.

[32] J. Ahnn and M. Potkonjak, “VeSense: Energy-Efficient Vehicular Sensing,” *Cs.Ucla.Edu*, no. 20112465, pp. 1–5, 2013.

[33] S. Abdelhamid, H. S. Hassanein, G. Takahara, and H. Farahat, “Caching-assisted access for vehicular resources,” in *39th Annual IEEE Conference on Local Computer Networks*, 2014, pp. 28–36.

[34] S. . Olariu, T. . Hristov, and G. . Yan, “the Next Paradigm Shift: From vehicular Networks To Vehicular Clouds,” in *Mobile ad hoc networking: cutting edge directions*, Second., Mobile Ad Hoc Networking: Cutting Edge Directions, 2013, pp. 645–700.

[35] W. Kim and M. Gerla, “NAVOPT: Navigator Assisted Vehicular route OPTimizer,” *Proc. - 2011 5th Int. Conf. Innov. Mob. Internet Serv. Ubiquitous Comput. IMIS 2011*, pp. 450–455, 2011.

[36] M. Gerla, J. T. Weng, and G. Pau, “Pics-on-wheels: Photo surveillance in the vehicular cloud,” *2013 Int. Conf. Comput. Netw. Commun. ICNC 2013*, pp. 1123–1127, 2013.

[37] M. Sookhak *et al.*, “Fog Vehicular Computing: Augmentation of Fog Computing Using Vehicular Cloud Computing,” *IEEE Veh. Technol. Mag.*, vol. 12, no. 3, pp. 55–64, 2017.

[38] K. Mershad and H. Artail, “Finding a STAR in a vehicular cloud,” *IEEE Intell. Transp. Syst. Mag.*, vol. 5, no. 2, pp. 55–68, 2013.

[39] H. R. Arkian, R. E. Atani, and S. Kamali, “FcVcA: A fuzzy clustering-

based vehicular cloud architecture," *2014 7th Int. Work. Commun. Technol. Veh. Nets4Cars-Fall 2014*, pp. 24–28, 2014.

[40] C. Caballero-Gil, P. Caballero-Gil, and J. Molina-Gil, "Self-Organized Clustering Architecture for Vehicular Ad Hoc Networks," *Int. J. Distrib. Sens. Networks*, vol. 2015, 2015.

[41] S. Vodopivec, J. Bester, and A. Kos, "A survey on clustering algorithms for vehicular ad-hoc networks," *2012 35th Int. Conf. Telecommun. Signal Process.*, pp. 52–56, 2012.

[42] K. Zheng, Q. Zheng, P. Chatzimisios, W. Xiang, and Y. Zhou, "Heterogeneous Vehicular Networking: A Survey on Architecture, Challenges, and Solutions," *IEEE Commun. Surv. Tutorials*, vol. 17, no. 4, pp. 2377–2396, 2015.

[43] A. Nasr and S. Mohamed, "Accurate Distance Estimation for VANET Using Nanointegrated Devices," *Opt. Photonics J.*, vol. 2, no. June, pp. 113–118, 2012.

[44] W. Zhu, D. Gao, and C. H. Foh, "An Efficient Prediction-Based Data Forwarding Strategy in Vehicular Ad Hoc Network," *Int. J. Distrib. Sens. Networks*, vol. 2015, 2015.

[45] C. Han, M. Dianati, R. Tafazolli, R. Kernchen, and X. Shen, "Analytical study of the IEEE 802.11p MAC sublayer in vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 873–886, 2012.

[46] P. Lahdekorpi, M. Hronec, P. Jolma, and J. Moilanen, "Energy efficiency of 5G mobile networks with base station sleep modes," *2017 IEEE Conf. Stand. Commun. Networking, CSCN 2017*, pp. 163–168, 2017.

[47] M. Soyturk, K. N. Muhammad, M. N. Avcil, B. Kantarci, and J. Matthews, "From vehicular networks to vehicular clouds in smart cities," in *Smart Cities and Homes: Key Enabling Technologies*, 2016.

[48] T. K. Refaat, B. Kantarci, and H. T. Mouftah, "Virtual machine migration and management for vehicular clouds," *Veh. Commun.*, vol.

4, pp. 47–56, 2016.

[49] E. Al-Rashed, M. Al-Rousan, and N. Al-Ibrahim, “Performance evaluation of wide-spread assignment schemes in a vehicular cloud,” *Veh. Commun.*, vol. 9, pp. 144–153, Jul. 2017.

[50] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, “Vehicular Fog Computing: A Viewpoint of Vehicles as the Infrastructures,” *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 3860–3873, 2016.

[51] A. M. Mustafa, O. M. Abubakr, O. Ahmadien, A. Ahmedin, and B. Mokhtar, “Mobility Prediction for Efficient Resources Management in Vehicular Cloud Computing,” in *Proceedings - 5th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering, MobileCloud 2017*, 2017, pp. 53–59.

[52] M. Aloqaily, I. Al Ridhawi, B. Kantraci, and H. T. Mouftah, “Vehicle as a resource for continuous service availability in smart cities,” *IEEE Int. Symp. Pers. Indoor Mob. Radio Commun. PIMRC*, vol. 2017-Octob, pp. 1–6, 2018.

[53] M. Aloqaily, “User Experience-Based Provisioning Services in Vehicular Clouds,” Université d’Ottawa/University of Ottawa, 2016.

[54] S. Abdel Hamid, M. Abu-Elkheir, H. S. Hassanein, and G. Takahara, “Towards provisioning vehicle-based rural information services,” in *Proceedings - Conference on Local Computer Networks, LCN*, 2012, pp. 835–842.

[55] K. Toczé and S. Nadjm-Tehrani, “A Taxonomy for Management and Optimization of Multiple Resources in Edge Computing,” *Wirel. Commun. Mob. Comput.*, 2018.

[56] J. Zare, S. Abolfazli, A. Alwadain, M. Shojafar, and A. Kamsin, “Resource Scheduling in Mobile Cloud Computing: Taxonomy and Open Challenges,” in *2015 IEEE International Conference on Data Science and Data Intensive Systems*, 2015, pp. 594–603.

[57] A. Atrey, N. Jain, and I. N. C. S. N, “A Study on Green Cloud

Computing," *Int. J. Grid Distrib. Comput.*, vol. 6, no. 6, pp. 93–102, 2013.

[58] J. M. H. Elmirghani *et al.*, "GreenTouch GreenMeter Core Network Energy-Efficiency Improvement Measures and Optimization," *J. Opt. Commun. Netw.*, vol. 10, no. 2, p. A250, 2018.

[59] L. Nonde, T. E. H. El-Gorashi, and J. M. H. Elmirghani, "Energy Efficient Virtual Network Embedding for Cloud Networks," *J. Light. Technol.*, vol. 33, no. 9, pp. 1828–1849, 2015.

[60] X. Dong, T. E. H. El-Gorashi, and J. M. H. Elmirghani, "On the energy efficiency of physical topology design for IP over WDM networks," *J. Light. Technol.*, vol. 30, no. 12, pp. 1931–1942, 2012.

[61] A. Q. Lawey, T. E. H. El-Gorashi, and J. M. H. Elmirghani, "Distributed energy efficient clouds over core networks," *J. Light. Technol.*, vol. 32, no. 7, pp. 1261–1281, 2014.

[62] X. Dong, T. El-Gorashi, and J. M. H. Elmirghani, "IP over WDM networks employing renewable energy sources," *J. Light. Technol.*, vol. 29, no. 1, pp. 3–14, 2011.

[63] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker, "Fog computing may help to save energy in cloud computing," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1728–1739, 2016.

[64] F. Jalali, R. Ayre, A. Vishwanath, K. Hinton, T. Alpcan, and R. Tucker, "Energy Consumption of Content Distribution from Nano Data Centers versus Centralized Data Centers," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 42, no. 3, pp. 49–54, 2014.

[65] W. Ejaz, M. Naeem, A. Shahid, A. Anpalagan, and M. Jo, "Efficient Energy Management for the Internet of Things in Smart Cities," *IEEE Commun. Mag.*, vol. 55, no. 1, pp. 84–91, 2017.

[66] E. Miluzzo, R. Cáceres, and Y.-F. Chen, "Vision mClouds – Computing on Clouds of Mobile Devices," in *Proceedings of the third ACM workshop on Mobile cloud computing and services - MCS '12*, 2012, p.

9.

[67] Y. Cui, X. Ma, H. Wang, I. Stojmenovic, and J. Liu, “A survey of energy efficient wireless transmission and modeling in mobile cloud computing,” *Mob. Networks Appl.*, vol. 18, no. 1, pp. 148–155, 2013.

[68] D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming*, Fourth., vol. 228. Springer, 2016.

[69] B. K. R Fourer, DM Gay, *AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press/Brooks/Cole Publishing Company, 2002.

[70] L. Gu, D. Zeng, S. Guo, and B. Ye, “Leverage parking cars in a two-tier data center,” *IEEE Wirel. Commun. Netw. Conf. WCNC*, pp. 4665–4670, 2013.

[71] S. Choo, I. Jang, J. Koo, J. Kim, and S. Pack, “Reliable vehicle selection algorithm with dynamic mobility of vehicle in vehicular cloud system,” *19th Asia-Pacific Netw. Oper. Manag. Symp. Manag. a World Things, APNOMS 2017*, pp. 319–321, 2017.

[72] L. Aminizadeh and S. Yousefi, “Cost minimization scheduling for deadline constrained applications on vehicular cloud infrastructure,” *Proc. 4th Int. Conf. Comput. Knowl. Eng. ICCKE 2014*, pp. 358–363, 2014.

[73] H. Zhang, Q. Zhang, and X. Du, “Toward vehicle-assisted cloud computing for smartphones,” *IEEE Trans. Veh. Technol.*, vol. 64, no. 12, pp. 5610–5618, 2015.

[74] Z. Zhou, P. Liu, Z. Chang, C. Xu, and Y. Zhang, “Energy-efficient workload offloading and power control in vehicular edge computing,” *2018 IEEE Wirel. Commun. Netw. Conf. Work. WCNCW 2018*, pp. 191–196, 2018.

[75] S. Midya, A. Roy, K. Majumder, and S. Phadikar, “Multi-objective optimization technique for resource allocation and task scheduling in vehicular cloud architecture: A hybrid adaptive nature inspired

approach," *J. Netw. Comput. Appl.*, vol. 103, no. July 2017, pp. 58–84, 2018.

[76] C. Zhu *et al.*, "Folo: Latency and Quality Optimized Task Allocation in Vehicular Fog Computing," *IEEE Internet Things J.*, vol. PP, no. c, p. 1, 2018.

[77] T. Adhikary *et al.*, "Quality of Service Aware Reliable Task Scheduling in Vehicular Cloud Computing," *Mob. Netw Appl*, vol. 21, pp. 482–493, 2016.

[78] F. Sun *et al.*, "Cooperative Task Scheduling for Computation Offloading in Vehicular Cloud," *IEEE Trans. Veh. Technol.*, vol. PP, no. c, pp. 1–1, 2018.

[79] A. Vishwanath Member, K. Hinton, R. W. A. Ayre, and R. S. Tucker, "Modeling energy consumption in high-capacity routers and switches," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 8, pp. 1524–1532, 2014.

[80] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright, "Power Awareness in Network Design and Routing."

[81] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, "A power benchmarking framework for network devices," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5550 LNCS, pp. 795–808, 2009.

[82] A. Shehabi *et al.*, "United States Data Center Energy Usage Report," *Lawrence Berkeley Natl. Lab. Berkeley, CA, Tech. Rep.*, no. June, pp. 1–66, 2016.

[83] D. Meisner, B. T. Gold, and T. F. Wenisch, "PowerNap: Eliminating server idle power," *ACM SIGPLAN Not.*, vol. 44, no. 3, pp. 205–216, 2009.

[84] D. A. Patterson and J. L. Hennessy, *Computer Organization & Design: The Hardware/Software Interface*, Third Edit. Morgan Kaufmann, 2016.

[85] C. Sawmilling and W. Planing, "Cisco Industrial Benchmark," *Cisco*,

2016. [Online]. Available: https://www.cisco.com/c/dam/global/da_dk/assets/docs/presentations/vBootcamp_Performance_Benchmark.pdf. [Accessed: 29-May-2019].

[86] “Intel® Xeon® Processor E5-2680 v2 (25M Cache, 2.80 GHz) Product Specifications.” [Online]. Available: https://ark.intel.com/content/www/us/en/ark/products/75277/intel-xeon-processor-e5-2680-v2-25m-cache-2-80-ghz.html. [Accessed: 30-Jun-2019].

[87] “Intel® Xeon® Processor E5-2630 v4 (25M Cache, 2.20 GHz) Product Specifications.” [Online]. Available: https://ark.intel.com/content/www/us/en/ark/products/92981/intel-xeon-processor-e5-2630-v4-25m-cache-2-20-ghz.html. [Accessed: 02-Jan-2020].

[88] “Intel® Xeon® Processor E5-2609 v4 (20M Cache, 1.70 GHz) Product Specifications.” [Online]. Available: https://ark.intel.com/content/www/us/en/ark/products/92990/intel-xeon-processor-e5-2609-v4-20m-cache-1-70-ghz.html. [Accessed: 02-Jan-2020].

[89] Rs-Components, “Raspberry Pi Model B,” 2019. [Online]. Available: https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2711/rpi_DATA_2711_1p0_preliminary.pdf. [Accessed: 05-Jan-2020].

[90] “MobiWAVE On-Board-Unit (OBU) MW-1000 OBU Family Leading-Edge Technology.” [Online]. Available: http://savari.net/wp-content/uploads/2017/05/MW-1000_April2017.pdf. [Accessed: 15-Nov-2017].

[91] “Cisco CRS-1 4-Slot Single-Shelf System.” [Online]. Available: https://www.cisco.com/c/en/us/products/collateral/routers/carrier-routing-system/product_data_sheet0900aecd804ff54e.pdf. [Accessed: 09-Nov-2017].

[92] Cisco, “Cisco Network Convergence System 5500 Series: Modular Line Cards NC55-36X100G,” 2018. [Online]. Available:

https://www.cisco.com/c/en/us/products/collateral/routers/network-convergence-system-5000-series/datasheet-c78-739657.html. [Accessed: 20-Nov-2019].

[93] Cisco, “Cisco Nexus 9300-EX Series Switches,” 2019. [Online]. Available: https://www.cisco.com/c/en/us/products/collateral/switches/nexus-9000-series-switches/datasheet-c78-742283.pdf. [Accessed: 28-Feb-2020].

[94] Huawei, “Huawei SmartAX MA5600T Global First All-in-one Access Platform,” 2015. [Online]. Available: http://huaweigpon.cz/wp-content/uploads/MA5600T.pdf. [Accessed: 20-Dec-2019].

[95] “FTE7502 10G ONU,” 2017. [Online]. Available: https://www.sumitomoelectric.com/wp-content/uploads/2016/01/OAN-7502-12212017-1.pdf. [Accessed: 15-Jul-2019].

[96] “ExtremeWireless ™ 3917i/e Outdoor Access Point,” 2018. [Online]. Available: https://cloud.kapostcontent.net/pub/a37e9d6b-44d9-4a54-b45d-d97b8c4ac7a0/ap3917-data-sheet.pdf?kui=GMr4scLR2vUnszuOwbPTug. [Accessed: 10-Jan-2019].

[97] “Cisco Nexus 9300-FX Series Switches,” 2020. [Online]. Available: https://www.cisco.com/c/en/us/products/collateral/switches/nexus-9000-series-switches/datasheet-c78-742284.pdf. [Accessed: 05-Nov-2019].

[98] “Cisco Network Convergence System 5500 Series: Modular Line Cards NC55-18H18F,” 2018. [Online]. Available: https://www.cisco.com/c/en/us/products/collateral/routers/network-convergence-system-5500-series/datasheet-c78-737776.pdf. [Accessed: 02-Jan-2020].

[99] “Cisco Nexus 9200 Platform Switches Data sheet Cisco public,” 2020. [Online]. Available: https://www.cisco.com/c/en/us/products/collateral/switches/nexus-

9000-series-switches/datasheet-c78-735989.pdf. [Accessed: 02-Jan-2020].

[100] "DIGI Wireless Vehicle BUS Adapter," 2017. [Online]. Available: https://www.digi.com/pdf/ds_wva.pdf. [Accessed: 14-Jan-2019].

[101] S. Zhao *et al.*, "Understanding energy efficiency in IoT app executions," in *Proceedings - International Conference on Distributed Computing Systems*, 2019, vol. 2019-July, pp. 742–755.

[102] G. Macario, M. Torchiano, and M. Violante, "An in-vehicle infotainment software architecture based on Google Android," *Proc. - 2009 IEEE Int. Symp. Ind. Embed. Syst. SIES 2009*, pp. 257–260, 2009.

[103] L. A. Adamic and B. A. Huberman, "Zipf's law and the Internet," *Glottometrics*, vol. 3, no. 1, pp. 143–150, 2002.

[104] S. Banerji and R. S. Chowdhury, "On IEEE 802.11: Wireless Lan Technology," *Int. J. Mob. Netw. Commun. Telemat.*, vol. 3, no. 4, pp. 45–64, 2013.

[105] C. White, *Data communications and computer networks: A business user's approach*. Cengage Learning, 2015.

[106] G. Kramer, B. Mukherjee, and G. Pesavento, "Ethernet PON (ePON): Design and Analysis of an Optical Access Network," *Photonic Netw. Commun.*, vol. 3, no. 3, pp. 307–319, 2001.

[107] K. C. Reichmann *et al.*, "160-km transparent metro WDM ring network featuring cascaded erbium-doped waveguide amplifiers," *IEEE Photonics Technol. Lett.*, vol. 13, no. 10, pp. 1130–1132, 2001.

[108] H. Q. Al-Shammari, A. Lawey, T. El-Gorashi, and J. M. H. Elmirghani, "Energy efficient service embedding in IoT networks," *2018 27th Wirel. Opt. Commun. Conf. WOCC 2018*, no. September, pp. 1–5, 2018.

APPENDIX A
Power Consumption and Processing Allocation Results

This Appendix includes extra scenarios to support the results of the Cloud-Fog-VEC architecture with multiple zones, explained in Section 4.64.6.3 (Chapter 4).

A.1 Scenario1: One task generated from one zone

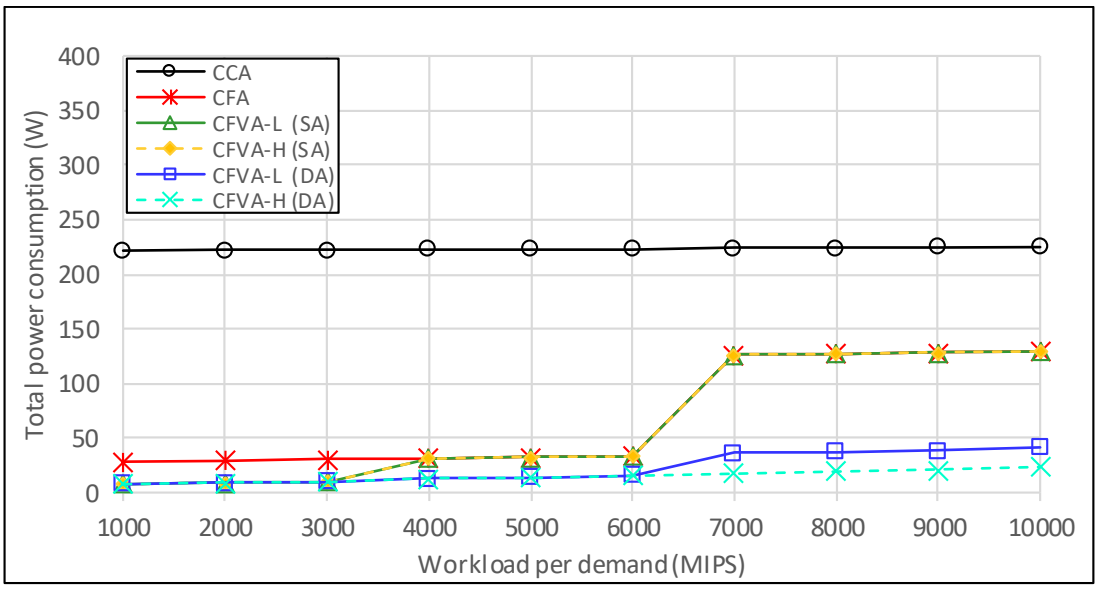

Figure A.1 Total power consumption, in Scenario 1 with multiple zones.

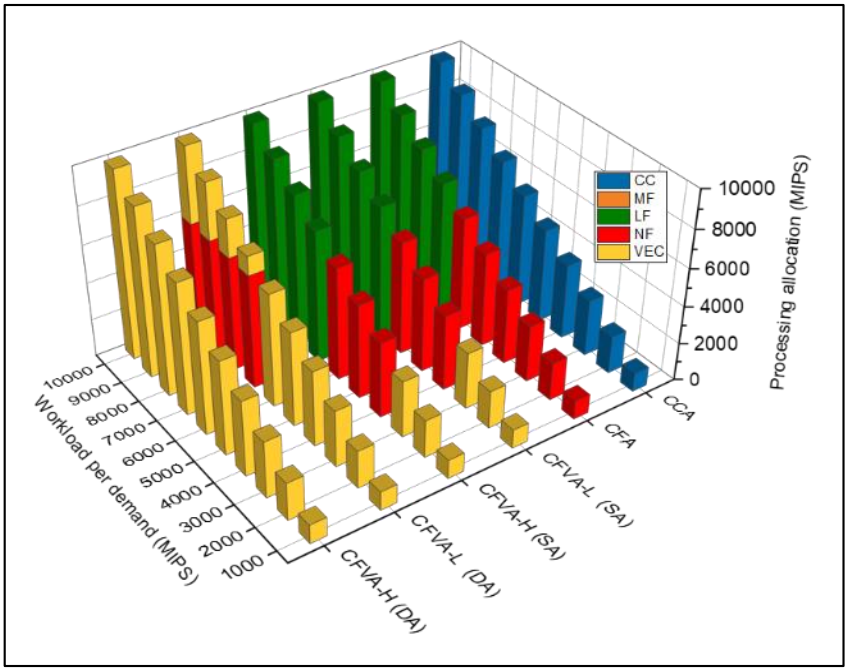

Figure A.2 Processing allocation in each PN, in Scenario1 with multiple zones

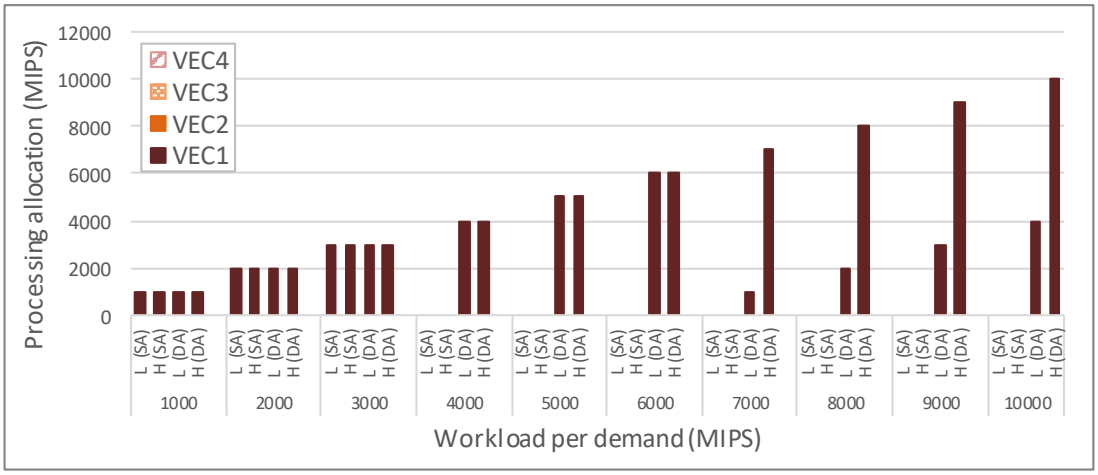

Figure A.3 Processing allocation in each VEC in CFVA (SA) and CFVA (DA), in Scenario 1 with multiple zones.

A.2 Scenario2: One task generated from each zone

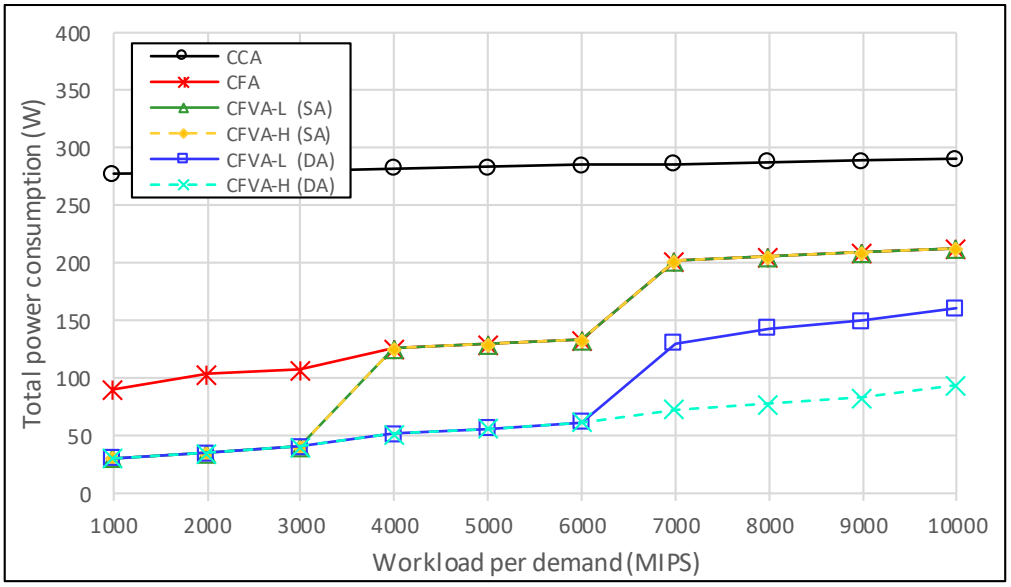

Figure A.4 Total power consumption, in Scenario 2 with multiple zones.

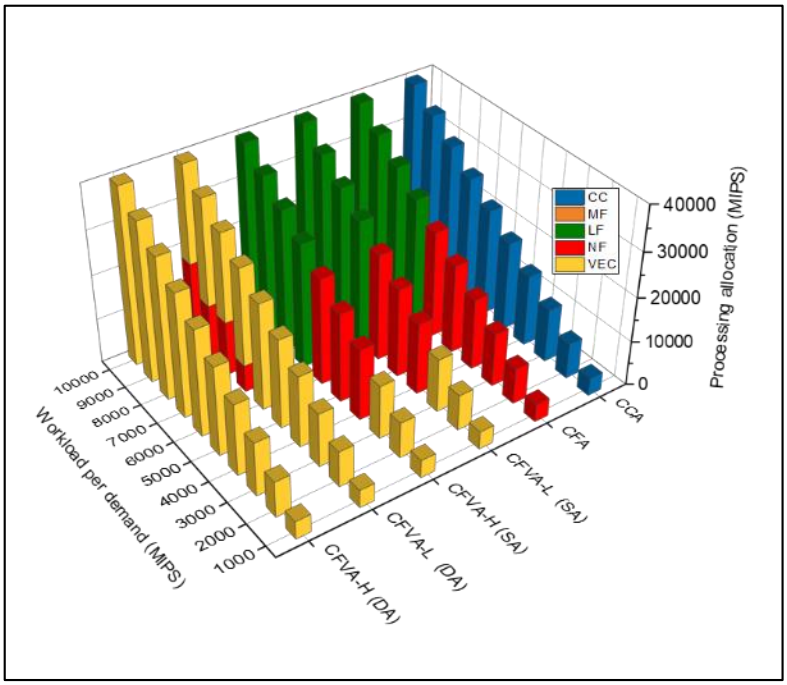

Figure A.5 Processing allocation in each PN, in Scenario 2, with multiple zones

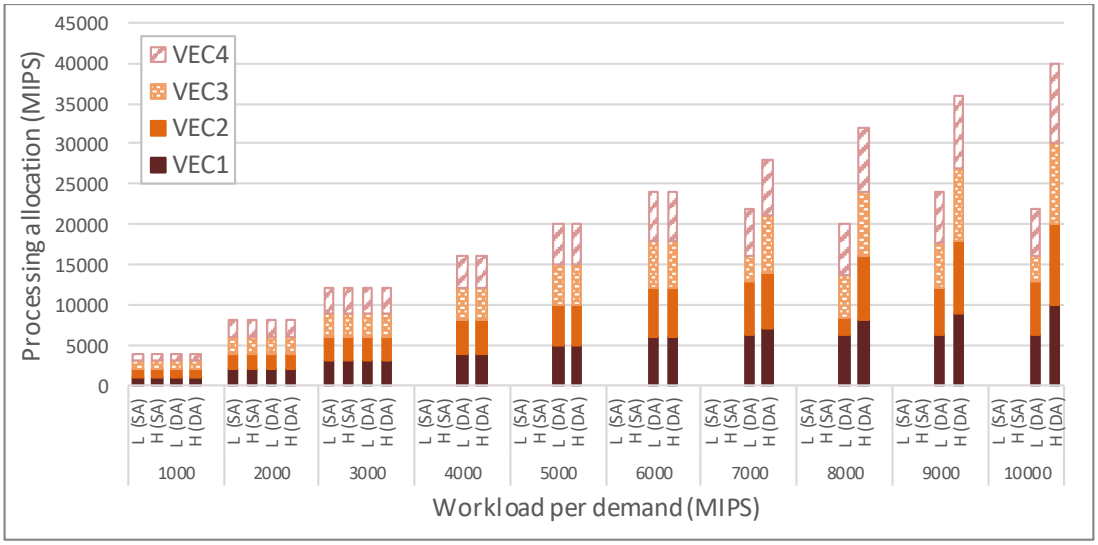

Figure A.6 Processing allocation in each VEC in CFVA (SA) and CFVA (DA), in Scenario 2 with multiple zones.

A.3 Scenario3: Five tasks generated from one zone

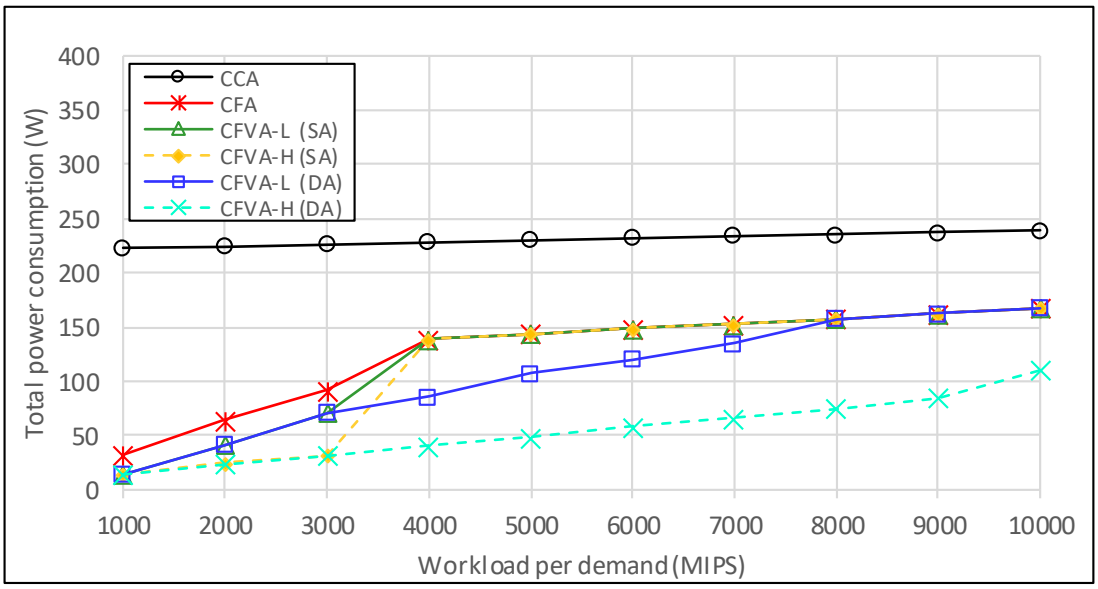

Figure A.7 Total power consumption, in Scenario 3 with multiple zones.

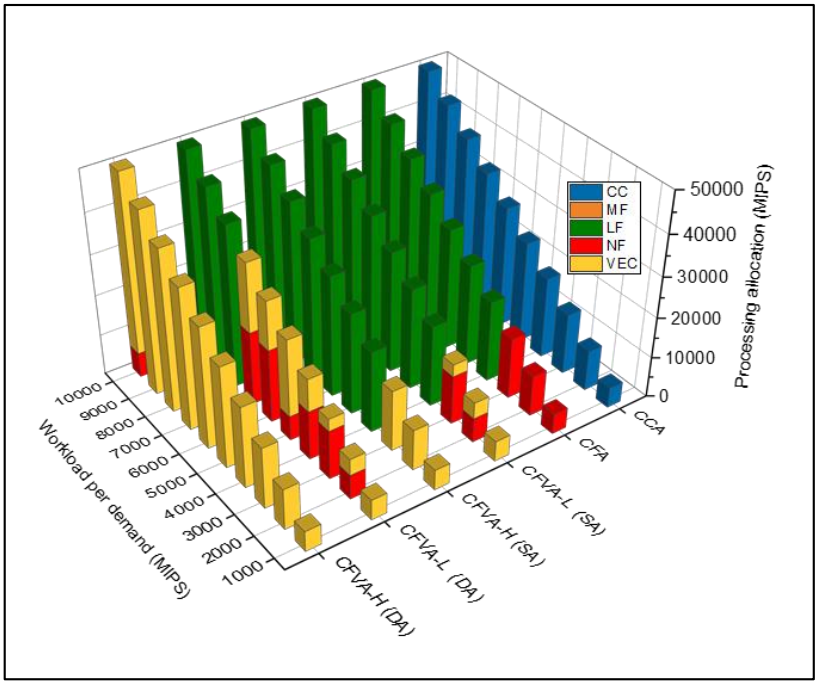

Figure A.8 Processing allocation in each PN, in Scenario 3 with multiple zones.

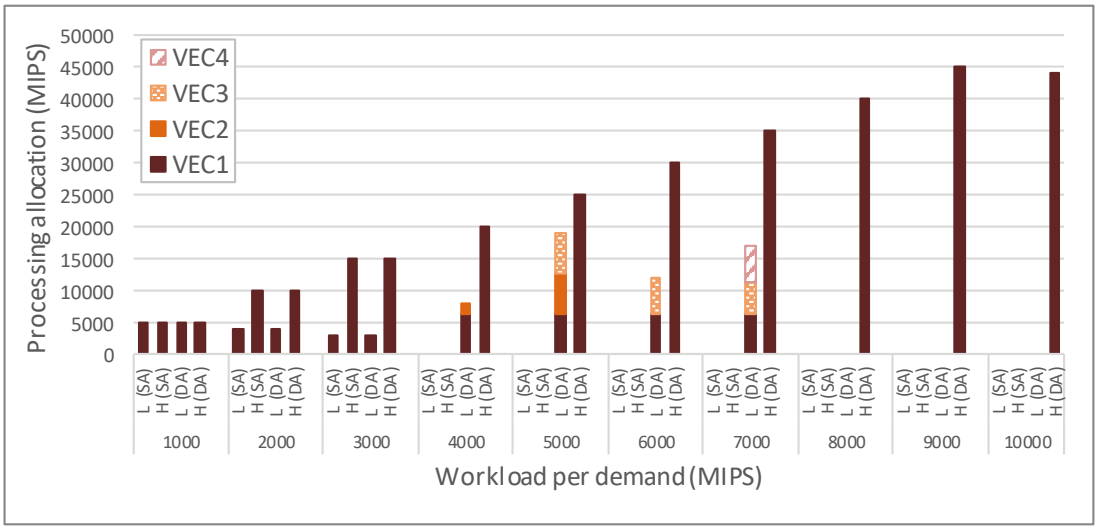

Figure A.9 Processing allocation in each VEC in CFVA (SA) and CFVA (DA), in Scenario 3 with multiple zones.

A.4 Scenario4: Five tasks generated from each zone

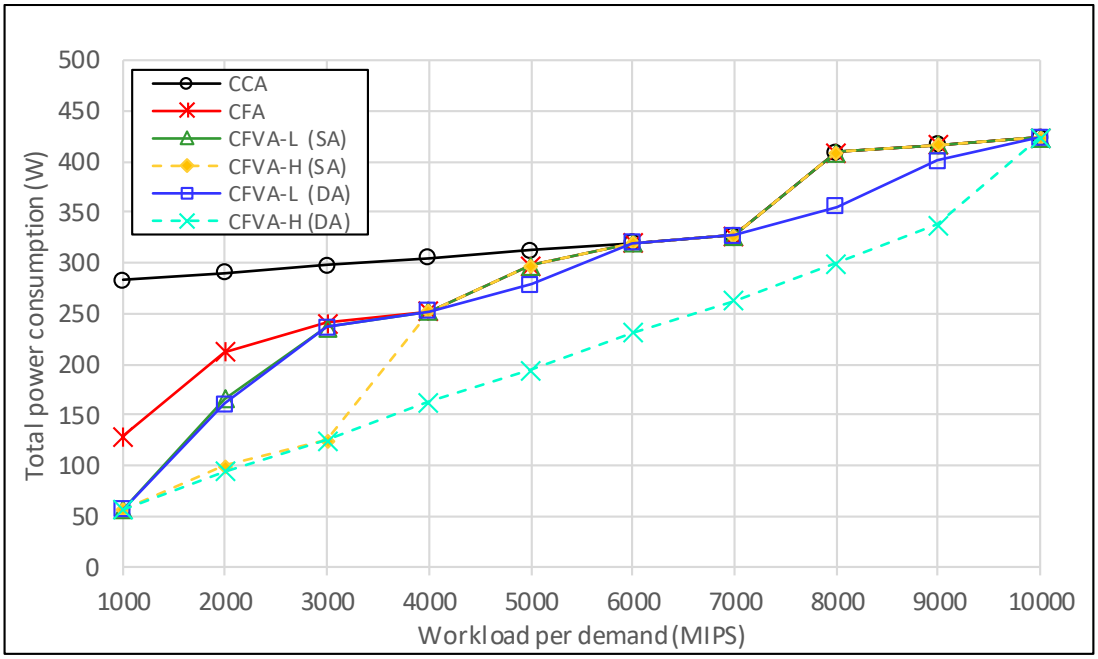

Figure A.10 Total power consumption, in Scenario 4 with multiple zones.

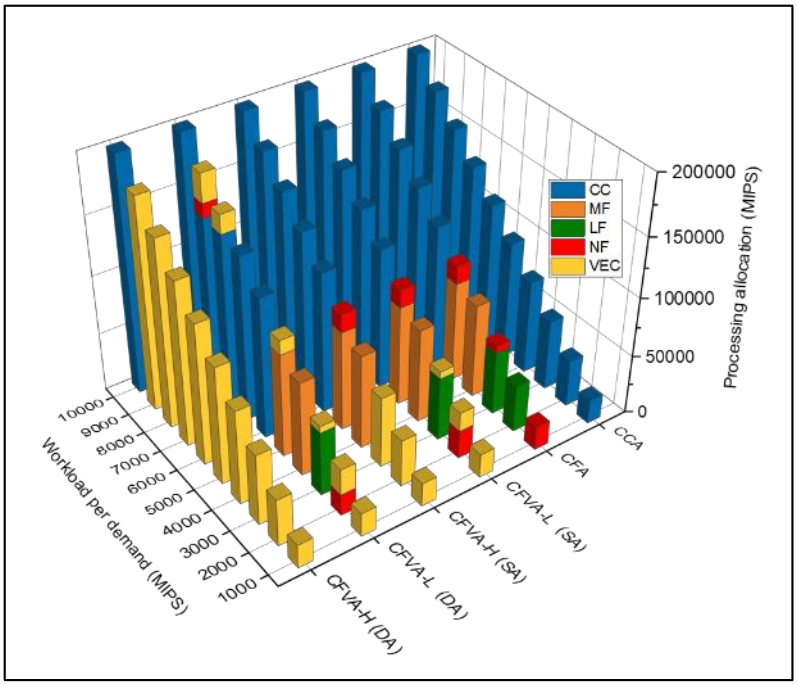

Figure A.11 Processing allocation in each PN, in Scenario 4 with multiple zones.

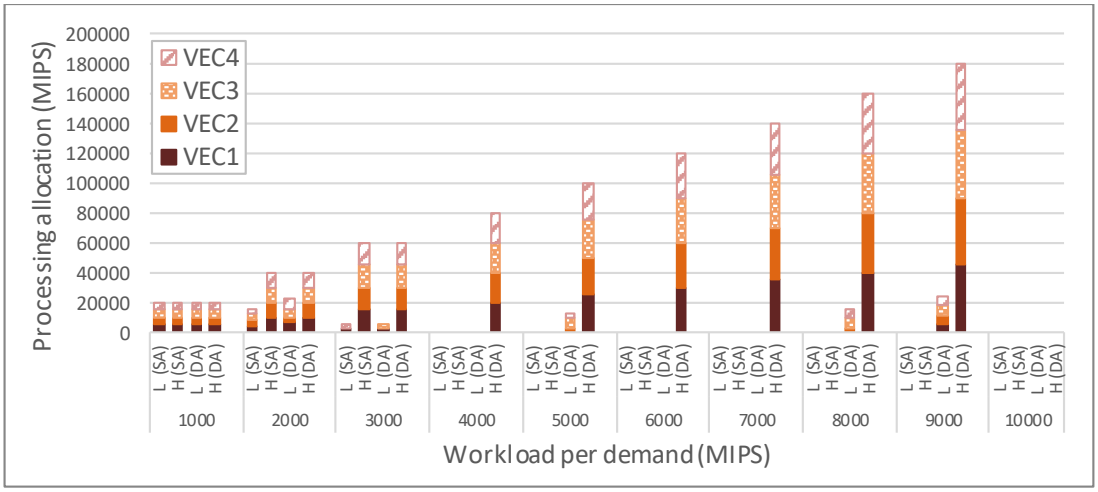

Figure A.12 Processing allocation in each VEC in CFVA (SA) and CFVA (DA), in Scenario 4 with multiple zones.

APPENDIX B

Lookup Tables for The Queuing Delay Linerisation

This Appendix includes the pre-defined lookup tables used in the delay-based MILP model in Chapter 6.

B.1 Lookup tables for the 100 Mb/s generated traffic

Possible allocated traffic Mb/s	Calculated delay (ms) at each service rate		
	40 Mb/s	40 Mb/s	40 Mb/s
10	0.0250063	0.1001001	0.2004
20	0.0250125	0.1002004	0.2008
30	0.0250188	0.1003009	0.2012
40	0.0250250	0.1004016	0.2016
50	0.0250313	0.1005025	0.2020
60	0.0250376	0.1006036	0.2024
70	0.0250438	0.1007049	0.2028
80	0.0250501	0.1008065	0.2033
90	0.0250564	0.1009082	0.2037
100	0.0250627	0.1010101	0.2041

B.2 Lookup tables for the 200 Mb/s generated traffic

Possible allocated traffic Mb/s	Calculated delay (ms) at each service rate		
	40 Mb/s	40 Mb/s	40 Mb/s
20	0.0250125	0.1002004	1.0204
40	0.0250250	0.1004016	1.0417
60	0.0250376	0.1006036	1.0638
80	0.0250501	0.1008065	1.0870
100	0.0250627	0.1010101	1.1111
120	0.0250752	0.1012146	1.1364
140	0.0250878	0.1014199	1.1628
160	0.0251004	0.1016260	1.1905
180	0.0251130	0.1018330	1.2195
200	0.0251256	0.1020408	1.2500

B.3 Lookup tables for the 300 Mb/s generated traffic

Possible allocated traffic Mb/s	Calculated delay (ms) at each service rate		
	40 Mb/s	40 Mb/s	40 Mb/s
30	0.0250188	0.1003009	1.0309
60	0.0250376	0.1006036	1.0638
90	0.0250564	0.1009082	1.0989
120	0.0250752	0.1012146	1.1364
150	0.0250941	0.1015228	1.1765
180	0.0251130	0.1018330	1.2195
210	0.0251319	0.1021450	1.2658
240	0.0251509	0.1024590	1.3158
270	0.0251699	0.1027749	1.3699
300	0.0251889	0.1030928	1.4286

B.4 Lookup tables for the 400 Mb/s generated traffic

Possible allocated traffic Mb/s	Calculated delay (ms) at each service rate		
	40 Mb/s	40 Mb/s	40 Mb/s
40	0.0250250	0.1004016	1.0417
80	0.0250501	0.1008065	1.0870
120	0.0250752	0.1012146	1.1364
160	0.0251004	0.1016260	1.1905
200	0.0251256	0.1020408	1.2500
240	0.0251509	0.1024590	1.3158
280	0.0251762	0.1028807	1.3889
320	0.0252016	0.1033058	1.4706
360	0.0252270	0.1037344	1.5625
400	0.0252525	0.1041667	1.6667

B.5 Lookup tables for the 500 Mb/s generated traffic

Possible allocated traffic Mb/s	Calculated delay (ms) at each service rate		
	40 Mb/s	40 Mb/s	40 Mb/s
50	0.0250313	0.1005025	1.0526
100	0.0250627	0.1010101	1.1111
150	0.0250941	0.1015228	1.1765
200	0.0251256	0.1020408	1.2500
250	0.0251572	0.1025641	1.3333
300	0.0251889	0.1030928	1.4286
350	0.0252207	0.1036269	1.5385
400	0.0252525	0.1041667	1.6667
450	0.0252845	0.1047120	1.8182
500	0.0253165	0.1052632	2.0000

B.6 Lookup tables for the 600 Mb/s generated traffic

Possible allocated traffic Mb/s	Calculated delay (ms) at each service rate		
	40 Mb/s	40 Mb/s	40 Mb/s
60	0.0250376	0.1006036	1.0638
120	0.0250752	0.1012146	1.1364
180	0.0251130	0.1018330	1.2195
240	0.0251509	0.1024590	1.3158
300	0.0251889	0.1030928	1.4286
360	0.0252270	0.1037344	1.5625
420	0.0252653	0.1043841	1.7241
480	0.0253036	0.1050420	1.9231
540	0.0253421	0.1057082	2.1739
600	0.0253807	0.1063830	2.5000

B.7 Lookup tables for the 700 Mb/s generated traffic

Possible allocated traffic Mb/s	Calculated delay (ms) at each service rate		
	40 Mb/s	10	40 Mb/s
70	0.0250438	0.1007049	1.0753
140	0.0250878	0.1014199	1.1628
210	0.0251319	0.1021450	1.2658
280	0.0251762	0.1028807	1.3889
350	0.0252207	0.1036269	1.5385
420	0.0252653	0.1043841	1.7241
490	0.0253100	0.1051525	1.9608
560	0.0253550	0.1059322	2.2727
630	0.0254001	0.1067236	2.7027
700	0.0254453	0.1075269	3.3333

B.8 Lookup tables for the 800 Mb/s generated traffic

Possible allocated traffic Mb/s	Calculated delay (ms) at each service rate		
	40 Mb/s	40 Mb/s	40 Mb/s
80	0.0250501	0.1008065	1.0870
160	0.0251004	0.1016260	1.1905
240	0.0251509	0.1024590	1.3158
320	0.0252016	0.1033058	1.4706
400	0.0252525	0.1041667	1.6667
480	0.0253036	0.1050420	1.9231
560	0.0253550	0.1059322	2.2727
640	0.0254065	0.1068376	2.7778
720	0.0254582	0.1077586	3.5714
800	0.0255102	0.1086957	5.0000

B.9 Lookup tables for the 900 Mb/s generated traffic

Possible allocated traffic Mb/s	Calculated delay (ms) at each service rate		
	40 Mb/s	40 Mb/s	40 Mb/s
90	0.0250564	0.1009082	1.0989
180	0.0251130	0.1018330	1.2195
270	0.0251699	0.1027749	1.3699
360	0.0252270	0.1037344	1.5625
450	0.0252845	0.1047120	1.8182
540	0.0253421	0.1057082	2.1739
630	0.0254001	0.1067236	2.7027
720	0.0254582	0.1077586	3.5714
810	0.0255167	0.1088139	5.2632
900	0.0255754	0.1098901	10.0000

B.10 Lookup tables for the 1000 Mb/s generated traffic

Possible allocated traffic Mb/s	Calculated delay (ms) at each service rate		
	40 Mb/s	10 Mb/s	1 Mb/s
100	0.0250627	0.1010101	1.1111
200	0.0251256	0.1020408	1.2500
300	0.0251889	0.1030928	1.4286
400	0.0252525	0.1041667	1.6667
500	0.0253165	0.1052632	2.0000
600	0.0253807	0.1063830	2.5000
700	0.0254453	0.1075269	3.3333
800	0.0255102	0.1086957	5.0000
900	0.0255754	0.1098901	10.0000
1000	0.0256410	0.1111111	1000.0000