Models and Algorithms for Efficient Data Processing in Fog Computing Supported Disaster Areas

Yu Wang

A DISSERTATION
SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
IN COMPUTER SCIENCE AND ENGINEERING

Data Networking Laboratory
Graduate Department of Computer and Information Systems
University of Aizu, Japan

September 2019

The thesis titled

Models and Algorithms for Efficient Data Processing in Fog Computing Supported Disaster Areas

by

Yu Wang

is reviewed and approved by:

Chief referee *Associate Professor* Junbo Wang	Junbo Wang	王
Professor Tuan Anh Pham	Anh T. PHAM	[illegible]
Senior Associate Professor Cong-Thang Truong	Thang	Truong
Associate Professor Peng Li	Peng Li	[illegible]

University of Aizu

September 2019

Dedicated to

my family

Models and Algorithms for Efficient Data Processing in Fog Computing Supported Disaster Areas

Yu Wang

Submitted for the Degree of Doctor of Philosophy
September 2019

Abstract

Mobile devices are commonplace today, and smartphones have enabled big data analytics. The spreading mobile phones allows us to easily collect human activities and generate a large quantity of mobile data, such as health data, commuting route or restaurant recommendations. Big data analysis is especially important in disaster scenarios. During disaster scenarios, big data analysis can be used to detect obstructions and dispatch rescue in a timely manner.

Communication infrastructure can be destroyed by the disaster, especially earthquakes which occur in or near major cities that cause a lot of damage. It is critical to immediately recover the communication system after a disaster occurs. Movable base stations (MBSs) can be positioned to reestablish an emergency communication network after a disaster.

The above emergency communication network brings new challenges for big data analytics because big data is often analyzed in a cloud center to save processing time with high-performance PCs and servers in a general case. The transmission delay will be extremely large when collecting data from mobile phones to cloud center via an MBS-based network. We need to process small amounts of data before they get to the server in order to handle the size of data that is collected.

In this thesis, we first proposed a set of delay models for spatial data processing networks, which specifically targeted disaster-stricken areas. We also have implemented a genetic algorithm (GA) solution which showed to have a reduced maximum end-to-end (E2E) delay over various network sizes. We researched some realistic constraints, and tested some of the conventional methods with MATLAB simulator, and modeled their worst-case delays. The results showed that none of the conventional cases matched the capabilities of the GA for increased computation or increased transmission rates.

Second, because the GA solution had a significant computation time, we proposed the disaster area adaptive delay minimization algorithm (DAADM). The goal of this new algorithm was to run in real-time. We also present a detailed mathematical model to represent data processing and transmission in an ECN fog network and an NP-hard proof for the problem of optimization the overall delay. We evaluated the systems across various transmissions speeds, processing speeds, and network sizes. Furthermore, we tested calculation time, accuracy, and percent error of the systems. Through evaluation, we found that the proposed DAADM algorithm can be implemented in a real-time system, which had a major advantage over the GA.

All emergency communication networks have performance drawbacks. In order to improve the network performance even further, we proposed an algorithm that determines the best way for the MBSs to be connected. We assume that each of the fog node has traffic in addition to the data that is used for the algorithm that needs to be processed, and thus the system needs to be account for a queueing delay, both on the processor and transmitter. The delay models were run in a genetic algorithm which solved for a delay-optimized solution for the network. The results shows that the GA outperforms the greedy algorithm.

Finally, the proposed architectures and algorithms were simulated with MATLAB, which realized effective data processing and transmission in disaster scenarios. Results show that the proposed system was able to achieve a higher performance with minimal delay for the overall system and can be run in real-time. The resulting system is able to address the problems of network topology as well as optimize processing delay and transmission delay. The resulting system is more suitable for deployment than previous existing systems.

Declaration

The work in this thesis is based on research carried out at the Data Networking Laboratory at the University of Aizu, Japan. No part of this thesis has been submitted elsewhere for any other degree or qualification and it is all my own work unless referenced to the contrary in the text.

Acknowledgements

I would like to express my sincere gratitude to thank my advisor, Associate Professor Junbo Wang, for his guidance over the past three years. He has guided me on not only just writing and research, but also on professional expectations of doctors.

I would also like to thank Professor Tuan Anh Pham, Senior Associate Professor Cong-Thang Truong, Associate Professor Peng Li of the University of Aizu for taking the time to review my thesis and give out comments.

I would like to thank my husband Michael Conrad Meyer for all his love, support and encouragement. He has a important meaningful for my doctoral life and experience.

I would like to thank all my friends, at home and in Japan. Current and previous members of the Data Networking Laboratory at the University of Aizu have helped me with my life in Japan, and have made the whole experience more enjoyable.

Lastly, I would like to thank my family for all their love and encouragement. For my parents who raised me with a love of science and supported me in all my pursuits.

Contents

Abstract **iv**

Declaration **viii**

Acknowledgments **ix**

1 Introduction **1**

1.1 Background 1

1.1.1 Computing 1

1.1.2 Cloud Computing 4

1.1.3 Big Data Processing 6

1.1.4 Internet of Things 8

1.1.5 Fog Computing 9

1.1.6 Emergency Networks 10

1.1.6.1 Disaster Area 10

1.1.6.2 Wireless Disaster-area Networks 11

1.2 Thesis Objectives 12

1.3 Thesis contributions 12

1.4 Thesis Outline 13

2 Related Works **14**

2.1 Edge Computing Networks 14

2.2 Fog Network Optimization 15

2.3 Chapter Summary 18

3 Model and Genetic Algorithm for Data Processing Ratio 19

3.1 Introduction 19

3.2 System Model and Problem Formulation 20

3.2.1 System Model 20

3.2.2 Problem Formulation 21

3.2.2.1 Delay for Processing Data in Node *k* 21

3.2.2.2 Delay for Receiving Data in Node *k* 22

3.2.2.3 Overall Delay and Problem Formulation 23

3.3 Proposed Solution (GA) 23

3.4 Simulations and Performance Evaluation 26

3.4.1 Evaluation Across Various Network Sizes 26

3.4.2 Evaluation Across with Varied Bandwidth Constraints 26

3.5 Chapter Summary 29

4 Real-time Solution for the Data Processing Ratio 31

4.1 Introduction 31

4.2 System Model and Problem Definition 33

4.2.1 System Model 33

4.2.2 Problem Formulation 35

4.2.3 NP-hardness of DMP-WNDA 37

4.2.4 Proposed Algorithm (DAADM) 38

4.3 Simulations and Performance Evaluation 41

4.3.1 Evaluation Methodology 41

4.3.2 Evaluation Across with Varied Bandwidth Constraints 42

4.3.3 Time Complexity Evaluation 44

4.4 Chapter Summary 48

5 Base Station Allocation for Users with Overlapping Coverage 49

5.1 Introduction 49

5.2 System Model and Problem Formulation 50

5.2.1 System Model 50

5.2.2 Processing Delay 51

5.2.3 Transmission Delay 53
5.2.4 Delay Model 54
5.3 Proposed Solution 55
5.4 Simulations and Performance Evaluation 56
5.4.1 Evaluation Across Processing Parameters 56
5.4.2 Evaluation Across various Channel Capacities 59
5.4.3 Cloud System Evaluation 61
5.5 Chapter Summary 61

6 Conclusion and Discussion 63
6.1 Contributions 63
6.2 Results Summary 64
6.3 Discussion 65

List of Figures

1.1 A Chinese Abacus 2

1.2 Trend of Processing Elements on a Chip [1]. 4

3.1 Different layers of a fog computing network. 20

3.2 System Model 21

3.3 Genetic algorithm flow chart 24

3.4 Sample ideal $\mathbb{X}$ values for a small network 25

3.5 Delay evaluation on a 11 node network. 26

3.6 Delay evaluation on a 35 node network. 27

3.7 Delay evaluation on a 154 node network. 27

3.8 Delay values of a 154 node network with varying μ_k values. 28

3.9 Delay values of a 154 node network with varying $\mathcal{R}$ values. 29

3.10 Delay values of a 154 node network with varying ρ values. 30

4.1 System Model 33

4.2 Data Structure Change from Spatial Data Processing on a Tweet 34

4.3 Node Model 36

4.4 Delay results while varying μ_k values on a large network 43

4.5 Delay results while varying μ_k values on a small network 43

4.6 Delay results while varying $\mathcal{R}_{ik}$ values on a large network 44

4.7 Delay results while varying $\mathcal{R}_{ik}$ values on a small network 45

4.8 Time consumed by calculating, Accuracy, and Percent error of each allocation schemes of a large network 46

4.9 Time consumed by calculating, Accuracy, and Percent error of each allocation schemes of a medium network 47

4.10 Time consumed by calculating, Accuracy, and Percent error of each allocation schemes of a small network 47

5.1 Structure of the Network in an Overlapped Area 50
5.2 Average Delay Results for Variations in Node Processing Speed 56
5.3 Worst-Case Delay Results for Variations in Node Processing Speed 57
5.4 Average Delay Results for Variations in the Data Compression Ratio . . . 58
5.5 Worst-case Delay Results for Variations in the Data Compression Ratio . 59
5.6 Average Delay Results for Variations in the Channel Capacity 60
5.7 Worst-case Delay Results for Variations in the Channel Capacity 60
5.8 Cloud System Average Delay Results for Variations in the Channel Capacity 61
5.9 Cloud System Worst-case Delay Results for Variations in the Channel Capacity 62

List of Tables

3.1 System Configuration. 25

4.1 Variable Definitions 35

5.1 System Variables 52

List of Abbreviation

MBSs:	Moveable Base Stations
GA:	Genetic Algorithm
E2E:	End to End
DAADM:	Disaster Area Adaptive Delay Minimization
ECN:	Emergency Communication Network
RFID:	Radio-frequency Identification Readers
IoT:	Internet of Things
AWS:	Amazon Web Services
WLANs:	Wireless Local Area Networks
WPANs	Wireless Personal Area Networks
MDRU:	Movable and Deployable Resource Unit
WNDA:	Wireless Network Disaster Area
SDP-WNDA:	Spatial Data Processing in Wireless-networked Disaster Areas
MCPSs:	Medical Cyber-physical Systems
MINLP:	Mixed Integer Non-liner Problem
LP	Linear Programming
MCC:	Mobile Cloud Computing
QoE	Quality of Experience
VANETs	Vehicular ad Hoc Networks
DMP:	Delay Minimization Problem
DMP-WNDA:	Delay Minimization Problem in Wireless Networked Disaster Area
DAADM:	Disaster Area Adaptive Delay Minimization
MECs:	Mobile Edge-clouds
MDP:	Markov Decision Process

Chapter 1

Introduction

During the recent years, technology is playing an increasingly important role in our lives. It exists everywhere in our daily life. Technology not only belongs to researchers, engineers or developers, but is also used by everyone else for their life activities and work. Technology provides a great advantage for our quality of life, and users enjoys using the technology. With the rapid growth in the use of technology, the requirement of its quality increased. Users usually determine quality based on the performance and durability. Many things today use wireless networks, thus a reliable communication system is becoming more necessary for daily life. This field is likely to impact other technological fields due to the increase in IoT devices.

1.1 Background

1.1.1 Computing

Computers have had many forms over the years. Determining when computing started is heavily dependent on what you define as computing. To help give an overview of the history of computing, I will cover some of the first computers in each category.

Computing in its simplest definition is any device that calculates. This means that the first calculators were computers, and the first known device used for calculating is the abacus [2]. An example of an abacus can be seen in Fig. 1.1, but they can take many different forms based on where they are used. The first known use of an abacus was around 2500BC in Sumeria. The devices are still used in some countries to this day. These devices had beads which could represent numbers. The Chinese abacus, or suanpan, in Fig. 1.1 has each row represent one digit of a number.

Figure 1.1: A Chinese Abacus

The next major step in computing was completed by Charles Babbage. He created several calculating machines in the 1800s, but his crowning achievement was the Analytical Machine [3]. It was a mechanical calculating device which could handle arithmetic, loops, and conditional branching. The machine stored numbers with 40 decimal digits of accuracy, and had enough 'memory' to store one thousand numbers. Different values could be input into the system via punch cards.

The next major development in the computing world was the conversion to electrical signals. While it is not the first electronic computer, the first Turing-complete non-mechanical computer was ENIAC [4]. ENIAC stood for Electronic Numerical Integrator and Computer. Its construction was completed in 1945, and could be programmed using patch cables and switches. Because of the electronic nature of the calculations, it could perform the same process approximately 1000 times faster than its mechanical-based competitors.

One of the biggest things holding back the ENIAC were its vacuum tubes. The answer to this was transistors [5]. In their initial form, they were not a significant space-saver because the vacuum tubes were capable of analog computation, until the invention of integrated circuits [6]. This technology allowed for the semiconductors to be arranged in a way that would create several prearranged transistors on a single chip. His circuit at the time was germanium-based, future ICs would be made of silicon, which allowed for P-N junction Isolation. Suddenly, circuits that would take up whole buildings before could be built in things the size of a dresser. Additionally, because the space reduction, the wasted heat could be reduced, and the speed of the computer could be increased. In one technological advancement, computers became smaller, faster, lower power, and cheaper to make.

Modern computers follow the Von Neumann architecture [7]. This means that there are 5 separate categories of components: a processing unit, a control unit, memory, storage, and I/O. The processing unit is typically an Arithmetic Logic Unit (ALU), and is controlled by the control unit. The control unit interprets the instruction and controls the ALU accordingly. Memory is where the instructions and data are stored, while the storage is where whole programs and other machine readable data is kept. I/O stands for Input and Output, and accounts for any component that helps the computer interact with the outside. This includes keyboards and displays that help a human interact with the computer as well as any internet connection that the computer may have.

One heavily referred to law in the world of computing is Moore's law [8]. This famous law states that the number of transistors in a dense circuit doubles every two years. This trend can be seen in fig. 1.2 This has for the most part held true for the last 4 decades, but is starting to show signs of failing. The way that this is usually achieved is by reducing the size of a transistor, which has several effects on the chip that it makes. In terms of performance beyond just the clock speed improvements, the smaller transistors mean that more cores and features can be added to the chip, thus Moore's law can be extended to include computer performance doubling every two years.

As stated earlier, Moore's law is starting to level out. This is due to the fact that fabricating smaller transistors is becoming more difficult simply because they have gotten so small that quantum physics is starting to affect them and cause leaks.Lately, increasing the core count has gotten popular, but the way that these cores communicate with each-other in a bus is inefficient, so an on-chip network was proposed for their communications [9]. This will improve the performance the chip somewhat, but architectural improvements will only go so far, so alternative communication mediums such as Photonics [10] and Quantum Entanglement [11] are being investigated. The problem with these technologies

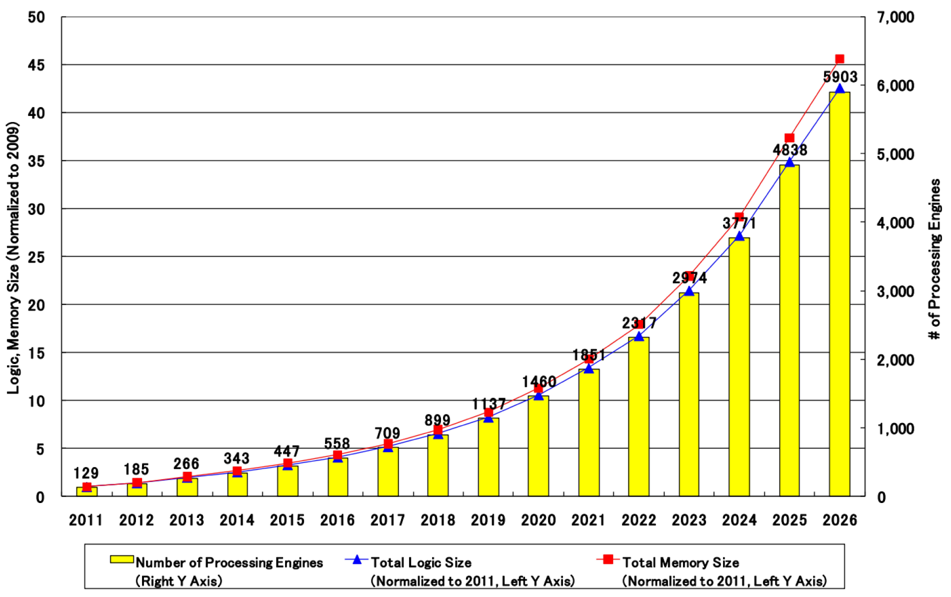

Figure 1.2: Trend of Processing Elements on a Chip [1].

is that because their basic switching element is no longer a transistor, the entire structure of the IC has to change.

1.1.2 Cloud Computing

One solution to improve the computing power of consumers was a technique called Cloud Computing [12]. Cloud computing is separated into different categories based on what type of service, but what they have in common is that the computation is performed at a remote data center and the results are forwarded to the user's device. The NIST [13] has 5 basic characteristics which define cloud computing services: on-demand self-service; broad network access; resource pooling; rapid elasticity; and measured service. On-demand self-service means that a user can access the cloud without interacting with another human. Broad network access means that it is not device limited, and can be accessed over a standard network connection such as a mobile phone network or home internet. Resource pooling is defined by the utilization of the same resources amongst several users. Rapid elasticity means that the resources can be reallocated to another user at any time. The last requirement can be handled in several ways. Some services measure the amount of data stored in Gigabytes, some measure the computing time that is used, while others measure your bandwidth that you utilize. In order to have a true cloud computing system all of these characteristics must be met.

The first category of cloud computing is Platform as a Service (Paas), and covers all cases where according to NIST [13], ”The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment”. The second common form of cloud computing is Software as a Service (SaaS), which is defined as ”The capability provided to the consumer is to use the provider’s applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings” [13]. The third common type of cloud computing is Infrastructure as a Service (IaaS), and is defined as ”where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components” [13]. There are other less common forms of cloud computing, and the list would be too long if we defined them all.

Cloud computing provided a way to use communication to handle computation for users through the internet. Users can access servers, storage, databases and application services over the internet. Amazon Web Services (AWS) [14] is a cloud service platform, it owns all the hardware and software services which are related to the service applications. Cloud services of AWS provide computing power, database storage, content delivery, and other functionality to help business scale and develop. The coverage area of AWS crosses 61 available zones within 20 geographic regions around the world. They also have announced plan for 12 more available zones and 4 more regions.

Cloud computing systems have already greatly expanded the capabilities of smartphones. In terms of storage-based computing, it allows users to keep several photos and videos on the web, and save important space on the phone, all while making it simultaneously available to the users device. In terms of processing capabilities, a user can upload a photo to social media, and the cloud can scan the faces in the photo and detect who is in it, but the phone itself does not do any of the facial recognition because it would take a long time. Engineers have also utilized cloud resources to execute long simulations, which would take several days on their workstations but minutes or hours on a cloud service.

Cloud computing is not perfect. In areas where the internet service is poor, the delay for results can be quite large. Additionally, if the user's internet service is interrupted for a period of time, then the cloud is inaccessible for that same period. This means that your performance and experience is entirely dependent on another service working properly. Additionally, user customizability of a service tends to be low because the business model needs to fit everyone, which makes it difficult to get a perfect fit. The biggest concern with cloud services recently has been security. If a user stores personal items on the cloud, and they get hacked and leaked, it can be very embarrassing as evidenced by several famous leaks making the news. It is however much worse if an IP algorithm is leaked and copied by a competitor, who can then release a competing product sooner.

1.1.3 Big Data Processing

With the rapid growth in use of mobile devices, cameras, radio-frequency identification (RFID) readers and wireless sensor networks, datasets have become easier to be gathered. A report from IDC predict that the global data quantities will grow from 4.4 zettabytes to 44 zettabytes between 2013 and 2020 [15].

Big data is a massive dataset which is large in scale and too complex to use normal processing application softwares to handle them. Big data is usually related with a few key words: massive, variety, speed. The challenge of big data includes: speed, storage, analysis, transfer, visualization, querying, security, etc. Each of these big data challenges require us to come up with corresponding solutions.

Because of the cloud, it is now possible to provide a way to process more data at one time. The big data resources can be sufficiently utilized. It promoted big data computing abilities and processing capabilities.

There have been many types of big data repositories. In [16], the importance of big data technology on the modern world and existing projects were introduced.Even though the large size of the datasets is very difficult to process, big data processing is able to offer many market advantages to several organizations. The paper also discusses several tools which are used for dealing with big data. Additionally, some good big data practices were discussed.

There are several approaches for spatial big data analytics [17], including spatial prediction for understanding the situation in the areas that are not covered by the network: spatial outlier detection to find abnormal situations, spatial co-location feature detection and spatial clustering. Spatial big data analysis is especially important in disaster scenarios. For example, kwan et al. [18] explored handling emergency response services by detecting obstructions caused by a major disaster, such as a hurricane, by using LiDAR data in a big data algorithm. This makes being able to properly service big data processing in a disaster is more important than at a regular time.

In [19], the relationship between big data and cloud computing, big data storage systems, and Hadoop are introduced. The quantity of data is still increasing at a exponential rate because cloud computing is be able to perform massive scale and complex computing. This is means that cloud computing and big data are conjoined. It brings a significant benefit for machine maintenance and space occupation. The massive scale of data that is gathered through the cloud has already been proved to be enough for many applications. Determining the success of data processing and data analysis is heavily depended on the processing time that they require. This is true of most big data processing because with enough time massive amounts of data can be processed, but usually that data may no longer be relevant by the time the data has finished processing.

In [20], the security problems for cloud computing, big data, map reduction and the Hadoop environment has been introduced. They mainly focus on the security problem between cloud computing and big data. Cloud computing security includes: computer security, network security, information security and data privacy. This paper covers many of the security concerns and possible solutions for those concerns.

1.1.4 Internet of Things

The Internet of Things (IoT) is a term used for devices that extend internet range from the traditional computing model into everyday objects. IoT devices are often have their data processed in the cloud because the scale of the data can be very large, and the IoT devices typically have very weak processors. IoT devices are embedded with electronics, internet connectivity and others hardware like sensors. Those devices are connected and interact with others through the internet. They can be remotely monitored and controlled. The contribution of IoT consists of: wireless sensor networks, control systems, smart devices, smart homes and smart buildings.

The IoT applications have been widely used in our daily life. The smart home concept including lighting control, heating control, and air conditioning control. The NEST learning thermostat [21] can automatically learn and control your home temperature when you enter the home. It can adjust the temperature after few times of running the self-learning model. This smart object can also help you save energy. This works by turning off or reducing the strength of the air-conditioning while you are at work, and putting it back on a few hours before you return, so that you can arrive to an appropriate-temperature home, but it does not need to be running at full power all the time.

The Instant Pot [22] is an IoT pressure cooker. There are several smart cooking devices on the market but this one in particular is packed with many features. It is capable of cooking rice and stews like most pressure cookers, but it has a few tricks up its sleeve. One of its first major features is that through your phone, you are able to look up a recipe in an online database, put in the appropriate ingredients, and hit start. The smart device will take care of the rest of the cooking. It can cook meat at a specific pressure and temperature for one hour and then shift to another temperature for another period of time. This takes a lot of skill out of cooking, and brings restaurant quality food to novice level chefs. The next major feature is that a user can put all of the ingredients in the pot before they leave for work, and then start cooking it through a command that they send from their phone at work, and can arrive to a meal that was appropriately timed to be consumed at their desired dinner time, even if it requires several hours of cooking time, which the user does not have between arriving at home and dinner time.

The Ring [23] is a home security IoT doorbell. Home security is one field where IoT is blooming because of the natural connection because it allows you to see your home in real time, and alert authorities even if you are not at home. The Ring can also interface with other IoT cameras to set up a joint video feed. One of the main features is that if someone such as a delivery man rings your doorbell, you can talk to them via the cellphone and see them. This allows you to instruct the delivery man an appropriate place to leave the package to avoid having it be stolen, which is a common problems in America and Europe with the recent popularity in the Amazon shopping service. Additionally, it can record video of a few seconds every time that the bell rings. If a suspicious person is ringing the bell at a certain time of day, he is likely to be a thief who is casing the joint by checking what times the owner is away from home. But with this video, and a camera in the doorbell, most thieves are not only deterred, they can also be caught in the future as you will have photo evidence of a likely suspect.

IoT smart devices are making life more convenient, but are also placing a large burden on our communication infrastructure. If each device has its own learning algorithm that is constantly running, and each person has a handful of devices, this equates to a large demand on computing power.

1.1.5 Fog Computing

Because of internet of things (IoT), there was more demand for cloud computing. In order to reduce the delay of a cloud computing systems, fog computing [24] was proposed as a possible solution, which offloaded some computation tasks from the cloud to edge devices, e.g., routers, which are closer to the users. This new computing system was a distributed cloud service, which still had a main cloud destination, but some key nodes had some processing power of their own. The proposed idea greatly reduced the strain at the center of the node, and increased response time to users near the the cell edge. Medium-load tasks are typically assigned to the fog nodes, because their performance is more based on the latency of the communication, so being closer to the user is more beneficial than having a higher performance processor. Larger tasks, such as heavy simulations, are more dependent on the processing power, and less on the network speed, and therefore are commonly allocated to the cloud, but determining the ideal allocation schemes is a common research problem as shown in [25–27]. One major practicality aspect is that edge users who may not live in a large enough city to have their own cloud center before, can now have their own fog node, and save on communication distance simply because the cost to make a fog node is much lower than a cloud center.

Because it is an extension of cloud computing, fog computing suffers from a lot of the same drawbacks, such as security, and internet-dependence. Security may be an even larger concern because it is separated across several physical locations which makes it more vulnerable than a single high-security location. One nice security benefit is that even though there are more vulnerable locations, the amount of data stored at each one is reduced. It also increases the complexity of the system because instead of sending all of the data to a single location, each task needs to be individually allocated to one of several fog nodes, or the cloud center. The biggest downside is the power consumption, which is increased because of the inefficiency of fog nodes.

1.1.6 Emergency Networks

An emergency communication network can provide a communication system after the regular communication infrastructure has been destroyed in a disaster.

1.1.6.1 Disaster Area

A disaster area is a location or region that has been heavily damaged by either natural, technological or social hazards. Disaster areas affect the local or nearby population, national economy, industrial economy, and natural economy. The disasters can also cause discontinuity of services, lost energy, shortage of food, an increase in the risk of diseases spreading, etc. An area struck by a natural disaster will affect quality of life for the citizens and the environment. Natural disasters includes: tornadoes, hurricanes, floods, and earthquakes. Hurricanes or typhoons can be refereed to as the same thing, the only difference being location.

Earthquakes are a result of the shaking of the earth's crust, which is caused by underground volcanic forces of breaking and shifting rock beneath the earth's surface. When a major earthquake happens, most of the utilities will be damaged, especially communication facilities, which can easily be knocked down. The victims of the earthquake can not send out any message to their family members or rescue teams.

Big data analysis is especially important in disaster scenarios. Strategic ambulance deployment can be decided by big data processing algorithms. Additionally, efficient evacuation requires traffic knowledge, which is also handled by big data processing. It is critical to immediately recover the communication system when a disaster happens.

1.1.6.2 Wireless Disaster-area Networks

In order to repair communication networks rapidly after they are damaged by a disaster, many companies deploy wireless networks to patch up the damaged areas of the traditional network. Wireless networks are computer networks that use wireless data connections between network nodes. Wireless networks avoid to install or connect with the cables for internet connection. This implementation occur at the physical layer of the network structure.

Wireless network include cell phone networks, wireless local area networks (WLANs), wireless sensor networks, satellite communication networks, and terrestrial microwave networks.

The first wireless network was developed in 1969 at the university of Hawaii and became operational in June 1971. In 1991, 2G cell phone network started. In June of 1997, the ”Wi-Fi” protocol was first released.

Wireless networks include a few types. Wireless personal area networks (WPANs) provide internet for a small area, that is usually within a person’s reach. For example, Bluetooth connections, etc.

Wireless local area network (WLAN) provide a connection for two or more devices in a short distance. A cellular network or mobile network is a radio network distributed over land areas called cells, this enables a large of number nodes to communicate with each transceiver, known as cell site or base station.

Wireless disaster-area networks is a type of cellular communication network. MBSs can be positioned to reconstruct an emergency communication network after a disaster. One currently available MBS is the movable and deployable resource unit (MDRU) from NTT in Japan. This type of base station can help to rebuild the communication network in disaster areas. This MDRU assists in the rescue efforts for the local people.

1.2 Thesis Objectives

The regular communication infrastructure can be damaged by disasters. NTT provided an easily deployable solution to construct an emergency communication network (ECN), but ECNs are slow at propagating big data due to their limited transmission capabilities. One major issue is efficiently integrating data processing in the ECN to realize effective data processing and transmission in disaster scenarios.

1.3 Thesis contributions

In order to improve the performance using the current technology, we propose a few contributions in this thesis. The main contributions of this research are as follows:

- The first contribution of this paper was a set of models for fog computing on a wireless network. In Chapter 3, we started with a simplistic model and used it to evaluate our algorithm. In Chapter 4, we proved that the model was NP-Hard. In Chapter 5, the model was improved by accounting for queuing delay in the processor and the communication networks. Additionally, we accounted for average delay.

- We proposed a genetic algorithm which was able to solve for the ideal set of data processing ratios for each node to reduce the maximum delay that the system. We started with a random population, and ran each of the genes through the models, and selected the best results, and performed crossover and mutation with them. This is repeated for 80 generations, but a solution usually settled before 40 generations.

- We also presented a real-time algorithm in Chapter 4 that could solve the same problem as the GA from Chapter 3. This new DAADM algorithm would account for the network structure and performance metrics. While this is running, it is assumed that the structure is not changing in real-time, but the performance of the links connected to the current node can change in real-time. This allows for the algorithm to adapt the processing ratio in real time, without receiving additional information from other nodes.

- New models were made to find the average delay of each node in the system, which is more applicable to a herd-satisfaction model, and we attempted to solve for the ideal mapping connections for each node. This was also handled by a GA, which functioned similarly to the one used in the second contribution.

1.4 Thesis Outline

The rest of the thesis is organized as follows:

- Chapter 2 presents some of the works that are related to communication network infrastructure optimization.
- Chapter 3 introduces the system model and genetic algorithm for determining the optimal data processing ratio.
- Chapter 4 presents a real-time solution for the data processing ratio, as well as the proof for the NP-hardness of the problem.
- Chapter 5 introduces the queueing delay model and covers the fog node topology optimization algorithm, and the methodology and results are also described in each sections.
- Chapter 6 demonstrates a system model that starts with individual users, and allocates them to MBSs according to a genetic algorithm.
- Finally in Chapter 7, we highlight the advantages and summarize the main conclusions of this thesis, and introduce some future works to optimize the contents even more.

Chapter 2

Related Works

In this chapter, we will first discuss some edge computing networks. We will then have a section detailing some of the optimization schemes which have been used in fog networks.

2.1 Edge Computing Networks

Recent studies on cloudlets [28] provide possible solutions for offloading tasks from the cloud to the devices near users by deploying special servers in the urban areas. A cloudlet network is a concept related to fog computing. A cloudlet is basically a small cloud that provides service in a similar way to traditional cloud computing. Cloud computing offered seemingly unlimited resources, but cloudlets run with limited resources. The main benefit of cloudlets is that they can be located closer to users, so users can directly connect to other devices in a cloudlet. The response time is therefore significantly reduced.

The optimal deployment of a cloudlet server has become one key research problem, as studied in [29]. Our solution can be considered to be an extension for big data analysis work on connected cloudlet servers.

Fog computing and its non-stationary counterpart, mobile fog computing, have been proven to be an efficient way to process big data algorithms. In [30], the authors proved that mobile fog networks had three main advantages: high-performance, low operational costs, and high elasticity/adaptivity. They applied a data flow algorithm to the network and then optimized the computation partitioning. This allowed them to achieve a higher throughput. Fog computing is also called edge computing because fog resources are placed near the edge of the service area. Fog computing facilities assist a cloud center in processing IoT services. Fog networks can match the demand of high-speed mobile devices and reduce the bandwidth load on the network core.

Ad-hoc computing networks do not rely on a pre-existing infrastructure, such as routers or access points. Each node forwards data to other nodes, thus the data transmission is based on the network connections and routing algorithms.

Ad-hoc networks are suitable for a variety of applications, such as Vehicular ad hoc networks (VANETs). Authors in [31] presented a survey on VANETs. They describe the network architecture and communication domains of VANETs. They also discuss the wireless access technologies that can be used to establish the communication of the network. Network challenges, requirements and simulation tools are also presented in the paper.

2.2 Fog Network Optimization

One of the major research problems in fog computing is task offloading/scheduling, as shown in [25–27]. However, the recent solutions may not fully satisfy a disaster scenario when reconstructing an emergency communication network with MBSs, since most of them only consider data transmission. Additionally, spatial big data is collected from a majority of smart phones, which makes the solution too complex if we see each smart phone as the basic unit for task distribution. Finally, besides 0/1 decisions (fully in cloud or fully local), it is necessary to investigate partial allocation solutions for data processing locally or in the cloud in this study.

Authors in [25] discussed a Fog Computing Supported Medical Cyber-Physical System (MCPSs), in which task distribution and VM placement problems are formalized as a Mixed Integer Non-liner Problem (MINLP).To support MCPSs, cloud resources are used to process the medical sensing data. Meanwhile, a high quality-of-service is required by MCPSs between the cloud data center and medical devices. To solve this issue, mobile edge cloud computing, or fog computing, offloads the computation resources to the edge nodes. Thus, the fog computing supported MCPS were built. More specifically, they jointly investigate base station association, task distribution, and virtual machine placement toward cost-efficient fog-based MCPS. To address the computation complexity, they also propose a Linear programming (LP) based two-phase heuristic algorithm. They show that their proposed algorithm has a performance advantage over the greedy one.

Similar work has been done in [26], where the task scheduling problem has been involved. They have discussed about which data should be analysed in which fog node, and which fog node should have a VM deployment to minimize total cost. Both of them have been solved through linearization of the problem.

A computation offloading game was designed in [32] for MCC based on game theory. They formulated the decentralized computation offloading decision among mobile device users. The results shows that their system achieves efficient computation offloading performance especially as the system size increases.

Rapidly delivering related information in ECNs can be a major challenge, because the data right after a disaster can sometimes be critical for life. A rapid and flexible disaster recovery network is required because traffic loads can change unpredictably [33]. A truly flexible network is able to take full advantage of all resources in real-time after the loads change.

Authors in [34] focus on the interplay and cooperation between the edge (fog) and the core (cloud). They investigated the tradeoff between power consumption and transmission delay in fog-cloud computing systems. They formulate the workload allocation problem and approximately decompose the primal problem into three subproblems, which can be, respectively, solved within corresponding subsystems. Based on the simulation and results, they presented that sacrificing a modest amount of computation resources could save communication bandwidth and reduce transmission latency, and that fog computing would significantly improve the performance of cloud computing. As such, fog computing complements cloud computing toward low-latency high-performance services for mobile users.

The delay benefits of using a fog-computing architecture for smart-cities has been demonstrated in [35]. The system leveraged closer processing nodes to reduce the overall system delay and energy consumption. But, this paper did not consider the possibility that partially processing the data in the cloud could result in an even lower delay.

Managing the computation distribution is a technique used in this field to efficiently use the limited computation and communication resources in the fog layer, as studied in [25,27,36,37].

In [37], the authors first studied the multi-user computation offloading problem for mobile-edge cloud computing in a multi-channel wireless interference environment, then designed a distributed computation offloading algorithm. They mainly showed that computing a centralized optimal solution is NP-hard, and hence adopt a game theory approach for achieving efficient computation offloading in a distributed manner. Numerical results verified that the proposed algorithm can achieve superior computation offloading performance and scale well as the number of users increases.

There are several studies on efficient computation distributions for mobile edge-clouds (MECs) computing. In [36], the authors studied the dynamic service migration problem in mobile edge-clouds that host cloud-based services at the network edge, and formulate a sequential decision-making problem for service migration using the framework of Markov Decision Process (MDP). The results provide an efficient solution to service migration in MECs. This offers benefits in the form of a reduction in network overhead and latency.

The authors in [27] optimized the users' quality of experience (QoE) and network performance. They are addressed the issue of load balancing in fog computing. The challenging of this paper is that when multiple users requiring computation offloading, where all requests should be processed by local computation clusters resources. They proposed a customizable algorithm for fog clustering. Their results showed that their proposed algorithm has a higher performance.

The non-stationary counterpart of fog computing has given an efficient solution for processing big data algorithms. The authors in [38] illustrated that mobile fog networks had three major merits: high-performance, low operational costs and high adaptivity. They optimized the computation partitioning by applying a data flow algorithm to the network.

In [39], authors provide a high-performance computation system for mobile devices by introducing the optimal tunable-complexity bandwidth manager (TCBM). They tested an optimal bandwidth manager for live migration of virtual machines in wireless channels from smartphone to access point. In order to reduce the energy consumption, a novel approach for bandwidth management was proposed using live migration of virtual machines in a wireless network context. The results showed a significant improvement when using TCBM.

Authors in [40] introduced that femtocells are a effective way to build up communication structure and system capacity. The benefits and drawbacks of the different access methods, including open access, closed access and hybrid access methods, were also elaborated in the paper. Closed and open access both have disadvantages, cross-tier brings lower network performance for closed access, and fewer customers' acceptance for open access, as well as a large number of handovers. Hybrid access offers a full range of algorithm that can be defined in order to control who accesses the femtocell and how the connection is configured.

In [41], authors found that some of media application have unnecessary energy consumption, thus, they introduced a new energy adaptive (EA) objective. They formulate the energy-efficient downlink resource allocation problem. With the required user's data rate base station's available transmit power, they minimize the total energy consumption of smart devices. The results shows that their proposal is very effective for reducing energy consumption of smart devices.

2.3 Chapter Summary

In this chapter, we discussed some of the important related works that deal with computing network architectures. More specifically, we covered edge computing networks, and fog network optimization. Several authors address the problem of task-allocation, but none of the works covered in this chapter consider users in overlapping areas or the data processing ratios of each node. Future algorithms will have to extend these existing works, and combine optimization of the task allocation with these aspects so that future fog networks can run as efficiently as possible.

Chapter 3

Model and Genetic Algorithm for Data Processing Ratio

3.1 Introduction

Spatial big data analytics has become possible with the data collected from the sensors in smart phones, which can support decision-making in disaster scenarios. However, sometimes the regular communication infrastructure can be destroyed after disasters. Movable base stations (MBS), as studied by the company NTT, offer an easily deployable solution to construct an emergency communication network, but are not suitable for transmitting big data from sensing devices to the cloud for data processing in the cloud. To solve this issue, we have studied a novel algorithm to process spatial big data efficiently in a wirelessly networked disaster area that uses multiple MBSs. More specifically, we proposed a novel algorithm to minimize overall delay for spatial data processing in wirelessly-networked disaster areas (SDP-WNDA), to enable quick responses to data analysis.

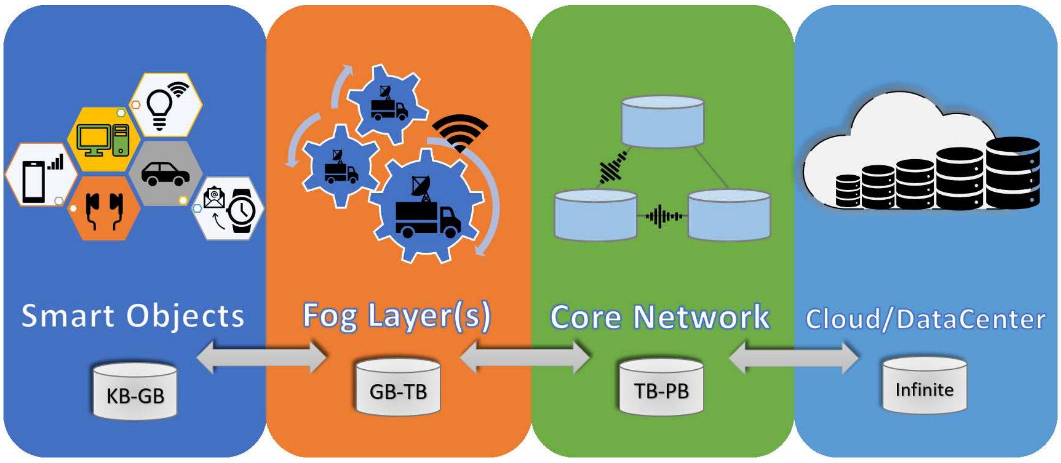

Figure 3.1: Different layers of a fog computing network.

3.2 System Model and Problem Formulation

3.2.1 System Model

Fog computing gave a possible solution by offloading computation tasks from cloud to edge devices. Fig 3.1 shows the different layers of a fog computing network. Fig. 3.2 represents a system model for a SDP-WNDA. After a disaster, movable base stations (MBSs) can be deployed to the disaster-stricken areas, to reconstruct a communication network. Users can upload data through the MBSs.

We used k to denote a MBS, which integrates both wireless communication and computation functions. First, smart phone users connect with k and upload data with spatial information, where the data size is represented as d_k. Then we assumed a part of the SBDA algorithm has been deployed in the MBS beforehand, and that the data size compression ratio after processing in node k is ρ, where $0 < \rho \leq 1$. The processing rate in each MBS k is represented as μ_k.

We also assumed that the MBSs are connected via a wireless medium, and that the communication rate is represented by $\mathcal{R}$. We assumed all MBSs send data into a special node j, which is located at the edge of the disaster area and can transmit data to the cloud quickly with a wired connection. To simplify the discussions, we considered a single j node case, but a multiple j node case can be modeled and solved similarly. Generally speaking, data analysis in cloud centers will not start until all the data is collected from the whole area. To enable quick decision-making in the disaster scenario, in this section, we investigated the minimal overall delay between mobile phones and the node j. Meanwhile, we supposed that for each end node i, there is a designated path from i to j, denoted as $\mathbb{P}_i$. We also grouped all child nodes of k in the set $\mathbb{C}_k$, and all nodes in the set $\mathbb{N}$.

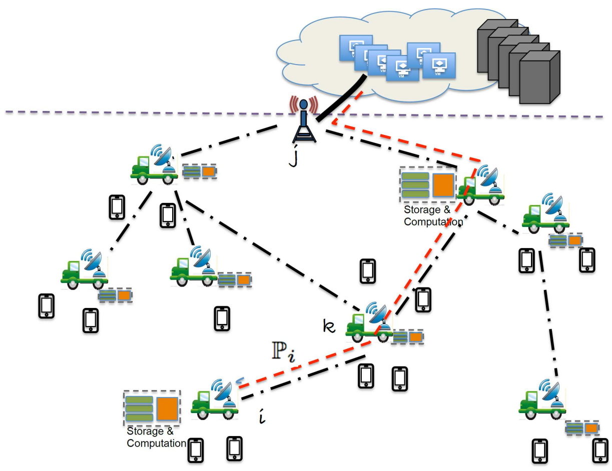

Figure 3.2: System Model

3.2.2 Problem Formulation

Based on the system model in Fig. 3.2, we studied the minimal delay from each end node to j for transmission to the cloud center. By adjusting the data processing ratio, denoted as x_k, in each MBS k. $\mathbb{X}$ is the set including the processing ratios for all MBSs.

The delay of each path $\mathbb{P}_i$ can be calculated based on the processing delay and transmission delay in each relay node k, which are formalized as follows.

3.2.2.1 Delay for Processing Data in Node k

To study the processing delay of each node k, we first formalized the size of the output data after processing as follows.

The size of the output data after processing in node k is denoted as O_k, and can be represented by two parts. One is the size of processed data and the other part is the size of the unprocessed data. They are different, since the data size will be compressed by ρ after data processing in the MBSs.

$$O_k = O_k^p + O_k^u \tag{3.2.1}$$

where O_k^p represents the processed data and O_k^u represents the unprocessed data from node i.

Then O_k^u is calculated as follows:

$$O_k^u = (d_k + \sum_{i \in \mathbb{C}_k} O_i^u) * (1 - x_k) \tag{3.2.2}$$

where O_k^u depends on the amount of data collected in node k and the amount of unprocessed data from k's child nodes $\mathbb{C}_k$.

Then, O_k^p is formalized as follows:

$$O_k^p = (d_k + \sum_{i \in \mathbb{C}_k} O_i^u) * \rho * x_k + \sum_{i \in \mathbb{C}_k} O_i^p \tag{3.2.3}$$

where the data processed in k's child nodes $\mathbb{C}_k$ are considered in the equation.

Generally speaking, the data processing often starts collecting data from all data resources, so the processing delay in node k is calculated as follows:

$$D_k^p = \frac{(\sum_{i \in \mathbb{C}_k} O_i^u + d_k) * x_k}{\mu_k} \tag{3.2.4}$$

3.2.2.2 Delay for Receiving Data in Node k

The second type of delay in node k comes from receiving data from k's child nodes, $\mathbb{C}_k$. Here we assumed the data collection and transmission in the network are well scheduled, so that the receiving delay in node k can be represented as:

$$D_k^r = \sum_{i \in \mathbb{C}_k} \frac{O_i}{\mathcal{R}} \tag{3.2.5}$$

3.2.2.3 Overall Delay and Problem Formulation

Then the overall delay in the node k can be represented as:

$$D_k = D_k^p + D_k^r \tag{3.2.6}$$

For each end node i, the overall delay of sending data to j can be represented as

$$D_i^o = \sum_{k \in \mathbb{P}_i} D_k \tag{3.2.7}$$

Therefore, the total delay of the network is represented as:

$$D^o = \max_{i \in \mathbb{N}} D_i^o \tag{3.2.8}$$

Then, the delay minimization problem (DMP) for spatial data processing in wireless networked disaster areas (WNDA) can be formalized as follows.

DMP-WNDA: *In wireless connected disaster areas, the goal is to minimize the overall delay to send all data outside under the system model in section 4.2.1, by adjusting the processing ratio of each MBS.*

The mathematic representation is:

Find the set $\mathbb{X}$ to $\min(D^o)$

3.3 Proposed Solution (GA)

The goal of our system is to find the set $\mathbb{X}$, which yields the minimum system delay. To achieve this, we have implemented a genetic algorithm [42], which is a widely

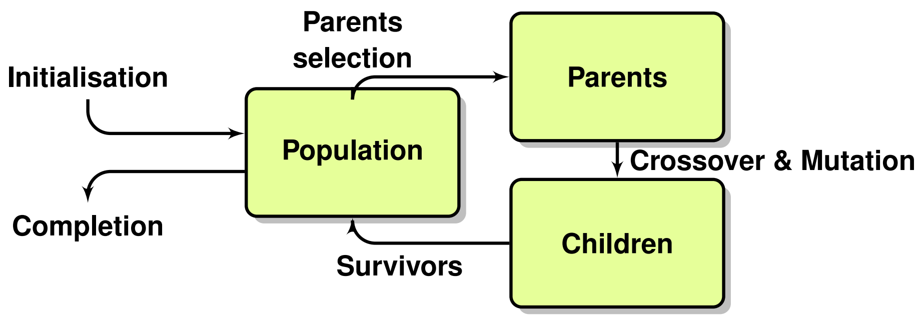

Figure 3.3: Genetic algorithm flow chart

used optimization solution that can be applied to a wide variety of problems. Our genetic algorithm's flow chart can be seen in Fig. 3.3. Each simulation starts with an initial population, called generation one. Then the algorithm selects the ideal candidates, performs genetic crossover and then mutation, creating the next generation. The algorithm perform the selection, crossover and mutations 80 times, until a solution is settled. We compared the results of the genetic algorithm to some baseline cases. The first baseline case was when all $\mathbb{X}_k$ values are set to 0, which represents a typical cloud computing system where all of the data is processed at the cloud [12]. The next case covered situations where each node processes all of the data that it receives immediately. Finally, we created a case where 50% of the data is processed in the fog [43], and 50% is handled by the cloud. We figured that these are some of the standard cases, representing the conventional, greedy, and balanced algorithms.

The modeling and calculations were all coded in MATLAB. We first attempted to determine the capabilities of each algorithm across different network sizes. We did this by first generating 3 random networks with 3 random sizes. The small network was an 11 node system. The medium sized system was 35 nodes. The large system was 154 nodes. The large system is approximately equivalent to the island of Okinawa, which according to the 2016 report by the Japanese Ministry of Internal Affairs and Communications has stated to have 105 LTE base stations [44]. In order to let the evaluation match with real cases, we surveyed some realistic values for $\mathcal{R}$, μ_k, and ρ. Most 5G antennas currently in development suggest that the wireless speeds will reach 20Gbps [45]. Hinitt et al. [46] have estimated that a GPU based processor for wireless networks is capable of processing speeds of at least 4Gbps. The amount of processing that a standard computer can handle is still scaling very rapidly, so we expected this number to increase rapidly as well. Guo et al. [47] suggest that LTE compression is capable of a one third ratio, so we adopted this value for evaluation. Furthermore, for a variety of applications, ρ can be significantly different, from a very small value (e.g. feature extraction from data) to a bigger value (e.g. data compression). We considered the above issue and have performed evaluation accordingly to different ρ in Fig. 3.10. All of these values, and other ones used for testing

Table 3.1: System Configuration.

Design Parameters	Description and Values
O_k	the output data after processing in node k
O_k^p	the processed data from node i
O_k^u	the unprocessed data from node i
D_k^p	the processing delay in node k
D_k^r	the receiving delay in node k
D_k	overall delay in node k
D^o	total delay of whole network
Nodes	11,35,154
ρ	1/3
$\mathcal{R}$	20Gbps
μ_k	4Gbps
Node Patterns	Random with max of 5 child nodes
d_k	Random with a max value of 200 Gb
Generations	80

the scalability of the algorithms can be seen in Table 3.1.

An output of $\mathbb{X}$ for the small system is shown in Fig. 3.4.

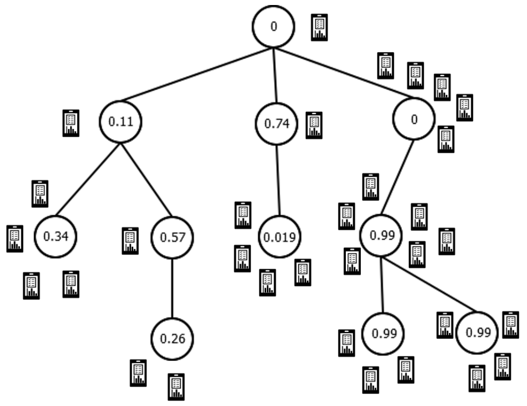

Figure 3.4: Sample ideal $\mathbb{X}$ values for a small network

In the next experiment, we attempted to see how the results change as we vary certain constraints. We first vary the μ_k, expecting the cases with faster speeds to have less delay. We then performed similar tests for $\mathcal{R}$ and ρ values.

3.4 Simulations and Performance Evaluation

3.4.1 Evaluation Across Various Network Sizes

One important aspect of the GA solution is that it is scalable in terms of the network's size. It still needs to perform better than the baseline algorithms. If it performs worse than any one of them, then there is no reason to implement it. Fig. 3.5, 3.6, and 3.7 show the worst case delay results for the small, medium, and large networks, respectively.

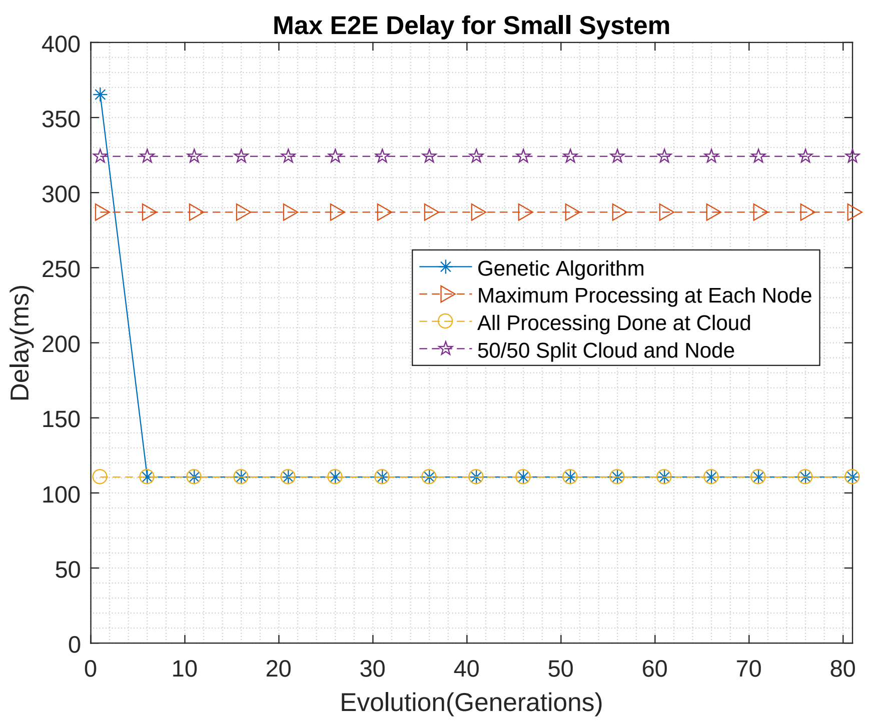

Figure 3.5: Delay evaluation on a 11 node network.

We can see that for the smallest network size, with the researched values, the cloud and the GA reach the same solution. As the network size is increased, such as Fig. 3.7 the benefit is more pronounced. These results suggest that even for a very large network, there is a benefit, in terms of system delay, to use the GA's solution.

3.4.2 Evaluation Across with Varied Bandwidth Constraints

The next step is to ensure that the GA's solution is the ideal one when the bandwidth parameters are varied. Fig. 3.8 shows the response that the delays have to these changes.

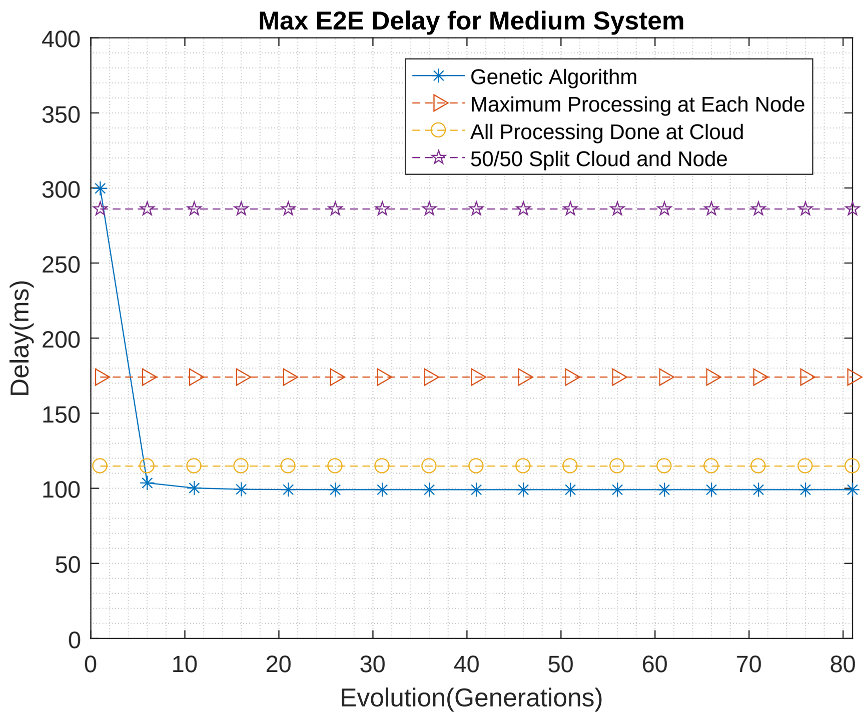

Figure 3.6: Delay evaluation on a 35 node network.

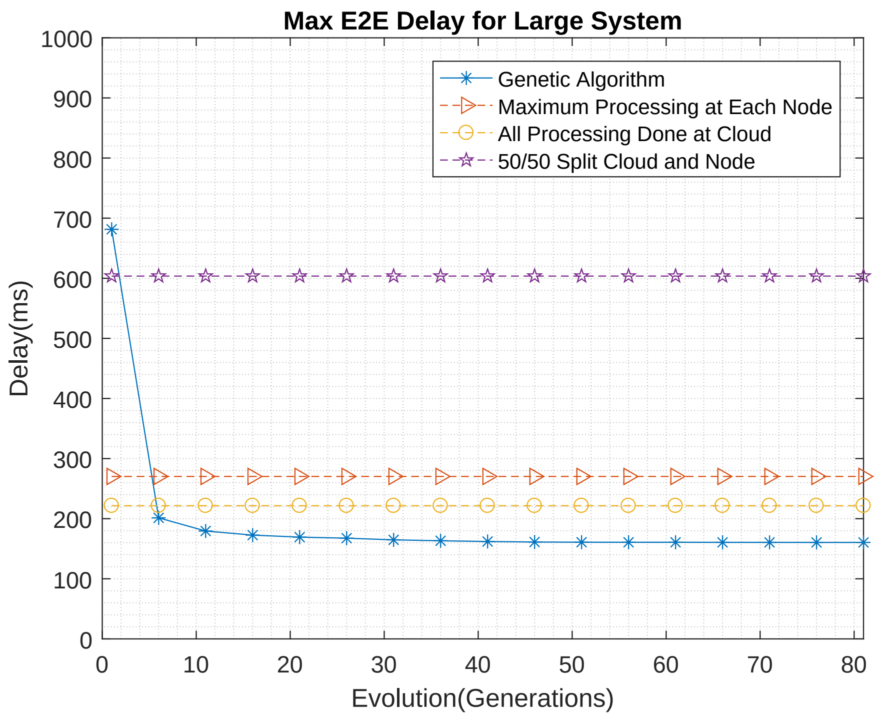

Figure 3.7: Delay evaluation on a 154 node network.

Fig. 3.9 shows the response that the delay has to changes in $\mathcal{R}$. The constraints that aren't varying for the tests are constant, and based off of realistic values found in other publications mentioned above. We only show the results for the largest system, because this is the system that should have the greatest sensitivity to changes in the bandwidth values.

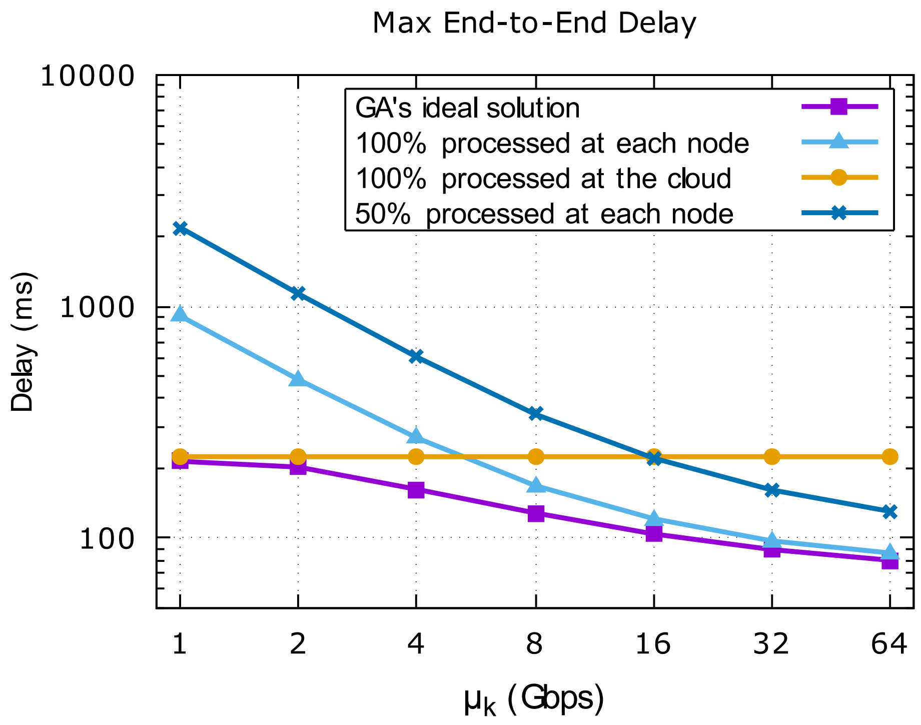

Figure 3.8: Delay values of a 154 node network with varying μ_k values.

From Fig. 3.8, we can see that as μ_k increases, all of the delays decrease, except for the cloud-based solution, which does no processing at the nodes, and is unaffected by the value of μ_k. One other thing to notice is that for the lower values, such as 4Gbps, the GA solves the problem significantly better than the conventional methods. If we designed these nodes with more processing power, then fog computing significantly outperforms cloud computing.

As $\mathcal{R}$ values increase, all of the delays decrease. The cloud computing value in particular decreases significantly. The $\mathbb{X}_k$ values yielded by the GA for the ideal case are lower as $\mathcal{R}$ increases, suggesting that as the transmission rate increases significantly in the future without a significant improvement of the computational power of the nodes, then

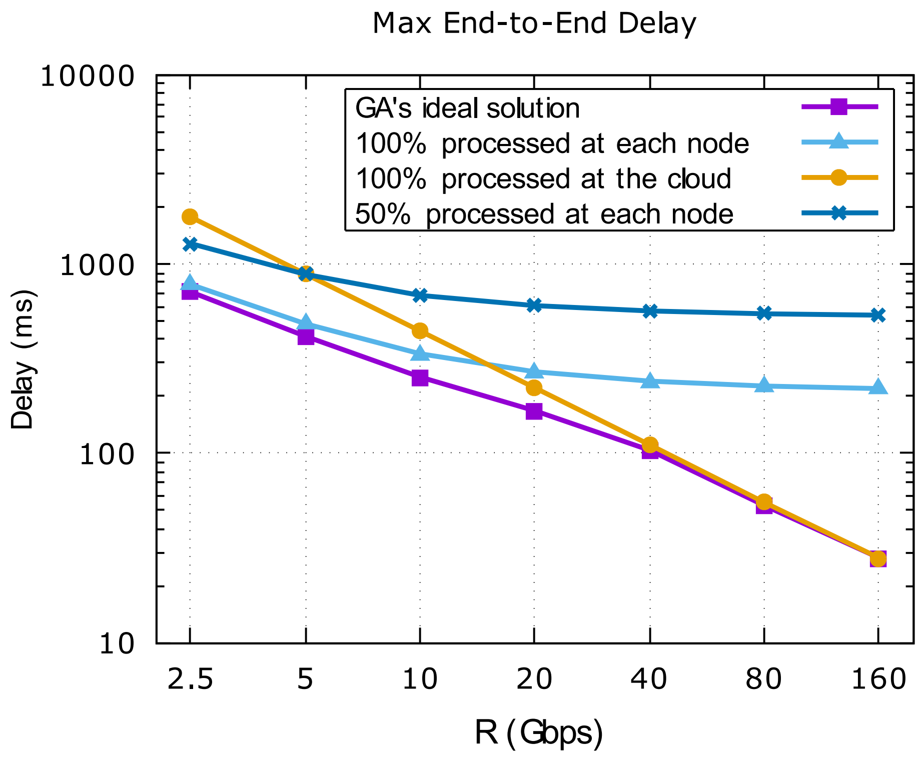

Figure 3.9: Delay values of a 154 node network with varying $\mathcal{R}$ values.

cloud computing will likely be the most efficient solution.

Fig. 3.10 shows the delay values as ρ is varied. As expected, on the high end of the ρ spectrum, the GA's solution aligns with the fully cloud processed solution. Towards the left, the GA outperforms all conventional methods. The research results show that the proposed solution is significantly better than the conventional methods for the applications with a high data reduction/compression ratio, because of the way spatial big data analysis works in.

3.5 Chapter Summary

In this paper, we presented a set of delay models for spatial data processing networks, which was specifically targeted at disaster-stricken areas. This model accounts for both the transmission delay and processing delay at each connection and node, respectively. We researched some realistic parameters, and put them into an in-house simulator, which tested some of the conventional methods, and modeled their worst-case delays. The sim-

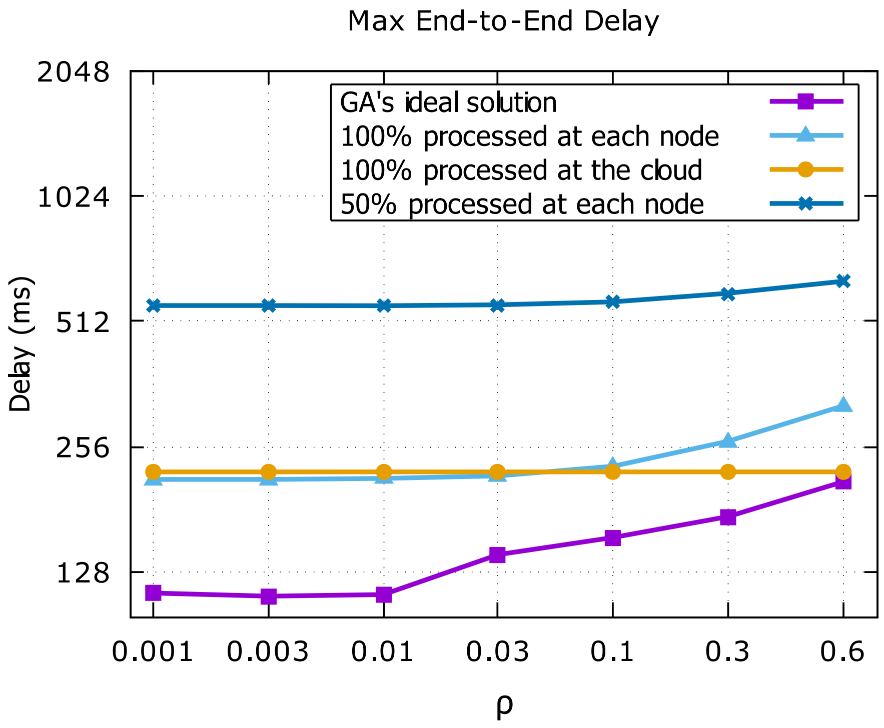

Figure 3.10: Delay values of a 154 node network with varying ρ values.

ulator also ran a genetic algorithm which found the ideal situation in each case.

Evaluation results show that for the realistic parameters, the genetic algorithm was able to solve the ideal case after approximately 40 generations for the largest system. With our researched values, the cloud system was closest to the GA. However, as μ_k was increased, the maximum processing algorithm got much stronger. But, as the $\mathcal{R}$ was increased, the cloud computing solution was more successful. The current GA solution shows that the amount of processing that needs to be done by each node changes with the $\mathcal{R}$ and μ_k values greatly, and that with the standard test parameters, the $\mathbb{X}_k$ value was heavily based on the number of hops that the node was from the cloud. This is possibly based on a transmission-compression trade-off. Additionally, because the ρ value can vary so much between compression and data processing cases, the target application varies the results greatly.

In the next chapter, we will cover an algorithm which can determine the ideal configuration in real time.

Chapter 4

Real-time Solution for the Data Processing Ratio

4.1 Introduction

In Chapter 4, we presented some models and a GA-based solution. This chapter extends that work by improving the algorithm so that it can be run in real time and adapt as the network speeds change. This chapter is relates to the work published in [48].

Fog computing (FC) is a disruptive technology in the big data analytics area. Smartphone users and organizations use cellular services, which can support decision-making in disaster scenarios with the data that has been collected. Nevertheless, the regular communication infrastructure can be damaged by disasters. NTT provided an easily deployable solution to construct an emergency communication network (ECN), but ECNs are slow at propagating big data due to their limited transmission capabilities. One major issue is efficiently integrating data processing in the ECN to realize effective data processing and transmission in disaster scenarios. In this chapter, we present: a detailed mathematical model to represent data processing and transmission in an ECN fog network; an NP-hard proof for the problem of optimizing the overall delay; and a novel algorithm to minimize the overall delay for wirelessly-networked disaster areas (WNDA) that can be run in real-time. We evaluated the systems across various transmission speeds, processing speeds, and network sizes. We also tested the calculation time, accuracy and percent error of the systems.

It is our goal to reduce the processing speed incurred by the genetic algorithm solution, and still maintain a result that has lower or equal latency, when compared to conventional algorithms, such as Fog [24] and Cloud [12] Computing. It is key to have better results than the conventional algorithms, otherwise, there is no reason to implement an adaptive one.

We attempt to minimize the effect that each node has on the system delay. One important aspect of each algorithms' efficacy is its execution time. The conventional methods (fog and cloud) determine the data processing ratio before running and they can not adapt to changes in transmission speed and processing speed, but at least they will not delay the system while trying to calculate how much data to process. Genetic algorithms, such as in [49], can yield a final solution that minimizes the system's delay, but they may take a significant amount of time to coverage. Thus, our proposed algorithm will attempt to get better results, in particular, be able to adapt to changing processing and transmission rates, but not require a large amount of processing power, which is feasible with a simple calculation.

The main contributions of this section are summarized as follows:

- We developed an MBS-based wireless network model that considers real-time computations for big data collection and analysis. The problem model is demonstrated to be a non-deterministic polynomial-time hardness (NP-Hard) problem.
- We propose an adaptive algorithm for fog-layer big data analysis which determines the data processing ratio for real-time wireless computation structures. The optimization objective is to achieve delay minimization for data processing of a WNDA in a real-time manner.
- We demonstrate the instant response of the proposed scheme through theoretical and simulation studies, which prove that the proposed adaptive algorithm can achieve high-performance results.

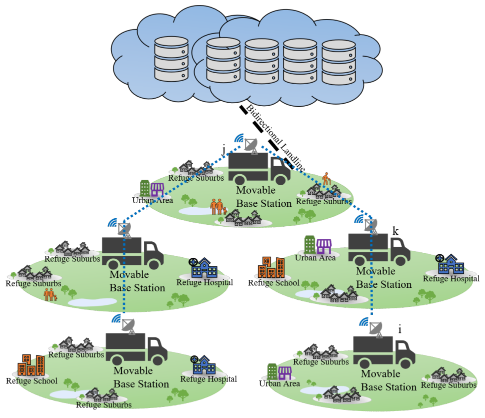

Figure 4.1: System Model

4.2 System Model and Problem Definition

4.2.1 System Model

Fig. 4.1 represents the system model for a WNDA. It is created by deploying MBSs to a disaster area in order to patch up the holes in the network. From a user's point-of-view, the network should work seamlessly, but slower. The proposed scenario is similar to [50], but the solution proposed in this section is solvable in real-time. .1

A single node which has computation and communication capabilities is denoted by the letter k. Smartphones connect with k, and upload large amounts of data. The total amount of data that k receives from these smartphones is denoted as d_k. The big-data processing algorithm that we are trying to optimize has a data compression ratio of ρ, where $0 < \rho \leq 1$. The algorithm can be run at a rate of μ_k Gbps on the MBSs. The communication speed between a node k, and another node i, is denoted as $\mathcal{R}_{ik}$.

Due to the aggregation needs of most big-data processing, we assume that all the data must be collected by a single node, j, which is located at the edge of the disaster, and is capable of transmitting data to the cloud in a quick and efficient manner because the wired connection from that point is still intact. This paper considers and discusses a case with a single j node, but a multiple-j-node case can be solved similarly. Because of the nature of big-data algorithms, the data analysis generally does not begin until all the data from the area has been collected into a single node, but some of the processing can be done to reduce the size of the data that is transmitted. This means that the algorithm cannot begin its final step until the data from the last node is received at the cloud. The time between starting to send data, and the last bit of data being received by the cloud is called the overall delay, and this paper attempts to minimize that overall delay using a real-time algorithm. To do this we assume that for each node i, there is a path $\mathbb{P}_i$ that is predetermined and terminates at j. Each node k has a set of child nodes $\mathbb{C}_k$ based on all of $\mathbb{P}_i$'s for each node in $\mathbb{N}$, the set that contains all nodes in the system.

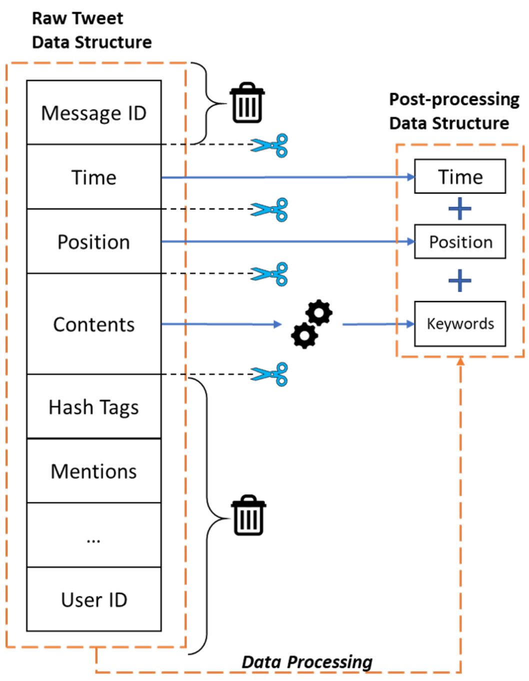

Figure 4.2: Data Structure Change from Spatial Data Processing on a Tweet

To help understand how data processing can achieve a significant size reduction, Fig. 4.2 demonstrates one way to handle spatial data processing. We can see that the raw tweet structure contains several bytes of information related to twitter and linking to other tweets. The algorithm only saves a few pieces of this data to make a single data point, namely: Time, Position, and Keywords. The time and position values are carried from the raw structure, but keywords must be generated using an algorithm that is specific to the application. The remaining data that is not used, such as hash-tags, can be discarded.

The authors in [51] have used spatial data to track human mobility throughout the day. Instead of analyzing tweets, they take data submitted by taxis and subways, and perform some analysis. Through the spatial analysis they are able to determine the travel distance, trip duration and trip interval. This information led the authors to conclude things such as whether people travel more for work or for pleasure on the subway, but that taxi users do not lean one way or another. This is one such algorithm that could run on an MBS-based Fog system in order to track a city's return to work after a disaster.

4.2.2 Problem Formulation

Based on the system model in Fig. 4.1, we came up with a series of equations in [50]. We attempt to solve the same model, but with an additional guarantee that the calculation time is small enough to be run in real-time. We also assume that no node knows the traffic of any other node, this way none of the bandwidth of the infrastructure is used to transfer traffic information.

Table 4.1: Variable Definitions

Design Parameters	Definition
O_k	Node k total output
O_i^p	Node i processed output
O_i^u	Node i unprocessed output
$\mathcal{I}_k^u$	Node k total unprocessed input
$\mathcal{D}_k^p$	Delay for processing data in node k
$\mathcal{D}_k^r$	Delay for receiving data in node k
$\mathcal{D}^o$	Network's overall delay
$\mathbb{C}_k$	Set of all child nodes of k
ρ	Data compression ratio
$\mathcal{R}_{ik}$	Communication rate between i and k
μ_k	Processing rate of node k
d_k	Cellular data input at node k
$\mathcal{H}_k$	Number of hops remaining in the path
$\Delta\mathcal{D}$	Delay effect of each node

Given the abbreviations in Table 4.1, we consider the output data to be:

$$O_k = O_k^u + O_k^p = [\mathcal{I}_k^u * (1 - \mathrm{X}_k)] + [\mathcal{I}_k^u * \rho * \mathrm{X}_k + \sum_{i \in \mathbb{C}_k} O_i^p] \tag{4.2.1}$$

Figure 4.3 demonstrates the basic data flow, and the adopted abbreviation. Node k receives raw data from nearby cellphones, labeled as d_k, as well as any unprocessed data (O_l^u and O_m^u) from the child nodes ($\mathbb{C}_k \in \{l, m\}$). All of these points combine to make the unprocessed input to node k ($\mathcal{I}_k^u$). The child nodes also output some processed data to node k (O_l^p and O_m^p). Only the unprocessed data can be processed in node k, because the same data can not be processed twice by the algorithm. Node k then outputs some processed (O_k^p) and unprocessed (O_k^u) data.

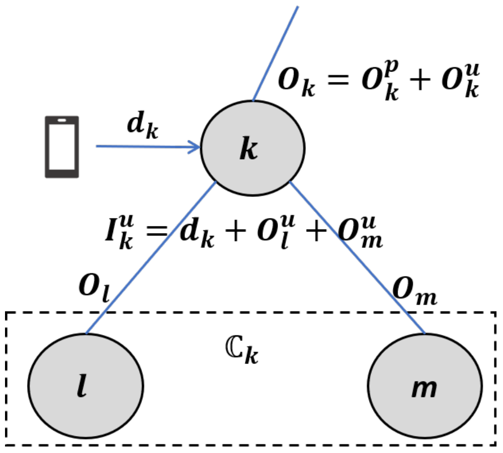

Figure 4.3: Node Model

Then we consider the delay measurement of a single node to be:

$$\mathcal{D}_k = \mathcal{D}_k^p + \mathcal{D}_k^r = [(\mathcal{I}_k^u * \mathrm{X}_k) * {\mu_k}^{-1}] + [\sum_{i \in \mathbb{C}_k} \frac{O_i}{\mathcal{R}_{ik}}, \forall i \in \mathbb{C}_k] \tag{4.2.2}$$

The delay for i to process and send its data to j can be calculated as:

$$\mathcal{D}_i^o = \sum_{k \in \mathbb{P}_i} \mathcal{D}_k, \forall i \in \mathbb{C}_k \tag{4.2.3}$$

And because the system's overall delay is based on the time to receive the last bit of data, the overall delay of the system is:

$$\begin{aligned} \mathcal{D}^o &= \max_{i \in \mathbb{N}} \mathcal{D}_i^o \\ s.t. \quad & (4.2.1), (4.2.2), \; and \; (4.2.3) \end{aligned} \tag{4.2.4}$$

We then formalize the delay minimization problem for wirelessly networked disaster areas (DMP-WNDA) as:

DMP-WNDA: *Minimize the overall delay for sending all data to the cloud by adjusting the data processing ratios of the nodes in the system.*

Find the set $\mathbb{X}$ to $\min(\mathcal{D}^o)$

4.2.3 NP-hardness of DMP-WNDA

Before proposing the algorithm, we first analyze the NP-hardness of the DMP-WNDA problem in this section, by first proving that the DMP-WNDA problem is equivalent to the Partition Problem (PP), we first introduce the following definition 1.

Definition 1: The Partition Problem (PP): decide whether a given multi-set $\mathcal{S}$ of positive integers can be partitioned into two subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ such that the sum of $\mathcal{S}_1$ is equal to the sum of $\mathcal{S}_2$.

Instance 1: Given a list of positive integers $\{A_1, A_2, ..., A_n\}$, determine whether there is a set of $\mathrm{I} \subseteq \{1, 2, ..., n\}$ such that $\sum_{i \in \mathrm{I}} A_i = \sum_{i \notin \mathrm{I}} A_i$.

The PP has been proven to be an NP-hard problem [52]. In the following *Proof 1*, we show that the DMP-WNDA problem is an expansion of the PP.

Proposition 1: DMP-WNDA is an NP-hard problem.

Proof 1: We start by assuming that the DMP-WNDA can be simplified from the original problem as a system which has one fog node $\mathcal{F}$ and one cloud node C. Second, we denote the data collected from devices as $\mathbb{D} = \{d_1, d_2, ..d_e, .., d_n\}$, where d_e is the size of data block e, and there are n data blocks in total. Then, the original problem can be redefined to find the minimal delay based on the data block allocation instead of based on the data processing ratio. We build a WNDA fog computing system instance as $\{C, \mathcal{F}, \mathcal{S}, \mathbb{D}\}$ to evaluate the delay according to the aforementioned conditions.

We can formalize these notions as follows: $\mathbb{S} = \{\mathrm{s}_e\}^n$ is a set to denote the data distribution strategy for n blocks data, where s_e is a binary value: 1 denotes fog and 0 represents cloud allocation respectively, and a system delay $\mathcal{D}(\mathbb{S})$ that is based on the s_e values. Given that $\mathrm{s}_e \in \mathbb{S}$, $1 \leq e \leq n$, determine if there are two subsets $\mathbb{S}_1, \mathbb{S}_2 \subseteq \mathcal{S}$ and $\mathbb{S}_1 \cup \mathbb{S}_2 = \mathcal{S}$, $\mathbb{S}_1 \cap \mathbb{S}_2 = \varnothing$, such that $\sum_{s \in \mathbb{S}_1} \mathcal{D}(S) = \sum_{s \in \mathbb{S}_2} \mathcal{D}(S)$. Then we consider the two disjoint distribution strategies, $\mathbb{S}_1 = \{s_1^1, s_2^1, ..s_e^1, ..s_n^1\}$ and $\mathbb{S}_2 = \{s_1^2, s_2^2, ..s_e^2, ..s_n^2\}$, where $s_e^1 =! s_e^2$, for each block of data e.

Due to Eq. (4.2.4), and considering one fog node system, $\mathcal{D}^o$ can be represented as

$$\mathcal{D}^o = \frac{D_T * X^a}{\mu} + \frac{\rho * D_T * X^a + (1 - X^a) * D_T}{\mathcal{R}} \tag{4.2.5}$$

where $D_T = \sum_{e \in n} d_e$ is the total generated data and X^a is the distribution ratio between the cloud and the fog layer, which is equivalent to X_k in the original DMP-WNDA problem, where a can be 1 or 2 to represent them based on which distribution strategy $\mathbb{S}_1$ or $\mathbb{S}_2$ is used. X^1 and X^2 can be further represented as follows,

$$X^1 = \frac{\sum_{e \in N}[d_e * s_e^1]}{D_T} \quad , \quad X^2 = \frac{\sum_{e \in N}[d_e * s_e^2]}{D_T} \tag{4.2.6}$$

When considering $X^1 = X^2$, which yields the minimum system delay as long as $\mathcal{R} = (1 - \rho) * \mu$, we can achieve two equal system delays for $\mathbb{S}_1$ and $\mathbb{S}_2$ as

$$\frac{\sum_{e \in N} d_e * s_e^1}{\mu} + \frac{(\rho - 1) * \sum_{e \in N}[d_e * s_e^1] + \sum_{e \in N} d_e}{\mathcal{R}} = \frac{\sum_{e \in N} d_e * s_e^2}{\mu} + \frac{(\rho - 1) * \sum_{e \in N}[d_e * s_e^2] + \sum_{e \in N} d_e}{\mathcal{R}} \tag{4.2.7}$$

The above statements combined with the fact that X^1 is X^2's complement means that they are both equal to $\frac{1}{2}$. In such cases, the DMP-WNDA problem can be represented as:

$$\sum_{e \in N}[d_e * s_e^1] = \sum_{e \in N}[d_e * s_e^2] = \frac{1}{2} * \sum_{e \in N} d_e \tag{4.2.8}$$

which is the definition of the PP problem as long as we assume that the data block sizes are integers, therefore DMP-WNDA is an NP-hard problem.

4.2.4 Proposed Algorithm (DAADM)

Since the problem is NP-Hard, we settled for an adaptive algorithm that accounts for the number of hops to reach node j and various network parameters and has each node calculate its own X_k. We relaxed the problem for our proposed solution, so we expected that the delay results would not be as good as the GA. We still demanded that the solution is never worse than the conventional methods for every variation of topology and network parameters. We propose a solution that minimizes the delay effect of each node ($\Delta\mathcal{D}_k$), which is the amount of delay that comes from processing the data in node k affects the $\mathcal{D}_i^o$ of $\mathbb{P}_i$. The model does not account for transmission delay of data that has already been processed in a previous node ($\sum_{i\in\mathbb{C}_k} O_i^p/\mathcal{R}_{ik}$), because the current node can not process it any further or do anything that would affect the delay caused by that data, but can only forward the data to the next node in the path. The number of hops to reach node j is multiplied by the estimated delay per hop, so we can get an estimate for the effect that processing the data in the current node has on the transmission delay for all hops in the path $\mathbb{P}_i$. The set of unprocessed data from node i ($i \in \mathbb{C}_k$) denoted as $\mathcal{I}_k^u$, where $\mathcal{H}_k$ represents the number of hops that remain in the path from i to j in $\mathbb{P}_i$ ($i \in \mathbb{C}_k$), and the $\Delta\mathcal{D}_k$ is defined as:

$$\begin{aligned}\Delta\mathcal{D}_k &= \mathcal{D}_k^p + \mathcal{H}_k(\mathcal{D}_k^r - \frac{\sum\limits_{i\in\mathbb{C}_k} O_i^p}{\mathcal{R}_{ik}}) \\ &= \frac{\mathcal{I}_k^u * \mathrm{X}_k}{\mu_k} + \mathcal{H}_k(\frac{\mathcal{I}_k^u * \rho * \mathrm{X}_k}{\mathcal{R}_{ik}} + \frac{\mathcal{I}_k^u * (1 - \mathrm{X}_k)}{\mathcal{R}_{ik}})\end{aligned} \tag{4.2.9}$$

After reduction, we can see that $\Delta\mathcal{D}_k$ is linearly related to X_k. Algorithm 1 provides the X_k that will yield the minimum $\Delta\mathcal{D}_k$ when it is bounded by [0,1]. This does not perfectly align with the problem statement where partial processing ratios are available, but this is true for minimizing the $\Delta\mathcal{D}_k$, which means that the GA is a more accurate solution to the problem. This is a small sacrifice for achieving much faster calculation times.

By performing a derivative of (4.2.9), the $\Delta\mathcal{D}_k{}'$ is defined as:

$$\Delta\mathcal{D}_k{}' = \frac{\mathcal{I}_k^u}{\mu_k} + \frac{\mathcal{H}_k * \mathcal{I}_k^u * \rho}{\mathcal{R}_{ik}} - \frac{\mathcal{H}_k * \mathcal{I}_k^u}{\mathcal{R}_{ik}} \tag{4.2.10}$$

We use Algorithm (1) to have each node calculate their own X_k value within a few cycles. This new algorithm is called the Disaster Area Adaptive Delay Minimization (DAADM)

Algorithm.

Algorithm 1: Disaster Area Adaptive Delay Minimization Algorithm

```
Data: $\Delta\mathcal{D}'_k$: objective function;$\mathcal{H}_k$: hops;$\rho$: compression ratio;$\mu$: processing rate;
      $\mathcal{R}_{ik}$: communication rate; $\mathcal{I}^u_k$: unprocessed input in node $k$;
Result: $X_k$, the processing ratio for the current node;
1 initialization;
2 while True do
3     calculate $\Delta\mathcal{D}'_k$ using equation (4.2.10);
4     if $\Delta\mathcal{D}'_k < 0$ then
5         $X_k \leftarrow 0$;
6     else
7         $X_k \leftarrow 1$;
8     end
9 end
```

The purpose of the algorithm is to calculate an ideal set $\mathbb{X}$ that will yield the minimal overall system delay, while keeping the algorithm simple enough to be useful for decision making on a fog node. The two aforementioned equations should be calculable with only a few cycles on the processor, which is orders of magnitude faster than a Genetic Algorithm.

To make sure that the complexity matches up with our goal, we perform some basic Big-O analysis. We will do the complexity analysis from the view of whatever is actually performing the calculation of the algorithms. The Fog-based, Cloud-based, and the proposed DAADM solutions calculate or set X_k on a node-by-node basis, but the GA solution must be calculated as a whole system. The Fog-based and the Cloud-based solutions preset X_k to 0 or 1, and require no calculations at all. Therefore, they are independent of the network size, which is the problem size 'n'. The complexity analysis of the Fog-based and Cloud-based solutions are both O(1). The GA solution from chapter 4 is run multiple times based on the network size. First, the length of a chromosome is n*8 (one byte per node). This alone would not affect the complexity, but the amount of chromosomes in the pool must be increased so that the crossover and mutation applications can have enough genetic diversity to be effective. Additionally, each gene needs to be run individually, and accounts for calculations of the whole network. Therefore, the GA solution has a complexity of O(n^2). The proposed DAADM algorithm is shown in algorithm 1, and is run one time which calculates $\Delta\mathcal{D}_k{}'$ once. The calculation of $\Delta\mathcal{D}_k{}'$ is unrelated to the network size, thus it is O(1), making the DAADM algorithm have a complexity of O(1).

4.3 Simulations and Performance Evaluation

4.3.1 Evaluation Methodology

In order to evaluate our DAADM algorithm, we tested it in a simulator, and compared it to conventional methods and a genetic algorithm (GA). The first conventional method is one where all of the processing is done in the cloud [12], and each node just forwards the data. The second conventional method is the one where all of the data is processed entirely in the fog, and the results are sent to the cloud. The genetic algorithm is run exhaustively and represents the baseline solution. The biggest problem with the GA is its run time.

The mathematical models and computations were all done in MATLAB. We test each system across a variety of parameters such as: network size, R_{ik}, and μ_k. For network size, 3 networks were randomly generated with a random number of nodes, which were randomly scattered around the map, and links were create. The 3 random sizes ended up being 11, 35, and 154 to represent the small, medium and large networks, respectively. All networks were formed using a tree topology, witch each node having no more than 6 child nodes. The largest system is 50% larger than the network coverage of the island of Okinawa in Japan which has 105 base stations [44]. The other network constraints were taken to match realistic cases, which were surveyed from other papers. The raw data input values varied randomly for each node. The communication speed was assumed to match 5G speeds, which was 20Gbps according to [45]. A GPU-based processor for wireless networks had a demonstrated performance capability of 4Gbps [46] This number is expected to increase rapidly as time goes on, especially as the next generation of computation-based GPUs was just announced [53]. The data compression ratio varies greatly between data processing and data compression algorithms. Authors in [47] estimate that a one third ratio is possible for LTE data compression. Most data processing algorithms are able to almost completely reduce the data down to a fraction of a percent. The previously mentioned spatial big data processing algorithm is capable of a data compression ratio of 0.01 from our tests. For each of the tests, we held all values at their default, except for the variable being tested. All tests comparing the different data processing ratio algorithms used the same network map and data input values.

4.3.2 Evaluation Across with Varied Bandwidth Constraints

Next, we need to determine how successful the adaptive algorithm is at achieving the minimum delays across the different bandwidth parameters. Figure 4.4 and 4.5 show the response that the delays have to changes in the processing speed on the large and small systems, respectively. Figure 4.6 and 4.7 show the response that the delay has to changes in the transmission rate for the large and small systems, respectively. In order to perform the tests, the network constraints that were not being tested were set to the default values, which correspond to the surveyed values. In the following tests, we only show the results for the large and the small system because the largest system is the one that is most likely to react to small changes in bandwidth, while the small network has the least sensitivity.

Figure 4.4 shows that all networks perform better as the processing capabilities are increased except for the cloud-based solution. One particular area to notice is the transition of the ideal case between the cloud and the fog. The GA is very apt at finding specific cases and can find the perfect combination of nodes that will yield the best possible solution. The adaptive algorithm is able to switch between the two main schemes, one node at a time, but this is handled less gracefully. As we shift to the small system in Fig. 4.5, we can see that the DAADM handles the transition much better, and is able to perform very similarly to the GA.

As expected, all of the systems other than the fog-based one perform better as the communication speeds are increased. The X_k values for the GA and the DAADM shift closer to 0 as the R_{ik} increases, which suggests that if the transmission speeds can be significantly improved in the future, then cloud computing will once again become the best solution. Unfortunately, the opposite seems to be true, and the computation speeds are rapidly increasing, while our wireless communication speeds are lagging behind. In Fig. 4.7, we can see that the DAADM algorithm has a much better transition than in the

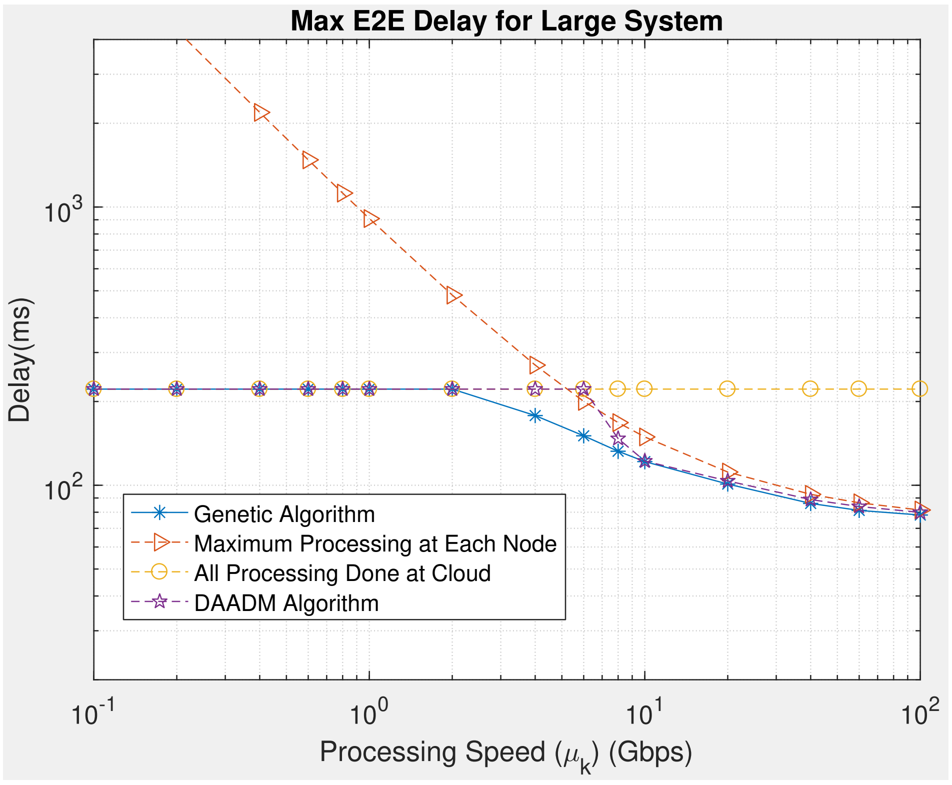

Figure 4.4: Delay results while varying μ_k values on a large network

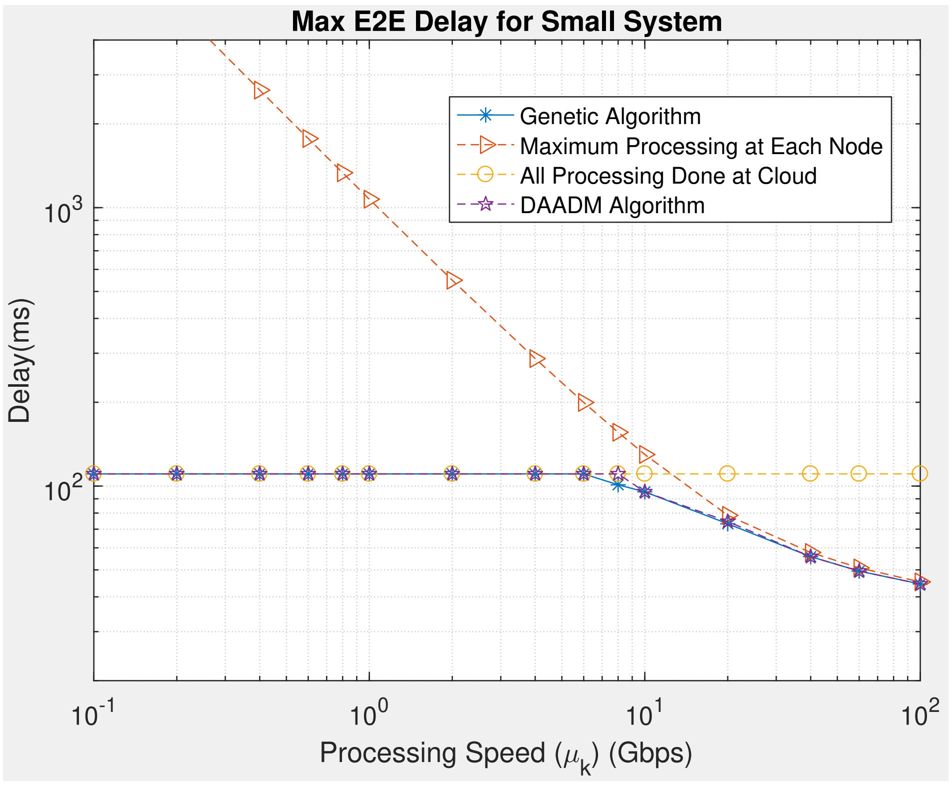

Figure 4.5: Delay results while varying μ_k values on a small network

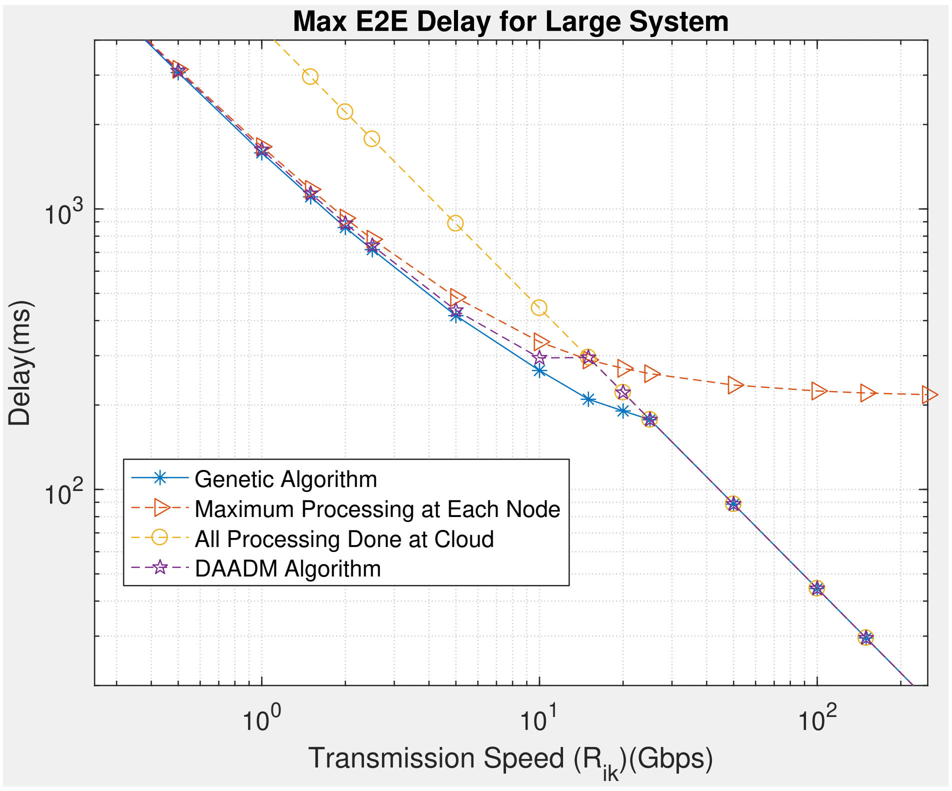

Figure 4.6: Delay results while varying $\mathcal{R}_{ik}$ values on a large network

large system (Fig. 4.6).

4.3.3 Time Complexity Evaluation

As the final and most important test, we needed to confirm that the complexity of our algorithm could be handled in an appropriate time frame and with appropriate accuracy and error. We measured the accuracy as $\sum_{i=1}^{n} Ideal/(Achieved \times n)$, where n is the number of data points collected. For both the accuracy and the percent error, we assume that the genetic algorithm was able to find the ideal result. We understand that increasing the number of data points, and the data point selection itself will affect the results of these two values, but they do help to give an overall idea of how the different algorithms compare overall. The accuracy represents how often the result aligns with the ideal, so higher is better (max of 100%). The percent error more accurately represents how far off a target can be from the ideal result, and so lower is better.

Figure 4.8 shows the results of our experimentations, and how long each algorithm

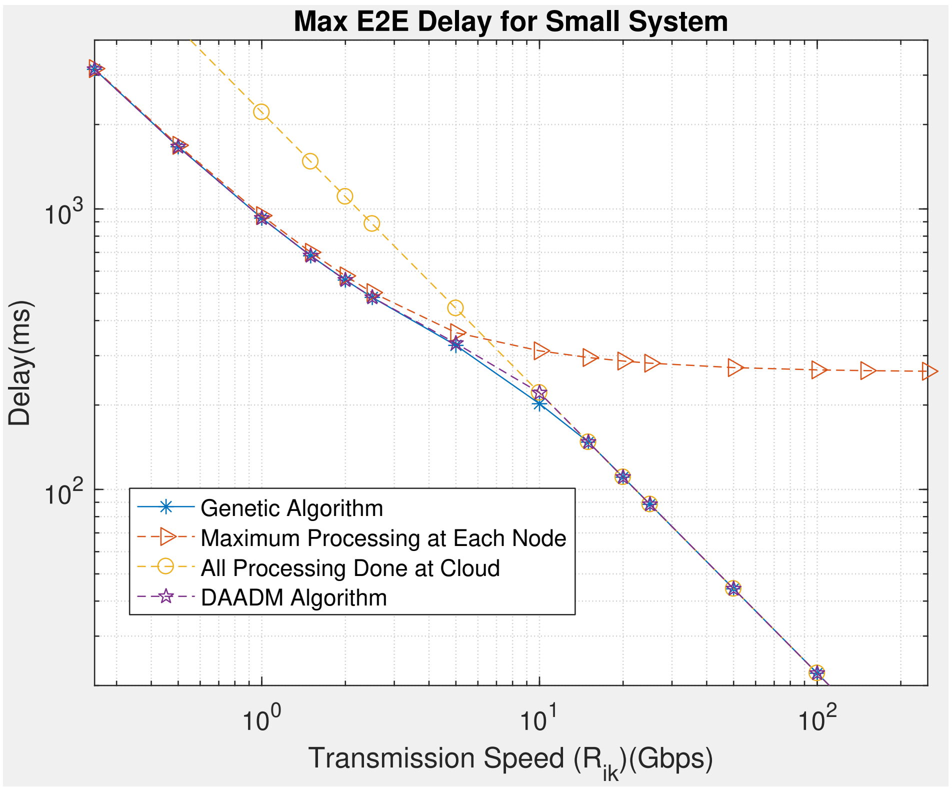

Figure 4.7: Delay results while varying $\mathcal{R}_{ik}$ values on a small network

took to calculate their respective results. The cloud and fog based solutions took exactly one cycle to assign either fully unprocessed or fully processed, respectively. The proposed DAADM took 3 cycles, which is still fast enough for a real-time system. However, the GA required over 8000 cycles per node (8.4s). While different processors will yield different times per cycle, these results should accurately approximate the ratio of the processing times between the different solutions.

As far as accuracy and error go, the GA was assumed to be the ideal solution, and so it had 100% accuracy, and 0% error. Because the cloud and fog solutions did not adapt well, they both had accuracies of around 65% of the time, but would have even worse results if we took data points from a wider range. This is because as the data points expand in either direction, either the fog or the cloud solution will have points that miss the ideal value. The percent error shows that the fog solution tends to miss the goal by a larger amount, with a percent error greater than 160%. The DAADM algorithm, which only took 3 cycles to calculate, had 95.2% accuracy and had a percent error of 5.7%. Additionally, taking more data points into consideration would improve the results of the DAADM. These results are promising for the use of algorithms such as DAADM to determine proper data processing ratios at each node.

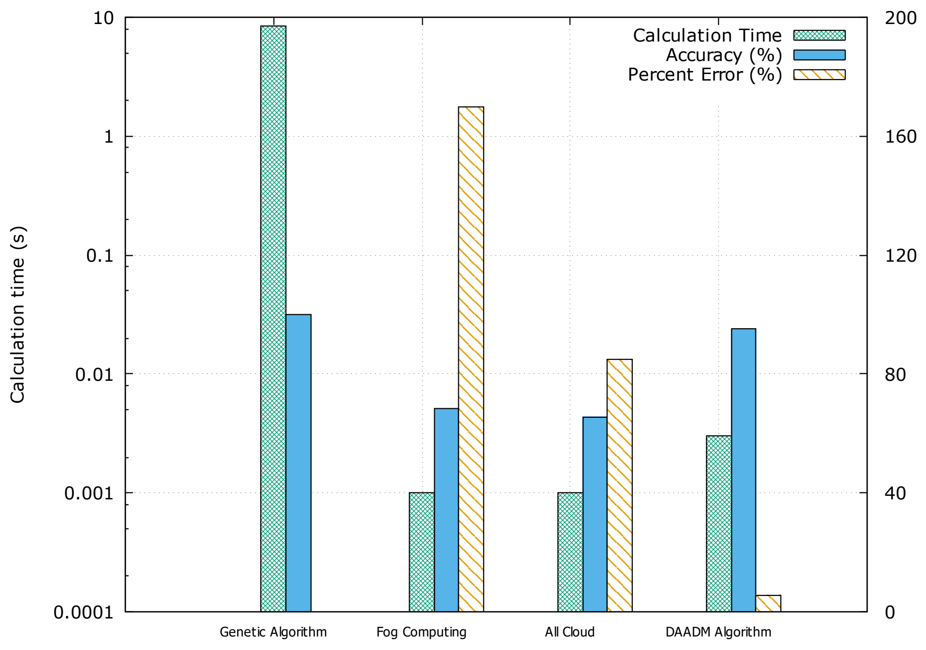

Figure 4.8: Time consumed by calculating, Accuracy, and Percent error of each allocation schemes of a large network

As we make the network slightly smaller, it reduces the runtime of the GA because it required a smaller gene pool. The results are still approximately 1.4 seconds, which is too slow to be used in real-time. It's also important to notice that for this particular network size and configuration that the percent error of the Fog-computing solution exceeded 600%. This likely happens on a case by case basis and does not reflect the efficacy of the algorithm at all times, but does show how the conventional algorithms can be susceptible to network structures. Again the DAADM algorithm solved the systems with a reasonable time and accurate results.

On the small-sized network, we can see that the cloud-computing solution is becoming more feasible, but still has a fairly low accuracy of 85%, while the DAADM has 99% accuracy. As the networks get smaller, the DAADM seems to have even less error, with only 0.8%. Then when we compare the calculation times, the DAADM has a 235x reduction compared to the GA.

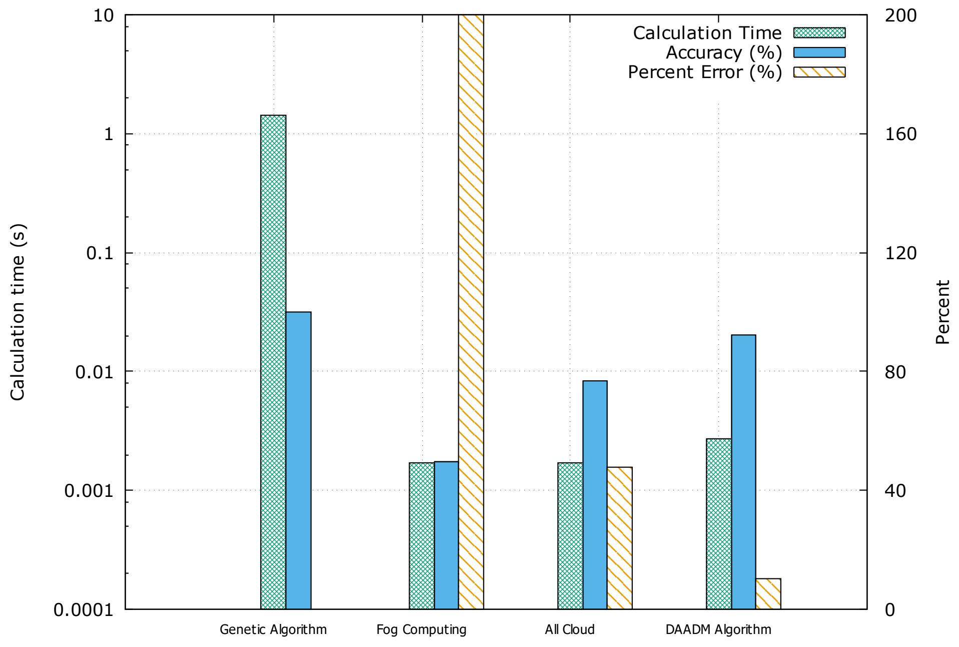

Figure 4.9: Time consumed by calculating, Accuracy, and Percent error of each allocation schemes of a medium network

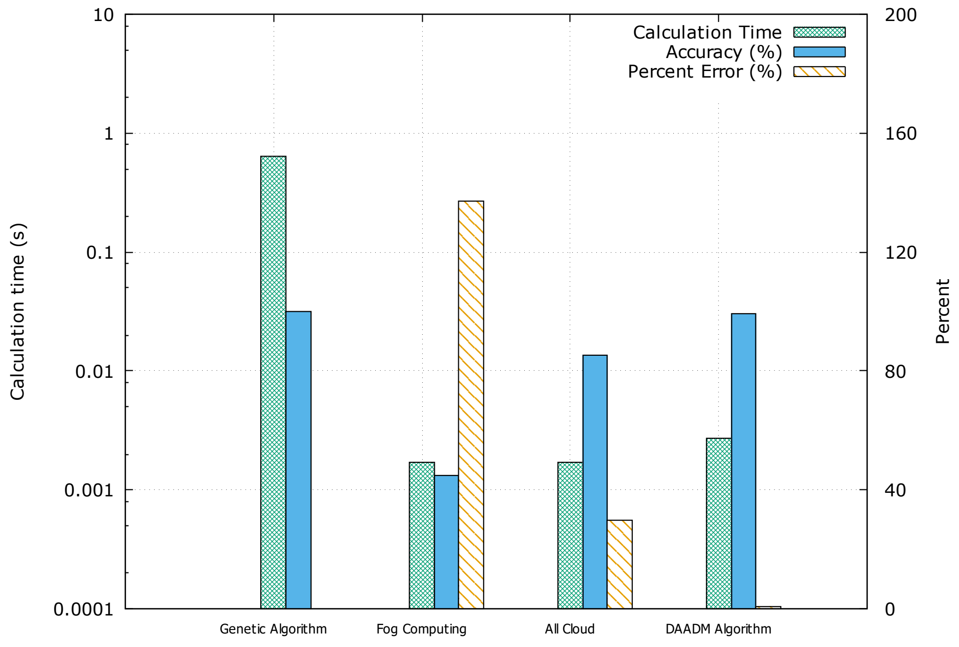

Figure 4.10: Time consumed by calculating, Accuracy, and Percent error of each allocation schemes of a small network

4.4 Chapter Summary

This chapter presented delay models for the various delays of big data applications seen in a WNDA. The models account for the time spent on processing as well as the time spent on transmitting the data at each of the nodes. After collecting some realistic network parameters, we simulated our proposed algorithm on various network sizes, and compared it to a GA and the two most common conventional methods. The DAADM was proposed to significantly reduce the calculation time that the genetic algorithm has, and still maintain better results than the conventional methods.

Evaluation results show that for the realistic parameters, the DAADM algorithm could be solved with minimal processing, enough to be considered a real-time algorithm. This means that it can be run constantly, and the results can adapt as the network does. With our researched values, for conventional algorithms, the cloud system was closest to the GA, but the DAADM algorithm matched the curve shape of the GA. However, as μ_k was increased, the maximum processing algorithm got much stronger, and the DAADM and GA adjusted accordingly.

The proposed DAADM solution shows that the appropriate X_k value is greatly determined by the processing speed, communication speed, and the number of hops that the current node is away from node j. There is a tradeoff between the time for transmitting raw data versus processing at the fog node, which is only amplified the further away a node is from the cloud. While the ρ value did affect the delays, the trend between different solutions was not greatly affected once ρ was small enough. This means that a smaller ρ value did improve the cloud, DAADM and GA's delays, but affected them all in a similar manner as long as ρ was not too large. Additionally, we saw the best results on a small network, which is more likely to be the case for an MDRU deployment as it would cover a small and dense area, not a county or something larger.

Chapter 5

Base Station Allocation for Users with Overlapping Coverage

5.1 Introduction

After major disasters, temporary deployable cellular networks are often used to construct an emergency communication network. These networks do not have the same level of performance as a typical fog or cloud network, and in order to service the users in a way that is sustainable, utilization of optimization algorithms must be considered. We proposed using a genetic algorithm (GA) to optimally allocate these users in the overlapping areas to a base station so that the system could provide an improved user experience. We tested our proposed algorithm against a greedy algorithm, a random algorithm, and allocating users to the closest MBS. The greedy algorithm outperformed the two other baseline algorithms, but the proposed algorithm was able to reduce the average and worst-case delay of the system by 80% compared to the greedy algorithm. The genetic algorithm had completely settled by 150 generations. This algorithm provided such an advantage that it warrants deeper study.

The main contributions of this paper are summarized as follows:

- We study a delay optimization problem under the MBS-fog-based wireless network model for big data collection and analysis. The network model problem mainly consists of processing delay, and transmission delay.

- We propose an overlapping wireless network structure which have different communication rate between users and MBSs. The optimization objective is to minimize the average delay for achieving the higher performance for all users.
- To solve this problem, we verify our framework through a genetic algorithm. Simulation results show that our proposed wireless network structure can achieve high-performance results.

5.2 System Model and Problem Formulation

In this section, we introduce the system model, the processing and transmission delay model. The average latency model will also be presented. We give the list all the major variables in Table 5.1.

5.2.1 System Model

Before presenting our proposal in detail, we will first introduce the system structure for computing the delay under our considered scenario.

Fig.5.1 shows a wirelessly networked disaster area (WNDA) fog based network. The movable base stations (MBSs) are deployed to the disaster area to rebuild communication network. Users are randomly distributed in the overlapping areas and non-overlapping areas. Refugees can upload data through the MBSs to the cloud.

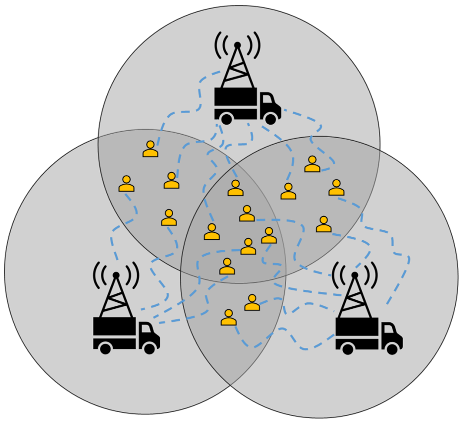

Figure 5.1: Structure of the Network in an Overlapped Area

We consider each MBS as having its own communication and computation capabilities. Users access the MBS and upload a large amount of data. We use γ_c to denote the data size collected by node c. ρ denote the compression ratio for spatial data processing algorithm, $0 < \rho \leqslant 1$. Meanwhile, the processing and communication rate at node c denote as μ_c, r_c, respectively.

Based on the spatial big data processing algorithms, we consider a system where all MBSs send data to a node c_e which is located at the edge of the disaster area. We assume the wired connected between c_e and the cloud is still intact, thus, c_e is able to transfer data rapidly and efficiently. In this paper, we introduce and discuss a single c_e node situation, but a multiple c_e node situation can be modeled and solved similarly.

According to the different distances between individual users and MBSs, the communication rate of each link is also different. Therefore, we consider that a communication rate follows Shannon's channel capacity. During the processing and transmission process, the average queueing time has been considered in the networking model.

For typical big data analysis, the algorithm cannot start until all the data has been gathered by the last node and has been sent to the cloud. The time between the data starting to transfer and the last bit of data being received by the cloud for each link called the path delay. The path between the starting node and the edge node c_e is denoted by $\mathbb{P}_{(i)}$. We attempt to minimize the average delay of all users in the system by properly allocating base stations in the overlapping areas.

5.2.2 Processing Delay

In this section, we present the processing delay model. We introduce the computation model for output data after processing through the equation below. After processing data in node c, the data can be separated into processed data and unprocessed data. We assume that the processed data cannot be processed second , which is why the two data types must be accounted for separately.

The raw cellular data collected by node c is defined by γ_c, where γ_c follows the Normal Density Function. We use $\hat{\alpha}$ to denote the mean value of γ_c and σ^2 denotes the variance of γ_c. Therefore, the probability value of γ_c can be given by:

Table 5.1: System Variables

Design Parameters	Descriptions
$\mathbb{N}$	set of all nodes
c	current node
c'	child node
γ_c	raw cellular data size
ρ	compression ratio
$\hat{\alpha}$	mean value of γ
σ^2	variance of γ
x_c	data processing ratio at node c
μ_c	processing rate
r_c	communication rate
Φ_c^P	processing arrival rate at node c
Φ_c^T	transmission arrival rate at node c
O_c^u	unprocessed output data from node c
O_c^p	processed output data from node c
$\hat{\mathrm{W}}_c^{p(q)}$	average queuing waiting time of processing data
$\hat{\mathrm{W}}_c^{T(q)}$	average queuing waiting time of transmitting data
D_c^p	processing delay at node c
D_c^T	transmission delay at node c
D_c^{path}	totally delay for initial node c on path $\mathbb{P}_{(i)}$
$\mathrm{D}_{average}$	average delay

$$P\,(\gamma_c \mid \hat{\alpha},\ \sigma^2)\ =\ \frac{1}{\sqrt{2\pi\sigma^2}}\ e^{-\frac{(\gamma_c-\hat{\alpha})^2}{2\sigma^2}} \tag{5.2.1}$$

Then, the data processing ratio at node c is represented by x_c, we group all child nodes of c in the set $\mathbb{C}_c$, and all nodes in the set $\mathbb{N}$. The unprocessed output data from node c can be calculated as:

$$\mathrm{O}_c^u\ =\ (\ \gamma_c\ +\ \sum_{c'\in\mathbb{C}_c}^{\mathbb{N}} \mathrm{O}_{c'}^u\ )\ \cdot\ (\ 1-\mathrm{x}_c\ ),\ \forall c\in\mathbb{N} \tag{5.2.2}$$

Next, we use ρ to denote the data compression ratio, since the total output data after processing consists of two parts, we then can compute the processed output data from node c as follows:

$$\mathrm{O}_c^p\ =\ (\ \gamma_c\ +\ \sum_{c'\in\mathbb{C}_c}^{\mathbb{N}} \mathrm{O}_{c'}^u\ )\ \cdot\ \rho\ \cdot \mathrm{x}_c\ +\ \sum_{c'\in\mathbb{C}_c}^{\mathbb{N}} \mathrm{O}_{c'}^p,\ \forall c\in\mathbb{N} \tag{5.2.3}$$

For the processing delay at each node c, we also consider the time spent waiting to process the data. The data arrival rate follows a Poisson process which is denoted by Φ, then the processing arrival rate at each node can be expressed as Φ_c^p.

We consider the processing order of data sent to node c by users follows a M/M/1 queue model. The processing rate is denoted as μ_c. Thus, the average queueing time for processing data can be calculated as below:

$$\hat{\mathcal{W}}_c^{p(q)} = \frac{\Phi_c^p}{\mu_c(\mu_c - \Phi_c^p)}, \ \forall c \in \mathbb{N} \tag{5.2.4}$$

Therefore, we can compute the processing delay at node c as follows:

$$\mathrm{D}_c^p = \frac{(\gamma_c + \sum\limits_{c' \in \mathbb{C}_c}^{\mathbb{N}} \mathrm{O}_{c'}^u) \cdot \mathrm{x}_c}{\mu_c} + \hat{\mathcal{W}}_c^{p(q)}, \ \forall c \in \mathbb{N} \tag{5.2.5}$$

5.2.3 Transmission Delay

The transmission model for a wireless network area is presented as follows. We assume r_c denote the set of communication rates for each channel. W means the channel bandwidth. p_c denote the MBS c's transmission power, h represents the channel gain for the link between the MBS and the mobile user. $(\lambda/4\pi d)^2$ denotes the loss of signal power for each channel. The distance between user i and MBS c is defined as $\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}$. $\mathcal{N}_0$ denotes the signal noise at the receiving end, and is composed of both the Gaussian noise and the interference.

$$r_c = \mathrm{W} \log_2 (1 + \frac{p_c h(\lambda/4d\pi)^2}{\mathcal{N}_0}) \tag{5.2.6}$$

We then consider that transmitting data between nodes or from users to the MBS follows a M/M/1 queue model. Φ_c^T denote the arrival rate of transmission data at node c. Thus, we can compute the average queueing time for transmitting data from node c as:

$$\hat{\mathcal{W}}_c^{T(q)} = \frac{\Phi_c^T}{r_c(r_c - \Phi_c^T)}, \ \forall c \in \mathbb{N} \tag{5.2.7}$$

Therefore, we can calculate the transmission delay as:

$$\begin{aligned} \mathrm{D}_c^T &= \frac{\mathrm{O}_c^u + \mathrm{O}_c^p}{r_c} + \hat{\mathcal{W}}_c^{T(q)}, \ \forall c \in \mathbb{N} \\ \text{s.t. } & (5.2.2), (5.2.3), (5.2.6) \text{ and } (5.2.7) \end{aligned} \tag{5.2.8}$$

5.2.4 Delay Model

Next, we will introduce the delay model. We can compute the relevant delay for any node c following the path $\mathbb{P}_{(c)}$ in terms of processing delay model and transmission delay as:

$$\begin{aligned} \mathrm{D}_c^{Path} &= \mathrm{D}_c^p + \sum_{c \in \mathbb{P}_{(i)}} \mathrm{D}_c^T, \ \forall c \in \mathbb{N} \\ \text{s.t. } & (5.2.4), (5.2.5), (5.2.6), (5.2.7) \text{ and } (5.2.8) \end{aligned} \tag{5.2.9}$$

According to the communication model of wireless networks, we are attempting to achieve the best performance for all users, thus by (5.2.9), the average latency of the network can be calculated as following:

$$\mathrm{D}_{average} = \frac{\sum_{c \in \mathbb{N}} \mathrm{D}_c^{Path}}{|\mathbb{N}|} \tag{5.2.10}$$

Wirelessly Networked Disaster Area Base Station Allocation for Minimal Delay (WNDA-BAMD): In overlapping wireless network areas, the goal is to achieve the best performance for all users by allocating them in a manner that minimizes the average delay of all users, which is shown in Eq. 5.2.10.

5.3 Proposed Solution

The goal of our system is to realize the ideal allocation of users to yield the minimum delay for the system. To achieve this, we have implemented a genetic algorithm [42] (GA), which has been applied to a wide variety of problems and is able to solve many more in the future. Each simulation starts with an initial population, called generation one. Then the algorithm selects the ideal candidates, performs genetic crossover and then mutation, creating the next generation. The algorithm performs the selection, crossover and mutation 50 times. The pseudo code for our GA can be seen in Algorithm 2.

Algorithm 2: Genetic Algorithm

```
1  $Gen \leftarrow 1$
2  $Pop \leftarrow InitialPop()$
3  while $Gen < MaxGenerationCount$ do
4      $F \leftarrow CalcFitness(Pop)$
5      $P_S \leftarrow Selection(P, F)$
6      $P_C \leftarrow Crossover(P_S)$
7      $P_M \leftarrow Mutation(P_C)$
8      $Pop \leftarrow P_M$
9      $Gen \leftarrow Gen + 1$
10 end
11 $IdealGene \leftarrow SelectBest(Pop)$
```

We compared our proposed system to 3 conventional methods: nearest, greedy, and random. The random algorithm randomly chose one of the MBSs whose area covers the user in the overlapping area. The nearest algorithm calculates the physical distance between the user and any MBS which cover that position, and allocates that user to the MBS that has the smallest distance. The greedy algorithm chooses the node, which will give that individual user the least amount of communication delay.

The modeling and calculations were all coded in MATLAB. We first attempted to determine the capabilities of each algorithm across different network sizes. In order to let the evaluation match with real cases, we surveyed some realistic values for $\mathcal{R}$, μ_k, and κ.

Most 5G antennas currently in development suggest that the wireless speeds will reach 20Gbps [45]. Hinitt et al. [46] have estimated that a GPU based processor for wireless networks is capable of processing speeds of at least 4Gbps. Guo et al. [47] suggest that LTE compression is capable of a one-third ratio, so we adopted this value for evaluation.

The amount of processing that a standard computer can handle is still scaling very rapidly, so we expected this number to increase rapidly as well. All of these values, and other ones used for testing the scalability of the algorithms can be seen in Table 5.1.

All systems were evaluated according to their average and worst-case delays. The average delay is more applicable to IoT device and everyday use, whereas many big data processing algorithms are bottle-necked by what time it takes for the last bit of data to be aggregated, and are thus better represented by the worst-case delay. The worst-case delays are expected to have less smooth curves because of nature of how easily one user can affect the results.

5.4 Simulations and Performance Evaluation

5.4.1 Evaluation Across Processing Parameters

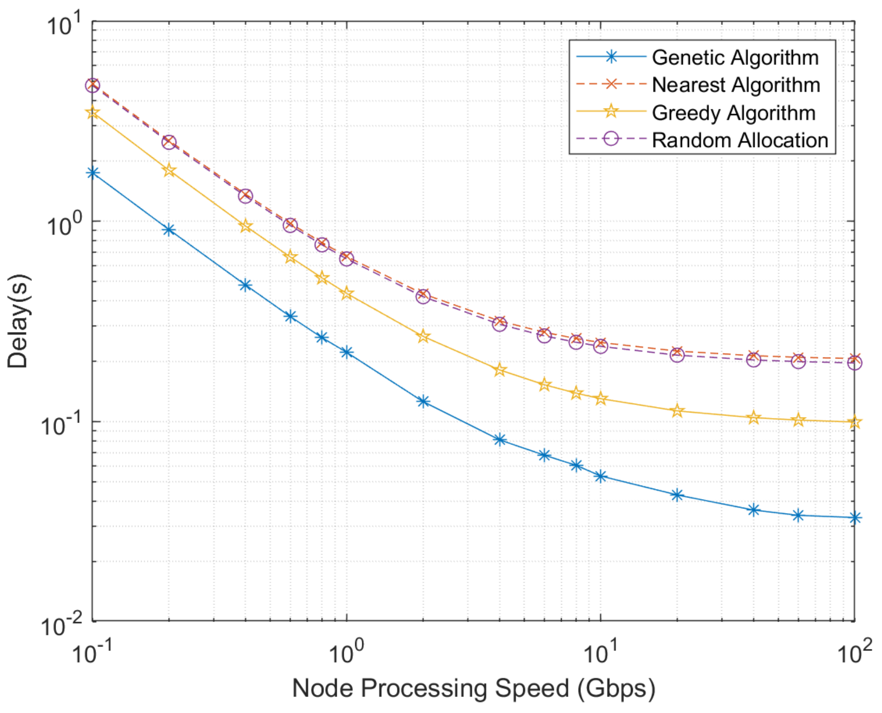

Figure 5.2: Average Delay Results for Variations in Node Processing Speed

The results in Figures 5.2 and 5.3 show how a Fog system reacts to changes in the processing speeds of the fog nodes. As expected, all systems decrease their Average

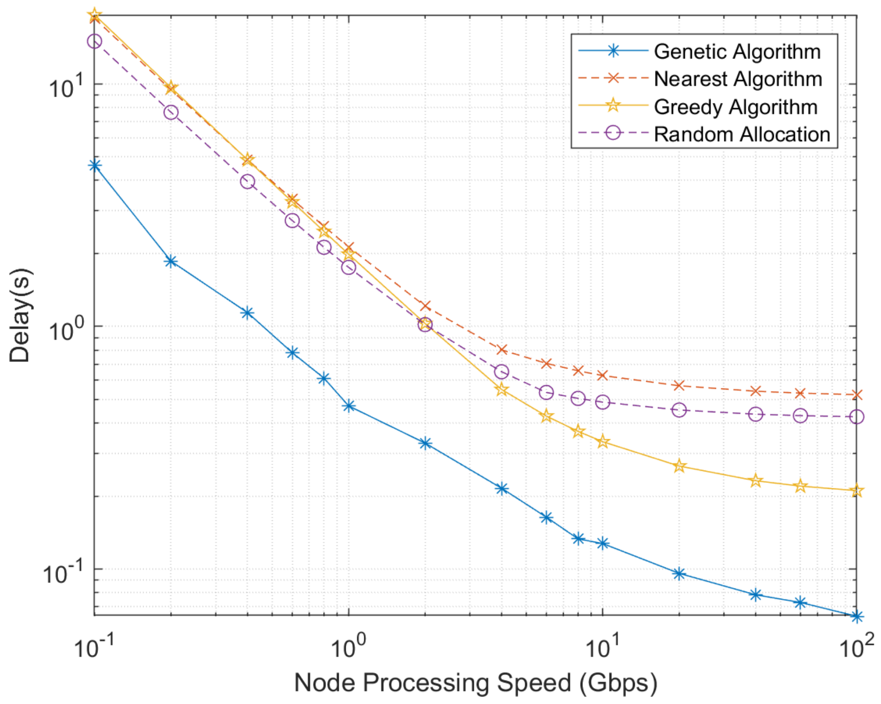

Figure 5.3: Worst-Case Delay Results for Variations in Node Processing Speed

Delay as the processing capabilities are increased, until each of them plateaus when the communication delay becomes the bottle neck. The average delay of the GA seems to start to plateau at the limit of our simulation, but the worst-case delay still appears to be improving. It is worth noting that the limits of these simulations still far exceed our current technology, but it is nice to know that this algorithm will still be applicable as technologies improve. In terms of average delay, the algorithm which allocates users to their closest MBS works approximately as well as randomly allocating users, but the worst-case user ends up being slightly worse when only considering position. The Greedy algorithm, which attempts to connect each user to the base station which can offer that user the best connection has an improved average latency over all node processing speeds, but the worst-case delay for low processing speeds can be even worse than the nearest node as many users in the same overlapping area will all choose the same node to be processed at, increasing the queuing delay at that node. The GA has a curve that is vastly below that of the other algorithms and often results in its delay having a 70% or greater reduction in latency compared to the next best algorithm.

The results for changes in the data compression ratio can be seen in Figures 5.4 and

5.5. Values of 10^{-1} and less are examples of rates we can expect from efficient processing algorithms. 0.25 to 0.5 are values we would expect to see from compression algorithms. Values greater than 1 show examples where processing increases the size of the data, which is very inefficient and rare done in fog systems, but is still good to look at for evaluation.

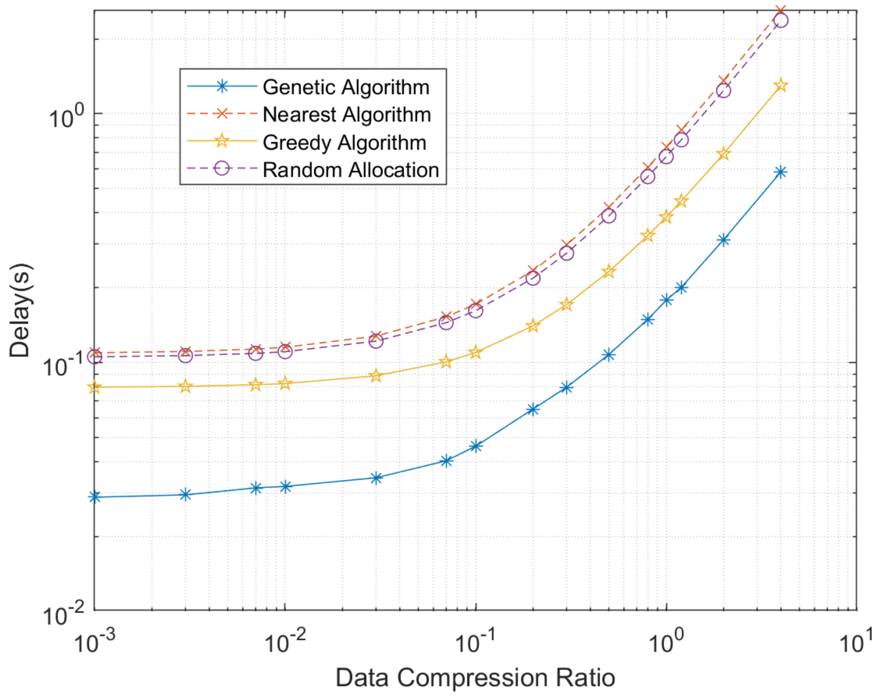

Figure 5.4: Average Delay Results for Variations in the Data Compression Ratio

The random and nearest algorithms performed similarly in terms of average latency, but the random algorithm was the most successful conventional algorithm for the worst-case delay because it balances the load evenly across the network. This does not hold true when running an inefficient BDP algorithm on the fog nodes, which leads the greedy algorithm to have the best worst-case delay. The Greedy algorithm was the most successful algorithm in terms of average latency, but was approximately 4x slower than the GA. The GA was approximately 3x better than the best conventional algorithm for all data compression ratios, in terms of worst-case and average delay, even when the best conventional method changes.

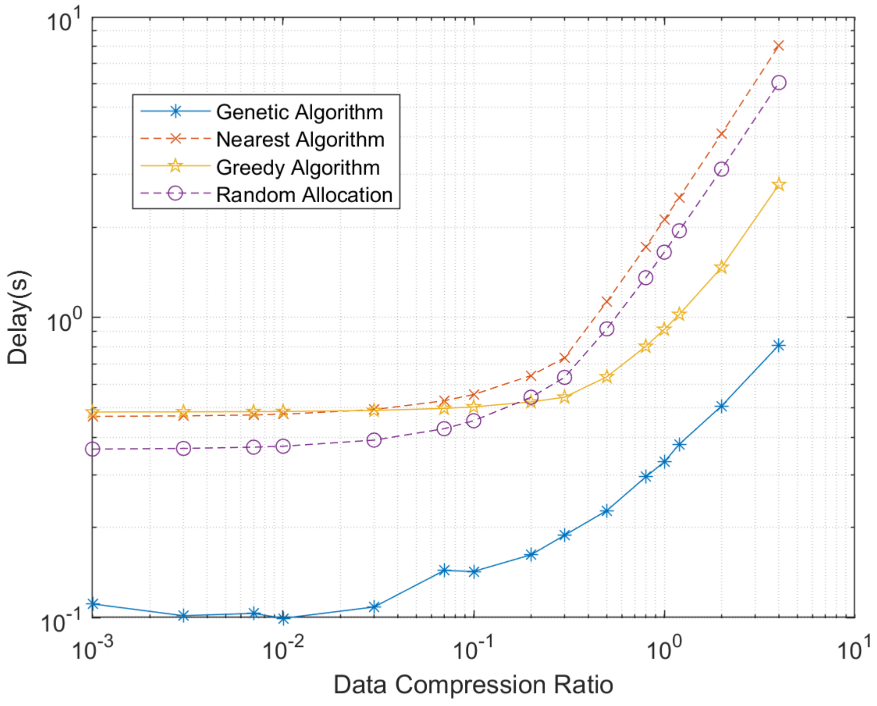

Figure 5.5: Worst-case Delay Results for Variations in the Data Compression Ratio

5.4.2 Evaluation Across various Channel Capacities

Perhaps the most important metric for this algorithm is its capability to adapt to varying communication rates. MBSs will likely soon be equipped with improved 5G antennas which have a higher channel capacity than current 4G antennas. This means that this technology will be significantly improved in the near future.

To show how our system reacts to changes in communication rates, we varied the channel capacities, and the results can be seen in figures 5.6 and 5.7. As we have seen in the past simulations, the average delay of the random and the nearest algorithms are similar, with the greedy algorithm giving a slight improvement and the GA gave a significant improvement. Overall the GA typically held a 4x improvement over the greedy algorithm, but this advantage took a dip around 20Gbps, where the advantage was only 2.8x. This still suggests that the GA is far superior to the deterministic methods. In terms of the worst-case latency, seen in figure 5.7, the GA maintains a 6x improvement over the greedy algorithm.

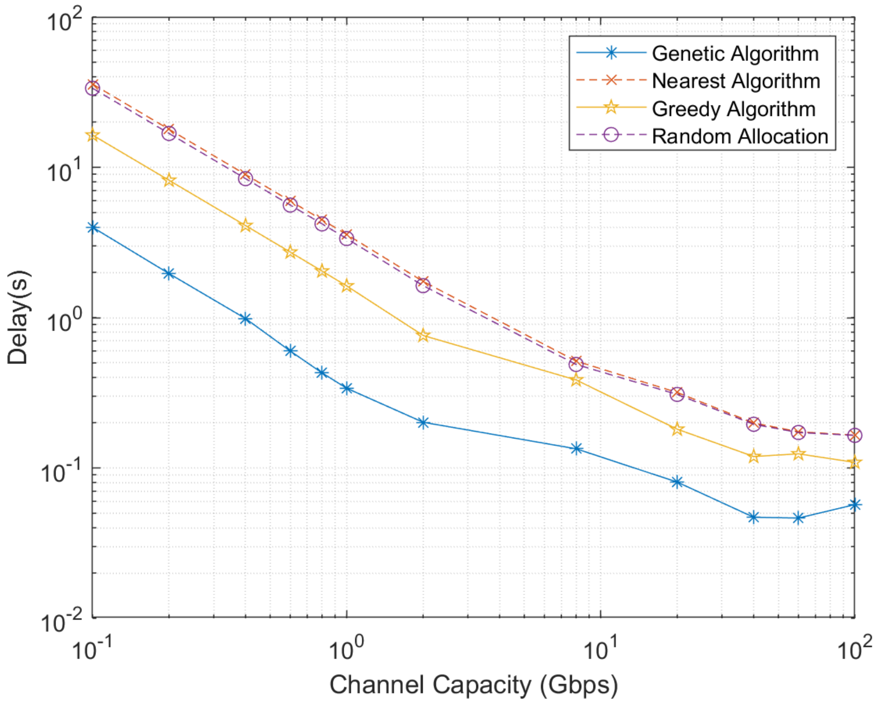

Figure 5.6: Average Delay Results for Variations in the Channel Capacity

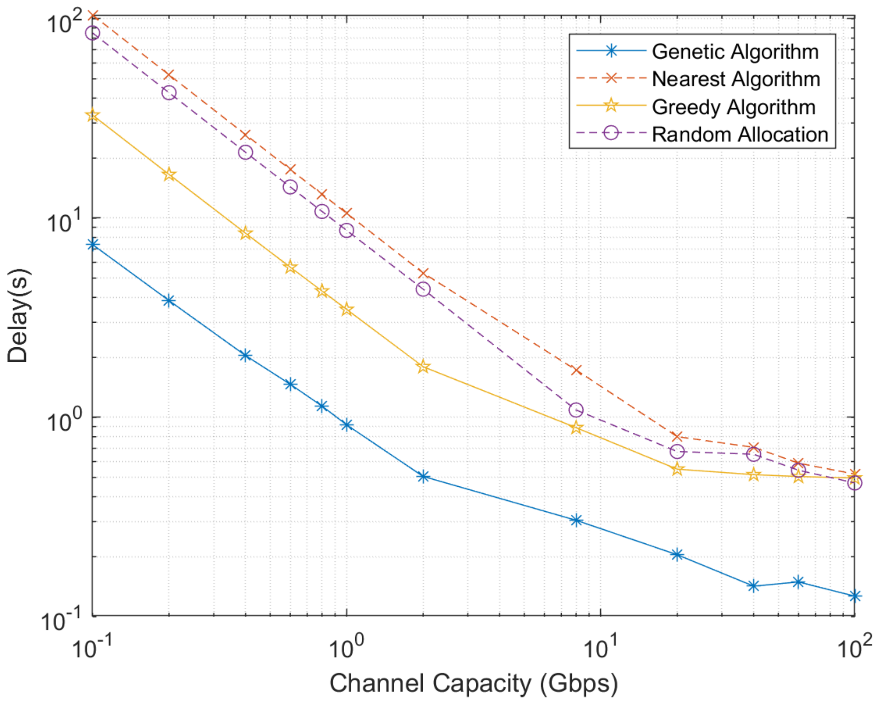

Figure 5.7: Worst-case Delay Results for Variations in the Channel Capacity

5.4.3 Cloud System Evaluation

Sometimes, MBS stations are set up to function as relay points, and then the MBS system functions like a cloud computing system. This is why it was important to verify the algorithms capabilities in a cloud network.

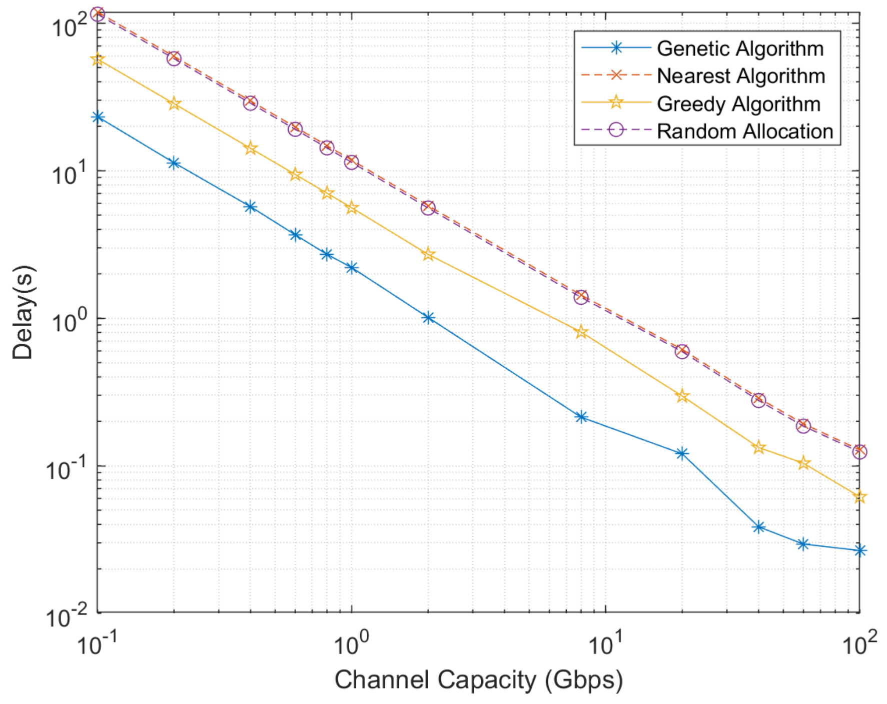

Figure 5.8: Cloud System Average Delay Results for Variations in the Channel Capacity

The results for varying the channel capacity in a cloud system can be seen in figures 5.8 and 5.9. For a cloud system, the results are much more linear due to the lack of a processing delay. The random and nearest algorithms follow similar speeds for the average delay, but the random algorithm performs slightly better in terms of worst-case latency. The greedy algorithm is always the second best for both the average and worst-case, with the GA performing approximately 2.5x better across all values.

5.5 Chapter Summary

This paper presented a genetic algorithm which was capable of solving for the ideal combination base station allocations to reduce the network's overall delay. Users who

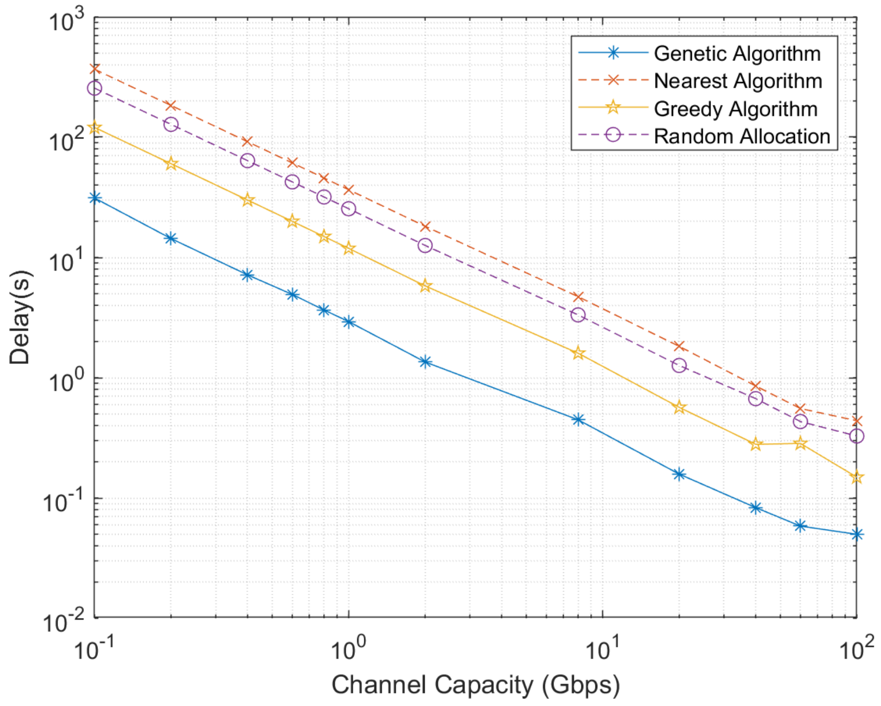

Figure 5.9: Cloud System Worst-case Delay Results for Variations in the Channel Capacity

were in areas that could be covered by two or more MBSs would be allocated in a way that was faster and improved the overall user experience for everyone in the coverage area.

The proposed algorithm provided the minimum worst-case and overall delay for the entire system across all network parameters. It was approximately at least 70% better than the next most efficient algorithm across all processing speeds. Across most of the Fog System evaluations, the GA maintained a 4x improvement in performance compared to the greedy algorithm, which was the best performing conventional algorithm. In a cloud system, it performed approximately 2x better than the greedy algorithm across all channel capacities.

Chapter 6

Conclusion and Discussion

This dissertation concludes with a summary of the main contributions and a discussion of future research. We create an overview of the results of each contribution. The future works will mostly consist of design considerations that were not implemented.

6.1 Contributions

This thesis presents four main contributions: (1) a set of models for fog computing on a wireless network; (2) a genetic algorithm to optimize the data processing ratio of each node; (3) a real-time algorithm to solve for a data processing ratio that will result in a reduced latency for the fog network; and (4) a genetic algorithm which can solve for the ideal mapping configuration of the network.

The first contribution of this paper was a set of models for fog computing on a wireless network. In chapter 4, we started with a simplistic model and used it to evaluate our algorithm. In chapter 5, we proved that the model was NP-Hard. In Chapter 6, the model was improved by accounting for queuing delay in the processor and in the communication networks. Additionally, we accounted for average delay

In chapter 4, we proposed a genetic algorithm which was able to solve for the ideal set of data processing ratios for each node to reduce the maximum delay that the system. We started with a random population, and run each of the genes through the models, and select the best results, and perform crossover and mutation with them. This is repeated for 80 generations, but a solution usually settled before 40 generations.

Chapter 5 presented a real-time algorithm that could solve the same problem as the GA from chapter 4. This new DAADM algorithm would account for the network structure and performance metrics. While this is running, it is assumed that the structure is not changing in real-time, but the performance of the links connected to the current node can change in real-time. This allows for the algorithm to adapt the processing ratio in real time, without receiving additional information from other nodes.

Chapter 6 attempted to solve a different problem. The models were shifted to find the average delay of each node in the system, which is more applicable to a herd-satisfaction model, and attempted to solve for the ideal mapping connections for each node. This was also handled by a GA, which functioned similarly to the one used in chapter 4.

6.2 Results Summary

Evaluation results show that for the realistic parameters, the genetic algorithm was able to solve the ideal data processing ratios after approximately 40 generations for the largest system. With our researched values, the cloud system was closest to the GA. However, as μ_k was increased, the maximum processing algorithm got much stronger. But, as the $\mathcal{R}$ was increased, the cloud computing solution was more successful. The current GA solution shows that the amount of processing that needs to be done by each node changes with the $\mathcal{R}$ and μ_k values greatly, and that with the standard test parameters, the $\mathbb{X}_k$ value was heavily based on the number of hops that the node was from the cloud. This is possibly based on a transmission-compression trade-off. Additionally, because the ρ value can vary so much between compression and data processing cases, the target application varies the results greatly.

Evaluation results show that for the realistic parameters, the DAADM algorithm could be solved with minimal processing, enough to be considered a real-time algorithm. This means that it can be run constantly, and the results can adapt as the network does. With our researched values, for conventional algorithms, the cloud system was closest to the GA, but the DAADM algorithm matched the curve shape of the GA. However, as μ_k was increased, the maximum processing algorithm got much stronger, and the DAADM and GA adjusted accordingly.

Evaluation results show that the GA was able to reduce the average network delay by approximately 40% with ideal connections. Further simulations showed that the GA's results had a worst-case delay that was approximately equivalent to the greedy algorithm's average delay. The actual run-time of the GA was quite small, because it only needed to run for a few generations. The results of the GA were resilient to changes in the processing speeds, transmission speeds, and data compression ratios.

6.3 Discussion

Even though GAs and the DAADM algorithm showed promising results, there are still some points that we would like to consider in future research.

As a future work, we plan to implement the DAADM algorithm on a real MBSs network system. Furthermore, we will determine the ideal paths for a set of coordinates for MBSs. We can also calculate the packet loss rate while we are generating the maps, because we can calculate the signal loss based on the paths and distances. Additionally, we would like to account for the signal strength of each connection. This can possibly affect the amount of times that the data needs to be resent, ultimately affecting the transmission speed of a link.

In order to improve the mapping scheme, we plan to propose a real-time algorithm which is able to replace the GA. The current algorithm does not run for a long time, approximately two seconds on a standard PC, but this could be taxing on a Fog network if it is to be continuously run so that the network could adapt to changes in real time. Assuming that the real-time algorithm is successful, we would like to test it on an MDRU network.

Bibliography

[1] *International Technology Roadmap For Semiconductors: Chapter: Interconnect*, 2011. [Online]. Available: www.itrs.net

[2] G. Ifrah and E. Harding, *The universal history of computing: From the abacus to the quantum computer*. John Wiley & Sons, Inc., 2001.

[3] "Babbage's analytical engine, 1834-1871. (trial model)." [Online]. Available: http://collection.sciencemuseum.org.uk/objects/co62245/babbages-analytical-engine-1834-1871-trial-model-analytical-engines

[4] A. W. Burks, "Electronic computing circuits of the eniac," *Proceedings of the IRE*, vol. 35, no. 8, pp. 756–767, Aug 1947.

[5] W. F. Brinkman, D. E. Haggan, and W. W. Troutman, "A history of the invention of the transistor and where it will lead us," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 12, pp. 1858–1865, 1997.

[6] J. S. Kilby, "Miniaturized electronic circuits," Jun. 23 1964, uS Patent 3,138,743.

[7] J. von Neumann, "First draft of a report on the edvac, contract no. w-670-ord-402 moore school of electrical engineering, univ. of penn., philadelphia. reprinted (in part) in randell, brian. 1982. origins of digital computers: Selected papers," 1945.

[8] "Replacing Silicon with Carbon Nanotubes: Why its Still Worth Considering," http://www.eeweb.com/blog/aaronfranklin, 2012, [Online; accessed 1-April-2015].

[9] A. Habibi, M. Arjomand, and H. Sarbazi-Azad, "Multicast-aware mapping algorithm for on-chip networks," in *2011 19th International Euromicro Conference on Parallel, Distributed and Network-Based Processing*. IEEE, 2011, pp. 455–462.

[10] M. "Meyer, Y. Okuyama, and A. B. Abdallah, “”microring fault-resilient photonic network-on-chip for reliable high-performance many-core systems”,” *"The Journal of Supercomputing"*, pp. "1–33", "2016". [Online]. Available: "http://dx.doi.org/10.1007/s11227-016-1846-0"

[11] P. Selinger, “A brief survey of quantum programming languages,” in *International Symposium on Functional and Logic Programming*. Springer, 2004, pp. 1–6.

[12] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, “A view of cloud computing,” *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010. [Online]. Available: http://doi.acm.org/10.1145/1721654.1721672

[13] P. Mell, T. Grance *et al.*, “The nist definition of cloud computing,” 2011.

[14] C. Newcombe, T. Rath, F. Zhang, B. Munteanu, M. Brooker, and M. Deardeuff, “How amazon web services uses formal methods,” *Communications of the ACM*, vol. 58, no. 4, pp. 66–73, 2015.

[15] M. S. Hajirahimova and A. S. Aliyeva, “About big data measurement methodologies and indicators,” *International Journal of Modern Education and Computer Science*, vol. 9, no. 10, p. 1, 2017.

[16] A. Katal, M. Wazid, and R. H. Goudar, “Big data: Issues, challenges, tools and good practices,” in *2013 Sixth International Conference on Contemporary Computing (IC3)*, Aug 2013, pp. 404–409.

[17] J. Wang, Y. Wu, N. Yen, S. Guo, and Z. Cheng, “Big data analytics for emergency communication networks: A survey,” *IEEE Communications Surveys Tutorials*, vol. 18, no. 3, pp. 1758–1778, thirdquarter 2016.

[18] M.-P. Kwan and D. M. Ransberger, “Lidar assisted emergency response: Detection of transport network obstructions caused by major disasters,” *Computers, Environment and Urban Systems*, vol. 34, no. 3, pp. 179–188, 2010.

[19] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. U. Khan, "The rise of "big data" on cloud computing: Review and open research issues," *Information systems*, vol. 47, pp. 98–115, 2015.

[20] V. N. Inukollu, S. Arsi, and S. R. Ravuri, "Security issues associated with big data in cloud computing," *International Journal of Network Security & Its Applications*, vol. 6, no. 3, p. 45, 2014.

[21] Y. Matsuoka, R. Sharan, M. D. Stefanski, and J. A. Ruff, "Intelligent controller providing time to target state," Jun. 24 2014, uS Patent 8,761,946.

[22] J. Xiao, "America's number 1 pressure cooker, multicooker," Mar 2019. [Online]. Available: https://instantpot.com/

[23] "Protection at every corner." [Online]. Available: https://ring.com/

[24] 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*, ser. MCC '12. New York, NY, USA: ACM, 2012, pp. 13–16. [Online]. Available: http://doi.acm.org/10.1145/2342509.2342513

[25] L. Gu, D. Zeng, S. Guo, A. Barnawi, and Y. Xiang, "Cost efficient resource management in fog computing supported medical cyber-physical system," *IEEE Transactions on Emerging Topics in Computing*, vol. 5, no. 1, pp. 108–119, Jan 2017.

[26] 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, no. 12, pp. 3702–3712, 2016.

[27] J. Oueis, E. C. Strinati, and S. Barbarossa, "The fog balancing: Load distribution for small cell cloud computing," in *Vehicular Technology Conference (VTC Spring), 2015 IEEE 81st*. IEEE, 2015, pp. 1–6.

[28] S. Clinch, J. Harkes, A. Friday, N. Davies, and M. Satyanarayanan, "How close is close enough? understanding the role of cloudlets in supporting display appropriation by mobile users," in *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on*. IEEE, 2012, pp. 122–127.

[29] M. Jia, J. Cao, and W. Liang, “Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks,” *IEEE Transactions on Cloud Computing*, 2015.

[30] L. Yang, J. Cao, Y. Yuan, T. Li, A. Han, and A. Chan, “A framework for partitioning and execution of data stream applications in mobile cloud computing,” *SIGMETRICS Perform. Eval. Rev.*, vol. 40, no. 4, pp. 23–32, Apr. 2013. [Online]. Available: http://doi.acm.org/10.1145/2479942.2479946

[31] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan, “A comprehensive survey on vehicular ad hoc network,” *Journal of network and computer applications*, vol. 37, pp. 380–392, 2014.

[32] X. Chen, “Decentralized computation offloading game for mobile cloud computing,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974–983, 2015.

[33] Y. Kikuchi and Y. Shibata, “Mobile cloud computing for distributed disaster information system in challenged communication environment,” in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops*, March 2015, pp. 512–517.

[34] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, “Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption,” *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1171–1181, Dec 2016.

[35] P. G. V. Naranjo, Z. Pooranian, M. Shojafar, M. Conti, and R. Buyya, “Focan: A fog-supported smart city network architecture for management of applications in the internet of everything environments,” *Journal of Parallel and Distributed Computing*, 2018.

[36] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, “Dynamic service migration in mobile edge-clouds,” in *2015 IFIP Networking Conference (IFIP Networking)*, May 2015, pp. 1–9.

[37] X. Chen, L. Jiao, W. Li, and X. Fu, “Efficient multi-user computation offloading for mobile-edge cloud computing,” *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, October 2016.

[38] L. Yang, J. Cao, S. Tang, T. Li, and A. T. S. Chan, “A framework for partitioning and execution of data stream applications in mobile cloud computing,” in *2012 IEEE Fifth International Conference on Cloud Computing*, June 2012, pp. 794–802.

[39] D. Amendola, N. Cordeschi, and E. Baccarelli, “Bandwidth management vms live migration in wireless fog computing for 5g networks,” in *2016 5th IEEE International Conference on Cloud Networking (Cloudnet)*, Oct 2016, pp. 21–26.

[40] G. de la Roche, A. Valcarce, D. Lopez-Perez, and J. Zhang, “Access control mechanisms for femtocells,” *IEEE Communications Magazine*, vol. 48, no. 1, pp. 33–39, January 2010.

[41] Y. Yu, A. Pang, P. Hsiu, and Y. Fang, “Energy-efficient downlink resource allocation for mobile devices in wireless systems,” in *2013 IEEE Global Communications Conference (GLOBECOM)*, Dec 2013, pp. 4692–4698.

[42] H. Bremermann, “A method of unconstrained global optimization,” *Mathematical Biosciences*, vol. 9, pp. 1–15, 1970.

[43] I. Stojmenovic and S. Wen, “The fog computing paradigm: Scenarios and security issues,” in *2014 Federated Conference on Computer Science and Information Systems*, Sept 2014, pp. 1–8.

[44] M. of Internal Affairs and Communications, “Information & Communications Statistics Database H28.1Q Report,” http://www.soumu.go.jp/johotsusintokei/field/denpa01.html, 2016, [Online: accessed 29-March-2017].

[45] T. Obara, S. Suyama, J. Shen, and Y. Okumura, “Joint fixed beamforming and eigenmode precoding for super high bit rate massive mimo systems using higher frequency bands,” in *2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)*, Sept 2014, pp. 607–611.

[46] N. Hinitt and T. Kocak, “Gpu-based fft computation for multi-gigabit wirelesshd baseband processing,” *EURASIP J. Wirel. Commun. Netw.*, vol. 2010, pp. 30:1–30:13, Apr. 2010. [Online]. Available: http://dx.doi.org/10.1155/2010/359081

[47] B. Guo, W. Cao, A. Tao, and D. Samardzija, “Lte/lte-a signal compression on the cpri interface,” *Bell Labs Technical Journal*, vol. 18, no. 2, pp. 117–133, Sept 2013.

[48] Y. Wang, M. C. Meyer, and J. Wang, “Real-time delay minimization for data processing in wirelessly networked disaster areas,” *IEEE Access*, vol. 7, pp. 2928–2937, 2019.

[49] M. C. Meyer, Y. Wang, and J. Wang, “Cost minimization of data flow in wirelessly networked disaster areas,” in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.

[50] Y. Wang, M. C. Meyer, J. Wang, and X. Jia, “Delay minimization for spatial data processing in wireless networked disaster areas,” in *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 2017, pp. 1–6.

[51] F. Xia, J. Wang, X. Kong, Z. Wang, J. Li, and C. Liu, “Exploring human mobility patterns in urban scenarios: A trajectory data perspective,” *IEEE Communications Magazine*, vol. 56, no. 3, pp. 142–149, March 2018.

[52] J. Weinman, “Cloud computing is np-complete,” 2011.

[53] B. CAULFIELD, “Nvidia unveils geforce rtx, world’s first real-time ray tracing gpus,” Aug 2018. [Online]. Available: https://blogs.nvidia.com/blog/2018/08/20/gamescom-rtx-turing-real-time-ray-tracing/

List of Publications

Refereed Journals

1. **Wang, Y.**, Meyer, M. C., and Wang, J. (2019). Real-Time Delay Minimization for Data Processing in Wirelessly Networked Disaster Areas. IEEE Access, 7, 2928-2937.(Major)

2. Wang, J., Meyer, M. C., Wu, Y., and **Wang, Y.** (2019). Maximum Data-resolution Efficiency for Fog-Computing Supported Spatial Big Data Processing in Disaster Scenarios. IEEE Transactions on Parallel and Distributed Systems. (2019). (Major)

Refereed International conferences

1. **Wang, Y.**, Meyer, M. C., Wang, J., and Jia, X. (2017, December). Delay minimization for spatial data processing in wireless networked disaster areas. In GLOBECOM 2017-2017 IEEE Global Communications Conference (pp. 1-6). IEEE. (Major)

2. Meyer, M. C., **Wang, Y.**, and Wang, J. (2018, May). Cost minimization of data flow in wirelessly networked disaster areas. In 2018 IEEE International Conference on Communications (ICC) (pp. 1-6). IEEE. (Major)

3. Meyer, M. C., **Wang, Y.**, and Watanabe T. (2019, Jan). Wavelength-Selective Fog-Computing Network for Big-Data Analytics of Wireless Data. In 2019 International Conference on Electronics, Information, and Communication (ICEIC) (pp. 1-6). IEIE.

4. Meyer, M.C., **Wang, Y.**, and Watanabe T. (2019, July). Fault-tolerant Traffic-aware Routing Algorithm for 3-D Photonic Networks-on-chip. In 2019 IEEE International Symposium on Embedded Multicore/Many-core Systems-on-chip (MCSoc). (Accepted)