On the Planning and Design Problem of Fog Networks

by

Faisal Haider

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of the requirements for the degree of

Master of Applied Science

in

Electrical and Computer Engineering

Carleton University
Ottawa, Ontario

Abstract

Fog computing is a paradigm of geographically distributed computing residing at the edge of the network. Fog computing is made up of fog nodes providing compute, storage and networking services to end users. In this thesis, we propose an exact model for the planning and design problem of fog networks. The model simultaneously determines the optimal location, the capacity and the number of fog node(s) as well as the interconnection between the installed fog nodes and the cloud, while minimizing the delay in the network and the amount of traffic going to the cloud. To address this multiobjective problem, three multiobjective optimization methods (weighted sum, hierarchical and trade-off) are evaluated. The CPLEX solver was used to optimize the model for the three methods with different problem sizes and the results are analyzed. The results show that, as the input size increases, the delay and the traffic also increase in a linear form; whereas the solution time increases in non-polynomial time. The weighted sum method was able to achieve the best trade-off results for the delay and the traffic, whereas the hierarchical method was able to return minimum delay but with worse traffic going to the cloud. As the model considers realistic edge device traffic parameters and constraints, it can be helpful in deploying fog networks in the current cloud computing architecture.

Acknowledgments

First and foremost, I am extremely grateful to my supervisor Prof. Marc St-Hilaire for believing in me when I contacted him from the other side of the world. This work would not have been possible without his invaluable inspiration, guidance and friendly availability, in spite of his busy schedule. It has been great honour and privilege to work under your supervision.

I would like to thank Dr. Christian Makaya for his guidance and wise suggestions throughout the whole period of my research. I am also grateful to my friend and mentor Dr. Muhammad Raisul Alam for his insightful suggestions and input on optimization and his support to make Canada a second home for me. I acknowledge with gratitude Decheng Zhang for being extremely supportive and sharing his knowledge from the beginning of this research.

I would like to extend my thanks to all my friends and colleauges of our lab for providing a relaxed and engaging research environment. Special thanks goes to Ammar, Jean-Daniel and Afsane. My heartfelt appreciation goes to my friend Shoeb Shaikh and Jamil Sayeed, who always stood by me with smile and good wishes.

Finally, I owe the greatest debt to my parents for their never-ending affection, love and patience to become who I am today. It is my great pleasure to dedicate this thesis to Saif, Salma, Luthful and my parents.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AHP	Analytic Hierarchy Process
API	Application Program Interface
APN	Access Point Name
AWS	Amazon Web Services
B&B	Branch and Bound
BS	Base Station
CEP	Complex Event Processing
CI	Confidence Interval
CoT	Cloud of Things
DC	Data Center
EC2	Elastic Compute Cloud
ECG	Electrocardiogram
ETC	Electronic Toll Collection
FPP	Fog Planning Problem
GCE	Google Compute Engine
GH	Greedy Heuristic
GPON	Gigabit Passive Optical Network
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol

HV	Hypervolume
IaaS	Infrastructure-as-a-Service
IETF	Internet Engineering Task Force
IoT	Internet of Things
IP	Internet Protocol
IT	Information Technology
MCC	Minimum Correlation Coefficient
MDC	Micro Data Center
MEC	Mobile Edge Computing
MILP	Mixed Integer Linear Program
nDC	Nano Data Center
NIST	National Institute of Standards and Technology
NP	Nondeterministic Polynomial
OPL	Optimization Programming Language
OR	Operation Research
PaaS	Platform-as-a-Service
PKI	Public Key Infrastructure
PM	Physical Machine
QoS	Quality of Service
RAN	Radio Access Network
RFID	Radio-frequency Identification
SaaS	Software-as-a-Service
SDN	Software Defined Network
SLA	Service Level Agreement

TEE	Trusted Execution Environment
V2V	Vehicle-to-Vehicle
vCPU	Virtual Central Processing Unit
VM	Virtual Machine
WDM	Wavelength Division Multiplexing

Chapter 1

Introduction

Cloud computing is a computing paradigm, where infrastructure of dynamic scalability is provided for data, file storage and applications. Cloud computing is used by a large pool of connected private or public networks. With massive storage and computation capacity, cloud computing has been a driving force in Internet of Things (IoT) providing a variety of services. With the evolution of machine learning applications, big data and the increased number of connected devices, the cloud model alone is not applicable for time-critical operations. Issues of performance, security, reliability and high consumption of bandwidth in the cloud promoted the idea of fog computing.

Fog computing can be defined as a non-trivial extension of the cloud residing on the edge of the network. Fog computing provides the same services as the cloud in a limited or full service. It is a Micro Data Center (MDC) focused on applications and services which are broadly distributed. An MDC is a small, containerized Data Center (DC). Unlike the traditional cloud DC, an MDC can be as small as a19 inch box with less than 10 servers and 100 Virtual Machines (VM) [1]. Fog computing is a highly virtualized architecture providing services like computation, storage and networking between end-devices in an IoT and Cloud of Things (CoT). Unlike the centralized technology of cloud, fog computing is targeted towards widely distributed applications and services. Fog applications have a distributed directory system to communicate with mobile devices acknowledging mobility techniques. Fog computing supports low latency applications such as augmented reality, gaming, video streaming, etc. The widespread geographical distribution of fog nodes, defined as the location where several physical devices provide resources such as virtual Central Processing Unit (vCPU), memory and storage to various services [2], will deliver high

quality streaming services to mobile nodes and moving vehicles through proxies and access points positioned in vital places such as tracks and highways. The cloud will always have an important role in IoT. Hence, by providing an improved Quality of Service (QoS) and reducing the amount of traffic in the cloud, fog computing is not a replacement but a complement to the cloud architecture.

1.1 Problem Statement

With the complexity of networks and the large number of devices, service providers need efficient planning tools to design optimal networks. Efficient network planning tools can help in designing optimal networks guaranteeing better service to the end users. The fundamental goal of network planning is to implement an optimum network addressing realistic traffic parameters. Unlike earlier generations, computer-based automatic planning tools are now required for large networks. The network design problem can be solved by using simulation or solving mathematical models. Simulation is typically used when a mathematical model cannot be developed for a network design problem. A basic representation of the network is modeled in a tool where the simulated network is examined with different inputs and parameters. Unfortunately, most network planning problems are NP-hard [3, 4].

Network planning is a massive research field with many applications in IP networks, optical networks, wireless networks, cellular networks, cloud computing, etc. To the best of our knowledge, no work has considered the planning and design of fog networks in collaboration with the cloud. It can be recognized that the fog nodes will be the distributed fog network entities enabling the deployment of fog services. For a normal fog network, there will be a lot of fog nodes to be installed and to deliver the expected performance improvements, proper planning of these fog nodes is necessary. Consequently, many factors such as the number of nodes, locations, node sizing, connectivity between fog and cloud, traffic distribution and so on need to be considered. Given the large number of fog nodes in the network combined with a vast number of edge devices over the entirety of the geographical region of interest, the search space for an optimal solution is huge. Therefore, sophisticated methods and algorithms are required to help network planners in the decision making process.

The purpose of the fog network is to improve the end user experience by reducing parameters like delay in the network, the amount of traffic in the cloud and so

on. Optimizing more than one parameters converts the problem into a multiobjective optimization problem resulting in a number of non-dominated solutions. The multiobjective optimization problem can be studied from different viewpoints with different solution goals while solving it. Hence, it is very hard to find a best compromised solution for all the objective functions even with a comprehensive knowledge of the problem.

There are different methods to solve multiobjective optimization problems. However, it is a very complex task to choose the best method as different methods have various requirements and can perform differently with different problems. Ideally, for the best convergence towards the Pareto set, the optimization method that handles the minimum amount of complexity should be used [5, 6]. But defining the amount of complexity is not straightforward. To mitigate that, a number of methods can be used simultaneously to understand and solve the problem. Moreover, the planning problem can be solved with exact and approximate multiobjective algorithms, each having its own advantages and disadvantages.

The fog network planning is a complex but necessary step towards building efficient fog networks. To that end, this thesis addresses the above mentioned problems and proposes an exact mathematical model with different multiobjective optimization methods for the planning and design of fog networks.

1.2 Research Objectives

According to the problem statement explained in the previous section, the main objective of the thesis is to develop a tool to help network providers with the planning and design of fog networks. More precisely, we aim to achieve the following two sub-objectives:

- Propose a mathematical model for the planning and design of fog networks from a green field scenario. The model will simultaneously determine the optimal location, the number, and the capacity of the fog node(s) and the interconnection between the installed fog node(s) and the cloud while considering edge device requests. The goal of the model is to minimize the delay in the network and the amount of traffic going to the cloud.

- Evaluate the performance of the proposed model and analyze the results with different optimization methods. The three multiobjective optimization methods

are the weighted sum, the hierarchical and the trade-off method. The goal of this sub-objective is to compare the three multiobjective optimization methods and find the best method for appropriate scenarios.

1.3 Methodology

To achieve our objectives, we will first review the different concept of cloud computing. This will help in realizing the weakness of the current cloud architecture and the need of fog computing for IoT. The current contributions in fog computing will also be reviewed to have an updated view of the state-of-the-art. The following methodology will be used in approaching the planning problem of fog networks.

- Generate a snapshot of traffic with realistic parameters: The requested amount of traffic needs to be known to properly plan fog networks. To generate the traffic, a small program will be developed taking into consideration different aspects like number of packets, amount of memory/vCPU requested, etc.

- Develop the mathematical model: In order to develop the mathematical model, a set of mathematical equations will be written to represent the architecture of fog networks. The goal of the model will be to minimize the delay experienced by the users and minimize the amount of traffic that must go to the cloud. Different assignment, capacity and uniqueness constraints will be taken into consideration to create a realistic fog network scenario. A set of decision variables will be used in determining the type and number of fog node(s) to be placed in the network. Since we will cast the problem as a multiobjective optimization problem, a multiobjective Pareto-optimization approach will be adapted to find the optimal fog network deployment that considers both objective functions. The model will be written in OPL and a solver called CPLEX will be used to solve the problem. The solver will return the optimal solution or the best solution found when it reaches its time limit.

- Analyze the results of the fog planning problem: A detailed example will be used to explain the planning problem. Then, using CPLEX, we will solve the planning problem for the three methods with different problem sizes. The result analysis will investigate the comparison of the three optimization methods with

respect to traffic, delay and solution time. Next, the Pareto-efficiency of the three methods will be analyzed for each problem size.

1.4 Thesis Contributions

The main contribution of this thesis is to to develop an exact model for the planning and design of fog networks. The major contributions of this thesis are summarized as follows:

- A mathematical model is proposed for the planning and design of fog networks for a brand new infrastructure in current cloud architecture. The main contribution of the model is the consideration of realistic edge device request parameters like vCPU, memory, etc. in determining the optimal location, capacity, type and number of fog node(s). Moreover, the optimal interconnection between the fog node(s) and the cloud is also determined by the planning model. As the model considers the delay in the network and the traffic going to the cloud, the planning tool will help fog service providers to deploy fog node(s) in the existing cloud architecture in an efficient manner with an improved QoS.
- As there are two objective functions, the main intention is to study different multiobjective optimization techniques to address the Pareto optimality and trade-off between the objective functions. As a result, three multiobjective optimization methods are used in solving the fog planning problem. Then, the results are analyzed and compared for the three methods with the weighted sum method achieving the best trade-off results for the two objective functions.

1.5 Thesis overview

The rest of the thesis is organized as follows. In Chapter 2, a literature review of cloud and fog computing is conducted. Chapter 3 introduces the planning problem followed by the mathematical model to solve the fog planning problem. Then, the mathematical model is modified to the three multiobjective optimization models. Chapter 4 presents the simulation results for the three modified models compared in terms of objective functions, solution time and Pareto-efficiency. Finally, the thesis

is concluded in Chapter 5 by summarizing the findings and proposing future research directions.

Chapter 2

Background and Related Work

Since the concept of fog computing is an extension of cloud computing, it is important to understand different notions related to cloud computing. To that end, Section 2.1 first discusses the history and the current architecture of cloud computing and then introduces the concept of fog computing. Section 2.2 reviews some of the applications and current research specifically related to fog computing. Section 2.3 reviews the network planning and design process and finally we summarize the chapter in Section 2.4.

2.1 Introduction to Cloud Computing

Cloud computing is experiencing an exponential growth over the years. As the cloud service enables new and efficient business models, small and medium businesses are integrating to the cloud platform. The word cloud was commonly used as a metaphor for Internet and a network on telephone schematics was usually illustrated with a standardized cloud like shape. Later, it was used to label the Internet in computer network diagrams. Sometime around 1955, John McCarthy came upon with the time sharing theory, which is very similar to present day cloud computing. In 1970s, IBM released an operating system called VM that allowed users to have more than one virtual systems (or VMs) on a single physical node. The 50s time sharing model was taken to a whole new level by the VM operating system and many of the elemental functions in virtualization software that are presently used can be traced back to this early VM operating system. To provide more users through shared connection to the same physical infrastructure, in 1990s, virtualized private network connections were offered by telecommunication companies. This development facilitated a better

network equity and higher control over bandwidth usage by adjusting the traffic as required. In the meantime, with the fervent start in virtualization for PC-based systems, and with the availability of Internet,the integration of virtualization was the next logical step. In 1997, Professor Ramnath Chellappa coined the term cloud computing in a talk on new computing paradigm. In 2002, with the creation of Amazon Web Services (AWS), Amazon provided an advanced system of cloud services accommodating storage to computation. Later, in 2006, the Elastic Compute Cloud (EC2) was introduced by Amazon as a commercial web service. The EC2 enabled renting of computers by small companies on which their own computer applications could be run. In 2009, the Google App Engine brought low-cost computing and storage services, and Microsoft followed with Azure.

2.1.1 Definitions

Although cloud computing is widely acknowledged, many researchers suggested their own definition with respect to their research domain. The National Institute of Standards and Technology (NIST) defines cloud computing as a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [7]. Sotomayor et al. denote that cloud is more often used to refer to the IT infrastructure deployed on an infrastructure as a service provider data center [8]. Vasquero et al. mention that clouds are a large pool of easily usable and accessible virtualized resources (such as hardware, development platforms and/or services). These resources can be dynamically reconfigured to adjust to a variable load (scale), allowing also for an optimum resource utilization. This pool of resources is typically exploited by a pay as you go model in which guarantees are offered by the infrastructure provider by means of customized Service Level Agreement (SLA) [9]. As can be seen, it is hard to come up with a single definition of cloud computing. In fact, cloud computing is not a new technology, but rather brings together a number of standing technologies with a new operational model to run business in a new dimension. Certainly, many cloud computing technologies, such as virtualized environments and utility computing, are not new. Instead, these existing technologies are leveraged to fulfill the economic and technological demand required by today's network environment. [7].

2.1.2 Related Technologies

Cloud Computing emerged from years of research in various similar technologies. The following technologies are usually compared with cloud computing, each sharing some similarities with cloud computing.

Grid Computing

Grid computing is a model of distributed computing where computing, data, storage, application or network resources are coordinated and shared over dynamic and geographically dispersed organization [10]. There are some similar features between cloud computer and grid computer such as:

- Multitask and multitenancy are part of both computing technologies, implying that customers can perform various tasks, gaining access to single or multiple application instances. Peak load capacity and infrastructure costs can be reduced by sharing resources among a large number of users. Cloud and grid computing administer SLAs for guaranteed uptime availability, for instance, 99 percent. When the service declines below the level of guaranteed up time service, service credit is offered to the consumer for receiving late data.

- Scalability is another common feature between cloud and grid computing. Scalability is achieved through load balancing of application instances connected through web services and running separately on a variety of operating systems. CPU and network bandwidth are allotted and de-allotted on demand. Based on the number of users, instances, and the volume of data transferred at a particular time, the systems storage capacity fluctuates [11].

Similarly, the major differences between grid and cloud are as follows:

- Business Models Business models in grid computing are generally based on mutual agreements between the service provider and the consumer, whereas provision of resources in cloud computing constitutes of more differentiated business models ranging from service provider to service provider and mixed approach.

- Resource Management: While grids depend on batch systems, clouds resource management solution is represented by the application of virtualization technologies.

- Resource Provision Model: The communications in grid resource provision models are established offline relying on virtual organizations. On the otherside, the usage of SLAs, trust management and compliance are the key to resource provision models in clouds.

- Resource Availability: Sometimes, there are plenty of resources in grids which are idle and at some other times, there are no available resources, therefore the resources are shared on the best-effort delivery. The matter of concern in clouds are to find the balance between utilization of resources to facilitate productive consumption of resources and reducing energy consumption on the one hand and wasting resources due to the standby mode of devices and virtualization overhead on the other.

Utility Computing

Utility computing comprises the on demand basis of renting computer resources such as network bandwidth, hardware and software as-required [12]. There is a similarity between utility and cloud computing as both concepts entertain the idea of leasing computer technology. But they have some major differences between them. One of the fundamental differences between utility and cloud computing relates to the behavior of the leasing. While a third party is used for software and infrastructure for both technology, these services are much more directly accessed by utility computing. This form of computing makes the use of technology like another utility, such as gas or electricity, and by the end of the month, businesses would be charged according to the usage. Though, all the services in cloud computing are still rented, the source of the services are less known by the companies. Users still pay for what they use, but the company providing the service utilizes a much more complex system of infrastructure and software, usually involving a grid network that supports multiple tasks at once. Another difference is that utility computing is dependent on basic computing practices, often utilizing traditional programming styles in a well-established business context. Cloud computing, in contrast, involves creating an entirely distinctive virtual computing environment that empowers programmers and developers in new ways.

Autonomic Computing

Autonomic computing was introduced by IBM as a term to describe a self-managing system that tries to get involved in computing system. IBM divided the self-managing system into four categories: self-optimizing, self-healing, self-configuring, and self-protecting [13]. The main difference between autonomic and cloud computing is that autonomic computing is focused on reducing the system complexity. Whereas, the objective of cloud computing focuses on decreasing the resource cost rather than to diminish system complexity. As there will be a lot of pre-processing done in the fog nodes, the reduction of resource usage in the cloud will help in focusing to a more automated system.

Virtualization

Virtualization is a technology providing virtualized resources for high-level applications by abstracting away the specifics of physical hardware. The common name of a virtualized server is virtual machine or VM [7]. Virtualization abstracts compute resources as VMs with association of storage and networking connectivity. The cloud determines how those virtualized resources are allocated, delivered and presented. Virtualization facilitates rapid scaling of resources which is hard to do by non-virtualized environments.

2.1.3 Service Models

Cloud services can be classified into three distinct models: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS). Each one is described below.

Infrastructure-as-a-Service

IaaS is a vastly automated and standardized service where computer resources are hosted and owned by service providers and provided to customers on-demand, integrating storage and networking capabilities. This infrastructure can be self-provisioned by customers, using a Web-based graphical user interface that accommodates the overall environment as an IT operations management console [14].

Typical examples of IaaS include: Joyent, Google Compute Engine (GCE), Microsoft Azure, Cisco Metapod, Amazon Web Services (AWS) and IBM SoftLayer.

Common use-case: Expands present data center infrastructure for temporary workloads (e.g. increased Canada Day holiday site traffic).

Platform-as-a-Service

PaaS allows developers to build applications and services over the Internet by providing a platform and environment. PaaS services being hosted in the cloud can easily be accessed by users via web browsers. Users can create software applications with the help of tools supplied by the provider of the service. In PaaS, customers can subscribe to preconfigured features, giving the option to include their required features or discarding the unwanted ones [15].

Typical examples of PaaS include: Google App Engine, Etelos, Force.com, IBM Bluemix, Qrimp and AppJet.

Common use-case: Increases developer efficiency and work rates while also minimizing an applications time-to-market.

Software-as-a-Service

SaaS is the cloud-based delivery of complete software applications that run on infrastructure managed by the vendor. SaaS applications use the web to deliver applications whose interface is accessed on the clients side and handled by a third-party. Most SaaS applications can be run directly from a web browser without the requirement of downloads or installation, although some require plugins [16].

Typical examples include Cisco WebEx, Citrix GoToMeeting, Workday, Google Apps, Salesforce.

Common use-case: Replaces conventional on-device software.

2.1.4 Deployment Models

Public Cloud

Public clouds provide users with a lucrative pay-as-you-go model. Public clouds are owned and managed by third parties. The infrastructure costs are divided among a number of different users delivering remarkable economies of scale to customers. The same infrastructure is used by all customers with finite configuration and security protections operated and supported by the provider. Public clouds have the capability to scale smoothly, as they can be larger than enterprise clouds, if required.

Private Cloud

Private clouds are solely built for a single company. Private clouds are intended to acknowledge their interest on data security. Unlike public clouds, there is a better authority on security in private clouds. There are two different type of private clouds:

- On-premise private clouds: On-premise private clouds, also called internal clouds, are accommodated within the company, hosting its own data center. This cloud computing is best fitted for applications requiring complete management and configurability of the infrastructure and security by providing a more regulated protection. The capital and operational costs are incurred by the company itself.
- Externally hosted private clouds: In this type of private cloud, an exclusive cloud environment with maximum privacy is provided to the enterprise. The cloud provider hosts the cloud externally especially for the companies that avoid public cloud.

Hybrid Cloud

Hybrid clouds are the combination of both public and private clouds. Hybrid clouds improve the flexibility of computing, where infrastructures of third party can be used partially or fully by the cloud providers [17]. The hybrid cloud environment is capable of providing on-demand externally provisioned scale. Any unforeseen rise in workload can be negotiated by reinforcing a private cloud with the available infrastructure of public cloud.

2.1.5 VM Placement in Cloud

Similar to fog nodes in a fog network, the VM is one of the key entities in the cloud computing architecture. The planning of VM is one of the most challenging problem which aims to find the best PM to host the VMs. The VM placement and assignment has an effect on the overall resource utilization, performance, power consumption and cost reduction of data centers. With a massive number of possible optimization criteria, a number of formulations have been proposed over the years for the static and dynamic VM placement problem. For example, a stable traffic aware VM placement problem is studied in [18]. The authors defined the problem as Min Cut

Ratio-aware VM Placement (MCRVMP) and propose two heuristics called 2-Phase Connected Component-based Recursive Split (2PCCRS) and Greedy Heuristic (GH). The 2PCCRS adopts a two-stage approach and solves the problem with the aid of a mathematical solver. The first stage places the connected components on network switches whereas in the second step, the VMs are placed on PM. In GH, the VMs are placed greedily on the available hosts without using any mathematical programming. The results show that there is an increase in data center scalability while decreasing the number of dropped packets. A hierarchical energy-aware resource management solution is proposed in [19]. The resource demands are modeled as random variables considering the correlation among the variables. Compare to the non-hierarchical model, the hierarchical model achieves around 15% energy cost reduction. The authors in [20] propose a Mixed Integer Linear Program (MILP) formulation that aims to place the VMs in the data center with minimum power consumption. The objective function is to minimize the sum of the power consumption of all associated DCs, the power consumption in the Internet Protocol (IP) layer and the power consumption in the Wavelength Division Multiplexing (WDM) layer. The results show a reasonable amount of power saving in geographically distributed multiple medium size DCs.

One of the most common approaches for the optimization is multi-objective solved as mono-objective. The authors in [21] adopt a variation of the weighted sum method in optimizing virtual machine placement and traffic flow routing. The joint optimization demonstrates a reduction in network power costs in cloud data centers. In [22], the authors propose a VM placement scheme minimizing the number of active physical servers and network elements. The results are an optimized network traffic and a reduction in energy consumption. Static and dynamic server consolidations are proposed in [23] for the VM migration. For the dynamic consolidation, the authors adopt a hierarchical approach by introducing the previous mapping of VM in physical servers as a constraint. Initially starting with a zero mapping, after each iteration the previous mapping is added. An energy efficient approach based on the Minimum Correlation Coefficient (MCC) method for VM placement is proposed in [24]. A fuzzy Analytic Hierarchy Process (AHP) is used to make a trade-off between SLA and low energy. A suitable trade-off between SLA violation reduction and power efficiency in cloud data centers was evaluated from the simulation results.

2.1.6 Limitations

Though cloud computing comes with lot of advantages, a rapid growth of users and the demand of higher QoS results in following limitations.

- Performance: Cloud computing is established within the Internet, a large non-homogeneous network with attributes of numerous technologies, speeds, topologies with non-central control. Due to the heterogeneity and loosely controlled nature of Internet, there is a lot of performance related challenges in the cloud. One such issue that affects the performance is latency. Users who are located far away from cloud providers can experience high delays.

- Security and privacy: One of the biggest limitations of the cloud is security and privacy. When customer data is outside the firewall, the integrity and confidentiality of data is highly vulnerable. As the system responses and data transmission travel a long distance to and from the user device to cloud, different types of encryption are required. As cloud providers use diverse structures and independent locations in delivering the technology, there is less trust between customers and providers.

- Bandwidth cost: The bandwidth cost for data-intensive applications is very high. Although cloud computing reduces expenses on hardware and software, companies have to spend more on network bandwidth.

- Reliability: Cloud computing can cause significant time delays to companies resulting in financial losses due to network failures and outages. Moreover, when all the raw data generated by end devices is sent at the same time, it could create a bottleneck. Bottlenecks from high volume of data transfers are one of the most common cause of network crash and outages [25].

2.2 Fog Computing

To address some of the limitations mentioned previously, the concept of fog computing has been recently introduced. Fog computing is a flexible system of connectivity which brings the computational, storage and networking elements of the cloud near to the edge of the network. Fog computing can have an impact on the performance, security, bandwidth and reliability as described below:

- Performance: One of the mission critical applications in IoT will be tactile Internet which requires extremely low latency, high reliability and security with real time data processing. By performing the required processing near the devices, fog computing can provide real time response with reduced latency.

- Security and privacy: As fog computing is not centralized and it is closer to the edge devices, it enhances the security of encrypted data making it less vulnerable to hostile elements. This will allow the users to have their data and applications in close proximity, building more trust between the users and service providers. As the cloud is located in a remote location, the nearest cloud DC can be outside the country border. If a company wants to keep its data within the country, the fog can be a viable option compared to the remote cloud.

- Bandwidth cost: Instead of using the backbone network, users can download through local connections which can reduce the bandwidth cost significantly. Moreover, less data will be sent to the cloud since processing will be done locally.

- Reliability: With the help of fog localization and cloud centralization, there can be interplay of data processing between fog and cloud, hence reducing the burden of all the data processing in the cloud.

2.2.1 Architecture

Figure 2.1 shows a reference fog architecture consisting of different layers. The bottom-most layer comprises of end devices, gateways and sensors. Additional applications are installed for efficient use of the technology. The second layer is where the fog nodes are resided along with the core network. The last layer consists of cloud components. Below is a description of the three layers.

Layer 1

The first layer is composed of smart objects (IoT devices) that are generating data. The devices can be mobile phones, wireless sensors, vehicles, etc. with various applications installed transmitting and receiving data and information to the immediate upper layer and/or the cloud layer.

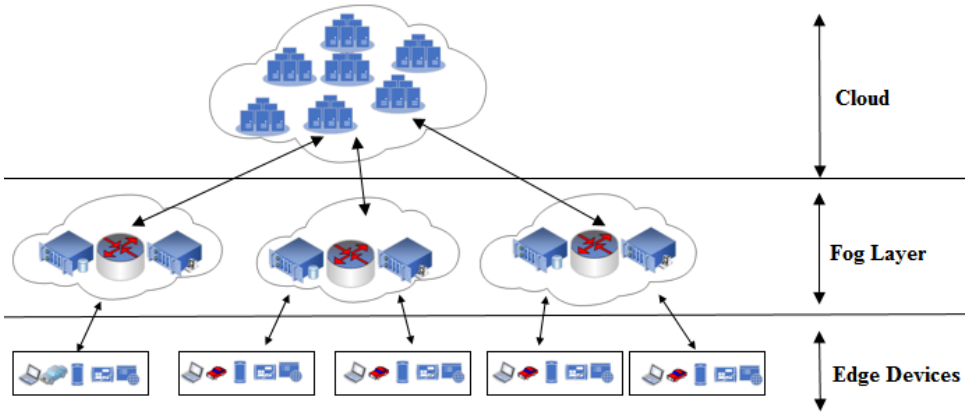

Figure 2.1: Reference architecture for fog computing

Layer 2

The fog computing layer is made up of fog nodes and smart devices such as intelligent gateways, routers and Access Point Name (APN). In this layer, the fog nodes and smart devices perform tasks like routing, data storing and preprocessing, caching and packet forwarding towards the cloud. Additionally, this layer incorporates resource management software while maintaining the entire infrastructure. This layer ensures the QoS to the different fog technology applications as well as the connectivity between the fog nodes and the cloud.

Layer 3

The cloud computing layer consists of servers and data centers with extensive storage capability. Applications and data which require permanent storage and powerful processing are redirected to the cloud. Unlike traditional cloud architecture, the cloud does not serve every individual query. The fog computing layer decides the accessibility and usage of the cloud in an efficient and disciplined manner.

2.2.2 Protocol

There is no implementation or suggestion of any specific protocol for fog computing. Since fog nodes will be providing real time service to numerous edge devices, it is imperative to look at the protocols with more details for a relaiable service. As fog nodes are nested in a middle layer, a number of IoT protocols, as mentioned below,

which are currently being developed or in use can be applied.

Data Distribution System

DDS is a middleware protocol and Application Program Interface (API) standard used for data-centric connectivity from the Object Management Group. DDS provides low-latency data connectivity and a scalable architecture making it useful for mission-critical IoT applications. It uses a publish/subscribe model making sure that data of one node is known by all other nodes [26].

Constrained Application Protocol

CoAP is a low cost software protocol designed to be used in small, low-power wireless sensor networks, switches and similar components that are controlled remotely. CoAP is designed by the Internet Engineering Task Force (IETF) and the interface is modeled after Internet Hypertext Transfer Protocol (HTTP) requests. CoAP provides low overhead and multicast support making it an efficient protocol [27].

Message Queue Telemetry Transport

MQTT was invented by Dr. Andy Stanford-Clark of IBM, and Arlen Nipper of Arcom in 1999. It is a publish-subscribe-based messaging protocol suitable for constrained devices and high-latency, low-bandwidth or unreliable networks. There are different modifications of MQTT such as IBM MessageSight, Mosquitto, emqttd, RabbitMQ, HBMQTT - which differ in features and may employ additional features upon requirement [28].

Advanced Message Queuing Protocol

AMQP is an open standard application layer protocol which provides interoperability between servers. It is a message-oriented middleware with a robust system guaranteeing endpoint to endpoint message delivery despite node failures. Despite being heavyweight and more established compared to other protocols, it can be used in the IoT framework where message reliability is the main focus [29].

Alljoyn

Alljoyn is a collaborative open source software through which compatible devices and applications communicate together [30]. It is a flexible framework promoting proximal network. Although the communication layer is limited to wi-fi, it can be used in fog network.

2.2.3 Application

The concept of fog computing has many different important applications. Below we review these applications and cite relevant work where fog is playing an important role.

Vehicular Networks and Smart Grid

The authors in [31] propose mobile fog, which is geospatially distributed, large-scale and latency sensitive on demand fog service for future Internet applications. Two large scale vehicle-to-vehicle (V2V) application scenarios are simulated. The first application is V2V video streaming and the second is mobility driven Complex Event Processing (CEP), where sensor data is requested for processing from a certain query range by a vehicle. The simulation results show that compare to cloud computing, the fog computing approach reduces network traffic and latency. A literature review by the author in [32] discusses the application of the fog computing framework to implement Software Defined Network (SDN) for vehicular networks. This framework may help in efficient traffic light control and accommodate a number of inputs like Electronic Toll Collection (ETC), Radio-frequency Identification (RFID) tags in cars by installing readers in traffic lights, cameras at traffic lights, etc. creating green traffic waves. Furthermore, the author proposes a multi-layered demand response management system connected to fog devices in a smart grid. Effective coalition between consumers, fog and cloud can minimize the overall power loss as well as reduce communication costs within a smart grid.

Healthcare

The authors in [33] propose a three-layer architecture integrating a role model, a layered cloud computing architecture, as well as a fog-computing-informed paradigm to provide a feasible architecture for healthcare and elderly-care applications. The

proposed model employs a service-oriented architecture for device mapping where the process flow of the healthcare application is modeled using Business Process Modeling Notation (BPMN). The fog computing layer improves the performance of the whole system by providing computing and storage, mobility support, security measures, location awareness and low latency. The validity of the architectural model was exhibited by a use-case as a template for a smart sensor-based healthcare infrastructure. In [34], FAST, a fog computing assisted distributed analytic system to monitor fall for stroke mitigation is proposed. A set of new fall detection algorithms was developed by the authors based on acceleration magnitude values and non-linear time series analysis techniques, as well as new filtering techniques to facilitate the fall detection process. Fog computing was used to design and employ a real-time fall detection system distributing the analytic throughout the network by dividing the detection between the edge devices and servers in cloud. The proposed system attains the high specificity and sensitivity when investigated with real-world data. Moreover, the response time and energy consumption of the system are similar to the most efficient current approaches. In [35], the authors conduct a case study on Electrocardiogram (ECG) feature extraction using fog computing at smart gateways. The fog layer improved on bandwidth utilization, QoS assurance and emergency notification. The experimental results show that more than 90% bandwidth efficiency and low-latency real time response could be achieved using fog computing.

Preprocessing and Content Delivery

The integration of IoT with cloud computing can be one of the primary use of fog computing. However, there are many challenges in this integration. The authors in [36] discuss one of the major challenges which is data trimming or preprocessing. There will be congestion in data center and core network when sending immense amount of raw data generated by IoT environments. A smart gateway-based communication is proposed for trimming or preprocessing data before sending it to the cloud. The IoT devices can send their generated data by using single-hop or multi-hop communication. In single-hop, the data goes directly to the smart gateway for necessary preprocessing whereas in the multi-hop, the data will go through the sink node to the smart gateway. The smart gateway with the help of fog computing can provide a better utilization of network and cloud resources. In [37], the authors discuss the use of edge servers in the fog computing architecture for web optimization. Users will be

connected to the Internet through fog nodes. Each HTTP request made by the user will go through the fog nodes where the fog nodes will reduce the time for web page to load by performing different optimization. General web optimizations like smartly organizing the web page composition, caching Hypertext Markup Language (HTML) components and minimizing the size of web objects can be done by the fog devices. In addition to that, the fog devices will perform optimizations on the basis of network conditions and user behavior. Moreover, the user machines can be monitored by fog devices and it can send graphics of suitable resolutions depending on the browser rendering time.

2.2.4 Current Research Related to Fog Computing

There are many organizations and academies focusing their research on fog computing. The open fog consortium is founded by members from Cisco, Dell, Intel, Microsoft, ARM and Princeton University [38]. It is working on influencing standards development through liaisons with other organizations as well as promoting innovation and industry interest in fog computing. In this section, we outline the current and future research directions in fog computing.

Programmability

Although the mobile fog in [31] is a fundamental development in fog and named mobile, the model is based on a tree topology where the fog nodes are fixed. Hence, a more general model may need to be proposed with diverse networks where fog nodes are decentralized and dynamically mobile. There are plenty of active research in computation offloading in the mobile and cloud computing domain [39, 40]. The important question in offloading in fog computing is how to accord with the three fold dynamic of a) radio access, b) nodes in the network and c) resources associated to it. A context adaptive model for offloading in Mobile Edge Computing (MEC) programming framework is presented in [41] named as Cloudaware. The authors present the offloading framework with the objectives of i) speed up computation, ii) save energy or bandwidth and iii) enable offloading for dynamic mobility. Further studies should be done in learning how to offload in the multi-layered device-fog-cloud infrastructures.

Energy Consumption

Since fog computing is a distributed technology with a large number of nodes, it can be less energy-efficient compared to the centralized cloud computing. In [42], the authors compare the energy consumption of applications using nano data centers (nDCs) in fog computing and using centralized DCs in cloud computing. The authors proposed time-based and flow-based energy consumption model. A set of experiments and measurements were performed to validate the models. The findings suggest that depending on several system design factors, the nDCs might result in energy saving. However, the energy savings are limited to infrequently accessed applications that generate and distribute large amount of data in end-user premises. Hence, the reduction of energy consumption in fog computing still need to be addressed. The authors in [43] study the tradeoff between power consumption and delay by mathematically formulating the workload allocation problem in fog-cloud computing system. Simulation and numerical results show that communication bandwidth and transmission latency can be reduced by drawing slightly more energy consumption.

Security and Reliability

Similar to current virtualized environments, fog computing is challenged with security and privacy issues. Authentication at different level of fog nodes is one of the major security issues identified by the authors in [44]. Public Key Infrastructure (PKI) involving multicast authentication [45] can be a solution to the problem. Authentication cost can be reduced by using measurement-based methods to identify unwanted fog nodes. Another promising solution for authentication can be the Trusted Execution Environment (TEE) technique [46]. The policy management issues in a fog environment were identified by the authors in [47], and a policy-based management framework and policy specification criteria is proposed. The efficiency of the proposed framework was highlighted by suggesting a set of use-case scenarios based on smart transportation systems.

Resource Management and Accountability

The authors in [48] investigate the optimization of maximum task completion time minimization problem in fog computing supported software-defined embedded system

(FC-SDES) with joint consideration of image placement and task scheduling. A low-complexity three-stage algorithm is proposed dealing with I/O and task computation time. The efficiency of the algorithm was validated by performance evaluation from the simulation based studies. A service oriented resource management model for fog deployment is presented by the authors in [49], which allow IoT devices to operate efficiently with fair management of resources. The proposed resource management framework is formulated on the basis of fluctuation in probability of resource utilization with respect to customers resource usage and allocates resources predicting the user behavior. Hermes, a framework that bridges the fog and cloud support analysis through federated query evaluations, is proposed in [50]. By selectively sampling the data generated by continuously sensing the environment, the framework reduces communication and memory consumption on fog nodes and cloud. Accrediting the users to share their resource in hosting applications will add dynamic to fog business models. The proposed user-in-the-loop approach by the authors in [51] can be used where incentives like reward programs, progressive tariffs or environmental indicators can be offered to users.

Mobile Edge Computing

MEC is a similar technology to fog computing. MEC brings the computational capacity within the Radio Access Network (RAN) to minimize delay and increase context awareness [52]. But there are some basic differences with fog computing. For example, the locationof the fog nodes can vary between the edge devices and the cloud with a proximity of one or more hop. In MEC, nodes are located in the macro Base Station (BS) with the proximity of a single hop only. The access mechanism between fog node and edge device is heterogeneous (Bluetooth, Wi-Fi or mobile networks) but the access mechanism for MEC is only mobile networks. Although the internode connection is possible in both technologies, unlike the nodes in MEC, fog nodes are able to do internode communication even without a direct connection between them with the help of the cloud.

2.3 Network Planning Process

Planning a network is a very complex task due to many factors such as the network size, the heterogeneous nature of the equipment, the dynamic behavior of the users,

the rapid changes in technology, etc. As a result, it is important to follow an organized methodolgy in planning and designing a network. The steps involved in planning and designing a network can be summarized as follows [53].

- Define design inputs: The important first step for network planning is to collect network input information and conduct requirement analysis. This step can be done by gathering comprehensive information on edge device traffic patterns and volume, estimated hardware type and cost, network capacity and QoS requirements. In general, the required data collection is complex and time consuming. Due to the unavailability of data, most of the time, the desired input is generated and considerable effort is put to validate the inputs with respect to real-time scenario.

- Network design: In this step, different design and dimensioning techniques are examined to create an appropriate algorithm for the network topology. This step includes recognizing the different network entities, geographical location, sizing and costing. Depending on the requirements, the output of the design can be the network topology, QoS, cost and so on.

- Optimization: The optimization of the network is done based on certain design precedence like delay, traffic, cost, power consumption, etc. The optimal solution is used as a benchmark and further improvement can be done. This step can be repeated with the evolution of network with different inputs and constraints.

- Performance analysis: After a number of iterations, a set of results can be obtained. This step requires the analysis and validation of the obtained results. The validation includes the sensitivity and statistical analysis of the uncertain variables.

There is lot of research focusing on the planning and design of networks in the field of computer and communication networks. For example, the authors in [54,55] address the cellular network planning problem using appropriate optimization techiniques. The UMTS network planning problem is addressed in articles [56, 57]. The authors in [58, 59] conducted a comprehensive research on the controller planning problem in SDN. Different planning tools for wireless networks as listed in [60] are developed by academics and industry researchers. The authors in [61] identify the Gigabit Passive

Optical Network (GPON) planning problem and proposes a Lagrangian heuristic algoruthm to find the optimal placement of distribution points. Recently, the network function virtualization placement problem is explored by many researchers. The authors in [62–65] propose different placement models and algorithms with various optimization criteria.

As can be seen from the above paragraph, network planning is a huge field within the field of Operation Research (OR). In this thesis, we focus on the planning and design of the emerging concept of fog networks.

2.4 Chapter Summary

The literature review shows the importance of fog computing especially in the context of IoT. Fog computing can be seen as a complement to cloud computing resulting in a better overall system for the end users. As an infant technology, fog computing has many challenges that still need to be addressed before its deployment. Most importantly, network planning mechanisms should be adopted which satisfies the end users as well as the requirements of the service providers hosting the fog nodes.

At the best of our review, there is no work on the planning and design of fog networks. In fact, most papers dealing with fog computing already assume that fog nodes are already deployed and available to use. As a result, the next chapter introduces a new mathematical model for the planning and design of fog networks.

Chapter 3

Mathematical Model for Fog Planning Problem

On a network of IoT devices with the coexistence of applications and heterogeneous services, it is not easy to find the best placement location for the installation of fog nodes. The planning problem can only be solved by the use of combinatorial methods and as the problem size gets larger, the number of possible combinations increases exponentially. For example, a network that has 5 possible locations and 200 edge device clusters, there are approximately 5^{200} different possible solutions for the planning problem. Given the limited capacity of fog nodes, assigning an edge device to a fog node for an optimal network is also a complex assignment problem. Furthermore, with the random distribution of edge device locations and their various requirements, it is challenging to determine the capacity of each fog node.

In this chapter, a mathematical model is formulated for the planning problem with a comprehensive explanation of the model. Given a set of edge device requests, the model simultaneously determines the optimal location, the number, and the capacity of the fog node(s) as well as the connection between the fog node(s) and the cloud. As the model has two different objectives (minimize delay and bandwidth), the Pareto optimality needs to be established, i.e., the delay in the network should not be worse off to minimize the traffic in the cloud.

The rest of this chapter is organized as follows. In the next section, we present the model formulation. Then, different techniques are described in order to solve optimization problems with multiple objectives functions.

3.1 Model Formulation

The translation process from analyzed information into mathematical representation is known as model formulation. The model can be formulated by defining the objective function, and the set of constraints and specifying the input to the problem that defines the search space. The objective and each constraint is expressed algebraically in terms of the decision variables. The objective is to minimize network delay and cloud traffic with respect to each constraint.

To model edge devices, we use the notion of clusters where each cluster represents an agglomeration of edge device requests. We are using this approach to reduce complexity as typically, several users (hundreds or even thousands) are using the cloud at the same time in a given area.

3.1.1 Assumptions

To formulate the mathematical model, we assume the following information is known:

- There is no existing infrastructure meaning that we do the planning from a green field scenario.
- The location of all edge devices and the possible locations of each fog node (i.e., x and y coordinates). For each edge device, the amount of generated traffic and the link speed are also known.
- The bandwidth availability and the cost of each link type for the connection between the fog nodes and the cloud.
- The characteristics (i.e., memory, vCPU, cost) of different types of fog nodes that may be installed in a network.
- The location of the cloud. We assume that the cloud is located in a remote location and if an edge device cannot be served by the fog, it will be connected to the cloud. We also assume that the cloud has unlimited memory and vCPU.
- The fog nodes send a fraction of its total traffic to the cloud. This assumption is based on the fact that if an edge device is served by a fog node, most of the work will be done at the fog node level. However, some data may still be sent

from the fog to the cloud for various reasons such as backup, enhanced services, etc.

The output of the planning problem focuses on the following:

- Selecting the optimal geographic location(s) to deploy fog node(s) in the network (i.e. where to deploy the fog nodes).
- Defining the optimal number of fog nodes that needs to be installed in the network.
- Specifying the type (capacity) of fog nodes to be installed in the network.
- Specifying the types of links used to interconnect the various network elements.

The primary goal of the Fog Planning Problem, denoted (FPP), is to find the best locations for installing fog nodes so that the network delay experienced by the users and the amount of traffic going towards the cloud are minimized.

3.1.2 Sets

- U, set of edge device clusters that are present in the network, $U = \{u_1, u_2, ...\}$
  - λ^u, the total amount of memory required by a cluster of edge devices of type $u \in U$
  - α^u, the total number of vCPU required by a cluster of edge devices of type $u \in U$
  - θ^u, the number of packets requested to the fog nodes or the cloud by a cluster of edge devices of type $u \in U$
  - κ^u, the link speed of a cluster of edge devices of type $u \in U$
- N, set of fog types that can be installed, $N = \{n_1, n_2, ...\}$
  - λ^n, the total amount of memory available for a fog of type $n \in N$
  - α^n the number of vCPU available for a fog of type $n \in N$
  - θ^n, the network interface capacity of a fog of type $n \in N$
  - β^n, the number of nodes available for a fog of type $n \in N$

  - ϕ^n, the cost for a fog of type $n \in N$

- L, set of possible link types that can be used to interconnect fog nodes and cloud, $L = \{l_1, l_2, ...\}$
  - ω^l, the bandwidth (in Gbps) of a link of type $l \in L$
  - ξ^l, the price (in dollars per meter) of a link of type $l \in L$
- P, set of possible locations to install the fog nodes, $P = \{p_1, p_2, ...\}$

3.1.3 Constants

- σ, average packet size (in bytes) sent by the edge devices.
- τ, maximum budget (in dollars) allowed for the deployment of the fog.
- t, speed of light ($3*10^8$ meters per second).
- h, average number of hops packets take from edge devices to fog nodes.
- k, average processing delay (in seconds) per hop.
- r, percentage of traffic going to the cloud from fog nodes.

3.1.4 Functions

- Distance (a, b) is a function that calculates the distance between points a and b. The value of points a and b are the x, y coordinates.
- ψ, transmission delay. It is a function that calculates the transmission delay (see Section 3.1.8).
- μ, propagation delay. It is a function that calculates the propagation delay (see Section 3.1.8).
- γ, processing delay. It is a function that calculates the processing delay (see Section 3.1.8).
- f(a, b) is a function that takes the input parameter a as the number of packets per second, b the packet length in bytes and converts to bytes per second.
- g(a) is a function that converts a value from Mbps to bytes per second.

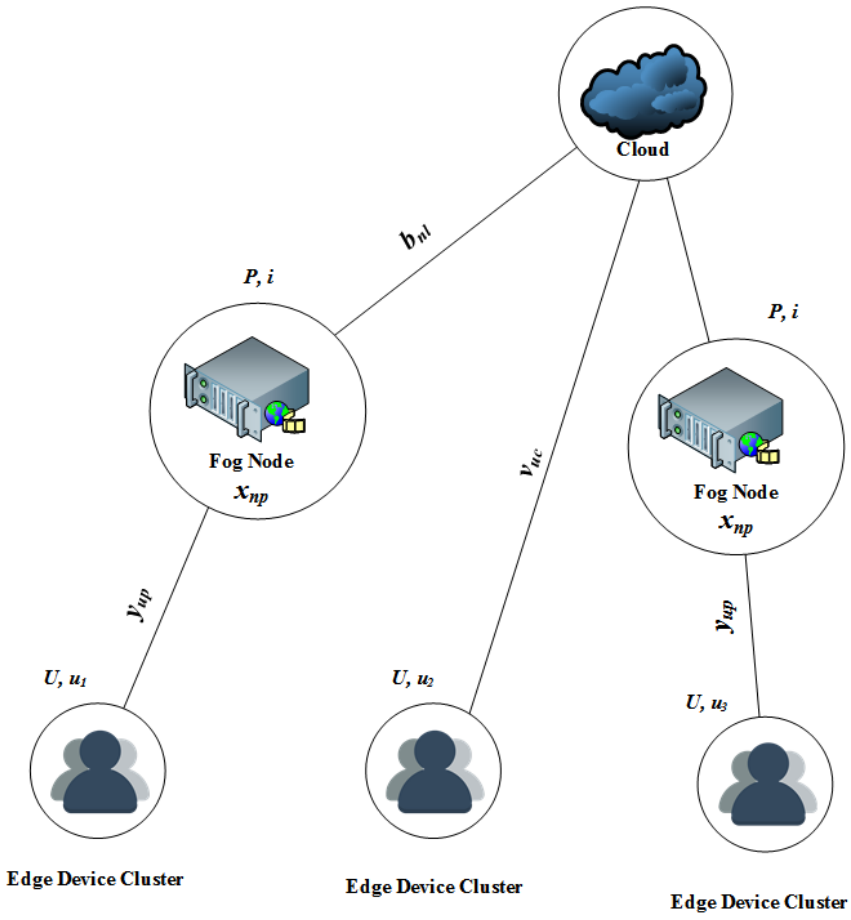

Figure 3.1: Graphical representation of the notations

3.1.5 Decision Variables

- x_{np}, a 0-1 variable such that $x_{np} = 1$ if and only if the fog node $n \in N$ is installed at location $p \in P$;
- y_{up}, a 0-1 variable such that $y_{up} = 1$ if and only if the edge device cluster $u \in U$ is connected to a location $p \in P$;
- v_{uc}, a 0-1 variable such that $v_{uc} = 1$ if and only if the edge device cluster $u \in U$ is connected to the cloud;
- b_{pl}, a 0-1 variable such that $b_{pl} = 1$ if and only if the fog node installed at location $p \in P$ is connected to the cloud with a link of type $l \in L$;

Figure 3.1 is a graphical representation of the network showing the notations described earlier. The figure illustrates a three-tier "edge device-fog-cloud" architecture, with links between each network elements. It can be seen that, depending upon the load (and/or the type of application/service), an edge device cluster can either be connected to a fog node or to the cloud. Moreover, when a fog node is installed at a possible location, it must be connected to the cloud.

3.1.6 Objective Function

As shown in Equation 3.1, the objective function of the FPP, is to minimize the total delay of the network and the amount of traffic sent to the cloud.

$$Minimize(D_{total}, Traffic) \tag{3.1}$$

As shown in equations 3.2 to 3.5, the total delay (D_{total}) is the sum of the transmission delay (D_t), propagation delay (D_n) and processing delay (D_p) in the whole network. On the other side, Equation 3.6 represents the total traffic going to the cloud which is the sum of the traffic from all edge devices connected to the cloud plus the traffic coming from the various fog nodes.

$$D_{total} = D_t + D_n + D_p \tag{3.2}$$

$$D_t = \sum_{u \in U} \sum_{p \in P} y_{up}\psi + \sum_{u \in U} v_{uc}\psi \tag{3.3}$$

$$D_n = \sum_{u \in U} \sum_{p \in P} y_{up}\mu + \sum_{u \in U} v_{uc}\mu \tag{3.4}$$

$$D_p = \sum_{u \in U} \sum_{p \in P} y_{up}\gamma + \sum_{u \in U} v_{uc}\gamma \tag{3.5}$$

$$Traffic = \sum_{u \in U} \sum_{l \in L} v_{uc}\theta^u + \sum_{p \in P} \sum_{l \in L} b_{pl}\theta^u r \tag{3.6}$$

3.1.7 Formulation of the Constraints

The model aims to minimize the network delay as well as the traffic entering the cloud. The set of constraints explained below restricts the minimization function.

- Uniqueness constraints: Uniqueness constraints enforce that at most one fog node is installed at a given location. In other words, we cannot install two or more fog nodes at the same location.

$$\sum_{n \in N} x_{np} \leq 1 \quad (p \in P) \tag{3.7}$$

- Edge device assignment constraints: The assignment constraints are used to make sure that each device is assigned to exactly one location (i.e., the fog or the cloud).

$$\sum_{p \in P} y_{up} + v_{uc} = 1 \quad (u \in U) \tag{3.8}$$

- Node assignment constraints: The assignment constraints enforce that each installed fog node is connected to cloud.

$$\sum_{n \in N} x_{np} = \sum_{l \in L} b_{pl} \quad (p \in P) \tag{3.9}$$

- Node capacity (vCPU) constraints: The capacity constraints for vCPU at the node level make sure that the number of vCPU required by the edge devices does not exceed the capacity of the fog node.

$$\sum_{u \in U} y_{up} a^u \leq \sum_{n \in N} x_{np} a^n \quad (p \in P) \tag{3.10}$$

- Node capacity (memory) constraints: The capacity constraints for memory at the node level make sure that the total amount of memory required by the edge devices does not exceed the capacity of the fog node.

$$\sum_{u \in U} y_{up} \lambda^u \leq \sum_{n \in N} x_{np} \lambda^n \quad (p \in P) \tag{3.11}$$

- Network Interface Capacity (NIC) constraints: The capacity constraints at the node interface level ensure that the total inbound bandwidth required by the edge devices does not exceed the fog node capacity.

$$\sum_{u \in U} y_{up} f(\theta^u, \sigma) \leq \sum_{n \in N} x_{np} g(\theta^n) \quad (p \in P) \tag{3.12}$$

- Link capacity constraints: The capacity constraints at the cloud interface level ensure that the total inbound bandwidth from the fog nodes to the cloud does not exceed the link capacity.

$$\sum_{u \in U} y_{up} f(\theta^u, \sigma) r \leq \sum_{l \in L} b_{pl} g(\omega^l) \quad (p \in P) \tag{3.13}$$

- Inventory constraints: Inventory capacity constraints of the fog nodes make sure that we do not use more than what we have in inventory for a fog type.

$$\sum_{p \in P} x_{np} \leq \beta^n \quad (n \in N) \tag{3.14}$$

- Budget constraint: The budget constraint makes sure that the total cost of the network does not exceed the maximum allowed budget.

$$\sum_{n \in N} x_{np} \phi^n + \sum_{l \in L} b_{pl} dist(p, c) \xi^l \leq \tau \quad (p \in P) \tag{3.15}$$

- Integrality constraints: States that the domain of the following decision variables is a set of binary numbers.

$$x_{np} \in B \quad (n \in N, p \in P) \tag{3.16}$$

$$y_{up} \in B \quad (u \in U, p \in P) \tag{3.17}$$

$$v_{uc} \in B \quad (u \in U, l \in L) \tag{3.18}$$

$$b_{pl} \in B \quad (p \in P, l \in L) \tag{3.19}$$

3.1.8 Delay Computation

The latencies for the network are computed using the transmission, propagation and processing delays. It would also be interesting to consider the queuing delay but

as this keeps changing over time, a static model like the one presented in this chapter does not allow to include it.

The transmission delay is the time taken for a process to send the information to the wire. This depends on the link speed that is used and the packet size that is to be sent to the edge device from the fog or cloud. The formula to calculate the transmission delay is shown below where σ is the amount of data (bytes) and w^l represents the link speed (bytes/sec).

$$Transmission\ delay\ (\psi) = \frac{\sigma}{w^l} \tag{3.20}$$

The propagation delay is the time taken to transmit a signal from a source to a destination. This depends on the medium used and the distance between the source and the destination. The propagation delay is calculated as distance divided by the speed at which the signal propagates in the medium. Depending on the medium that is used, the propagation speed varies. The propagation speed of wireless communications is the speed of light. For copper wires, the speed varies from $0.59t$ to $0.77t$ [66]. In the proposed model, we use $0.59t$ for the speed of copper wire which can be also be used in premises where fiber or other medium cannot be deployed [67].

$$Propagation\ delay\ (\mu) = \frac{Distance(a, b)}{0.59 * t} \tag{3.21}$$

Each router or switch in the data path adds a finite amount of delay as the packet is received, processed, and then forwarded. This includes the time taken at each layer of the TCP/IP down until the bit level layer. Using features that are supported with hardware assistance can greatly reduce the processing delay. Typically, the processing delay with a hardware-assisted switch will be in the 4-to-20 microseconds range [68]. In the worst case scenario, even when using a software assistance, the most reasonable processing delay that can be expected in practice should be 25 microseconds per hop (h), which has been used in our calculation.

$$Processing\ delay\ (\gamma) = k * h \tag{3.22}$$

3.2 Model with Pareto Optimality

Since the formulation of the problem is a multiobjective optimization model, we need to simultaneously minimize the two objective functions expressed in equations 3.2 and 3.6. Sometimes, the minimum network delay will increase the traffic in the cloud and vice versa. Therefore, to ensure the fairness of the network delay and the traffic in the cloud, we need to find the Pareto optimal solution. In this section, we present three different methods that we used to find solutions to our problem.

3.2.1 Weighted Sum Optimization

The weighted sum strategy converts the multiobjective problem of optimizing the main objective function into a scalar problem by building a weighted sum of all the objectives [69]. Therefore, the proposed objective function can be rewritten as:

$$Minimize\ (w_1 D_{total}^{norm} + w_2 Traffic^{norm}) \tag{3.23}$$

where w_1 and w_2 are weighted coefficient and D_{total}^{norm} and $Traffic^{norm}$ are the normalized objective functions as they have different scales. Equations 3.24 and 3.25 are used to normalize the objective functions.

$$D_{total}^{norm} = \frac{D_{total}^{max} - D_{total}}{D_{total}^{max} - D_{total}^{min}} \tag{3.24}$$

$$Traffic^{norm} = \frac{Traffic^{max} - Traffic}{Traffic^{max} - Traffic^{min}} \tag{3.25}$$

The weighted coefficients define the contribution of each objective function in the modified model. Even with a proper knowledge of the problem, it is not easy to specifically select the weight of the coefficients in the modified objective function. Therefore, an additional constraint 3.26 is added which states that the sum of the weighted coefficients w_1 and w_2 should be 1.

$$w_1 + w_2 = 1 \tag{3.26}$$

During simulations, both weighted coefficients are varied in steps of 0.1 with respect to the added constraint to obtain a Pareto front.

3.2.2 Hierarchical Optimization

Hierarchical optimization is generally solved by the sequential optimization method. Here, the objective functions are ranked in order of importance. From the literature review, it can be assumed that the fog nodes will be used extensively for latency sensitive applications [31, 33–35]. Hence, the model is solved using *Traffic* as the lone objective function with an additional constraint that do not allow the value of the delay function to exceed a set of prescribed fractions of its optimal value [70]. In other words, we try to improve the less important criteria through minimal loss of the most important criterion.

$$Minimize\ D_{total} \tag{3.27}$$

In the first step, we solve Equation 3.27 with respect to constraints 3.7-3.19. In the second step, we solve Equation 3.28 with the additional constraint 3.29.

$$Minimize\ Traffic \tag{3.28}$$

$$D_{total} \leq D_{total}^{min} * \delta \tag{3.29}$$

where δ is the added value for delay. We vary the δ with a set of values to obtain a Pareto optimal front.

3.2.3 Trade-Off Method

This method involves optimizing a primary objective, and expressing the other objectives in the form of inequality constraints [71]. In our proposed trade-off model, we consider D_{total} as the primary objective function due to the reason explained before and *Traffic* as an inequality constraint with specified values of traffic in the cloud. We solve the objective function show in Equation 3.27 subject to:

$$Traffic \leq \epsilon \tag{3.30}$$

and all other constraints 3.7-3.19. However, it is not easy to find the precise ϵ value for the optimal solution. For this, the range of $Traffic^{max}$ and $Traffic^{min}$ is divided into ten equal intervals and we use the eleven values as the varying ϵ value to obtain an optimal Pareto set curve [72].

3.3 Determining the Best Compromise Solution

From each method of the multiobjective model, we obtain a set of non-dominated solutions forming a Pareto front. Since there can be an uncertainty in decision makers preference, it can be assumed there is a fuzziness in the goal of each objective function. The membership functions define this fuzziness by representing the degree of fuzziness in some fuzzy sets using values in the range [0, 1].

In this thesis, we adapt a fuzzy-based mechanism [73] which is used to find out a compromised solution on the Pareto front of each problem. The fuzzy mechanism looks at the way the solutions are contributing to each objective and assigns a fuzzy variable. When the solutions in a Pareto front are very close together, the mechanism shows a possible way of finding the best compromise solution. In this mechanism, a membership value for the i^{th} objective of the j^{th} solution in the Pareto front is calculated using the membership function as:

$$\mu_i^j = \begin{cases} 1 & \text{if } F_i \leq F_i^{min} \\ \frac{F_i^{max} - F_i}{F_i^{max} - F_i^{min}} & \text{if } F_i^{min} < F_i < F_i^{max} \\ 0 & \text{if } F_i \geq F_i^{max} \end{cases} \tag{3.31}$$

μ_i^j indicates how well the j^{th} solution is able to satisfy the i^{th} objective in a Pareto optimal set. The sum of the membership value of all objectives of the j^{th} solution suggests how well it satisfies all the objectives. Given N solutions in a Pareto front and M objective functions for each solution, the achievement of each solution with respect to all the N solutions can be calculated by:

$$\mu^j = \frac{\sum_{i=1}^{M} \mu_i^j}{\sum_{j=1}^{N} \sum_{i=1}^{M} \mu_i^j} \tag{3.32}$$

The solution with a maximum value of μ^j is a compromise solution that can be accepted by the decision maker.

3.4 The Hypervolume Indicator

The Hypervolume (HV) indicator was first proposed in [74]. In recent years, the hypervolume indicator has been widely used in multiobjective optimization to evaluate various search algorithms.

If an objective space contains solution that are considered as points, the n-dimensional space that is contained by a solution set is called hypervolume. In other words, the hypervolume is the n-dimensional volume of the set relative to some reference point. The reference point usually refers to the worst possible point in the solution set. The hypervolume of a set is the total dominated space of the solutions in the set. The single unary value of HV gives a measure of the spread of the solutions along the Pareto front. A set with large HV is always desirable as it presents a better set of trade-offs as it covers a larger area in the solution space.

Given a non-dominated set of solutions S, for each solution $i \epsilon S$ there is a hypercube v_i with a reference point r where the solution i is the diagonal corners of the hypercube. With the union of all the hypercubes, the hypervolume can be calculated by:

$$HV = \cup_{i=1}^{|S|} v_i \tag{3.33}$$

Figure 3.2 shows an example of HV for a 2-dimensional minimization problem with a set of non-dominated solution S = $\{a_1, a_2, a_3, a_4\}$ and reference point r. As we can see from the figure, the yellow area is the hypervolume contributed by the four non-dominated solutions with respect to the reference point.

3.5 Chapter Summary

In this chapter, we formulated an exact mathematical model for the planning and design of fog networks. The objective of the model is twofold: 1) minimize the delay experienced by the users and 2) minimize the amount of traffic that must go to the cloud, subject to a set of realistic constraints. Since we formulated the problem as a multiobjective mathematical model, we reviewed three different methods and modified our model with additional constraints to obtain Pareto optimal solutions. Finally, two different methods and functions fuzzy-based mechanism and HV indicator were

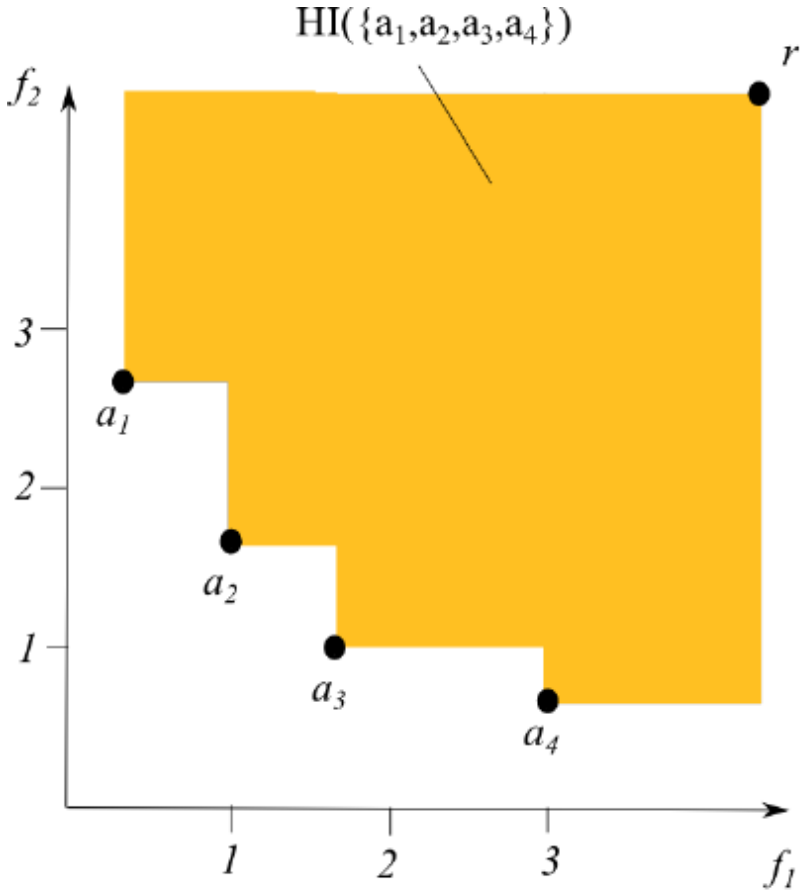

Figure 3.2: HV enclosed by non-dominated solution a_1, a_2, a_3 and a_4 [75]

introduced and explained in the last two sections of the chapter. The aforementioned methods are used to analyze and better understand the results. In the next chapter, we analyze the results and performance of the proposed methods with different inputs.

Chapter 4

Results and Analysis

In this chapter, we solve the planning problem modeled in Chapter 3 with different input. Then we do a complete analysis of the obtained results to better understand the performance of the planning model. More precisely, the chapter begins with a summary of the steps involved when solving the FPP. In the next section, we solve a detailed example to understand how each of the constraints work in the model. Then, we analyze the results from small scale to large scale problems followed by a study of the impact of increasing the number of possible placement locations. Finally, we summarize the chapter in the last section.

4.1 Framework for the Fog Planning Problem

There are many steps that need to be taken to find the optimal solution for FPP. Figure 4.1 summarizes the steps that we used for each problem in our planning model. First, the input information is generated and the problem is solved with the planning model in Chapter 3. The problem is solved for both single and multiobjective functions. For the multiobjective optimization, three different methods are used to generate a Pareto front. Next, the fuzzy-based mechanism described in Section 3.3 is used in determining the best compromise result from the Pareto set. The hypervolume of the Pareto front is calculated using Equation 3.33. Finally, the results obtained from all the steps are analyzed for better understanding of the FPP.

Figure 4.2 shows the flowchart for solving the FPP using the weighted sum method. In the first step, we generate the input for the planning problem. Initially, the value for w_1 is set to 1 and w_2 is set to 0. Then the modified model from Section 3.2.1 is

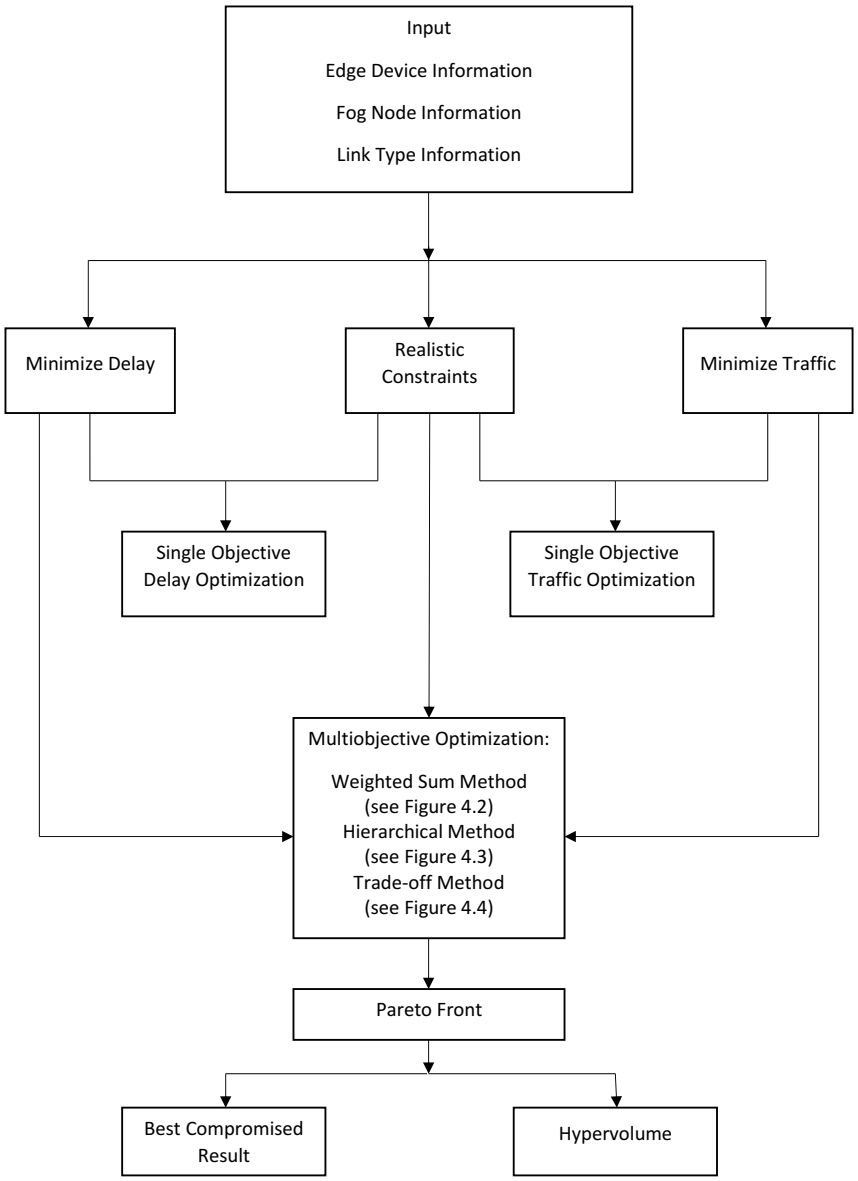

Figure 4.1: Steps to solve the FPP

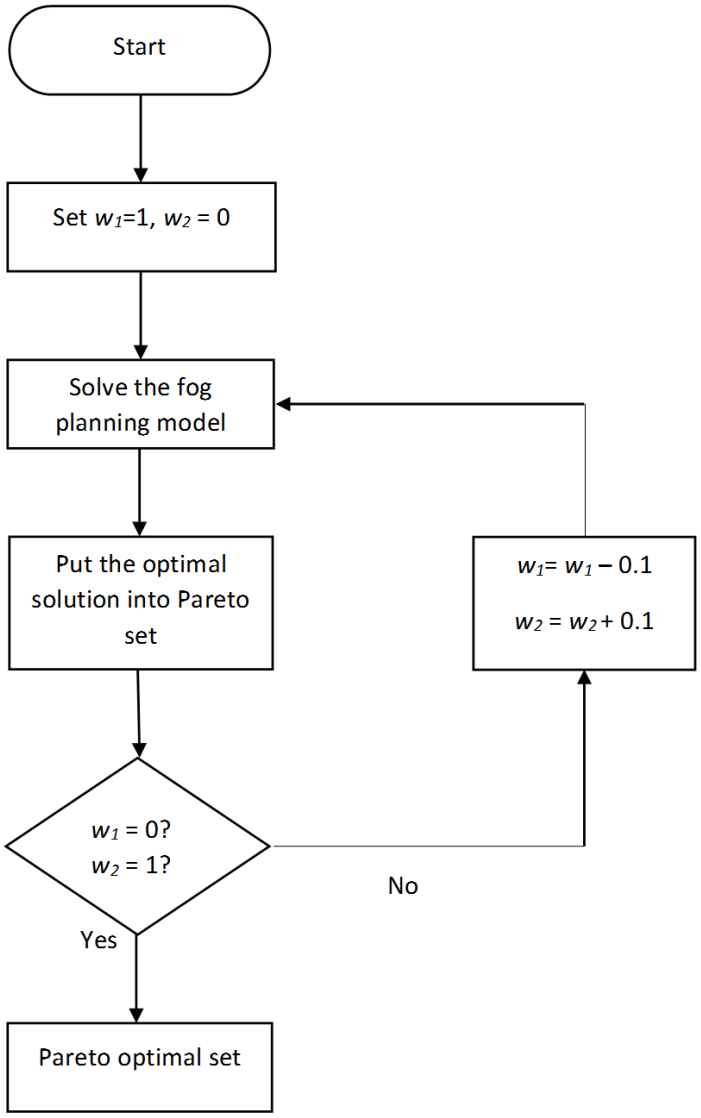

Figure 4.2: Steps to solve the FPP with the weighted sum method

solved and the returned objective function values are stored in the Pareto set. Then the terminating condition is checked and if it is not satisfied, the model is solved one more time with updated weighted coefficient values and the new solution being stored in the Pareto set. This step is repeated until the terminating condition is satisfied. Once the terminating condition is satisfied, a Pareto optimal set is obtained from the stored objective function values. The fuzzy-based mechanism described in Section 3.3 is used to extract the best compromised result from the Pareto optimal set, which is used as the final result for analysis. The procedure outlined in the flowchart is used each time we solve a fog planning problem with the weighted sum method.

The steps for solving the hierarchical method are shown in Figure 4.3. Initially, the inputs for the FPP are randomly generated. In the next steps, the value of δ is set to 1.0001 and the modified hierarchical model presented in Section 3.2.2 is solved. The solution results are stored in the Pareto optimal set. Next, the terminating condition is checked and if not satisfied, the δ value is updated with δ‘, where δ‘ equals to 0.0001 and the problem is solved one more time with the updated δ value and the results are stored in the Pareto set. This process is repeated until the terminating condition is satisfied, which is δ equal to 1.0008. When the terminating condition is satisfied, a Pareto optmal set is obtained from all the solutions from Pareto set. Finally, the best compromised result is obtained from the Pareto optimal set using the fuzzy decision approach. This procedure is used when solving all the problem in FPP.

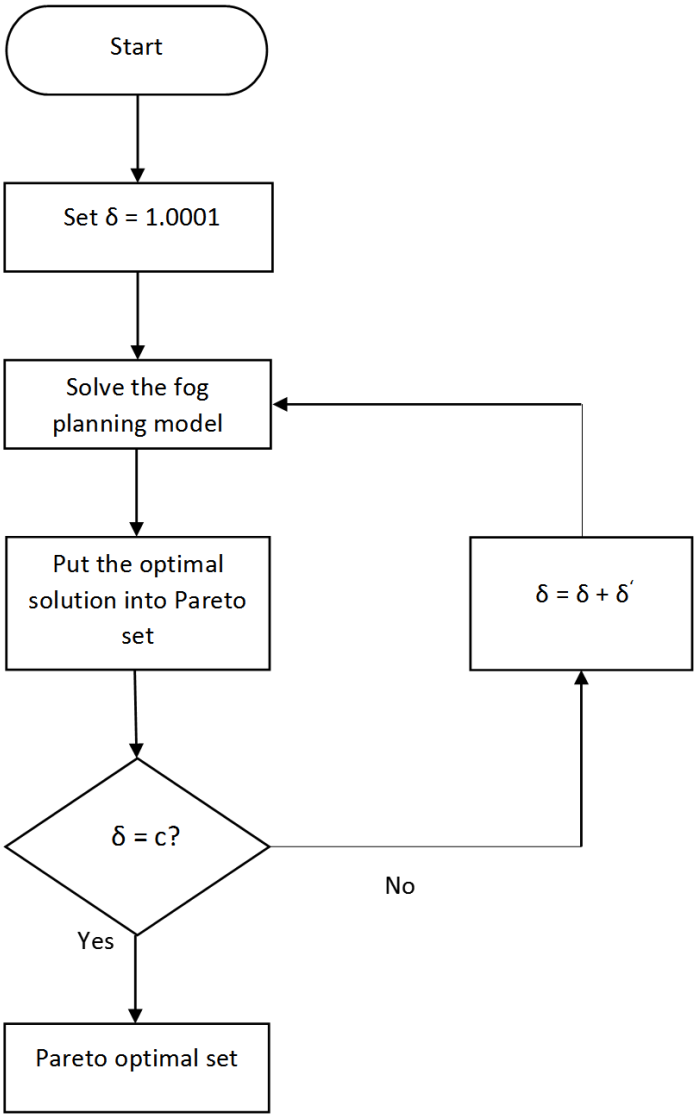

Figure 4.3: Steps to solve the FPP with the hierarchical method

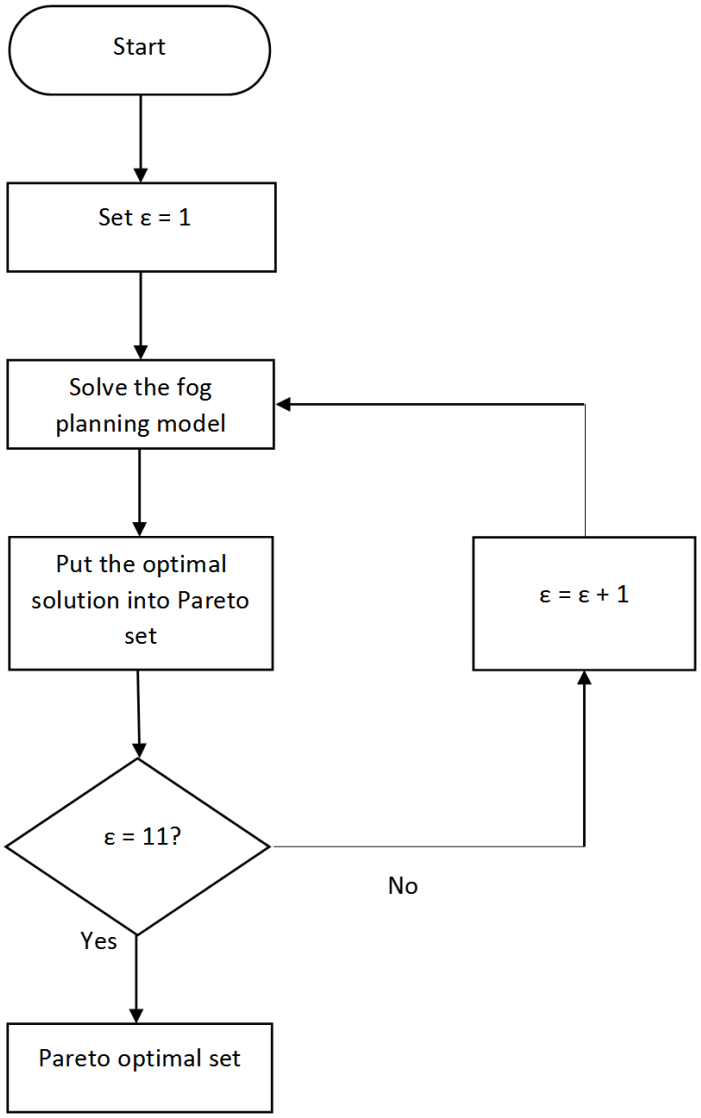

Figure 4.4: Steps to solve the FPP with the trade-off method

The flowchart to solve the FPP using the trade-off method is illustrated in Figure 4.4. The method uses the same randomly generated inputs used for the previous two multiobjective optimization methods. The ϵ value is set to 1 and the modified trade-off model in Section 3.2.3 is executed. The optimal solutions from the model

are stored in the Pareto optimal set. In the next step, the terminating condition is checked and if it is not satisfied, the problem is solved with a new ϵ value, which is a specified value of traffic in the cloud. The model is repeatedly solved until the terminating condition is satisfied. When the terminating condition is checked out, an optimal Pareto front is obtained. Finally, the fuzzy-based mechanism is used to obtain the best compromised result for the trade-off method.

4.2 Detailed Example

The simple example presented in this section is used to explain and show how the mathematical model works. In this example, we want to plan and design a brand new fog network in order to accommodate 10 edge device clusters by finding the optimal number and location for the fog node(s) while considering the nodes processing capacity, the bandwidth requirements of the edge device clusters and link and nodes inventory. Once the result is obtained, it will be analyzed and the final conclusion is made by showing the total network delay and traffic in the cloud with the optimal placement variables.

Figure 4.5 shows the area to be planed. As it can be seen, there are 10 edge device clusters that need to be connected to either a fog node or the cloud depending upon their request with 5 possible fog node placement locations. As we assume, that the cloud is located in a remote location, is not shown in the figure.

In this simple scenario, we have two types of fog nodes. The characteristics of both types are given in Table 4.1. Similarly, as shown in Table 4.2, there are two types of link with a bandwidth of 10 Mbps and 100 Mbps respectively. Table 4.3 shows the optimization input parameters with randomly generated edge device and fog node locations over an area of 100 km^2. The size of each requested packet is 1250 bytes and there are only three fog nodes available (two of type a and one of type b). The parameters of each cluster are given as input to the model and are listed in Table 4.4. The first column shows the edge device cluster index. Columns 2 and 3 show the location of the edge device cluster in the planned area, whereas columns 3-6 show the profile of each edge device cluster which includes the requested memory and vCPU,

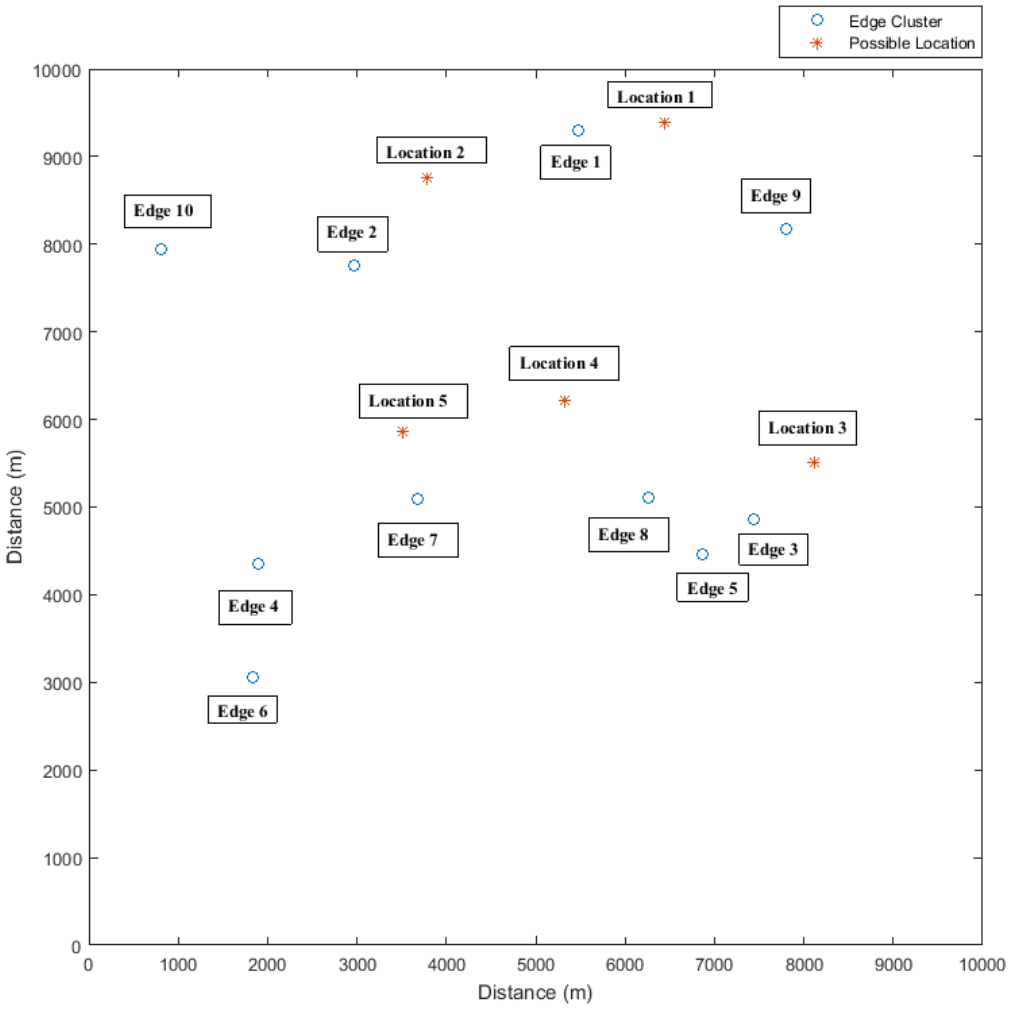

Figure 4.5: Edge device locations and potential locations of fog nodes

Table 4.1: Fog type features for the planning example

Fog Types	**a**	**b**
vCPU (α^n)	5	10
Memory (λ^n)	15	30
NIC (θ^n)	45	100
# of Nodes (β^n)	2	1
Cost (ϕ^n)	1000	2000

Table 4.2: Link type features for the planning example

Link Types	**a**	**b**
Bandwidth (ω^l)	10 Mbps	100 Mbps
Cost/metre (ξ^l)	\$0.05	\$0.25

Table 4.3: Topology and optimization input parameters for the example

Simulation area	10x10 km
Number of edge device clusters	10 (random)
Maximum Placements	5 (random)
Packet size (σ)	1250 Bytes

Table 4.4: Edge device input parameters for each cluster

Edge device cluster index	**Location(m)**		**Requested**			**Network Access Bandwidth (Mbps)**
	x	y	**Memory**	**vCPU**	**Traffic (Mbps)**	
1	5470	9294	6	2	12	10
2	2964	7757	5	1	16	21
3	7447	4868	9	4	13	14
4	1890	4359	3	12	16	17
5	6868	4468	10	3	33	28
6	1836	3064	18	1	26	8
7	3685	5085	6	3	41	11
8	6257	5108	5	3	11	22
9	7803	8176	35	2	23	14
10	812	7948	13	8	44	9

as well as the traffic and the network access bandwidth.

To solve the problem, we used CPLEX 12.7 [76]. All the parameters in the optimizer were set to default except that we set a Time Limit (TL) of 1.5 hours. This means that if the optimal solution is not found after 1.5 hours of computation, CPLEX will simply return the best solution obtained so far. We run the simulation on a PC with an Intel i5 processor running at 3.00 GHz with a total memory of 12 GB.

The goal of the model is to select the best locations to install the fog nodes such that all constraints are satisfied with minimum delay in the network and minimum amount of traffic going to the cloud. As it is a small problem, the solution search space is also small resulting in obtaining the same solution for every iterations for the three multiobjective methods. The optimal solution obtained from CPLEX has a total delay of 0.00702 second and traffic of 117.19 Mbps. From Figure 4.6, we can see that 6 edge device clusters are connected to fog nodes installed at 3 different locations. In the figure, the solid black lines show the connections between the edge device clusters and the fog nodes, the dash line shows the connection between the edge device clusters and the cloud and the dash-dot line presents the connection between the installed fog nodes and the cloud. The fog nodes that were not part of the optimal solutions (i.e. locations 1 and 4) were not connected as seen in the figure.

By visual observation from Figure 4.6, we can see that edge device cluster 3 is close to location 3 with 922 meters where a fog of type a is installed. Although edge device cluster 1 is close to location 1 (978 meters), installing a fog node there will be far for edge device cluster 2 increasing the overall network delay. So, a fog of type a is installed in location 2 connecting both edge device cluster 1 and 2. Placement location 5 is used to install a fog of type b as it is connected to three edge device clusters which are closer. All of the installed fog nodes are connected to the cloud which is approximately 90,000 meters away from each of the fog nodes.

Capacity Constraints

Constraints 3.10-3.12 state that the capacity of a fog node should be greater or equal to the sum of all edge device requirements that are connected to it. For example, edge device cluster 9 is closer to location 1, but it requests 35 MB of memory which is above the capacity of both fog types. Hence, it is connected to the cloud. The fog node installed at location 5 can accommodate edge device cluster 4 in terms of all

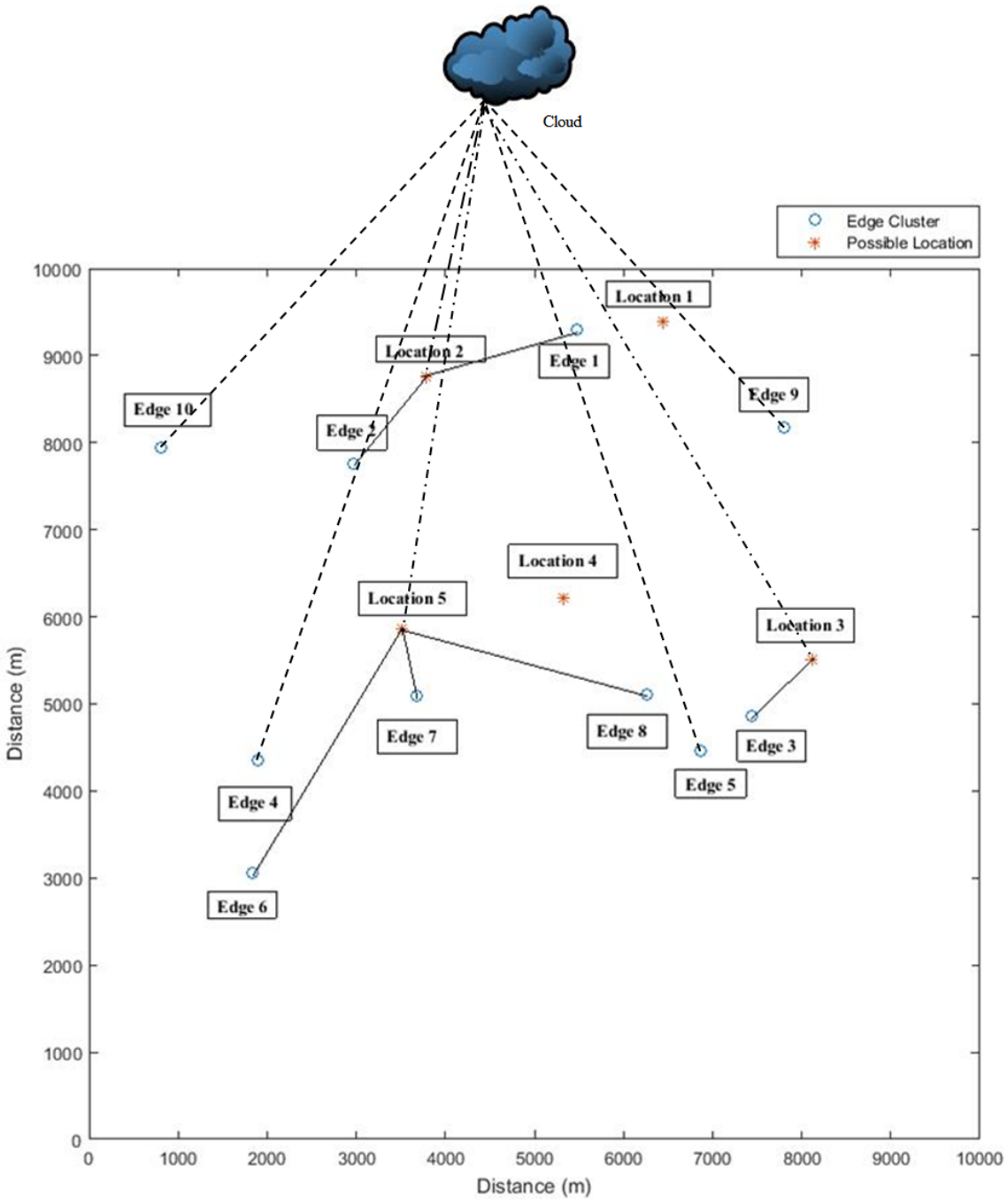

Figure 4.6: Optimal solution found by Cplex

the constraints except Equation 3.10. Therefore, edge device cluster 4 is connected to the cloud.

Inventory and Budget Constraints

Equation 3.14 limits the number of each fog node type that can be installed. In the final solution, the solver will make sure that each fog node type is not used more than the maximum number of nodes available as indicated in the fog nodes input information in Table 4.1. In this case, two fog nodes of type a and one fog node of type b are installed in the network. Constraint 3.15 restricts the solver to go over budget. For an ideal situation, there can be a fog node installed in every location, but as we have restricted the budget to 40% of the maximum cost, a limited number of fog node can be installed.

Link Bandwidth between Fog Nodes and Cloud

Equation 3.9 ensures that there is a connection between all installed fog node and the cloud. Links from Table 4.2 are in Mbps and need to be converted to bytes per second. Function $g(\omega^l)$ converts between the two units for link type ($l \in L$). In this simple example, we assume that only 1% of the traffic coming to a fog node is sent to the cloud for further processing. Since, we do not want to go over budget, the lowest link type is used.

Optimal Solution

In summary, the decision variables selected are:

- Fog node placement (x_{np}): x_{12}, x_{13}, x_{25}
- Edge device and fog node connection (y_{up}): y_{12}, y_{22}, y_{33}, y_{65}, y_{75}, y_{85}
- Edge device and cloud connection (v_{uc}): v_{41}, v_{51}, v_{91}, v_{101}
- Fog node and cloud connectivity (b_{nl}): b_{21}, b_{31}, b_{51}

Table 4.5: Delay formulation of the planning example

Transmission Delay (ψ)	**Propagation Delay** (μ)	**Processing** (γ)
$0.000125y_{12}$	9.98192E-06y_{12}	$0.000125y_{12}$
5.95238E-05y_{22}	7.32655E-06y_{22}	$0.000125y_{22}$
8.92857E-05y_{33}	5.20565E-06y_{33}	$0.000125y_{33}$
7.35294E-05y_{65}	1.84593E-05y_{65}	$0.000125y_{65}$
4.46429E-05y_{75}	4.54915E-06y_{75}	$0.000125y_{75}$
$0.00015625y_{85}$	1.61175E-05y_{85}	$0.000125y_{85}$
$0.000113636v_{41}$	$0.000564972v_{41}$	$0.00075v_{41}$
5.68182E-05v_{51}	$0.000564972v_{51}$	$0.00075v_{51}$
8.92857E-05v_{91}	$0.000564972v_{91}$	$0.00075v_{91}$
$0.000138889v_{101}$	$0.000564972v_{101}$	$0.00075v_{101}$
Total Delay = 0.00702 sec		

Table 4.6: Traffic formulation of the planning example

Edge Device Traffic (v_{uc})	**Fog Node Traffic** (b_{nl})
$0v_{11}$	$0b_{11}$
$0v_{21}$	$0.28b_{21}$
$0v_{31}$	$0.13b_{31}$
$16v_{41}$	$0b_{41}$
$33v_{51}$	$0.78b_{51}$
$0v_{61}$	
$0v_{71}$	
$0v_{81}$	
$23v_{91}$	
$44v_{101}$	
Total Traffic = 117.19 Mbps	

The solutions returned by the solver for the three methods was validated by solving the problem manually. Tables 4.5 and 4.6 were used to verify the results of the optimizer.

Since this is a small example, we are able to show partial delay function and traffic function which is exactly equal to the results obtained by the solver.

4.3 Results Analysis

The output of the problem is a direct reflection of the input. In the FPP, we are interested in analyzing the effect of increasing the number of edge device clusters and possible fog node placement locations. Typically, with the increase of input size, there will be more delay in the network and more traffic will go towards the cloud. There are different methods of measuring the quality of the solution of an optimization problem. The most common way to measure is to keep track of the objective functions and time taken by the solver to reach the optimal solution.

4.3.1 Input Parameters

In this section, we first present the different features of the fog nodes as well as the characteristics of the edge device traffic. All the planning problems are solved with respect to the input specified. Each fog node has a maximum capacity that cannot be crossed. In other words, if one of the requested parameters like memory is greater than the nodes capacity, then the capacity should be increased by using a different fog type or send the edge device cluster request to the cloud. The specification of each fog node depends on the service providers and there are many companies like IBM, Rackspace, Amazon AWS, etc. providing cloud services. Table 4.7 shows the four types of fog nodes that can be used by the solver.

The specifications of the different link types used to connect the fog node(s) to the cloud are shown in Table 4.8. The price of the links depends on their capacity and material which can be obtained from network sales website [77, 78].

The input parameters for the edge device clusters and each edge device within the cluster are tabulated in the tables 4.9 and 4.10 respectively. The simulation area is chosen to be 100 km by 100 km. This larger area is chosen because we

Table 4.7: Characteristics of the different fog types

Fog Type	# of CPU	Memory (GB)	NIC (Mbps)	Cost ($)	Inventory
a	90	480	360	67200	20
b	180	800	1024	120000	15
c	360	1600	1024	170000	10
d	720	3200	10240	250000	4

Table 4.8: Characteristics of the different link types

Link Type	Bandwidth	Cost/meter ($)
a	100 Mbps	0.25
b	1 Gbps	2
c	10 Gbps	200

envision the fog network to be spanning over a city and beyond. Moreover, some point of the simulation there will be 200 edge device clusters which will be hard to accommodate in a smaller area for visual representation. The locations (i.e. the x and y coordinates) of the edge device clusters and the possible locations are randomly generated within the simulation area. The location cost to install a fog node is set to be $1000. The maximum budget of the planning problem was set to 40% of the maximum cost the fog network can incur. Instead of a fixed value for the budget, we used a percentage of the maximum cost because with the increase of network size (possible placement locations) the maximum cost increases and the budget should also be increased proportionally instead of biasing the small scale problem with a large budget. The cost of a fog network includes the fog node cost, the location cost and the link cost between fog nodes and the cloud (i.e. the hardware cost). The maximum cost includes the installation of highest capacity of fog node(s) in all the optimal locations as well as connecting the fog node(s) and the cloud with the most expensive link type which has the highest speed. The location cost for a fog node depends on many factors. For example, the cost of renting or leasing a location in a commercial area is probably higher than the cost in a residential area. It also depends

Table 4.9: The edge device cluster input parameters

Simulation area	100x100 km
Packet size	1500 Bytes
# of edge device in a cluster	U(10,200)

Table 4.10: The edge device input parameters for each edge device in a cluster

# of CPU core	U(1,4)
Memory	U(1,40) GB
# of Packets sent per second	U(1,64)
Network access bandwidth	U(20,70) Mbps

upon the size and expandibility. To reduce complexity the same cost was used for all the possible fog node placement locations.

Two different sets of problems were solved using three different multiobjective optimization methods. The first set consists of small scale problems and the second set has larger scale problems. For each set, different problem sizes were generated, and for each size, four different instances of the problem were generated. In total, for each mulltiobjective optimization method, we optimized 32 different problem sizes over 4 instances resulting in solving 128 optimization problems. In order to obtain the Pareto front, the weighted sum and the trade-off methods were solved with 11 iterations for each problem (as shown in Figures 4.2 and 4.4), whereas the hierarchical method was solved with eight iterations for each problem (as explained in Figure 4.3). A time limit of 1.5 hours was set for each iteration. This means that if the optimal solution is not found after 1.5 hours of computation, CPLEX will simply return the best solution obtained so far for that iteration. For each problem, the membership functions 3.31 and 3.32 are used to evaluate each member of the Pareto front and the member with the highest membership function value is extracted as the best compromise result.

4.3.2 Small Scale Problems

In the first set of problems, the number of edge device clusters is varied between 10 and 200 with 5 possible placement locations. 21 different problem sizes (as shown in Table 4.11) are solved over the 4 different instances for a total of 84 problems. The results presented in Table 4.12 are the averages over the 4 different instances (refer to Appendix A for the results of each instance) and includes the best compromised total delay in the network, total traffic to the cloud and the solution time for each multiobjective optimization method. The same CPLEX and hardware configuration was used as described in Section 4.2.

The first column in Table 4.12 represents the problem number corresponding to the problem size presented in Table 4.11. The second, third and fourth columns represent respectively the total delay in the network, the total traffic going to the cloud and the solution time for the weighted sum optimization method; whereas the fifth, sixth and seventh columns represent respectively the total delay in the network, the total traffic going to the cloud and the solution time for the hierarchical method. Finally, columns 8, 9 and 10 show respectively the total delay in the network, the total traffic going to the cloud and the solution time for the trade-off method.

For the small scale problems, the lowest delay obtained was 0.111 seconds using the hierarchical method and the lowest traffic was 97 Mbps achieved by the weighted sum method.

The delay, traffic and solution time comparisons of the three methods are graphically represented in figures 4.7, 4.8 and 4.9 respectively. For our FPP, we used 95% Confidence Interval (CI) to examine the reliability of the results obtained. In figures 4.7 and 4.8, the vertical bars along the graphs show the CI. As it can be seen from the figures, there is a small difference between the three methods in terms of total network delay and traffic towards the cloud. However, the hierarchical method attains less delay and more traffic than the other two methods. This is because the hierarchical optimization is bounded by the delay constraint and we do not allow the objective function to go beyond a reasonable delay in the network which compromises the optimal traffic. With the increase of problem size, the objective functions also increase for all the three methods. This is because more edge device clusters are

Table 4.11: Problem size for the small scale FPP

Problem	**# of edge device clusters**	**# of possible placement locations**
1	10	5
2	15	5
3	20	5
4	25	5
5	30	5
6	35	5
7	40	5
8	45	5
9	50	5
10	55	5
11	60	5
12	65	5
13	70	5
14	75	5
15	80	5
16	85	5
17	90	5
18	95	5
19	100	5
20	150	5
21	200	5

Table 4.12: Results obtained from the solver for small scale problems (average over 4 instances)

	Weighted Sum			Hierarchical			Trade-off		
Problem	**Delay (s)**	**Traffic (Mbps)**	**Solution Time (s)**	**Delay (s)**	**Traffic (Mbps)**	**Solution Time (s)**	**Delay (s)**	**Traffic (Mbps)**	**Solution Time (s)**
1	0.113	97	2	0.111	110	3	0.113	122	2
2	0.193	175	2	0.191	175	4	0.193	175	3
3	0.266	401	3	0.265	416	5	0.266	401	3
4	0.343	391	3	0.34	408	7	0.343	391	4
5	0.419	604	3	0.414	635	8	0.42	599	4
6	0.576	919	4	0.573	949	12	0.576	919	6
7	0.572	898	7	0.573	906	16	0.572	903	14
8	0.669	1112	4	0.655	1168	19	0.659	1126	20
9	0.701	1234	7	0.694	1283	21	0.702	1231	26
10	0.822	1373	35	0.817	1419	29	0.822	1374	81
11	0.941	1533	24	0.931	1608	35	0.939	1543	73
12	0.999	1666	44	0.988	1743	38	0.998	1672	50
13	1.067	1777	907	1.078	1782	42	1.067	1786	998
14	1.169	1959	149	1.16	2030	68	1.171	1956	173
15	1.21	2021	18	1.203	2084	56	1.217	1991	776
16	1.332	2269	148	1.321	2338	61	1.331	2277	448
17	1.169	2242	706	1.356	2283	62	1.361	2262	242
18	1.359	2700	1010	1.354	2736	76	1.36	2704	438
19	1.617	3017	25	1.602	3102	74	1.613	3033	139
20	2.398	4015	513	2.392	4059	211	2.401	4001	365
21	3.22	5670	3789	3.213	5706	505	3.221	5672	3779

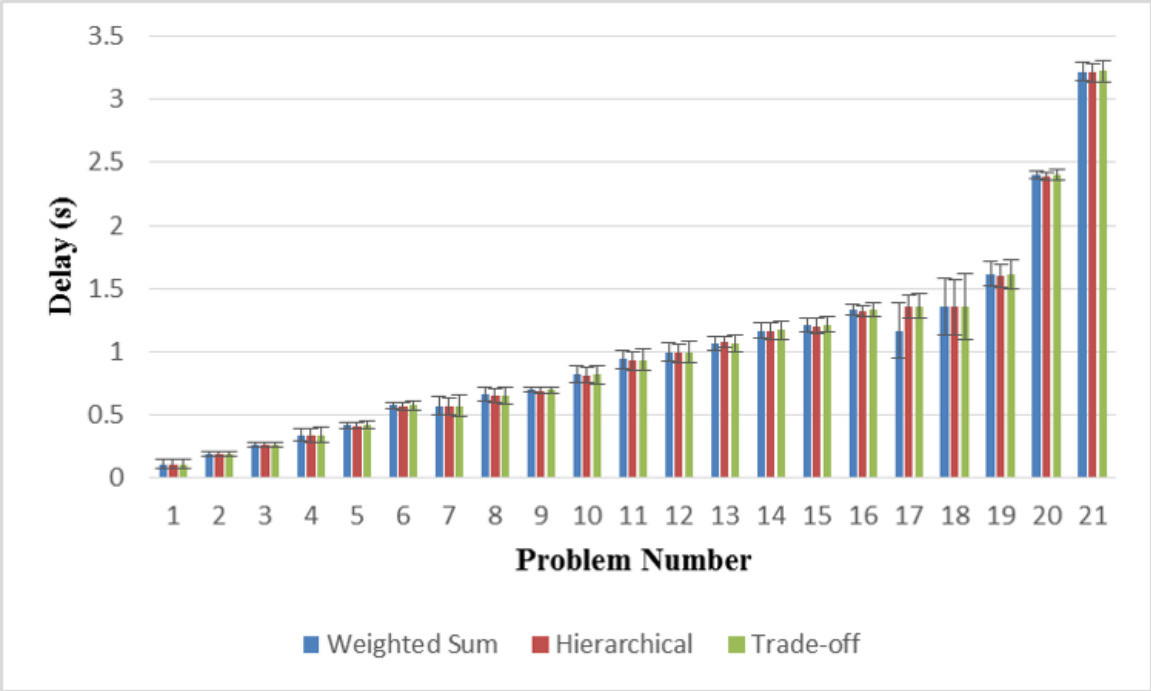

Figure 4.7: Delay comparison with 95% CI for small scale problems (average over 4 instances)

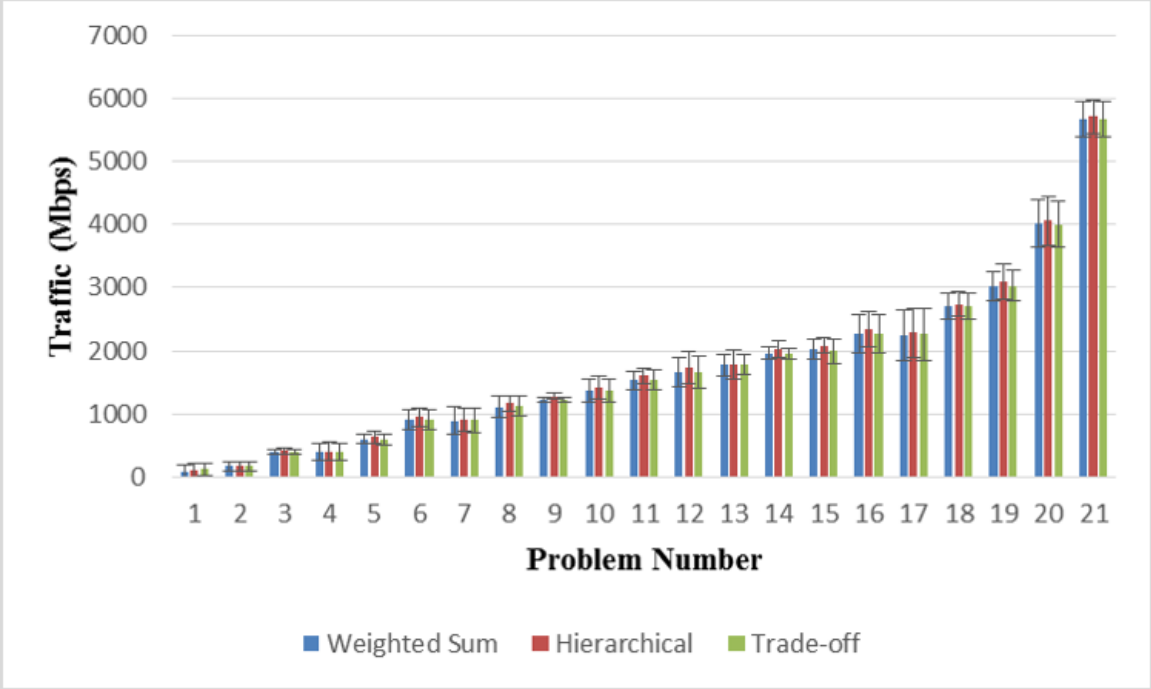

Figure 4.8: Traffic comparison with 95% CI for small scale problems (average over 4 instances)

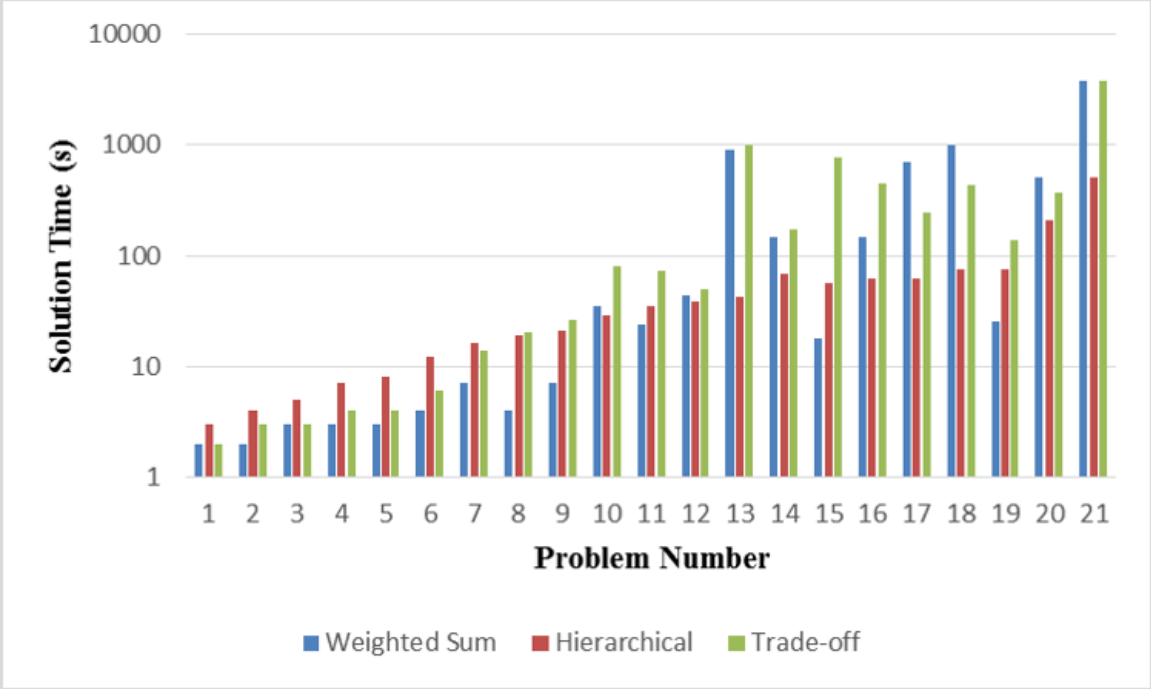

Figure 4.9: Solution time comparison for small scale problems (average over 4 instances)

introduced in the network resulting in more traffic requests from the fog nodes but the number of possible fog node placement location is the same. Hence, many of the edge device clusters are connected to the cloud. The weighted sum method and the trade-off method return similar results for delay and traffic due to the use of the maximum and minimum value of the objective functions as formulated in equations 3.25 and 3.30.

In general, with the increase of input size, the time taken to obtain an optimal solution must increase as the model must consider a lot of combinations of where to install the fog nodes, what type of fog node needs to be installed and which edge device cluster should be connected to it. From Figure 4.9, it can be seen that the solution time increases in a non linear pattern. This is due to the use of the Branch and Bound (B&B) algorithm [79] used by the solver [76]. In B&B, at first the search space is recursively split into smaller search spaces called branching and tries to find the minimum objective function in the smaller search space. To avoid the brute-force search and testing all the candidate solutions, the algorithm uses heuristic to keep track of bounds on the minimum that it is trying to find and these bounds are used to cut back the search space by eliminating the candidate solutions which cannot give

optimal solution. To add with that, for the same problem, different multiobjective optimization methods generate different search spaces. For instance, from problem seven onward, the hierarchical method takes a lot less time compared to at least one of the methods and in some cases both methods. The reason it takes less time is because the objective function is traffic towards the cloud which according to Equation 3.6 has less variables and constraint 3.29 provides a strong feasible initial solution for the particular method compared to the other two optimization methods. Moreover, for each problem, the hierarchical method runs for 8 iterations which is 3 less than the other two methods. Although the CI for the average solution time was calculated, it could not be represented in the figure because of the semi-log scale used in the plot.

Comparison to the Best

Table 4.13 show the comparison of the results among the three multiobjective optimization methods for small scale problems. The first column represents the problem number. The second and third columns illustrate the percentage gap for the delay and traffic between the weighted sum method and the minimum value obtained from all three methods. Similarly, columns 4 and 5 show the percentage gap for the delay and traffic between the hierarchical method and the minimum value achieved from all three methods. Finally, columns 6 and 7 represent the delay and traffic percentage difference between the trade-off method and the minimum objective function value returned by all three methods. A 0% implies that the delay or traffic value obtained by that method for the particular problem is the minimum among all the three methods, hence, there is no percentage difference. The mean percentage gap for the weighted sum method for the delay and traffic is 0.73% and 0.15% respectively. The mean of the delay gap for the hierarchical method is 0.82% and the traffic is 3.49%. For the trade-off method, the mean of the delay gap obtained was 1.46%, whereas, the mean of the traffic difference was 1.51%. From the results, it can be concluded that, for small scale problems, the weighted sum method returns the best optimal results among the three methods.

Table 4.13: Solutions comparison among the three methods over 4 instances

	Weighted Sum		**Hierarchical**		**Trade-off**	
Problem	**Delay Gap (%)**	**Traffic Gap (%)**	**Delay Gap (%)**	**Traffic Gap (%)**	**Delay Gap (%)**	**Traffic Gap (%)**
1	1.5	0	0	13	1.4	26
2	1	0.2	0	0	1	0.2
3	0.65	0	0	3.8	0.65	0
4	0.89	0	0	4.2	0.89	0
5	1.1	0.81	0	6	1.5	0
6	0.53	0	0	3.3	0.53	0.058
7	0.022	0	0.061	0.82	0	0.57
8	2.1	0	0	5.1	0.68	1.3
9	0.9	0.22	0	4.2	1.1	0
10	0.53	0	0	3.4	0.53	0.092
11	1	0	0	4.9	0.84	0.66
12	1.1	0	0	4.7	0.95	0.36
13	0.033	0	1.1	0.28	0	0.48
14	0.8	0.12	0	3.8	0.92	0
15	0.54	1.5	0	4.7	1.2	0
16	0.8	0	0	3	0.78	0.38
17	0	0	16	1.9	16	0.92
18	0.34	0	0	1.3	0.41	0.16
19	0.92	0	0	2.8	0.73	0.52
20	0.25	0.35	0	1.4	0.38	0
21	0.22	0	0	0.62	0.25	0.035
Mean	**0.73**	**0.15**	**0.82**	**3.49**	**1.46**	**1.51**

Table 4.14: Solutions comparison between single objective and multiobjective (over 4 instances)

	Single Objective		**Weighted Sum**		**Hierarchical**		**Trade-off**	
Problem	**Delay (s)**	**Traffic (Mbps)**	**Delay Gap (%)**	**Traffic Gap (%)**	**Delay Gap (%)**	**Traffic Gap (%)**	**Delay Gap (%)**	**Traffic Gap (%)**
1	0.111	95	1.8	2.1	0	16	1.8	28
2	0.191	163	1	7.4	0	7.4	1	7.4
3	0.265	390	0.38	2.8	0	6.7	0.38	2.8
4	0.34	370	0.88	5.7	0	10	0.88	5.7
5	0.414	570	1.2	6	0	11	1.4	5.1
6	0.573	899	0.52	2.2	0	5.6	0.52	2.2
7	0.566	851	1.1	5.5	1.2	6.5	1.1	6.1
8	0.655	1070	2.1	3.9	0	9.2	0.61	5.2
9	0.694	1197	1	3.1	0	7.2	1.2	2.8
10	0.817	1322	0.61	3.9	0	7.3	0.61	3.9
11	0.931	1484	1.1	3.3	0	8.4	0.86	4
12	0.988	1619	1.1	2.9	0	7.7	1	3.3
13	1.058	1712	0.85	3.8	1.9	4.1	0.85	4.3
14	1.158	1903	0.95	2.9	0.17	6.7	1.1	2.8
15	1.199	1936	0.92	4.4	0.33	7.6	1.5	2.8
16	1.319	2184	0.99	3.9	0.15	7.1	0.91	4.3
17	1.158	2174	0.95	3.1	17	5	18	4
18	1.351	2647	0.59	2	0.22	3.4	0.67	2.2
19	1.596	2962	1.3	1.9	0.38	4.7	1.1	2.4
20	2.383	3919	0.63	2.4	0.38	3.6	0.76	2.1
21	3.2	5592	0.63	1.4	0.41	2	0.66	1.4
Mean			**0.98**	**3.55**	**2.21**	**7**	**1.76**	**4.90**

Comparison to Single Objective Function for Small Scale Problems

Until now, we compared the optimal solutions and the time taken in obtaining the solution for different small scale problems but this does not explain the reliability of the methods and their Pareto front. There are different ways to measure the reliability of multiobjective optimization models. One of the ways is to optimize each objective function individually in order to study the diversity characteristics of the trade-off surface in a Pareto front. Table 4.14 represents a comparison between the results of single objective optimization and multiobjective optimization for 21 problems over the 4 different instances (refer to Appendix C for the results of each instance). The first column in the table represents the problem number. The optimal delay and traffic for each problem when optimized individually are given in columns 2 and 3. Columns 4, 5, 6, 7, 8 and 9 represent the percentage difference between the individual single objective optimal result and the compromised optimal values obtained using the three multiobjective optimization methods shown in Table 4.12. The mean difference in optimal delay for the weighted sum method is 0.98% and the traffic is 3.55%. For the hierarchical method, the mean difference in delay and traffic for all the small scale problems is 2.21% and 7% respectively. Whereas, the trade-off method attained a mean difference of 1.76% for optimal delay and 4.9% for optimal traffic compared to individual optimization. For half of the problems, the optimal delay obtained by the hierarchical method is exactly the same as the individually optimized delay but the difference in traffic is high compare to the other two methods. This is due to the tight delay constraint which ensures minimal sacrifice of optimal delay in the network. Overall, the difference in each of the result is not more than 7%, which shows that for small scale problems, all of the three methods are able to solve multiobjective functions, returning nearly the same result as the single objective optimization for each objective function.

The Hypervolume Indicator

The hypervolume indicator is another widely used performance metric for multiobjective optimization. Figure 4.10 shows the HV comparison between the three methods for the small scale problems. For different problem sizes we obtain different HV ranging from a lowest value of 0.274 to a highest value of 0.783. From the figure, it can be observed that the hierarchical method results in smaller HV compared to the other two methods. This is due to the lack of non-dominated member in the

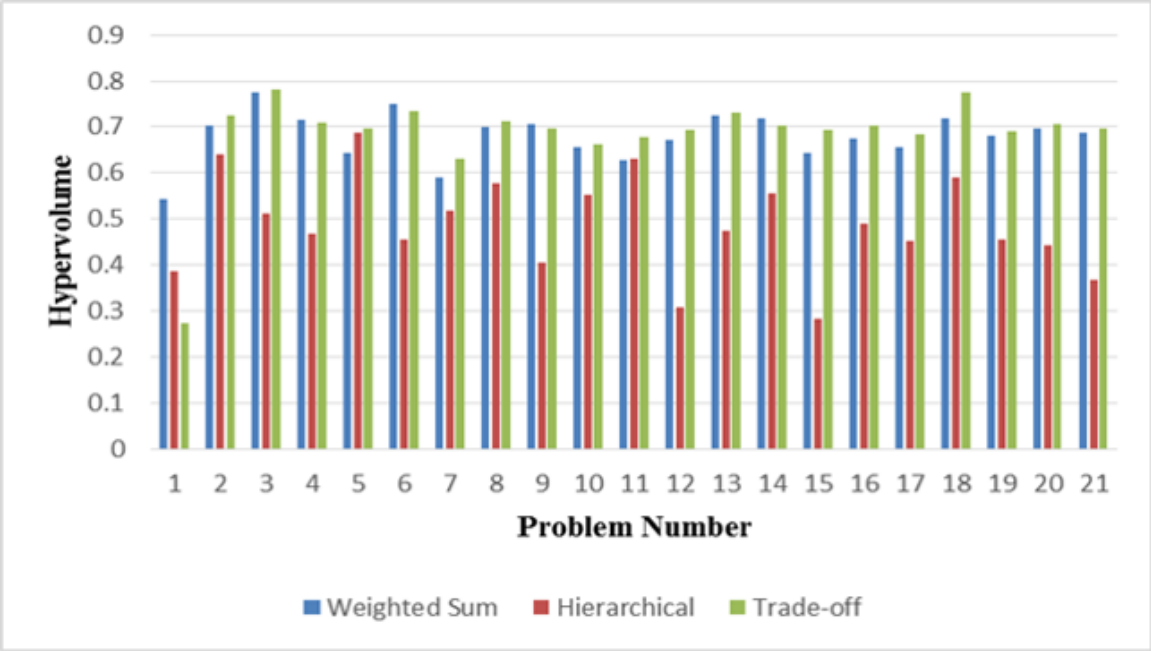

Figure 4.10: Average HV indicator for small scale problems

Pareto front, as the hierarchical method can at most generate only eight points or less in the Pareto front. Moreover, the delay constraint does not allow to explore as many combinations as the other two methods can do. The weighted sum method and the trade-off method have similar HV for its respective Pareto front for each problem. However, the trade-off method outperforms the weighted sum method by a slight margin proving that it has more diverse non-dominated solutions and better convergence towards the true approximated Pareto front among all the three methods.

The following section explores the effect of increasing the problem size to a larger scale.

4.3.3 Large scale problems

The second set of problems shows how the three methods behave when the number of possible placement locations is increased. The number of possible placement locations is increased from 5 to 10 for a range between 10 to 60 edge device clusters (see Table 4.15). The number of edge device cluster is stopped at 60 because after 45 edge device clusters, optimal solutions were not guaranteed as several iterations

Table 4.15: Problem size for the large scale FPP

Problem	**# of edge device clusters**	**# of possible placement locations**
22	10	10
23	15	10
24	20	10
25	25	10
26	30	10
27	35	10
28	40	10
29	45	10
30	50	10
31	55	10
32	60	10

where not returning the optimal solution within the time limit. A total of 44 problems were solved with 11 different problem sizes and the average of each problem size was used for the analysis of the problem set (refer to Appendix B for the results of all instances). We used the same CPLEX and hardware environment as defined for the example problem.

The first column in Table 4.16 represents the problem number with respect to the problem size presented in Table 4.15. The second, third and fourth columns represent respectively the total delay in the network, the total traffic going to the cloud and the solution time for the weighted sum optimization method; whereas the fifth, sixth and seventh columns represent respectively the total delay in the network, the total traffic going to the cloud and the solution time for the hierarchical method. Finally, columns 8, 9 and 10 show respectively the total delay in the network, the total traffic going to the cloud and the solution time for the trade-off method. It is important to note that some of the iterations could not find the optimal value within the time limit. Although these solutions are not optimal, they were still included in the Pareto front. The fuzzy-based mechanism which was used to find the best compromise result always returned the membership value which has been solved within the time limit.

Table 4.16: Results obtained from the solver for large scale problems (average over 4 instances)

	Weighted Sum			Hierarchical			Trade-off		
Problem	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)
22	0.119	127	2	0.119	127	5	0.119	129	2
23	0.197	157	19	0.196	164	13	0.197	166	6
24	0.261	226	258	0.259	235	124	0.261	237	2700
25	0.341	355	6020	0.34	371	807	0.343	344	10445
26	0.449	607	1230	0.444	630	320	0.447	613	936
27	0.492	576	1187	0.494	627	2504	0.495	612	801
28	0.589	713	2110	0.579	785	1753	0.585	731	946
29	0.691	840	4612	0.687	908	4383	0.691	845	6414
30	0.666	897	2001	0.66	972	2837	0.664	924	1615
31	0.783	944	6063	0.769	1058	3359	0.783	950	6080
32	0.876	1195	12054	0.856	1254	9009	0.864	1210	8370

As it is for a small number of problems, we cannot make sure that this will always be the case.

All the three methods returned a lowest delay value of 0.119 seconds, whereas the weiighted sum method and the hierarchical method attained the lowest traffic value of 127 Mbps.

Figure 4.11 graphically represents the delay comparison between the three methods for large scale problems over four different instances with a 95% CI. As it can be noticed from the figure, as the problem size increases, the delay in the network also increases. We can observe a linearity in the increment of delay for problems 22 to 29, but the pattern is disrupted by problem 30 for which the delay in the network is less than the previous problem size. The pattern continues for problem 31 and 32. However, we believe that, if we run each problem for more than four instances, a more linear pattern can be seen without any interruption.

The traffic comparison between the three methods for the four different instances of the second problem set is illustrated in Figure 4.12. The vertical bars on the columns show a 95% CI for each problem. The traffic obtained from the weighted sum and trade-off are very similar, whereas the hierarchical method attains a slightly higher traffic for every problem. This is because, in the FPP, the hierarchical method

Figure 4.11: Delay comparison with 95% CI for large scale problems (average over 4 instances)

was proposed with respect to a delay sensitive network. It can be clearly observed from the figure, that the traffic in the cloud increases linearly with the problem size for first 5 problems. However, problem 27 interrupts the pattern. The pattern again continues for the last 5 problems. If the number of instances is increased a smoother pattern can be observed.

The average solution times are illustrated in Figure 4.13 for the three methods over four instances. As the vertical axis in the figure is in logarithmic scale, the calculated CI could not be illustrated. The figure shows that there is a non linear increasing pattern for the average solution time for all the problems. The randomness in the solution time relates to the theory that the combinational space are shortened by the solver using heuristics. As the figure depicts, for the same problem size there is a visible difference in solution time for the three methods. For example, to solve problem 25, the weighted sum method takes 6020 seconds, the hierarchical method solves in 807 seconds and the trade-off method takes 10445 seconds. Each method generates a different search space for the same problem, hence solving the same problem size

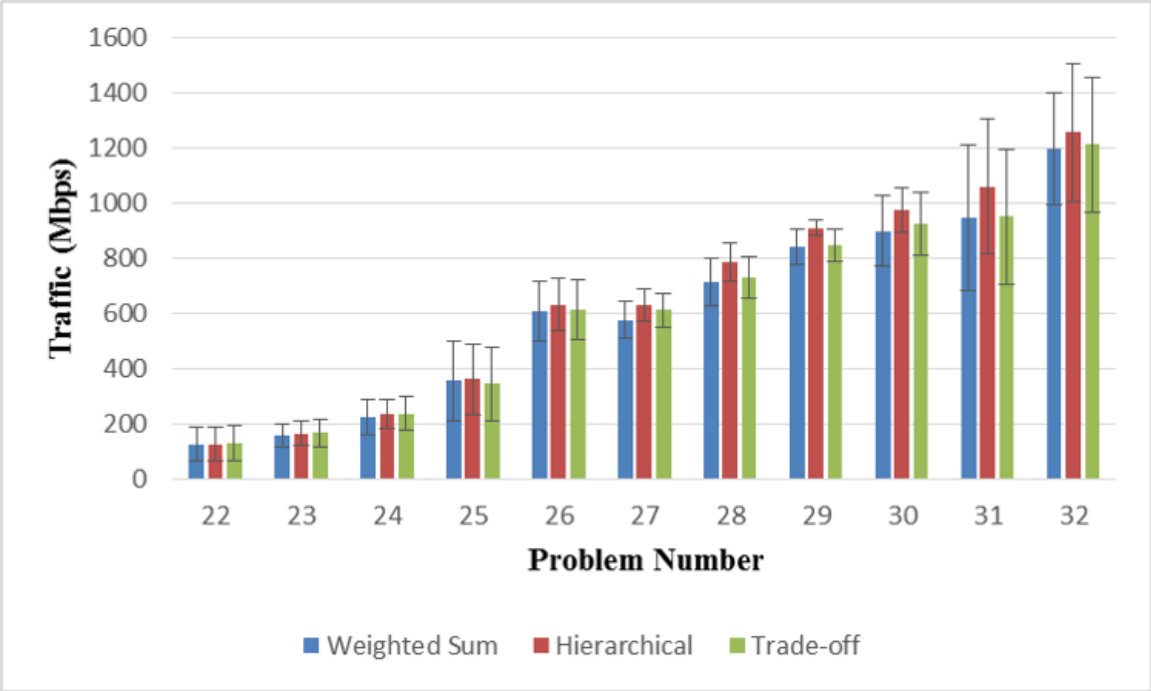

Figure 4.12: Traffic comparison with 95% CI for large scale problems (average over 4 instances)

Figure 4.13: Solution time comparison for large scale problems (average over 4 instances)

Table 4.17: Average solution comparison among the three methods over 4 instances for large scale problems

	Weighted Sum		Hierarchical		Trade-off	
Problem	**Delay Gap (%)**	**Traffic Gap (%)**	**Delay Gap (%)**	**Traffic Gap (%)**	**Delay Gap (%)**	**Traffic Gap (%)**
22	0.013	0	0.019	0	0	1.2
23	0.74	0	0	4.3	0.67	5.7
24	0.55	0	0	4.1	0.44	5
25	0.56	3.3	0	7.7	0.96	0
26	1	0	0	3.7	0.68	1
27	0	0	0.22	8.7	0.57	6.2
28	1.6	0	0	10	1.1	2.6
29	0.67	0	0	8.1	0.63	0.56
30	0.94	0	0	8.3	0.65	3
31	1.9	0	0	12	1.9	0.65
32	2.3	0	0	4.9	0.94	1.2
Mean	**0.93**	**0.3**	**0.022**	**6.53**	**0.78**	**2.46**

with different solution time.

Comparison to the Best

Table 4.17 represents the comparison of results among the three multiobjective optimization methods for large scale problems. Column 1 in the table shows the problem number. The second and third columns show the percentage difference for the average delay and traffic between the weighted sum method and the minimum value obtained from all three methods. Columns 4 and 5 represent the percentage gap for the average delay and traffic between the hierarchical method and the minimum value achieved from all three methods. Whereas, columns 6 and 7 illustrate the average delay and traffic percentage difference between the trade-off method and the minimum objective function value returned by all three methods. Although, the hierarchical method returns the best results for the delay value in the network, the weighted sum method performs the best when both objectives are considered, with a lowest mean percentage difference of 1.23 % when both objectives are combined.

Table 4.18: Solutions comparison between single objective and multiobjective over 4 instances for large scale problems

	Single Objective		Weighted Sum		Hierarchical		Trade-off	
Problem	Delay (s)	Traffic (Mbps)	Delay Gap (%)	Traffic Gap (%)	Delay Gap (%)	Traffic Gap (%)	Delay Gap (%)	Traffic Gap (%)
22	0.119	127	0	0	0	0	0	1.6
23	0.196	154	0.51	1.9	0	6.5	0.51	7.8
24	0.259	224	0.77	0.89	0	4.9	0.77	5.8
25	0.339	316	0.59	12	0.29	17	1.2	8.9
26	0.444	582	1.1	4.3	0	8.2	0.68	5.3
27	0.493	552	0.2	4.3	0.2	14	0.41	11
28	0.579	679	1.7	5	0	16	1	7.7
29	0.686	784	0.73	7.1	0.15	16	0.73	7.8
30	0.659	836	1.1	7.3	0.15	16	0.76	11
31	0.768	874	2	8	0.13	21	2	8.7
32	0.853	1144	2.7	4.5	0.35	9.6	1.3	5.8
Mean			**1.14**	**5.53**	**0.21**	**12.92**	**0.94**	**7.4**

Comparison to Single Objective Function for Large Scale Problems

To understand the reliability of the three methods for large scale problems, the two objective functions were optimized individually. The average results over the 4 different instances (refer to Appendix C for the results of each instance) are tabulated in Table 4.18 and compared with the multiobjective optimization results. In the table, column 1 represents the problem number referring to the the problem size from Table 4.15. The delay and traffic for each individually optimized problem is given in columns 2 and 3 respectively. Columns 4 and 5 show the delay and traffic gap between the weighted sum method and the single objective. The delay and traffic gap for the hierarchical method are given in columns 6 and 7. Finally, the eighth and ninth columns show the gap obtained from the trade-off method.

The weighted sum method returns a combined lowest percentage gap of 6.67% among the three methods. Overall, the percentage gap are very reasonable considering the trade-off, summarizing that for large scale FPP, the proposed methods can return multiobjective optimized solution as the near optimal solution of single objective functions.

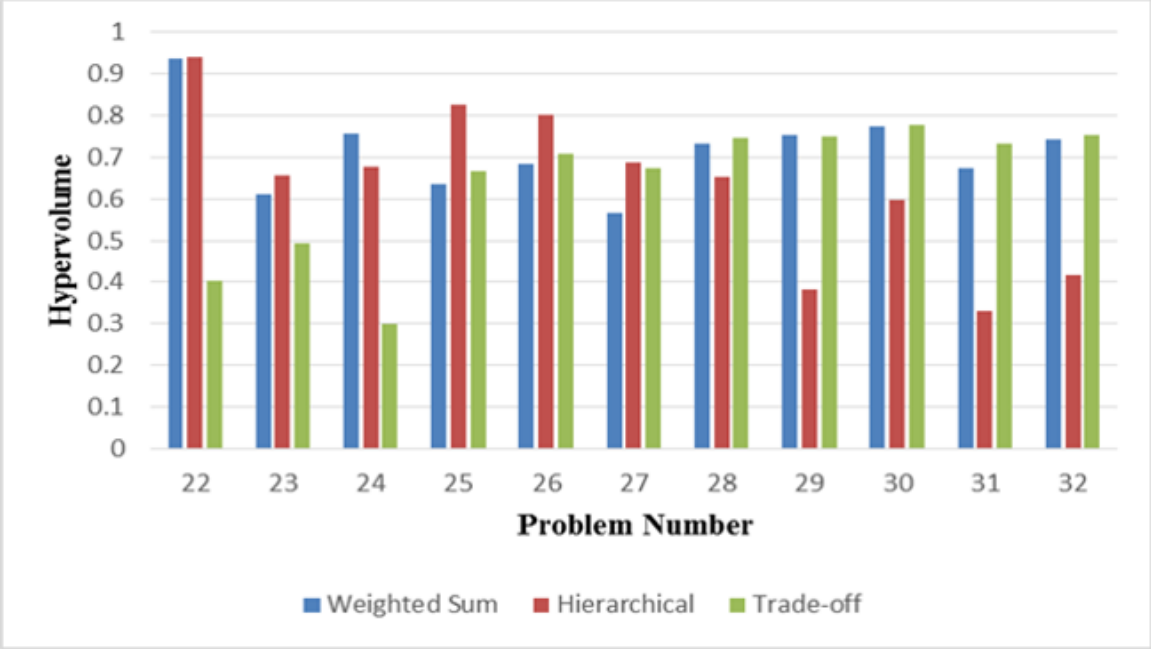

Figure 4.14: Average HV indicator for large scale problems

The Hypervolume Indicator

Figure 4.14 portrays the average HV for large scale FPP for the three methods. The highest HV of 0.9396 was achieved by the hierarchical method, whereas the trade-off method gives the lowest HV of 0.2993. It can be observed that the hierarchical and the trade-off method show an opposite pattern with the increase of problem size. For the first problem, the hierarchical method gives the best Pareto front but keeps on getting worse as the problem size increases and it is the opposite for the trade-off method. Compared to the other two methods, the weighted sum gives a more consistent set of non-dominated solutions for all the problem sizes. Hence, it can be deduced that the weighted sum generates the best dominated space of the solutions in a Pareto front.

4.3.4 Impact of the Number of Possible Placement Location

In the previous two sections, we analyzed the results obtained for different number of edge device clusters with a fixed number of placement locations i.e., 5 and 10. In this section, we investigate the effect of the two different number of possible placement locations when the number of edge device clusters is set to 10, 15, 20, 25, 30, 35, 40,

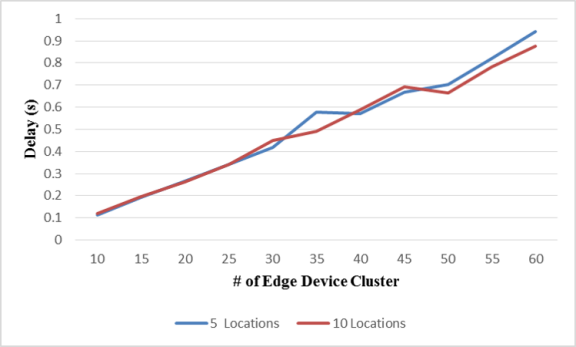

Figure 4.15: Delay comparison for different number of possible locations using the weighted sum method

45, 50, 55 and 60.

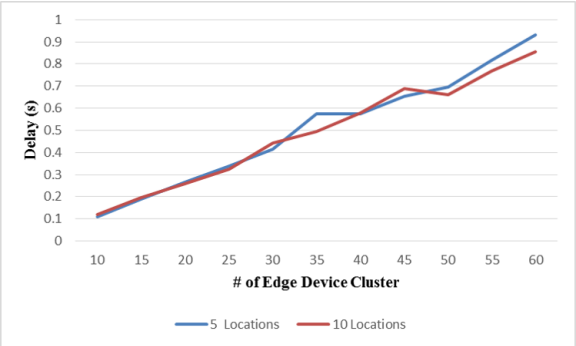

Figure 4.16: Delay comparison for different number of possible locations using the hierarchical method

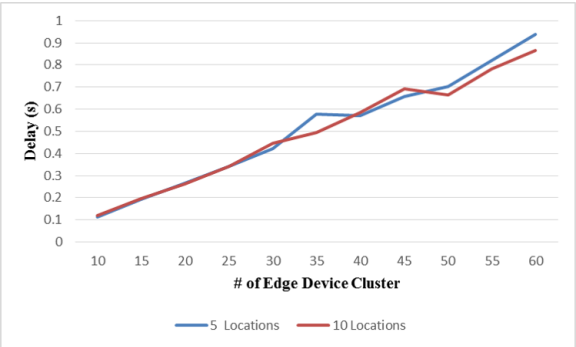

Figure 4.17: Delay comparison for different number of possible locations using the trade-off method

Figures 4.15 to 4.23 represent the results for each set of edge device clusters

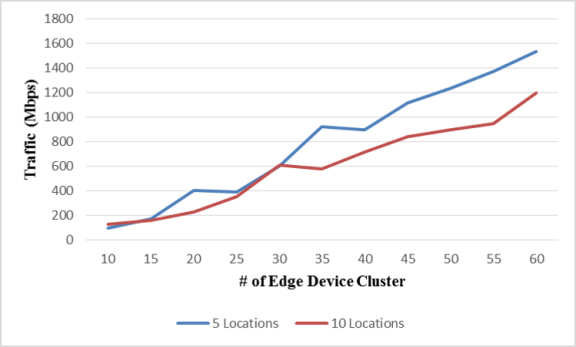

Figure 4.18: Traffic comparison for different number of possible locations using the weighted sum method

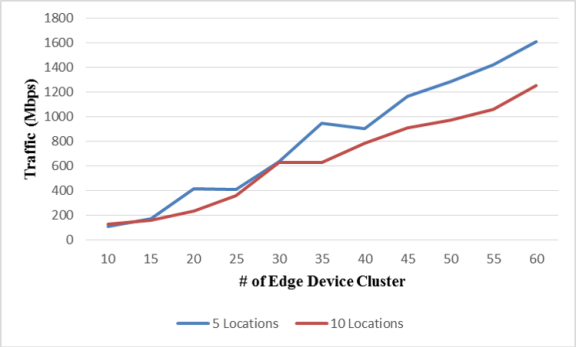

Figure 4.19: Traffic comparison for different number of possible locations using the hierarchical method

grouped by the number of maximum possible placement locations for the three methods respectively. From the figures, it can be seen that as the number of edge device clusters increases, the delay gap for the two different number of placement location is also increasing for all the three methods. To be exact, the delay is higher when only 5 possible placement locations are used. As more possible placement locations are added and for the same number of edge device clusters, more fog nodes can be installed in the network and better locations can be selected resulting in a decrease of the total network delay. Moreover, as the number of possible placement location is increasing, the budget is also increasing. In terms of the average traffic, a similar pattern can be observed. With the availability of more fog locations, less traffic is

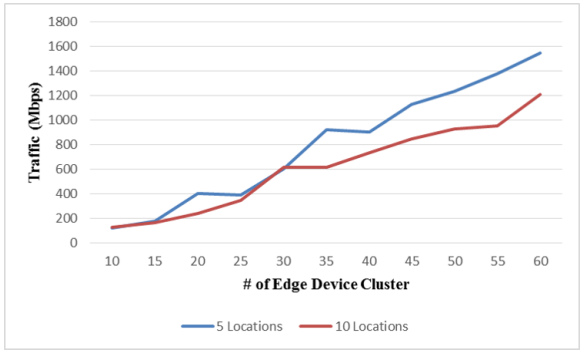

Figure 4.20: Traffic comparison for different number of possible locations using the trade-off method

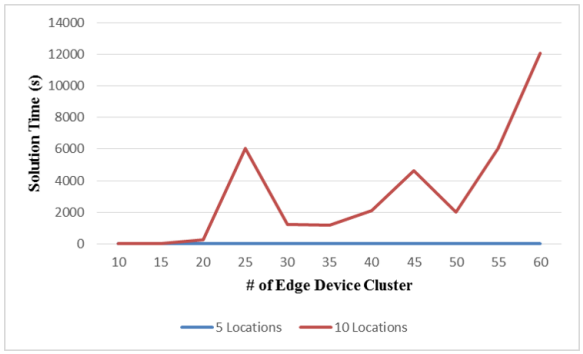

Figure 4.21: Solution time comparison for different number of possible locations using the weighted sum method

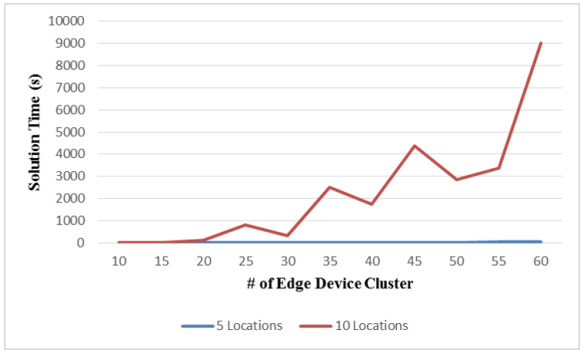

Figure 4.22: Solution time comparison for different number of possible locations using the hierarchical method

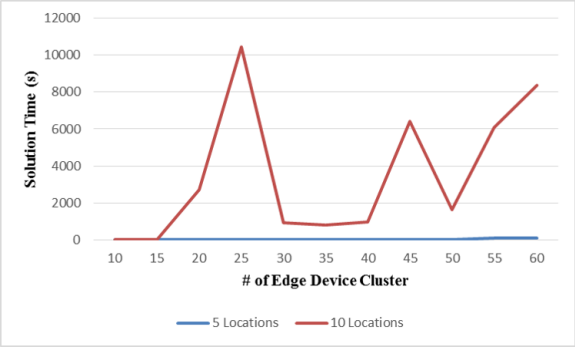

Figure 4.23: Solution time comparison for different number of possible locations using the trade-off method

sent to the cloud as most of the requests are processed at the fog layer.

Unlike the objective functions, the solution time plotted in figures 4.21 to 4.23 show more randomness. Moreover, a larger gap in the solution time can be observed for all the three methods. For example, the solution time for 60 edge device clusters and 5 possible placement locations is 24 seconds for the weighted sum, 35 seconds for the hierarchical and 73 seconds for the trade-off method. Whereas, for the same number of edge device cluster and 10 maximum placement locations, the solution time is 12054 seconds for the weighted sum, 9009 seconds for the hierarchical and 8370 seconds for the trade-off method. The larger gap is because the FPP is an integer linear programming problem which is NP-Hard [80].

4.4 Chapter Summary

Planning a fog network is a complex problem. Some of the difficult aspects of FPP include the finding of information about the input parameters, fog node features, finding the location to deploy fog nodes, the assignment of the edge device clusters, etc. In this chapter, we identified the features and characteristics of the proposed FPP based on the optimization results for different topologies. The results for the three multiobjective optimization methods were obtained in terms of total delay in the network, traffic in the cloud and solution time. The detailed example returned exactly the same result as the manually derived, helping in understanding the model.

To evaluate the FPP model, two different sets of problem sizes were generated. 21

different problem sizes were solved for the small scale problems and 11 different problem sizes were solved for the large scale problems. With the increase of problem size, delay, traffic and solution time also increases. The objective functions had a linear pattern in the increment, whereas the solution time increased in a non linear form. This illustrates that the FPP is a NP-hard problem and further increase in problem size will increase the solution time exponentially. Furthermore, the three methods were compared, in terms of the objective functions for all the problems. For most of the problem sizes, the hierarchical method was able to return good solutions for the delay in the network at the expense of traffic. This makes the hierarchical method the best method for a delay sensitive network. Whereas, the weighted sum method achieved the best compromised result among the three multiobjective optimization methods while considering the trade-off between the objective functions which should be used for delay-traffic aware networks.

The performance of the methods used for the model was also analyzed showing that the multiobjective methods are able to give similar results compared to the single objective optimization. Moreover, the HV value indicates that the weighted sum and trade-off methods are able to generate satisfactory Pareto fronts. Finally, we compared the results for two different numbers of possible fog node placement locations. It was seen that with the same number of edge device cluster, if we increase the maximum possible placement locations, the delay in the network and the traffic in the cloud decreases but the solution time increases significantly.

The next chapter summarizes the thesis and describes the different improvements that can be done as future work.

Chapter 5

Conclusions and Future Work

Since, fog networks are designed to be closer to the edge devices, it is very important to develop efficient network planning tools for the best service. This thesis studies the planning and design of a brand new fog networks by simultaneously finding the optimal location, number, capacity and the type of the fog nodes, as well as the interconnection between the fog nodes and the cloud. The optimization model minimizes the network delay and the traffic in the cloud which are two of the most important parameters when dealing with the fog networks. To obtain the optimal solutions, the memory, vCPU and the link speed of the edge devices were considered. The FPP is an integer linear problem that ensures the minimization of the objective functions with respect to uniqueness, capacity, assignment, budget and integrality constraints.

Before solving the planning model, different sizes of problems were generated with different edge device input parameters and randomized locations for edge device clusters and the possible fog node placements. A detailed example was solved to better understand the planning model. As the FPP contains two objective functions, three multiobjective methods were used to obtain optimal solutions. More precisely, the weighted sum, the hierarchical and the trade-off methods were used in solving the FPP and a fuzzy-based mechanism was used to obtain the best compromised results. Some general trends can be observed for the delay in the network, the traffic in the cloud and the solution time for the different topologies. As the input size increases, the objective functions and the solution time for each multiobjective optimization method also increases. However, for the same number of edge device clusters and a higher number of possible fog node placement locations, the objective functions decrease but the solution time increases. For example, the planning problem with 50

edge device clusters and 5 possible placement locations results in a minimum delay of 0.694 seconds and a minimum traffic of 1231 Mbps with a lowest solution time of 7 seconds. Whereas, for the same number of edge device clusters and 10 possible fog node placement locations the optimal solution for delay is 0.66 seconds and the traffic is 897 Mbps with a lowest solution time of 897 seconds.

In term of the methods, the weighted sum method was able to return the best trade-off results for the delay and the traffic. The hierarchical method was able to achieve better delay but the traffic was worse compared to the other two methods. For small scale problems, the trade-off method generated the best set of non-dominated solutions. It shows that, for small scale problems, the trade-off method can be used for situations were the decision maker can make their own choice of best result from the set of non-dominated solutions, instead of using the fuzzy-based approach. The weighted sum method generated the best Pareto optimal set for the large scale problems.

5.1 Future Works

To the extent of our knowledge, research in planning and design of fog networks is a very new topic. Hence, the work in this study can be extended to many possible directions. Some of them are as follows:

- We have restricted our delay equation to transmission, propagation and processing delays. For a better approximation of the end-to-end delay, the addition of queuing delay could be a major improvement. For the FPP, the budget is limited to a fixed percentage of the maximum cost, in the future the model can be analyzed by varying the budget percentage.

- One of the problems with the weighted sum method is that it is impossible to obtain points on non-convex region of the Pareto front in the criterion space [81]. The exponential weighted criterion method can be used to obtain the points on the non-convex region of the Pareto optimal set [82]. The hierarchical and the trade-off methods are modified according to a delay sensitive network, but sometimes it is hard for the decision makers to articulate a preference. In that case, different global criterion methods can be used to obtain a better trade-off between the objective functions.

- The solution time for the exact algorithm is very high. To decrease the solution time, we can run many problems and test different parameter settings in CPLEX for each problem. As the FPP is a NP-hard problem and the solution time increases exponentially with the increase of input size, obtaining the best parameters for the solver will provide a negligible improvement. A viable improvement can be the use of multiobjective heuristic algorithms like Strength Pareto Evolutionary Algorithm (SPEA) or Non-dominated Sorting Genetic Algorithm II (NSGA-II). Although the solutions obtained will not be optimal, a significant improvement can be achieved in terms of solution time.

- After deploying the fog network, the next step is to manage the resource and the workload. In a dynamic network where the edge device location and density is changing, efficient algorithms need to be developed to assign the edge devices to appropriate fog nodes. The SDN architecture can be used for enhancement of performance and betterment of traffic and fog node management.

- Another future direction can be the expansion of fog networks. Once the fog network is planed and implemented, over time, more edge device clusters may be added to the network. The proposed model can be extended to an expansion model, where new fog nodes can be added (or replaced) without changing the whole network infrastructure.

List of References

[1] "Micro-datacentre: What it problems it solves and what workload systems suit it." http://www.computerweekly.com/feature/Micro-datacentre-What-IT-problems-it-solves-and-what-workload-systems-is-it-b (Accessed on 12/22/2017).

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, MCC '12, (New York, NY, USA), pp. 13–16, ACM, 2012.

[3] S. Liu and M. St-Hilaire, "Cooperative algorithm for the global planning problem of umts networks," in *IEEE Global Telecommunications Conference GLOBECOM*, pp. 1–5, Dec 2010.

[4] S. Z. Ali and R. J. Read, "Design and performance evaluation of an optimization methodology for optimal solution of cellular layout design problems," in *Signals, Circuits and Systems. International Symposium on SCS*, vol. 2, pp. 521–524, 2003.

[5] P. Ngatchou, A. Zarei, and A. El-Sharkawi, "Pareto multi objective optimization," in *Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems*, pp. 84–91, Nov 2005.

[6] C. Zhu, L. Xu, and E. D. Goodman, "Generalization of pareto-optimality for many-objective evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, pp. 299–315, Apr 2016.

[7] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, pp. 7–18, May 2010.

[8] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster, "Virtual infrastructure management in private and hybrid clouds," *IEEE Internet Computing*, vol. 13, pp. 14–22, Sept 2009.

[9] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: Towards a cloud definition," *SIGCOMM Comput. Commun. Rev.*, vol. 39, pp. 50–55, Dec. 2008.

[10] I. Foster and C. Kesselman, eds., *The Grid: Blueprint for a New Computing Infrastructure*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999.

[11] L. K. Arya and A. Verma, "Workflow scheduling algorithms in cloud environment - a survey," in *Recent Advances in Engineering and Computational Sciences (RAECS)*, pp. 1–4, Mar 2014.

[12] I. Management Association, *Grid and Cloud Computing: Concepts, Methodologies, Tools and Applications: Concepts, Methodologies, Tools and Applications*. Premier reference source, Information Science Reference, 2012.

[13] M. R. Nami and K. Bertels, "A survey of autonomic computing systems," in *Autonomic and Autonomous Systems. Third International Conference on ICAS*, pp. 26–26, Jun 2007.

[14] R. Buyya, C. S. Yeo, and S. Venugopal, "Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities," in *High Performance Computing and Communications.10th IEEE International Conference on HPCC*, pp. 5–13, IEEE, 2008.

[15] T. Dillon, C. Wu, and E. Chang, "Cloud computing: Issues and challenges," in *24th IEEE International Conference on Advanced Information Networking and Applications*, pp. 27–33, Apr 2010.

[16] Y. Jadeja and K. Modi, "Cloud computing - concepts, architecture and challenges," in *International Conference on Computing, Electronics and Electrical Technologies (ICCEET)*, pp. 877–880, Mar 2012.

[17] S. Ramgovind, M. M. Eloff, and E. Smith, "The management of security in cloud computing," in *Information Security for South Africa*, pp. 1–7, Aug 2010.

[18] O. Biran, A. Corradi, M. Fanelli, L. Foschini, A. Nus, D. Raz, and E. Silvera, "A stable network-aware vm placement for cloud systems," in *12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012)*, pp. 498–506, May 2012.

[19] I. Hwang and M. Pedram, "Hierarchical virtual machine consolidation in a cloud computing system," in *IEEE Sixth International Conference on Cloud Computing*, pp. 196–203, Jun 2013.

[20] B. Kantarci, L. Foschini, A. Corradi, and H. T. Mouftah, "Inter-and-intra data center vm-placement for energy-efficient large-scale cloud systems," in *IEEE Globecom Workshops*, pp. 708–713, Dec 2012.

[21] W. Fang, X. Liang, S. Li, L. Chiaraviglio, and N. Xiong, "Vmplanner: Optimizing virtual machine placement and traffic flow routing to reduce network power costs in cloud data centers," *Computer Networks*, vol. 57, no. 1, pp. 179 – 196, 2013.

[22] J. Dong, X. Jin, H. Wang, Y. Li, P. Zhang, and S. Cheng, "Energy-saving virtual machine placement in cloud data centers," in *13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, pp. 618–624, May 2013.

[23] T. C. Ferreto, M. A. Netto, R. N. Calheiros, and C. A. D. Rose, "Server consolidation with migration control for virtualized data centers," *Future Generation Computer Systems*, vol. 27, no. 8, pp. 1027 – 1034, 2011.

[24] N. Kord and H. Haghighi, "An energy-efficient approach for virtual machine placement in cloud based data centers," in *The 5th Conference on Information and Knowledge Technology*, pp. 44–49, May 2013.

[25] P. Hofmann and D. Woods, "Cloud computing: The limits of public clouds for business applications," *IEEE Internet Computing*, vol. 14, no. 6, pp. 90–93, 2010.

[26] G. Pardo-Castellote, "Omg data-distribution service: architectural overview," in *23rd International Conference on Distributed Computing Systems Workshops. Proceedings.*, pp. 200–206, May 2003.

[27] Z. Shelby, K. Hartke, and C. Bormann, "The constrained application protocol (coap)," 2014.

[28] M. Singh, M. A. Rajan, V. L. Shivraj, and P. Balamuralidhar, "Secure mqtt for internet of things (iot)," in *Fifth International Conference on Communication Systems and Network Technologies*, pp. 746–751, Apr 2015.

[29] S. Vinoski, "Advanced message queuing protocol," *IEEE Internet Computing*, vol. 10, pp. 87–89, Nov 2006.

[30] D. Costa, E. Mingozzi, G. Tanganelli, and C. Vallati, "An alljoyn to coap bridge," in *IEEE 3rd World Forum on Internet of Things (WF-IoT)*, pp. 395–400, Dec 2016.

[31] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe, "Mobile fog: A programming model for large-scale applications on the internet of things," in *Proceedings of the Second ACM SIGCOMM Workshop on Mobile Cloud Computing*, MCC '13, (New York, NY, USA), pp. 15–20, ACM, 2013.

[32] I. Stojmenovic, "Fog computing: A cloud to the ground support for smart things and machine-to-machine networks," in *Australasian Telecommunication Networks and Applications Conference (ATNAC)*, pp. 117–122, Nov 2014.

[33] V. Stantchev, A. Barnawi, S. Ghulam Muhammad, J. Schubert, and G. Tamm, "Smart items, fog and cloud computing as enablers of servitization in healthcare," vol. 185, pp. 121 – 128, 02 2015.

[34] Y. Cao, S. Chen, P. Hou, and D. Brown, "Fast: A fog computing assisted distributed analytics system to monitor fall for stroke mitigation," in *IEEE International Conference on Networking, Architecture and Storage (NAS)*, pp. 2–11, Aug 2015.

[35] T. N. Gia, M. Jiang, A. M. Rahmani, T. Westerlund, P. Liljeberg, and H. Tenhunen, "Fog computing in healthcare internet of things: A case study on ecg feature extraction," in *IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, pp. 356–363, Oct 2015.

[36] M. Aazam and E. N. Huh, "Fog computing and smart gateway based communication for cloud of things," in *International Conference on Future Internet of Things and Cloud*, pp. 464–470, Aug 2014.

[37] J. Zhu, D. S. Chan, M. S. Prabhu, P. Natarajan, H. Hu, and F. Bonomi, "Improving web sites performance using edge servers in fog computing architecture," in *IEEE Seventh International Symposium on Service-Oriented System Engineering*, pp. 320–323, Mar 2013.

[38] "Openfog consortium —." `https://www.openfogconsortium.org/`. (Accessed on 11/09/2017).

[39] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *2012 Proceedings IEEE INFOCOM*, pp. 945–953, Mar 2012.

[40] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: Elastic execution between mobile device and cloud," in *Proceedings of the Sixth Conference on Computer Systems*, EuroSys '11, (New York, NY, USA), pp. 301–314, ACM, 2011.

[41] G. Orsini, D. Bade, and W. Lamersdorf, "Computing at the mobile edge: Designing elastic android applications for computation offloading," in *8th IFIP Wireless and Mobile Networking Conference (WMNC)*, pp. 112–119, Oct 2015.

[42] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker, "Fog computing may help to save energy in cloud computing," *IEEE Journal on Selected Areas in Communications*, vol. 34, pp. 1728–1739, May 2016.

[43] R. Deng, R. Lu, C. Lai, and T. H. Luan, "Towards power consumption-delay tradeoff by workload allocation in cloud-fog computing.," in *ICC*, pp. 3909–3914, IEEE, 2015.

[44] I. Stojmenovic, S. Wen, X. Huang, and H. Luan, "An overview of fog computing and its security issues," *Concurrency and Computation: Practice and Experience*, vol. 28, no. 10, pp. 2991–3005, 2016. cpe.3485.

[45] Y. W. Law, M. Palaniswami, G. Kounga, and A. Lo, "Wake: Key management scheme for wide-area measurement systems in smart grid," *IEEE Communications Magazine*, vol. 51, pp. 34–41, Jan 2013.

[46] C. Marforio and K. K. S. C. Nikolaos Karapanos, Claudio Soriente, "Smartphones as practical and secure location verification tokens for payments," in *21st Annual Network and Distributed System Security Symposium, NDSS 2014, San Diego, California, USA, February 23-26*, 2014.

[47] C. Dsouza, G. J. Ahn, and M. Taguinod, "Policy-driven security management for fog computing: Preliminary framework and a case study," in *Proceedings of the IEEE 15th International Conference on Information Reuse and Integration (IEEE IRI 2014)*, pp. 16–23, Aug 2014.

[48] D. Zeng, L. Gu, S. Guo, Z. Cheng, and S. Yu, "Joint optimization of task scheduling and image placement in fog computing supported software-defined embedded system," *IEEE Transactions on Computers*, vol. 65, pp. 3702–3712, Dec 2016.

[49] M. Aazam and E. N. Huh, "Fog computing micro datacenter based dynamic resource estimation and pricing model for iot," in *IEEE 29th International Conference on Advanced Information Networking and Applications*, pp. 687–694, Mar 2015.

[50] M. Malensek, S. L. Pallickara, and S. Pallickara, "Hermes: Federating fog and cloud domains to support query evaluations in continuous sensing environments," *IEEE Cloud Computing*, vol. 4, pp. 54–62, Mar 2017.

[51] R. Schoenen and H. Yanikomeroglu, "User-in-the-loop: spatial and temporal demand shaping for sustainable wireless networks," *IEEE Communications Magazine*, vol. 52, pp. 196–203, Feb 2014.

[52] D. Sabella, A. Vaillant, P. Kuure, U. Rauschenbach, and F. Giust, "Mobile-edge computing architecture: The role of mec in the internet of things," *IEEE Consumer Electronics Magazine*, vol. 5, pp. 84–91, Oct 2016.

[53] T. Piliouras, *Network design: management and technical perspectives*. Auerbach, 2005.

[54] H. D. Sherali, C. M. Pendyala, and T. S. Rappaport, "Optimal location of transmitters for micro-cellular radio communication system design," *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 662–673, May 1996.

[55] H. R. Anderson and J. P. McGeehan, "Optimizing microcell base station locations using simulated annealing techniques," in *Proceedings of IEEE Vehicular Technology Conference (VTC)*, pp. 858–862 vol.2, Jun 1994.

[56] M. St-Hilaire, S. Chamberland, and S. Pierre, "A tabu search algorithm for the global planning problem of third generation mobile networks," *Computers & Electrical Engineering*, vol. 34, no. 6, pp. 470 – 487, 2008.

[57] M. R. Pasandideh and M. St-Hilaire, "Automatic planning of 3g umts all-ip release 4 networks with realistic traffic," *Comput. Oper. Res.*, vol. 40, pp. 1991–2003, Aug. 2013.

[58] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, HotSDN '12, (New York, NY, USA), pp. 7–12, ACM, 2012.

[59] A. Sallahi and M. St-Hilaire, "Optimal model for the controller placement problem in software defined networks," *IEEE Communications Letters*, vol. 19, pp. 30–33, Jan 2015.

[60] "Ubc radio science lab." `http://rsl.ece.ubc.ca/planning.html`. (Accessed on 11/10/2017).

[61] L. Cen, K. F. Poon, Z. L. Yu, and A. Ouali, "Automatic planning of gpon/ftth networks based on lagrangian heuristic optimization," in *IEEE International Conference on Industrial Engineering and Engineering Management*, pp. 335–339, Dec 2013.

[62] B. Addis, D. Belabed, M. Bouet, and S. Secci, "Virtual network functions placement and routing optimization," in *IEEE 4th International Conference on Cloud Networking (CloudNet)*, pp. 171–177, Oct 2015.

[63] H. Moens and F. D. Turck, "Vnf-p: A model for efficient placement of virtualized network functions," in *10th International Conference on Network and Service Management (CNSM) and Workshop*, pp. 418–423, Nov 2014.

[64] C. Pham, N. H. Tran, S. Ren, W. Saad, and C. S. Hong, "Traffic-aware and energy-efficient vnf placement for service chaining: Joint sampling and matching approach," *IEEE Transactions on Services Computing*, vol. PP, no. 99, pp. 1–1, 2017.

[65] B. Blanco, I. Taboada, J. O. Fajardo, and F. Liberal, "A robust optimization based energy-aware virtual network function placement proposal for small cell 5g networks with mobile edge computing capabilities," *Mobile Information Systems*, vol. 2017, no. 2603410, p. 14, 2017.

[66] M. Cvijetic and I. Djordjevic, *Advanced Optical Communication Systems and Networks*. Artech House applied photonics series, Artech House, 2013.

[67] "Alcatel-lucent sets new world record broadband speed of 10 gbps for transmission of data over traditional copper telephone lines — nokia networks." `https://networks.nokia.com/press/2014/alcatel-lucent-sets-new-world-record-broadband-speed-10-gbps-transmission-dat` (Accessed on 12/20/2017).

[68] I. Cisco Systems, "Design best practices for latency optimization," tech. rep., Cisco Systems, Inc., CA, USA, 2007.

[69] W. Jakob and C. Blume, "Pareto optimization or cascaded weighted sum: A comparison of concepts," vol. 7, pp. 166–185, 03 2014.

[70] G. Anandalingam and T. L. Friesz, "Hierarchical optimization: An introduction," *Annals of Operations Research*, vol. 34, pp. 1–11, Dec 1992.

[71] V. Chankong and Y. Haimes, *Multiobjective decision making: theory and methodology*. North-Holland series in system science and engineering, North Holland, 1983.

[72] G. Mavrotas, "Effective implementation of the ϵ-constraint method in multi-objective mathematical programming problems," *Appl. Math. Comput.*, vol. 213, pp. 455–465, Jul 2009.

[73] B. Panigrahi, V. Pandi, R. Sharma, S. Das, and S. Das, "Multiobjective bacteria foraging algorithm for electrical load dispatch problem," vol. 52, pp. 1334–1342, 02 2011.

[74] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms - a comparative case study," in *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*, PPSN V, (London, UK, UK), pp. 292–304, Springer-Verlag, 1998.

[75] A. Custdio, M. Emmerich, and J. Madeira, "A robust optimization based energy-aware virtual network function placement proposal for small cell 5g networks with mobile edge computing capabilities," *Computational Technology Reviews*, vol. 5, pp. 1–30, Apr 2012.

[76] Ibm, *IBM ILOG CPLEX Optimization Studio CPLEX User's Manual*, 2011.

[77] I. A. Kateeb, K. F. AlOtaib, L. Burton, and M. S. Peluso, "Cooperative algorithm for the global planning problem of umts networks," in *Proceedings of the 2013 American Society for Engineering Education Pacific Southwest Conference*, pp. 452–467, 2013.

[78] "Managed dedicated server hosting — rackspace hosting experts." `https://www.rackspace.com/dedicated-servers`. (Accessed on 08/12/2017).

[79] J. Clausen, "Branch and bound algorithms – principles and examples," 1999.

[80] C. H. Papadimitriou, "On the complexity of integer programming," *J. ACM*, vol. 28, pp. 765–768, Oct. 1981.

[81] I. Das and J. Dennis, "A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems," vol. 14, pp. 63–69, 01 1997.

[82] T. W. ATHAN and P. Y. PAPALAMBROS, "A note on weighted criteria methods for compromise solutions in multi-objective optimization," *Engineering Optimization*, vol. 27, no. 2, pp. 155–176, 1996.

Appendix A

Results for Small Scale Problem

A.1 Instance 1

Table A.1: Result obtained from the solver for small scale problem: Instance 1

	Weighted Sum			Hierarchical			Trade-off		
Problem	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)
1	0.104	67	1	0.098	110	3	0.104	78	2
2	0.177	116	2	0.177	116	3	0.177	116	2
3	0.277	448	2	0.27	474	4	0.277	448	2
4	0.354	554	3	0.348	595	7	0.354	554	6
5	0.387	631	3	0.387	631	8	0.387	631	3
6	0.582	815	3	0.576	857	10	0.582	815	3
7	0.651	1130	16	0.651	1130	24	0.651	1130	31
8	0.664	953	2	0.62	1064	18	0.626	1011	11
9	0.671	1246	15	0.665	1319	20	0.671	1256	56
10	0.792	1224	119	0.786	1292	45	0.792	1224	265
11	0.996	1576	14	0.984	1643	33	0.99	1607	27
12	1.118	1853	10	1.106	1917	36	1.119	1850	44
13	1.06	1825	8	1.054	1914	37	1.06	1825	110
14	1.113	1879	10	1.107	1908	94	1.113	1879	232
15	1.305	2099	11	1.305	2102	46	1.304	2102	106
16	1.265	2269	8	1.254	2395	65	1.265	2298	76
17	1.479	2628	2801	1.479	2628	66	1.479	2630	693
18	1.401	2561	279	1.389	2643	83	1.401	2561	1375
19	1.538	2732	29	1.539	2732	74	1.538	2740	44
20	2.42	4504	83	2.415	4533	186	2.426	4481	130
21	3.13	5356	1548	3.124	5385	1107	3.135	5348	9019

A.2 Instance 2

Table A.2: Result obtained from the solver for small scale problem: Instance 2

	Weighted Sum			Hierarchical			Trade-off		
Problem	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)
1	0.068	11	1	0.068	11	2	0.068	101	2
2	0.225	277	1	0.225	277	3	0.225	277	2
3	0.269	420	2	0.269	420	4	0.269	420	2
4	0.4	435	3	0.4	435	6	0.4	435	2
5	0.444	491	3	0.444	491	8	0.45	469	5
6	0.549	910	4	0.543	990	14	0.549	910	14
7	0.634	937	3	0.629	961	12	0.64	925	5
8	0.584	987	3	0.578	1061	21	0.584	987	46
9	0.702	1247	5	0.695	1279	19	0.702	1248	9
10	0.817	1302	8	0.811	1374	22	0.817	1302	27
11	0.822	1312	14	0.816	1442	28	0.822	1309	164
12	0.916	1460	134	0.909	1511	44	0.916	1460	75
13	1.16	1995	91	1.153	2052	56	1.16	1995	178
14	1.177	2050	5	1.158	2215	67	1.177	2050	48
15	1.199	2212	24	1.192	2253	52	1.205	2189	2435
16	1.315	2044	557	1.309	2087	69	1.315	2044	221
17	1.277	1713	8	1.264	1778	57	1.276	1725	79
18	1.629	3007	207	1.624	3034	69	1.629	3007	118
19	1.55	3268	9	1.526	3340	74	1.543	3294	23
20	2.384	3939	149	2.374	4013	186	2.384	3942	759
21	3.318	5682	5484	3.311	5725	286	3.324	5665	1759

A.3 Instance 3

Table A.3: Result obtained from the solver for small scale problem: Instance 3

	Weighted Sum			Hierarchical			Trade-off		
Problem	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)
1	0.167	255	2	0.167	255	2	0.167	255	1
2	0.178	155	2	0.178	155	4	0.178	155	3
3	0.287	352	3	0.287	352	5	0.287	352	2
4	0.261	241	3	0.255	268	6	0.261	241	3
5	0.44	644	2	0.433	697	8	0.44	644	4
6	0.554	798	4	0.554	798	11	0.554	800	3
7	0.531	613	4	0.518	676	13	0.524	646	7
8	0.736	1306	6	0.736	1306	16	0.736	1306	12
9	0.733	1175	3	0.726	1211	24	0.733	1175	10
10	0.926	1633	4	0.921	1679	23	0.926	1633	10
11	1.016	1660	4	1.003	1738	28	1.016	1660	11
12	0.959	1450	9	0.952	1533	36	0.959	1433	36
13	0.995	1608	3502	1.026	1515	36	0.994	1633	3152
14	1.269	2031	472	1.269	2031	52	1.275	2013	99
15	1.182	1889	25	1.175	1967	68	1.182	1889	492
16	1.385	2715	16	1.385	2715	55	1.385	2715	1471
17	0.905	2128	6	1.279	2188	62	1.284	2155	159
18	0.996	2624	5	0.995	2660	68	0.994	2684	10
19	1.592	2958	49	1.581	3037	75	1.592	2958	411
20	2.354	3559	183	2.35	3572	178	2.354	3559	50
21	3.257	6026	7094	3.252	6046	298	3.25	6046	3075

A.4 Instance 4

Table A.4: Result obtained from the solver for small scale problem: Instance 4

	Weighted Sum			Hierarchical			Trade-off		
Problem	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)
1	0.111	56	1	0.111	64	2	0.111	56	1
2	0.192	153	2	0.184	152	3	0.192	153	2
3	0.232	384	4	0.232	420	5	0.232	384	3
4	0.358	334	3	0.358	334	6	0.358	334	5
5	0.405	648	3	0.393	719	8	0.405	651	3
6	0.62	1152	2	0.62	1152	13	0.62	1152	3
7	0.474	913	4	0.493	855	12	0.474	913	13
8	0.691	1200	3	0.685	1240	18	0.691	1200	8
9	0.697	1267	3	0.691	1321	20	0.703	1245	28
10	0.751	1332	8	0.751	1332	26	0.751	1337	20
11	0.929	1584	63	0.922	1612	51	0.928	1596	89
12	1.004	1900	20	0.985	2011	34	0.997	1943	45
13	1.053	1680	24	1.078	1646	38	1.053	1689	550
14	1.119	1875	106	1.106	1966	57	1.119	1883	312
15	1.154	1882	9	1.142	2014	57	1.179	1782	69
16	1.361	2046	9	1.336	2154	53	1.361	2052	22
17	1.014	2498	6	1.403	2539	61	1.403	2539	35
18	1.41	2607	3547	1.41	2607	81	1.416	2564	249
19	1.787	3111	12	1.762	3300	73	1.781	3140	75
20	2.433	4060	1634	2.428	4117	291	2.439	4023	521
21	3.175	5617	1029	3.164	5667	327	3.174	5630	1263

Appendix B

Results for Large Scale Problem

B.1 Instance 1

Table B.1: Result obtained from the solver for large scale problem: Instance 1

	Weighted Sum			Hierarchical			Trade-off		
Problem	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)
22	0.131	199	2	0.131	199	4	0.131	201	2
23	0.223	104	11	0.223	104	13	0.223	104	4
24	0.241	316	3	0.241	316	13	0.241	316	4
25	0.345	310	140	0.338	336	18	0.345	310	71
26	0.501	694	346	0.502	694	314	0.501	694	369
27	0.495	680	456	0.494	711	3215	0.495	690	851
28	0.548	648	404	0.529	783	1963	0.535	719	755
29	0.69	764	553	0.684	877	2501	0.69	771	2369
30	0.675	871	5258	0.675	880	4758	0.674	880	2781
31	0.726	1135	14804	0.714	1249	1144	0.726	1135	1514
32	0.833	893	30324	0.841	881	7023	0.845	853	6973

B.2 Instance 2

Table B.2: Result obtained from the solver for large scale problem: Instance 2

	Weighted Sum			Hierarchical			Trade-off		
Problem	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)
22	0.09	149	2	0.09	149	4	0.09	149	2
23	0.209	151	3	0.204	164	7	0.209	151	2
24	0.243	160	726	0.237	197	412	0.242	170	10754
25	0.294	479	7314	0.294	479	286	0.294	479	13442
26	0.404	594	63	0.392	650	25	0.398	618	62
27	0.54	545	3886	0.533	622	6190	0.533	622	1051
28	0.605	834	2172	0.6	883	648	0.605	834	1111
29	0.761	901	5401	0.755	943	2204	0.761	908	11549
30	0.71	1052	1216	0.71	1052	205	0.71	1052	679
31	0.841	1092	694	0.834	1178	7555	0.841	1092	6368
32	0.993	1350	8233	0.943	1425	11234	0.942	1425	23316

B.3 Instance 3

Table B.3: Result obtained from the solver for large scale problem: Instance 3

	Weighted Sum			Hierarchical			Trade-off		
Problem	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)
22	0.16	116	2	0.16	116	4	0.16	116	2
23	0.138	208	3	0.138	217	11	0.137	223	3
24	0.261	206	301	0.262	206	59	0.261	206	34
25	0.372	467	131	0.372	467	227	0.377	422	523
26	0.432	687	4450	0.432	687	379	0.432	687	3255
27	0.467	540	183	0.461	577	51	0.467	540	150
28	0.592	648	272	0.585	718	796	0.592	648	351
29	0.692	880	259	0.692	902	4250	0.692	880	504
30	0.637	738	93	0.624	923	353	0.636	785	275
31	0.782	552	1378	0.763	695	2297	0.776	587	9774
32	0.84	1302	2463	0.811	1405	1340	0.833	1326	1492

B.4 Instance 4

Table B.4: Result obtained from the solver for large scale problem: Instance 4

	Weighted Sum			Hierarchical			Trade-off		
Problem	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)	Delay (s)	Traffic (Mbps)	Solution Time (s)
22	0.094	45	2	0.094	45	6	0.094	48	2
23	0.22	165	57	0.22	170	19	0.22	185	13
24	0.298	222	2	0.298	222	12	0.298	257	6
25	0.355	166	16494	0.354	201	2694	0.355	166	27744
26	0.458	455	59	0.452	490	560	0.458	455	57
27	0.467	540	222	0.486	596	559	0.486	596	1152
28	0.61	721	5592	0.603	755	3602	0.61	723	1565
29	0.622	816	12234	0.616	908	8575	0.622	820	11233
30	0.642	928	1434	0.63	1031	6029	0.636	978	2723
31	0.784	998	7373	0.764	1111	2440	0.79	987	6664
32	0.837	1236	7195	0.831	1305	16439	0.837	1236	1698

Appendix C

Results for Single and Multiobjective

C.1 Instance 1

Table C.1: Result for single objective and multiobjective: Instance 1

	Single Objective		Weighted Sum		Hierarchical		Trade-off	
Problem	Delay (s)	Traffic (Mbps)	Delay Gap (%)	Traffic Gap (%)	Delay Gap (%)	Traffic Gap (%)	Delay Gap (%)	Traffic Gap (%)
1	0.098	67	6.3	0	0	65	6.2	16
2	0.177	111	0.29	4.9	0.34	4.9	0.29	4.9
3	0.27	443	2.5	1.1	0.0072	6.9	2.5	1.1
4	0.347	542	1.9	2.2	0.097	9.7	1.9	2.2
5	0.386	564	0.08	12	0.08	12	0.08	12
6	0.576	806	1.1	1.2	0.00034	6.3	1.1	1.2
7	0.65	1073	0.14	5.2	0.14	5.2	0.14	5.2
8	0.62	953	7.1	0	0.07	12	1	6
9	0.665	1191	0.96	4.7	0.049	11	0.92	5.5
10	0.785	1172	0.81	4.5	0.082	10	0.81	4.5
11	0.983	1523	1.4	3.5	0.097	7.9	0.74	5.6
12	1.106	1845	1.1	0.44	0.0091	3.9	1.2	0.29
13	1.054	1765	0.65	3.4	0.025	8.5	0.65	3.4
14	1.1	1826	1.1	2.9	0.61	4.5	1.1	2.9
15	1.298	2038	0.51	3	0.5	3.1	0.46	3.1
16	1.253	2165	1	4.8	0.13	11	0.99	6.1
17	1.472	2482	0.47	5.9	0.49	5.9	0.47	6
18	1.388	2518	0.96	1.7	0.08	5	0.96	1.7
19	1.531	2674	0.42	2.2	0.48	2.2	0.42	2.5
20	2.407	4383	0.54	2.7	0.35	3.4	0.81	2.2
21	3.11	5263	0.65	1.8	0.47	2.3	0.82	1.6
22	0.131	199	0.098	0	0.098	0	0.057	1.3
23	0.223	93	0.051	12	0.066	12	0.051	12
24	0.241	316	0.038	0	0.038	0	0.038	0
25	0.338	300	2	3.4	0.0091	12	2	3.4
26	0.501	672	0.068	3.3	0.088	3.3	0.068	3.3
27	0.494	647	0.2	5.2	0.089	10	0.2	6.7
28	0.529	622	3.6	4.1	0.099	26	1.3	16
29	0.684	705	0.84	8.4	0.083	25	0.83	9.4
30	0.673	807	0.26	8	0.19	9.1	0.17	9.1
31	0.713	1066	1.8	6.4	0.096	17	1.8	6.4
32	0.833	796	0.07	12	1	11	1.5	7.1

C.2 Instance 2

Table C.2: Result for single objective and multiobjective: Instance 2

	Single Objective		Weighted Sum		Hierarchical		Trade-off	
Problem	Delay (s)	Traffic (Mbps)	Delay Gap (%)	Traffic Gap (%)	Delay Gap (%)	Traffic Gap (%)	Delay Gap (%)	Traffic Gap (%)
1	0.068	11	0	0	0	0	0	0
2	0.224	250	0.1	11	0.1	11	0.1	11
3	0.269	412	0.038	2	0.038	2	0.038	2
4	0.4	406	0.048	7.1	0.048	7.1	0.048	7.1
5	0.444	447	0.046	10	0.046	10	1.4	5.1
6	0.543	894	1.2	1.8	0.022	11	1.2	1.8
7	0.628	888	0.98	5.6	0.088	8.3	1.9	4.2
8	0.578	932	1.1	0	0.04	14	1.1	5.8
9	0.695	1225	0.97	1.8	0.043	4.4	0.95	1.9
10	0.811	1223	0.77	6.5	0	12	0.77	6.5
11	0.816	1253	0.79	4.7	0	15	0.79	4.4
12	0.909	1414	0.75	3.2	0.049	6.9	0.75	3.2
13	1.153	1921	0.61	3.9	0.054	6.8	0.61	3.9
14	1.158	2000	1.6	2.5	0.02	11	1.6	2.5
15	1.186	2132	1	3.8	0.49	5.7	1.6	2.7
16	1.308	1968	0.5	3.9	0.046	6	0.5	3.9
17	1.258	1655	1.5	3.5	0.48	7.4	1.5	4.2
18	1.616	2972	0.79	1.2	0.49	2.1	0.79	1.2
19	1.519	3232	2.1	1.1	0.45	3.4	1.6	1.9
20	2.365	3880	0.82	1.5	0.41	3.4	0.82	1.6
21	3.305	5599	0.4	1.5	0.2	2.3	0.59	1.2
22	0.09	149	0	0	0.0068	0	0	0
23	0.204	151	2.8	0	0.0014	8.5	2.8	0
24	0.237	160	2.5	0	0	0	2	0
25	0.294	427	0.13	12	0.13	12	0.13	12
26	0.391	580	3.3	2.4	0.096	12	1.7	6.6
27	0.533	481	1.5	13	0.13	29	0.11	29
28	0.599	788	1	5.8	0.15	12	1	5.8
29	0.754	866	0.96	4	0.077	8.9	0.87	4.9
30	0.709	957	0.12	9.9	0.14	9.9	0.12	9.9
31	0.834	985	0.86	11	0.039	20	0.86	11
32	0.941	1350	5.5	0.000027	0.16	5.6	0.13	5.6

C.3 Instance 3

Table C.3: Result for single objective and multiobjective: Instance 3

	Single Objective		Weighted Sum		Hierarchical		Trade-off	
Problem	Delay (s)	Traffic (Mbps)	Delay Gap (%)	Traffic Gap (%)	Delay Gap (%)	Traffic Gap (%)	Delay Gap (%)	Traffic Gap (%)
1	0.167	255	0.075	0.0000079	0.075	0	0.039	0.0000079
2	0.177	145	0.39	7.1	0.43	7.1	0.39	7.1
3	0.287	341	0.074	3.2	0.074	3.2	0.074	3.2
4	0.255	226	2.4	6.8	0.026	18	2.4	6.8
5	0.433	630	1.6	2.2	0.057	11	1.6	2.2
6	0.554	766	0.077	4.3	0.095	4.3	0.062	4.5
7	0.518	586	2.4	4.6	0.018	15	1.2	10
8	0.736	1267	0.067	3	0.089	3	0.067	3
9	0.726	1154	0.9	1.8	0.028	4.9	0.9	1.8
10	0.92	1599	0.68	2.1	0.085	5	0.68	2.1
11	1.002	1637	1.3	1.4	0.038	6.2	1.3	1.4
12	0.952	1393	0.77	4.1	0.028	10	0.77	2.9
13	0.987	1515	0.79	6.1	3.9	0	0.66	7.8
14	1.268	1957	0.053	3.8	0.073	3.8	0.51	2.9
15	1.169	1824	1.1	3.6	0.5	7.8	1.1	3.6
16	1.379	2580	0.44	5.2	0.49	5.2	0.44	5.2
17	0.899	2076	0.69	2.5	42	5.4	43	3.8
18	0.993	2592	0.32	1.2	0.19	2.6	0.12	3.5
19	1.573	2901	1.2	1.9	0.49	4.7	1.2	1.9
20	2.342	3501	0.54	1.6	0.34	2	0.54	1.6
21	3.237	5940	0.6	1.4	0.46	1.8	0.4	1.8
22	0.16	116	0	0	0	0	0	0
23	0.137	208	0.52	0	0.5	4.5	0.33	7.4
24	0.261	198	0.07	3.7	0.097	3.7	0.07	3.7
25	0.372	384	0.1	22	0.1	22	1.6	9.8
26	0.431	671	0.11	2.2	0.14	2.2	0.11	2.2
27	0.461	538	1.4	0.44	0.0087	7.4	1.4	0.44
28	0.584	617	1.2	5.2	0.086	16	1.2	5.2
29	0.691	849	0.15	3.7	0.096	6.3	0.15	3.7
30	0.624	733	2.1	0.74	0.085	26	2	7.1
31	0.761	512	2.6	7.9	0.14	36	1.9	15
32	0.807	1272	4.1	2.4	0.47	10	3.2	4.3

C.4 Instance 4

Table C.4: Result for single objective and multiobjective: Instance 4

	Single Objective		Weighted Sum		Hierarchical		Trade-off	
Problem	Delay (s)	Traffic (Mbps)	Delay Gap (%)	Traffic Gap (%)	Delay Gap (%)	Traffic Gap (%)	Delay Gap (%)	Traffic Gap (%)
1	0.111	46	0.39	23	0.13	40	0.39	23
2	0.185	146	3.4	4.5	0	3.6	3.4	4.5
3	0.232	365	0.16	5.2	0.1	15	0.16	5.2
4	0.358	304	0.16	10	0.18	10	0.16	10
5	0.393	638	3.1	1.5	0	13	3.1	2
6	0.619	1131	0.053	1.9	0.056	1.9	0.053	1.9
7	0.467	855	1.4	6.8	5.5	0	1.4	6.8
8	0.685	1126	0.97	6.6	0.097	10	0.97	6.6
9	0.691	1217	0.94	4.1	0.031	8.6	1.8	2.3
10	0.751	1294	0.032	2.9	0.036	2.9	0.016	3.3
11	0.922	1523	0.75	4	0.058	5.8	0.7	4.8
12	0.984	1824	2	4.1	0.088	10	1.3	6.5
13	1.04	1646	1.2	2	3.7	0	1.2	2.6
14	1.106	1830	1.2	2.5	0.022	7.5	1.2	2.9
15	1.142	1750	1.1	7.6	0.013	15	3.3	1.8
16	1.336	2022	1.9	1.2	0.017	6.5	1.9	1.5
17	1.004	2485	1	0.53	40	2.2	40	2.2
18	1.409	2505	0.06	4.1	0.066	4.1	0.5	2.4
19	1.762	3041	1.4	2.3	0.019	8.5	1.1	3.3
20	2.419	3912	0.58	3.8	0.36	5.3	0.82	2.8
21	3.149	5567	0.82	0.91	0.47	1.8	0.79	1.1
22	0.094	45	0.025	0	0.05	0	0.016	8.1
23	0.22	165	0.22	0	0.18	3.1	0.091	12
24	0.298	222	0.057	0	0.079	0	0	16
25	0.354	155	0.41	7.1	0.17	30	0.41	7.1
26	0.452	406	1.5	12	0.14	21	1.5	12
27	0.485	545	3.8	8.2	0.097	9.4	0.097	9.4
28	0.603	690	1.1	4.6	0.017	9.5	1.1	4.9
29	0.615	719	1.1	14	0.14	26	1.1	14
30	0.629	847	2.1	9.5	0.2	22	1.1	15
31	0.763	933	2.7	6.9	0.13	19	3.5	5.7
32	0.829	1160	0.88	6.6	0.15	13	0.88	6.6