



KADIR HAS UNIVERSITY
SCHOOL OF GRADUATE STUDIES
PROGRAM OF COMPUTER ENGINEERING

**PEER TO PEER AND MASTER-SLAVE TOPOLOGIES
FOR EDGE COMPUTING IN INTERNET OF THINGS**

MOSTAFA ZIADOON IBRAHIM IBRAHIM

MASTER'S THESIS

ISTANBUL, AUGUST, 2019

MOSTAFA ZIADDOON IBRAHIM IBRAHIM

M.S. Thesis

2019

PEER TO PEER AND MASTER-SLAVE TOPOLOGIES FOR EDGE COMPUTING IN INTERNET OF THINGS

MOSTAFA ZIADOON IBRAHIM IBRAHIM



MASTER'S THESIS

Submitted to the School of Graduate Studies of Kadir Has University in partial fulfillment of the requirements for the degree of Master's in the Program of Computer Engineering

ISTANBUL, AUGUST, 2019

DECLARATION OF RESEARCH ETHICS /
METHODS OF DISSEMINATION

I, MOSTAFA ZIADOON IBRAHIM IBRAHIM, hereby declare that;

- this Master's Thesis is my own original work and that due references have been appropriately provided on all supporting literature and resources;
- this Master's Thesis contains no material that has been submitted or accepted for a degree or diploma in any other educational institution;
- I have followed "Kadir Has University Academic Ethics Principles" prepared in accordance with the "The Council of Higher Education's Ethical Conduct Principles"

In addition, I understand that any false claim in respect of this work will result in disciplinary action in accordance with University regulations.

Furthermore, both printed and electronic copies of my work will be kept in Kadir Has Information Center under the following condition as indicated below

- The full content of my thesis/project will be accessible from everywhere by all means.

MOSTAFA ZIADOON IBRAHIM IBRAHIM



DATE AND SIGNATURE

28/08/2019

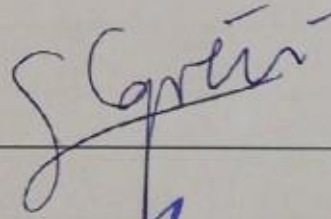
KADIR HAS UNIVERSITY
SCHOOL OF GRADUATE STUDIES

ACCEPTANCE AND APPROVAL

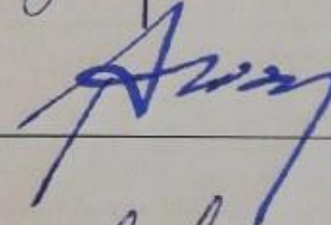
This work entitled **PEER TO PEER AND MASTER-SLAVE TOPOLOGIES FOR EDGE COMPUTING IN INTERNET OF THINGS** prepared by MOSTAFA ZIADOON IBRAHIM IBRAHIM has been judged to be successful at the defense exam held on **28/08/2019** and accepted by our jury as **MASTER 'S THESIS**.

APPROVED BY

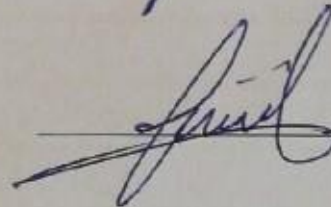
Asst. Prof. Dr. Arif Selçuk Öğrenci (Advisor)
Kadir Has University



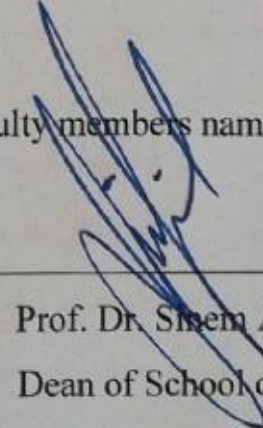
Asst. Prof. Dr. Taner Arsan
Kadir Has University



Asst. Prof. Dr. Figen Özen
Haliç University



I certify that the above signatures belong to the faculty members named above.



Prof. Dr. Sitem AÇIKMEŞE
Dean of School of Graduate Studies
DATE OF APPROVAL

TABLE OF CONTENTS

| | |
|--|------------|
| ABSTRACT | i |
| ÖZET | ii |
| ACKNOWLEDGEMENTS | iii |
| LIST OF TABLES | iv |
| LIST OF FIGURES | v |
| LIST OF SYMBOLS/ABBREVIATIONS | vi |
| 1. INTRODUCTION | 1 |
| 1.1 Objectives..... | 4 |
| 1.2 Motivation..... | 4 |
| 1.3 Research Questions | 4 |
| 1.4 Problem Statement..... | 5 |
| 1.5 Outline..... | 6 |
| 2. LITERATURE REVIEW | 7 |
| 2.1 Introduction | 7 |
| 2.2 Iot..... | 7 |
| 2.3 The Shift From Cloud To Edge Computing | 8 |
| 2.4 Mobile Cloud And Communication Resources | 9 |
| 2.4.1 Communication Resources | 10 |
| 2.4.2 Cloud Computing Resources..... | 11 |
| 2.5 Assignment Problems..... | 13 |
| 2.6 Scheduling And Planning Problem | 14 |
| 2.7 Edge Computing..... | 15 |
| 3. METHODOLOGY | 17 |
| 3.1 Peer to Peer (P2P) | 18 |
| 3.2 Master – Slave (M2S)..... | 19 |
| 4. RESULTS | 22 |
| 4.1 Peer to Peer (P2P) RESULTS | 22 |
| 4.2 Master - Slave (M2S) RESULTS | 28 |
| 5. CONCLUSIONS | 35 |

| | |
|------------------------------|-----------|
| 6. REFERENCES..... | 36 |
| CURRICULUM VITAE..... | 40 |



PEER TO PEER AND MASTER-SLAVE TOPOLOGIES FOR EDGE COMPUTING
IN INTERNET OF THINGS

ABSTRACT

The rapid increase in the number of devices connected to the Internet has led to a transition from cloud computing to the edge. Edge computing slowly replaces cloud computing due to the increased capacity of mobile devices such as memory, battery, and computing power. These computational and power resources in mobile devices may however become insufficient, especially in time-sensitive and computationally intensive applications. Advanced edge computing will not only improve the performance of your device applications but it will also reduce the power consumption of your devices, by extending battery life. However, the performance of edge based computing depends on the effective allocation of computer and communications resources. This work assesses the problem of improving resource allocation for sophisticated mobile computing systems. Resource allocation is implemented to reduce the cost associated with edge computing. In this research we link edge devices together with the application of resource assignment to other devices depending on the type of link. The main consideration is that each device is designed to perform a set of tasks while calculating the usage rate of both RAM and CPU. Two methods have been used to scale the connection of devices at the edge: peer to peer (P2P) and master-slave models. The use of communication models such as P2P and Master-Slave shows a significant improvement in task execution. Master-Slave model has a better performance and is therefore recommended for advanced computing.

Keywords: IOT, edge computing, mobile edge computing, cloud computing, fog computing, Peer to Peer, Master-Slave, CPU utilization, RAM utilization.

NESNELERİN İNTERNETİNDE KENARDA HESAPLAMA İÇİN EŞLER ARASI VE ANA-YARDIMCI TOPOLOJİLERİ

ÖZET

İnternete bağlanan cihaz sayısındaki hızlı artış bulutta hesaplamadan kenarda hesaplamaya geçişe yol açmıştır. Gezgin cihazların hafıza, pil ömrü ve hesaplama gücündeki artışlar sayesinde kenarda hesaplama yavaş yavaş bulutta hesaplamının yerini almaktadır. Ancak gezgin cihazlardaki bu hesaplama ve güç kaynakları, özellikle süreye hassas ve hesaplama yoğun uygulamalarda, yetersiz kalabilmektedir. Kenarda hesaplamının gelişmesi, sadece cihazların performanslarını iyileştirmeyecek, aynı zamanda cihazların güç harcamasını azaltarak pil ömrünü uzatacaktır. Öte yandan kenarda hesaplamadaki performans hesaplama ve iletişim kaynaklarının etkin atanmasına bağlıdır. Bu çalışmada gezgin hesaplama sistemlerindeki kaynak atama problemi ele alınmıştır. Kenarda hesaplamayla bağlantılı maliyetleri azaltmak için kaynak atama gerçekleştirilmektedir. Bu araştırmada cihazlar arasındaki bağlantı tipine göre yapılan atama ile cihazların birbirine bağlanması öngörülmüştür. Temel yaklaşım, her bir cihazın kendisine atanan belli sayıda görevi yerine getirirken hafıza ve işlemci kullanımını sürekli olarak hesaplamasıdır. kenarda cihazların bağlantılarını ölçeklerken iki yöntem kullanılmıştır: Eşler arasında (P2P) bağlantı ve ana-yardımcı bağlantı modeli. Bu tarz iletişim modellerinin görevlerin işlenmesinde anlamlı iyileştirmeler getirdiği gösterilmiştir. Ana-yardımcı bağlantı modelinin performansı daha iyidir ve bu sebeple kenarda hesaplama için önerilmektedir.

Anahtar Sözcükler: nesnelere interneti, kenarda hesaplama, gezgin kenarda hesaplama, bulutta hesaplama, siteme hesaplama, eşler arası, ana-yardımcı, işlemci kullanımı, hafıza kullanımı.

ACKNOWLEDGEMENTS

I would like to express my gratitude to the advisor of my thesis Asst. Prof. Dr. Arif Selçuk Öğrenci for giving me the opportunity to work on this most wonderful research and give him my thanks and pride and respect to him for his help from the title of the topic and the application of the idea until it became a complete and beautiful thesis.

I would like to express my deep appreciation to Kadir Has University which has helped me develop a strong knowledge base and acquire skills through courses and lectures. I would like to thank the distinguished professors of the discussion committee for accepting this thesis. I also thank my parents, wife, sister and, who are always encouraged and supported.

LIST OF TABLES

| | |
|---|----|
| Table 4.1 P2P EXECUTION OF TASK FOR P2P..... | 25 |
| Table 4.2 P2P Average RAM and CPU utilization..... | 27 |
| Table 4.3 Master-Slave average CPU and RAM utilization..... | 33 |
| Table 4.4 Master-Slave task execution..... | 35 |
| Table 4.5 Comparison between Peer-to-Peer and Master-Slave..... | 36 |



LIST OF FIGURES

| | |
|---|----|
| Figure 2.1 The design of mobile cloud computing | 10 |
| Figure 2.2 How edge computing works..... | 15 |
| Figure 3.1 Peer To Peer Flow chart..... | 18 |
| Figure 3.2 Master -Slave Flow chart..... | 20 |
| Figure 4.1 P2P CPU utilization for 1000 task and 10 nodes..... | 21 |
| Figure 4.2 Peer to Peer RAM utilization for 1000 task and 10 nodes..... | 22 |
| Figure 4.3 Peer to Peer subplot CPU utilization for 1000 task and 10 nodes..... | 23 |
| Figure 4.4 Peer to Peer subplot RAM utilization for 1000 task and 10 nodes..... | 24 |
| Figure 4.5 Master -Slave CPU UTILIZATION..... | 29 |
| Figure 4.6 Master -Slave RAM UTILIZATION..... | 30 |
| Figure 4.7 Master- Slave subplot CPU UTILIZATION..... | 31 |
| Figure 4.8 Master – Slave subplot RAM UTILIZATION | 32 |

LIST OF SYMBOLS/ABBREVIATIONS

| | |
|-------|---|
| 2G | 2th Generation Cellular Networks |
| 3G | 3th Generation Cellular Networks |
| 4G | 4 th Generation Cellular Networks |
| 5G | 5 th Generation Cellular Networks |
| CPU | Central Processing Unit |
| D2D | Device-To-Device Communication |
| DCF | Distributed Coordination Function |
| EC | Edge Computing |
| IaaS | infrastructure-as-a-service |
| IoT | Internet of Things |
| LTE | Long-Term Evolution |
| M2M | Machine to Machine |
| M2S | Master to Slave |
| MCC | Mobile Cloud Computing |
| MEC | Mobile Edge Computing |
| OFDMA | Orthogonal Frequency Division Multiple Access |
| P2P | Peer to Peer |
| PaaS | platform-as-a-service |
| PC | Personal Computer |
| RAM | Random Access Memory |
| RAS | Resource Allocation Scheme |
| RFID | Radio-Frequency Identification |

| | |
|-------|--|
| SaaS | software-as-a-service |
| SCADA | Supervisory Control And Data Acquisition |
| SOA | Service Oriented Architecture |
| SPE | Distributed Stream Processing Engines |
| TDMA | Time Division Multiple Access |
| UID | Unique Identifier |



1. INTRODUCTION

The move towards connected devices in Internet has led to the increased number of mobile devices such as tablets, smartphones, sensor, and other portable devices. A recent estimation by Cisco has revealed that the number of connected mobile devices will grow tremendously to 11.6 billion by 2021 (Gezer et al., 2018). This represents a compound annual growth rate of 8% for the period between 2016 and 2021 (Josilo, 2018). Although there has been a significant increase in the number of these devices, the computational capabilities are still limited compared to traditional servers and desktops. Considering that these mobile devices are battery powered then it is vital to note that most of the applications in these devices are profoundly limited by the clock speed of the processor. Internet of Things (IoT) may be a tricky topic but it is not a new principle. At the dawn of the second millennium, the concept of Internet of Things emerged. This concept is based on the requirement that each thing must have a wireless identifier and a connection to internet for data exchange. These things can be controlled or communicated through a computer. In 1990, Kevin Ashton wrote an article in RFID magazine in which he mentioned about connecting all we have with computers, that we can control everything and know everything through the information that will be collected without any human intervention in addition to the possibility of knowing the quantity and location. Thus, we can reduce costs and rates of loss and waste. We need to strengthen computers by means of data collection so that they can see, hear and identify the world on their own. Radio frequency identification, sensors and actuator technology gave acceptance for computers to display, recognize and know the world – without the conditions of data that are entered by ourselves. At that time, this sight required important technological advancement. Along the process, several important questions have arisen: How to connect the things on this planet? What kind of wireless communication should we build on devices? What changes should we make to the internet infrastructure to deal with billions of new devices? What kind of intensity can these devices hold? What should we evolve to make these solutions cost operative?

In 1999, the number of questions was huge than the number of responses to internet of things. All these questions represent problems in themselves and pose difficulties

in building the infrastructure of the IoT concept (Buyya and Dastjerdi, 2016). Therefore, to maintain the portability of those mobile devices, we need to ensure that the energy consumption is maintained at the lowest possible level allowing the devices to be physically light and small while transferring the limited computational capabilities and memory to the cloud (Dao et al., 2018). The mobile cloud is difficult to implement in real time due to the absence of access schemes, standards, and elasticity in the application model. Although the mobile cloud is associated with some issues, it is also better at energy management, task division, security and better services. The mobile cloud computing is the conventional accommodation of cloud computing and wireless networks to allow for mobility. Furthermore, the mobile cloud computing system is made up of three primary components: hardware, software and the communication network (Josilo, 2018). These elements allow the users to access computing functionalities from anywhere provided there is a full connection to the information space considering that each node in the information space has communication capabilities for processing information and delivering data via the voice or data channel.

Resource assignment (allocation) is the process of utilizing and allocating the limited computing resources contained within the environment to meet the needs of the user. The resources are assigned in amount and type to complete a certain task (Dao et al., 2018). For optimal resource assignment in cloud computing and connected devices (i.e. in IoT), it is vital to utilize an optimal resource allocation scheme (RAS) that will assist in preventing two applications from trying to access the same resource at the same time especially in cases of isolated unlimited resources (Josilo, 2018). The mobile cloud computing technology is a new approach that has been applied in closing the gap between the limited computational capabilities of mobile devices and the high requirements of certain applications. Resource assignment is a significant part of the application process (Prabhu and J, 2015). For any program or application to be run in a system that requires resources that's when an application is open, the device must allocate certain resources for it to run. In mobile cloud computing, this condition is satisfied through the mapping of virtualized resources to a physical application. Computing resources such as hardware and software are allocated to the cloud application which in turns provide unlimited virtualized resources. Mobile

cloud computing is part of scalable computing models in cases where virtual machines are entered therefore resource management system is vital to manage the allocation of physical resources to the cloud servers and the mobile devices (Gezer et al., 2018). The process is typically carried out in a on-demand fashion especially when the process progresses to execution. The management system should not allocate any resources prior to execution as this will reduce the number of resources available in times of provisioning.

Design optimization is the practice of determining the optimal set of design parameters that will be vital in attaining a certain objective. Optimization is vital in any design project especially in complicated problems (Gawanmeh et al., 2017). For instance, when designing for resource allocation in computing, it's vital to determine which resources will be used for provisioning to maximize the utilization of the limited computing resources. Optimization can be divided into functional or combinatorial optimization (Gawanmeh et al., 2017). The functional optimization is formulated as a continuous function for the design parameters such as hardware (which might be a continuous function of the computing resources available in a mobile device). In combinatorial optimization, the values of the parameters are investigated. In this case, parameters are typically discrete, i.e., the design problem can be accomplished in finite number of states. For complex design problems, the functional optimization might be inadequate; thus, in this case, the problem may be discretized to convert it to a combinatorial optimization problem (Weintraub and Cohen, 2017). Genetic algorithm and neural networks can be implemented for optimization. The genetic algorithm is the search technique utilized for minimizing inefficiencies and reducing production time while maximizing productivity. On the implementation side, software containers are emerging solutions. It is the most important characteristic of containers that the programmer or developer will be able to build installations and infrastructure in order to use resources and invest in a way to get rid of the constant need to buy corporate services such as software and other limited use, which needs a lot of time to be synchronized with the concept of the Internet of Things. Containers are the right solution for those who need to expand or develop the structures of the Internet of Things in a short period of time. In mobile cloud computing the optimization algorithm will be used to reduce the energy

consumption, and the cost of cloud services, optimize resource allocation and increase latency tolerance of some applications (Josilo, 2018).

1.1 Objectives

The primary aim of this thesis is to develop a methodological framework that will improve the utilization of nodes (computing elements) in edge computing when the nodes are overloaded by the tasks arriving to them. This will be accomplished through simulations in MATLAB where tasks are offloaded to other nodes available for processing. This research will evaluate two different topologies for resource allocation in edge computing, namely, the master-slave and peer-to-peer communication schemes.

1.2 Motivation

This thesis was motivated by the need for resource assignment process in the emerging technology of edge computing and the increase in the low latency requirement and delay tolerance in mobile applications. Generally, tasks require a certain amount of computational resources to be used where the arrival of subsequent tasks to a specific node may cause overloading in the node. The latency requirements cannot be achieved neither by a sequential processing nor by transferring the task(s) to the cloud as the latter would need extra latency for communication. Hence, a suitable task offloading strategy would be desirable to achieve system level service levels without further installations of hardware resources.

1.3 Research Questions

The simulation of the task offloading based resource assignment in edge computing will be the main focus in the thesis as it seeks to answer the following research questions:

- How to generate a set of tasks in a group of processing units that serve as edge nodes?
- How to achieve the task offloading among the nodes using peer-to-peer and master-slave topology?
- Which mode of communication among nodes is better in terms of utilization?

1.4 Problem Statement

The growth in the number of connected mobile devices over the years has also led to an increase in mobile data traffic. This increase in the mobile data traffic has increased stress on the mobile computational performance thus leading to the development of mobile cloud computing. This increased demand has also placed a lot of expectation on the cloud communication and computing resources; therefore, it is vital to evaluate resource allocation in these technologies to ensure the user experience is always maintained. The design of mobile cloud computing has always been faced by challenges primarily to constraints, i.e., maintaining user experience and cost (Gawanmeh et al., 2017). User experience is fundamental as each application should be concerned with the overall response time and energy consumption of the battery. The responses of any application in mobile communication are affected by access to the limited resources through the wireless network; therefore, application performance is typically constrained by external resources. Furthermore, the performance of applications dictates battery life as heavy applications consume more energy. The cost of using mobile applications is another constraint that will be evaluated as it is important in the selection of the cloud computing services. To address all these challenges, various cloud architectures have been considered with the common practice being that the mobile device users will access a commercial cloud infrastructure where they would offload their computational tasks. The recent development in the remote resourceful clouds such as Windows Azure and Amazon EC2 has been such that resource allocation accommodates for the extremely low latency requirements of the delay sensitive

applications used in mobile devices (Gawanmeh et al., 2017). The optimization efforts in this thesis will evaluate the proposal of edge computing bringing computational resources close to the edge of the network which is easily accessible through the emerging 5G network. Thus, the problem statement of the thesis can be formulated as follows. How can we simulate the resource allocation in edge computing? The simulation of the computing scenario will be done in MATLAB and optimized using two network topologies peer-to-peer and master-slave to increase the efficiency of resource allocation for low latency and delay-intolerant applications.

1.5 Outline

This thesis contains six chapters. Chapter 1 provides an introduction that outlines a brief background, motivation, goals and problem statement. Chapter 2 provides background knowledge of existing literature through a comprehensive literature review of the various aspects to be considered in this research. Chapter 3 provides the methodology that deals with problem formulation, algorithm simulation and method comparison. Chapter 4 presents simulation results and comparison of various topology techniques. Furthermore, it defines a detailed discussion of the results that give meaning to each value and relates it to the research questions. Chapter 5 will present the concluding observations of the thesis that provide a new way of research.

2. LITERATURE REVIEW

2.1 Introduction

This chapter provides a theoretical background to the knowledge and the concepts that are to be utilized in this thesis. The theoretical background and methods covered in this section will be utilized in the formulation of the problem and in the creation of the most efficient way of solving the problem.

2.2 IoT

IoT is the emerging technology that describes a system of interrelated computing devices, embedded systems, digital machines, microprocessor chips, people and mechanical machines that are assigned unique identifiers (UID) which can transfer data and interact over a network without the need for human-computer interaction (Hassan et al., 2015). IoT systems are made up of web-enabled smart devices that utilize sensors, communication hardware, and embedded processors to collect a huge amount of data in their respective environments. The collected data are shared among the connected devices via various IoT gateways or other edge devices that enable the data to be sent to the cloud (Josilo, 2018). Related devices can access this data, and most of these functionalities are carried out without human intervention. The internet of things is an extension to SCADA (Supervisory Control And Data Acquisition) which is a class of software used for process control and data gathering in real time especially from remote locations (Elijah et al., 2018). Similar to SCADA the IoT also include the hardware and software components but has been extended to collect a huge amount of data and transmit via the internet to cloud storage.

IoT has numerous real-world applications ranging from manufacturing, enterprise IoT, consumer IoT, industrial IoT, etc. (Elijah et al., 2018). The application of this technology spans various sectors both horizontally and vertically such as energy, automotive, business and telco. The consumer sector has seen the surge of smart homes that are typically equipped with smart devices such as thermostats, heating systems, electronic devices, lighting controls, smart appliances etc. which can be remotely controlled via a network connected device such as computers and

smartphones (Jindal, et al., 2018). Further wearable devices contain sensors and software that can transmit or analyze data providing real time experience to the user. These wearables have increased safety through improving first responders' response time especially in emergencies or in life threatening situations (Jindal et al., 2018). In healthcare, mobile devices with sensors have been used to monitor patients closely collecting data which can be analyzed and stored for future diagnosis and treatment. This increased collection of data and usage of software have led to the need for mobile edge computing technology to improve accessibility and analysis of data.

2.3 The Shift from Cloud to Edge Computing

Cloud computing has been at the center of the internet of things; conventionally it is the practice of utilizing remote network service via the internet to process and store data. Cloud computing has grown tremendously over the last decade from a business idea to an aspect of the information technology industry (Kang et al., 2015). A definitive advantage of cloud computing paradigm is the capability to deploy any application or service without the cost of local hardware. Further, the cloud offers the ability to scale the computing capacity of a mobile device which can either be scaled up or down in a "on demand" basis. This elasticity has improved the process of resource allocation as it is dependent on real time demand. This feature has become a significant form of mobile device users especially those in the internet of things that suffer from seasonal demand peak (Liu et al., 2017). This elasticity has seen a change in investment as initially there was the need for infrastructure such as servers with increased processing power and storage to be able to accommodate a huge amount of data as all this is easily accessible through the cloud services. Presently, the cloud can offer any connected user services based on a variety of pricing models that will fit the user.

Users and companies are becoming increasingly aware of the importance of cloud computing in business and the value it takes the transition to the cloud. A smooth transition can only be realized where the comprehension of the benefits and challenges involved in the cloud is achieved (Liang, 2017). Cloud services are expected to have the potential of exceeding service reliability, and availability requirements of the traditional deployment in mobile or business sectors.

Consequently, the primary concerns in edge computing are security and privacy as there is a fundamental shift in the traditional model of accessing resources from a centralized position to a distributed and decentralized system (Ahmed and Rehmani, 2017). This new paradigm is referred to as edge computing in which case the fundamental computing building blocks such as computational resources, network, and storage are brought closer to the consumer. However, edge computing is still in its infancy as various vendors are positioning it from a different perspective. The current cloud market is made of networked devices, public cloud, and automation companies; thus, edge computing can be divided into two, i.e., fog computing and edge computing (Ahmed and Rehmani, 2017). The fog computing technique entails the moving of intelligence down the local area network where data can be processed in the fog node. Edge computing pushes processing power, intelligence and communication capabilities towards connected mobile devices via the edge gateway. The movement of the computing nodes close to the origin of the data reduces latency in the round trip to the cloud, this advantage is being exploited in numerous areas such as gaming, healthcare, video streaming and internet of things (Liang, 2017). The internet of things is the primary driver of edge computing accelerating the pace of adoption of this technology, especially in the data driven applications. Most clouds are underpinned by a network of virtualized infrastructure that allows the optimization of hardware use. Virtualization technology that enables elasticity in cloud computing increasing flexibility in the speed of deployment, cloud management and dynamic auto provisioning of resources. Currently, multiple vendors offer the virtualization environments where the primary factor for the selection of the vendor is based on the requirements of the user.

2.4 Mobile Cloud and Communication Resources

The two primary types of resources that are contained the mobile cloud system, that is, computational and communication resources, are illustrated in Figure 2.1 below. The mobile devices will decide whether to utilize computational resources locally or to upload the task to external cloud storage where the communication network is used (Josilo, 2018). Offloading the tasks to computational resources requires an external resource manager. An increased competition for the resources might affect

the performance of both a mobile device and other devices in the system. Therefore, it's vital to evaluate the various access technologies, mobile cloud computing architectures and their effects on the mobile cloud computing systems.

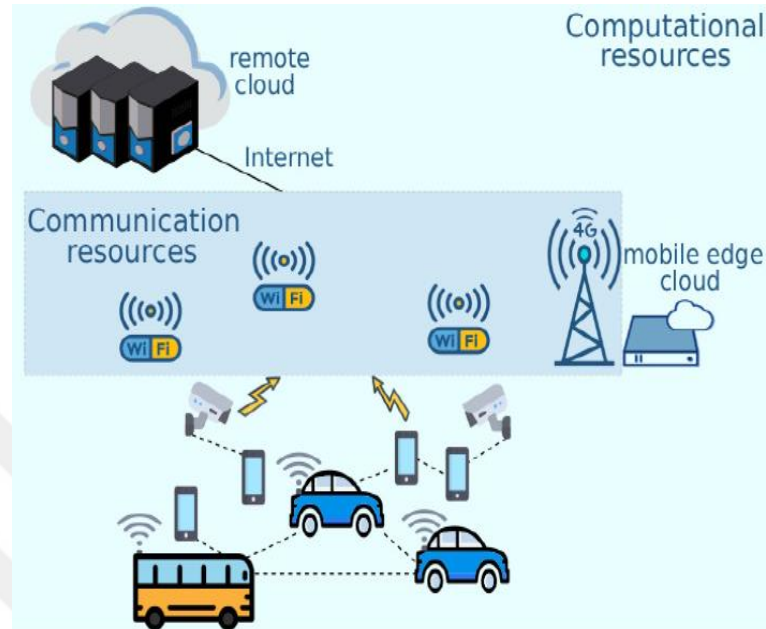


Figure 2.1 The design of mobile cloud computing (Josilo, 2018).

2.4.1 Communication Resources

The process of offloading specific tasks to external computational resources such as the cloud is highly reliant on wireless communication networks. There is a heterogeneity in the wireless networks available to mobile users hence they can select between the different radio access technologies such as 2G, 3G, 4G, 5G and Wi-Fi communication (Gusain and Kumar, 2014). The radio communication technologies may experience problems such as intermittent connectivity, limited bandwidth, and variable network conditions. Moreover, the sharing of the communication medium considerably affects the transmission rate which is highly dependent on the bandwidth allocation algorithm of the service provider, for instance, distributed coordination function (DCF) is a protocol used for CSMA/CA which uses the fair bandwidth sharing protocol (Sonkar and Kharat, 2015). Other alternatives, fair bandwidth sharing mechanisms include the ones used for time fair TDMA and OFDM where the medium access protocol can be computed using the mathematical relations, but in this case, it will be dependent on the total number of

users sharing the access point (Sonkar and Karate, 2015). The model can also be used to describe the proportional fair scheduling (PFS) which is a characteristic of the 3G networks (Jennings and Stadler, 2015). The overall growth rate of the mobile devices, the increase in the latency requirement, and delay sensitive applications have made it clear that it's vital for the communication resources stored in the cloud to be adequately managed (Jennings and Stadler, 2015). Recently, there has been the emergence of mechanisms for predicting network connectivity to the user based on the movement or database of network connectivity within a geographical region. The emergence of connected devices has improved the above mechanism using a collaboration of the mobile devices to determine the connectivity. This method of connected devices improves the bandwidth utilization and allows device-to-device communication (D2D) that might promote the development of 5G communication network (Nzanywayingoma and Yang, 2017). Additionally, the collaboration between the devices would lead to a highly distributed mobile cloud computing system that will promote edge computing.

2.4.2 Cloud Computing Resources

In mobile cloud computing, the computational resources may originate from various sources such as the cloud at the network, the edge, mobile devices and commercial clouds (Jennings and Stadler, 2015). The origin of the mobile cloud computational resources will affect the performance of applications in a mobile device; therefore, we evaluate the traditional centralized system and the emerging distributed mobile cloud system. Cloud computing, depending on its storage and processing capabilities, can be used to analyze the data generated by the IoT objects in batch and stream formats, and has reduced the pay-as-you-go model adopted by cloud computing providers. Data storage and analysis, as well as efficient procedure creation for the construction of IoT applications, and the advantage of cloud flexibility with distributed Stream Processing Engines (SPE) can implement important features such as fault tolerance for fault loads (Buyya and Dastjerdi, 2016).

Researchers who propose a framework that supports the collection of sensor data in a cloud based IoT context are based on SOA and event oriented, and define the benefits of the heuristic layer responsible for processing events and their logic.

Others suggest Multitenant, a platform as a service to deploy IoT applications, that provides users with an isolated virtual service that can be customized to their IoT devices while sharing the cloud infrastructure with other users.

2.4.2.1 Centralized Cloud System

The currently offered commercial cloud services by the four major providers, i.e., Amazon, Microsoft, Google, and IBM may either be; platform-as-a-service (PaaS), software-as-a-service (SaaS) and infrastructure-as-a-service (IaaS) (De Filippi and McCarthy, 2012). The PaaS and IaaS are characterized by flexibility, increased control and management, and therefore they are suitable for other providers that provide the mobile user with SaaS. It is evident that from the above perspective the SaaS is the most vital cloud computing service (Krogh, 2013). The commercial cloud services have huge transmission delays due to the distance between the user and the servers; thus, for mobile users, the commercial cloud will not offer the best solution in delay sensitive applications.

2.4.2.2 Fog Computing Resources

The fog computing technology is the beginning of a paradigm shift from the centralized cloud architecture to the distributed system that brings the resources close to the user (More and Kulkarni, 2017). The primary idea for computing is extending the existing centralized cloud computing architecture through a collaboration of the distributed cloud resources to the heterogeneous devices. This practice has increased the computational resources accessible to mobile devices through the pooling of resources (Silva et al., 2019). The emergence of IoT applications is preparing the reason for shifting towards the distributed architecture. This technology has increased the number of devices that access the cloud resources, therefore, making it difficult for a centralized cloud to be accessed through a limited network bandwidth. For computing, this provides advantages in both communicational and computational resources due to the collaboration of nearby devices through device to device (D2D) communication (Prakash et al., 2017). The D2D communication has improved bandwidth utilization but the challenge of integrating heterogeneous devices into a

single cloud computing platform while efficiently assigning tasks among the numerous devices, still exists.

2.4.2.3 Mobile Edge Clouds

The distributed mobile edge cloud (MEC) is a rather more popular alternative to the centralized cloud. The MEC advocates for the installation of computational and storage resources in the existing infrastructure such as mobile base stations (BS) (Ahmed and Rehmani, 2017). This practice is more applicable for network operators as they can profit from bringing the computational resources to the edge of the network. MEC has the potential of improving performance and resource allocations to users however the increased number of users accessing the MEC resources will lead to a downgraded performance. The quantity of the bandwidth allocated to a user may not be sufficient for transmission especially in offloading vast amounts of data (Ahmed and Rehmani, 2017). Secondly, the MEC provides limited storage and computational resources as compared to the commercial cloud. As a consequence, this limitation in the computing resources that can be assigned to the user has led to the development of the optimization technique to ensure that the resources are adequately and fairly allocated.

2.5 Assignment Problems

Assignment problems (AP) need an optimization technique that entails the optimal matching, i.e., assigning the components of two sets these are agents and tasks where each matching has a different weight (cost) (Chauvet et al., 2000). Problems in this class are combinatorial optimization problems and can be described mathematically in multiple ways such as a bijective mapping between two finite sets and flows in a network. The AP technique functions to assign m agents to n tasks; each task will be assigned to a single agent to minimize the cost of assignment (Dil Afroz and Hossen, 2017). It can be described as a one-to-one mapping such as a worker (agent) to a machine (task). The AP optimization technique has numerous real world applications such as assigning developers to a software project, military personnel to operations, resource scheduling in cloud, etc.

The main statement is the constraint of the problem and defines that each agent is assigned to a single task and in no instance will an agent be assigned to more than

one task (Singh et al., 2012). The AP can be naively solved through a comparison mechanism of all the possible assignments of tasks to agents, but this approach is computationally infeasible as it will cause a combinatorial explosion (Kabiru et al., 2017). For instance, consider a problem with several combinations for assigning more than 100 tasks to a similar number of agents. Increasing the number of nodes will also increase the possibilities of combinations exponentially. The access point approach has several flavors such as multiple agents per task, multiple tasks per agent and multidimensional assignment problems that require matching of members of two or more sets (Supian et al., 2018).

2.6 Scheduling and Planning Problem

Scheduling is defined as the assignment of resources over a period to perform a collection of tasks (Hyari et al., 2006). The primary aim of scheduling in any problem is to optimize one or more of the performance criteria while assigning the limited resources over a period to a sequence of activities. The most significant elements of scheduling are the sequence of activity and the timing on resources. Schedules can be categorized into two as dictated by the availability of tasks: either before or after the creation of the schedule, i.e., static or dynamic schedules (Hassani et al., 2018). In the static case, the tasks do not change once the process schedule has been defined, i.e., no new jobs can be added do the process (Hassani et al., 2018). In dynamic scheduling the arrival of tasks is unpredicted; thus, jobs can emerge at any point of the process and would require to be scheduled. This practice of scheduling emerging unpredictable activities in dynamic scheduling and tells the modification of the original schedule is known as rescheduling.

Rescheduling is an important element of individual and organizational decision-making processes in various fields. Rescheduling is an integral aspect of real-life problems such as rescheduling of a nurse shift due to an emergency, rescheduling of a trip due to a vehicle breakdown, rescheduling of human resources due to new activities, etc. (Hassani et al., 2018). Rescheduling may directly affect the different assigning criteria such as quality, cost, time and security. Planning in optimization

problems is related to determine the tasks that need to be performed. Scheduling differs from planning as it is primarily concerned with the assignment of limited resources in a period of sequence of activities while planning is best on the determination of the subset of events that will require to be performed to attain a particular objective (Ben Issa and Tu, 2017). The planning problem is concerned mostly with the selection of a subset of tasks which are selected from a set of given alternatives.

2.7 Edge Computing

First, the desire was driven using platforms that provide the facilities and tools that we have always used in cloud computing (Whitaker, 2017). Edge computing represents an environment for the distribution of information technology in which the client and user data are processed in less time from the network source. Through this computing we can reduce the consumption of hardware components and devices associated with the network in the Internet of Things. Edge computing can benefit from office environments and remote branches of companies, especially as these companies have a large database of users that are distributed in a sophisticated geographic manner. There is a need to improve the performance with the possibility of damaged data to work with on the devices (Yetimler, 2018). A general architecture of edge computing is given in Figure 2.2.

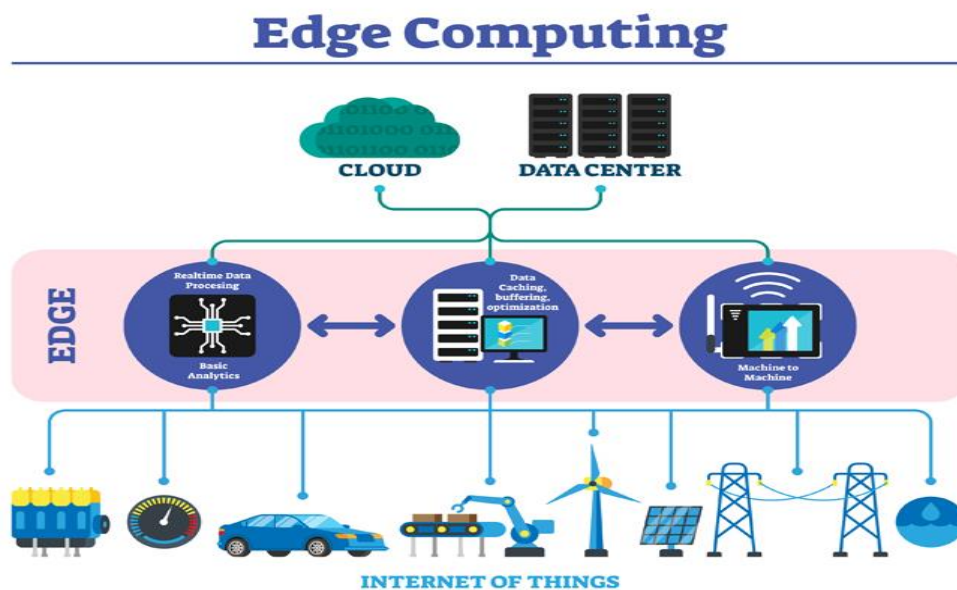


Figure 2.2 How edge computing works (Yetimlier, 2018)

In edge computing security has two sides: On the one hand, some argue that edge computing is the best and base their decision on the evidence that data is not transmitted over a network and that it remains at the source of its creation. The second aspect is that the security is weak in edge computing since the devices are the first elements exposed to risks (Brandon, 2017). Further considerations of network security in edge computing can be reviewed in (Benslimane, 2019).



3. METHODOLOGY

Each node at the edge has certain hardware related properties such as CPU capacity and RAM capacity. Tasks arriving at nodes also have the same set of resource usage requirements (CPU and RAM). In the standard approach, each node executes the tasks arriving to it; however, depending on the frequency of tasks arrived at a node, the capacity of the node may be exceeded, either for the CPU or for the RAM. In this case, the task has to wait until the resources will be available again. Hence, the execution time for the task will be extended. The proposed solution to this deficiency is task offloading which will be investigated in this thesis. The main idea of task offloading is to transfer (assign) a particular task along with its data to another node which has available resources for the specific task. Once the task is executed in the remote node, the result will be transferred back to the original node. The proper implementation of a task offloading strategy at the edge is necessary to prevent tasks to be sent to the cloud directly. This strategy should not impose further costs and latency issues than the case with direct cloud computing. Depending on the characteristics of the node devices at the edge which are the CPU capacity, RAM capacity, and the communication capability in terms of bandwidth, the methodology used has to yield an increased utilization with respect to the case without task offloading. The offloading methods have to be based on a sustainable communication mechanism which has to deal with the transfer of task data and task results among nodes. Furthermore, there is also the need to exchange the utilization levels of individual nodes so as to determine the available nodes for task offloading. This can only be achieved by devising a proper communication scheme for the edge network. In our work, two alternative methods have been used to assess the performance of task offloading:

1. Peer-to-peer communication of tasks and status information,
2. Master-slave system where a master node is responsible for the collection and distribution of data, and for the decision making.

The devices (nodes, computing or edge elements) are either connected to each other by a peer-to-peer scheme, or they communicate with the master node only.

3.1 Peer-to-Peer (P2P)

In P2P communication, each node informs all the other nodes in the network about the CPU and RAM capacity available at the sending node so that each node is aware of the available nodes for task offloading. In case of a congestion, the node will send the task forward to another node which is available and which has the ability to execute the task. The simplest way of selection would be based on a weighted scheme of availability of resources. The process of P2P task offloading is simulated according to the scenario described in the following steps where the time step is taken as 1ms in an arbitrary way.

- 1- Each task has an arrival time at nodes distributed between 1 s and 1000 s. The CPU and RAM loads of tasks are distributed between 5% and 50%, and their duration is distributed between 5 and 20 time units (s).
- 2- Each node has a CPU capacity and RAM capacity set to 100%.
- 3- Nodes will keep a list of tasks that are assigned from other nodes. If there is a task assigned, then the execution starts, and the node updates her capacity and informs about her current status to the other nodes.
- 4- If there is no task assigned from other nodes, the node executes her first task in the own list and calculates utilization (CPU and RAM) and updates her capacity and sends her current state to other nodes.
- 5- Each node has the utilization (CPU and RAM) status of other nodes and according to the status values, nodes will make forward or assign her task that cannot be executed locally.
- 6- Then, the utilization (CPU and RAM) of nodes can be obtained.

The flowchart in Figure 3.1 explains how the P2P system works.

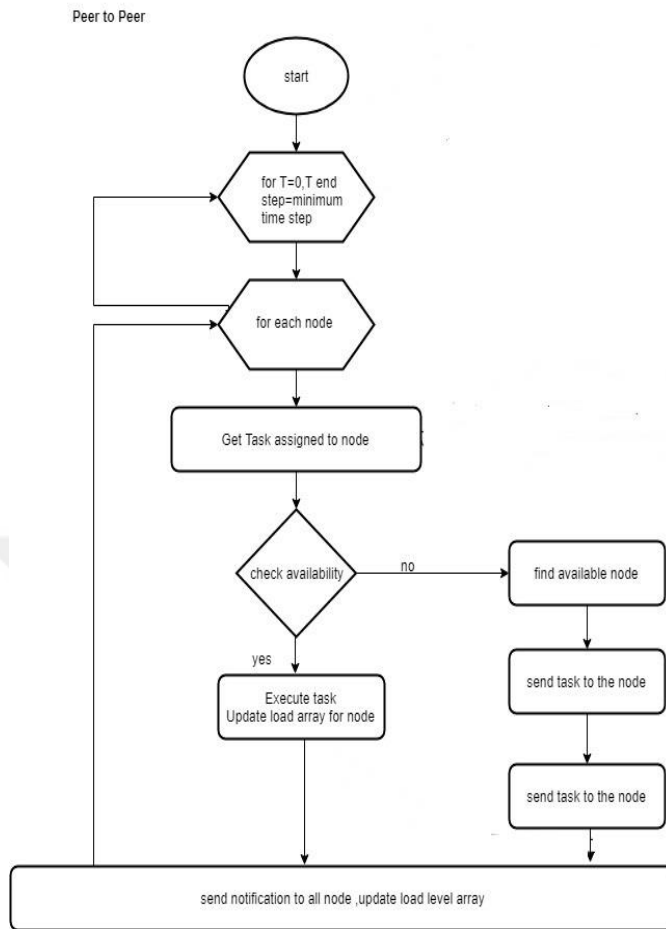


Figure 3.1 Peer-To-Peer flow chart

3.2 Master–Slave (M2S)

In this configuration, each node informs a “predefined” master node about their status. In the case of a congestion, the node also sends the task and the data to the master node which carries out the selection of the node which will execute the task. Hence the communication of the data will be done via the master node which is assumed to have a larger capacity of processing. The process of M2S task offloading is simulated according to the scenario described in the following steps where the time step is taken as 1ms in an arbitrary way.

- 1- Each task has an arrival time at nodes distributed between 1 s and 1000 s. The CPU and RAM loads of tasks are distributed between 5% and 50%, and their duration is distributed between 5 and 20 time units (s).
- 2- Each node has a CPU capacity and RAM capacity set to 100%.
- 3- Nodes will keep a list of tasks that are assigned from the master node. If there is a task assigned, then the execution starts and the node updates her capacity and informs about her current status to the master node.
- 4- If there is no task assigned from other nodes, the node executes her first task in the own list and calculates utilization (CPU and RAM) and updates her capacity and sends her current state to the master node.
- 5- Master node has the utilization (CPU and RAM) status of other nodes and according to the status values, master node will make forward or assign tasks that cannot be executed locally.
- 6- Then, the utilization (CPU and RAM) of nodes can be obtained.

The flowchart in Figure 3.2 explains how the master-slave system works.

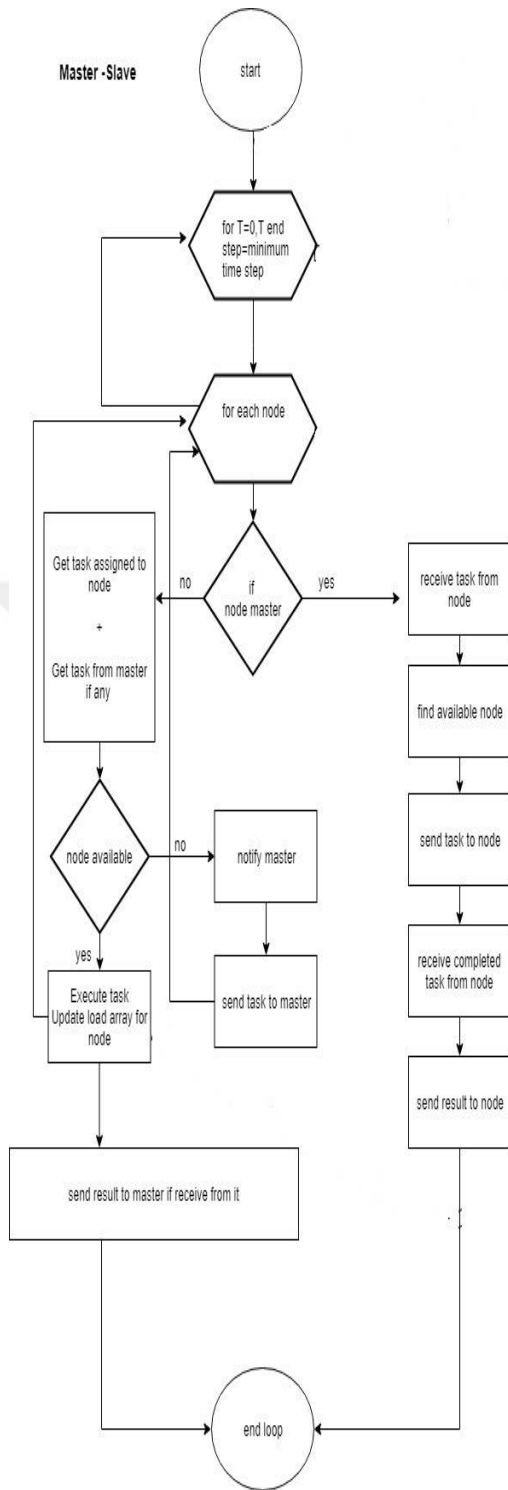


Figure 3.2 Master -Slave flow chart

4. RESULTS

This scenario is based on 1000 tasks distributed over 10 nodes so that each node has 100 tasks. These tasks come with random arrival times between 1 to 1000 seconds and a random duration period of 5s to 20s. More importantly, these tasks have CPU load values and RAM load values between 5% and 50%. The nodes also have 100% CPU capacity and RAM capacity.

4.1 Peer-to-Peer (P2P) results

Based on the simulation of the scenario, the CPU utilization for 10 nodes with 1000 tasks is displayed in Figure 4.1.

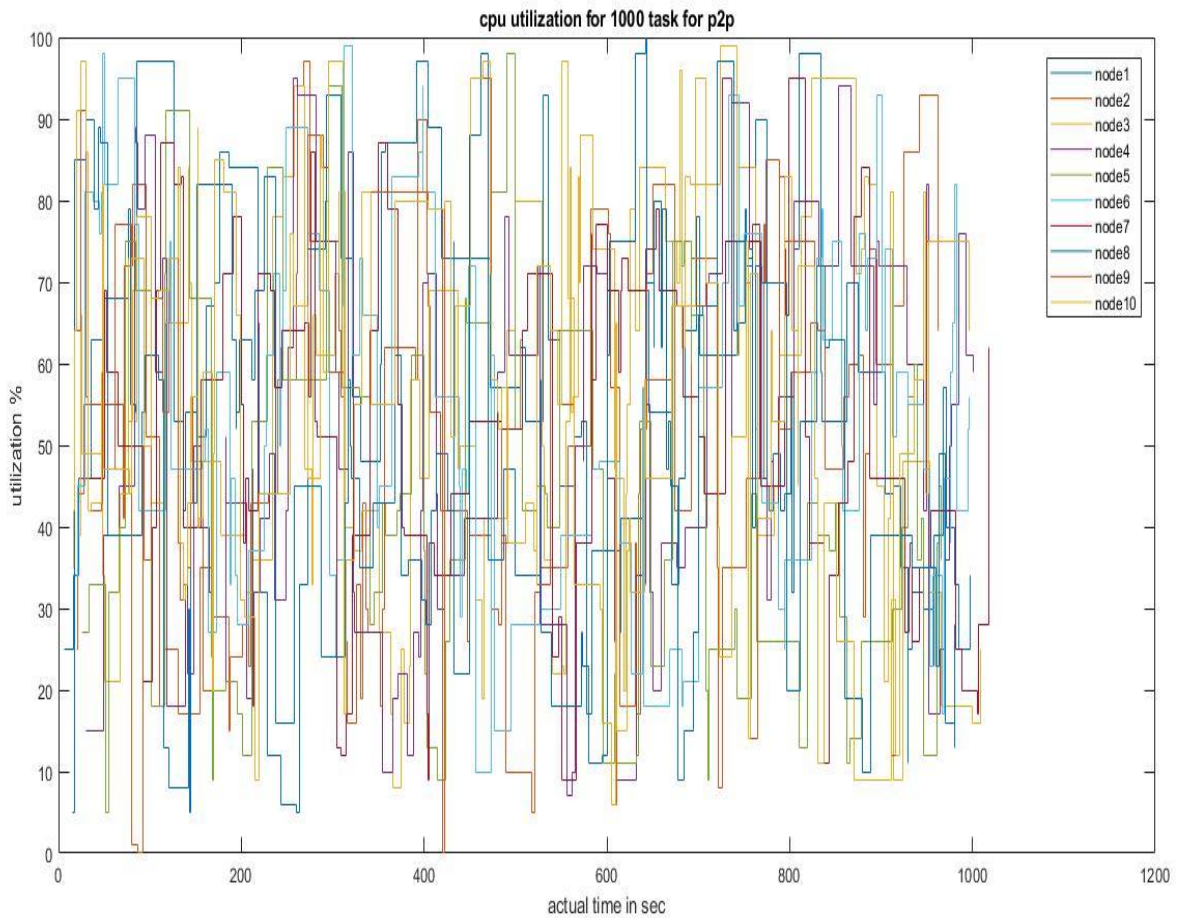


Figure 4.1 P2P CPU utilization for 1000 tasks and 10 nodes

It can be noticed that the total execution time exceeded 1100 seconds and that there are many nodes in which the utilization reached 100%. In Figure 4.2, the RAM usage over the task execution can be seen. As for the CPU, there are many nodes in which the RAM utilization reached 100%.

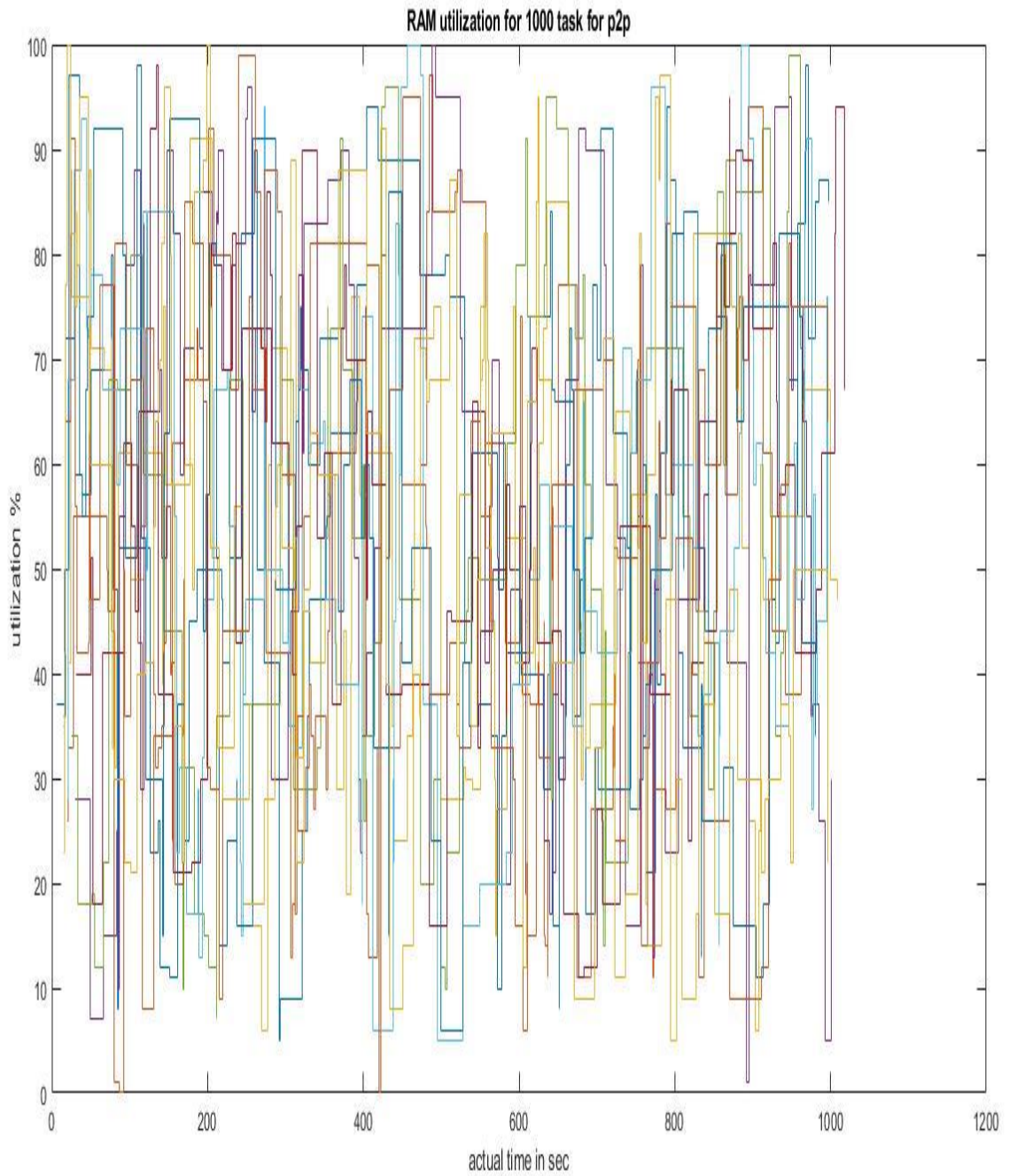


Figure 4.2 Peer-to-Peer RAM utilization for 1000 tasks and 10 nodes

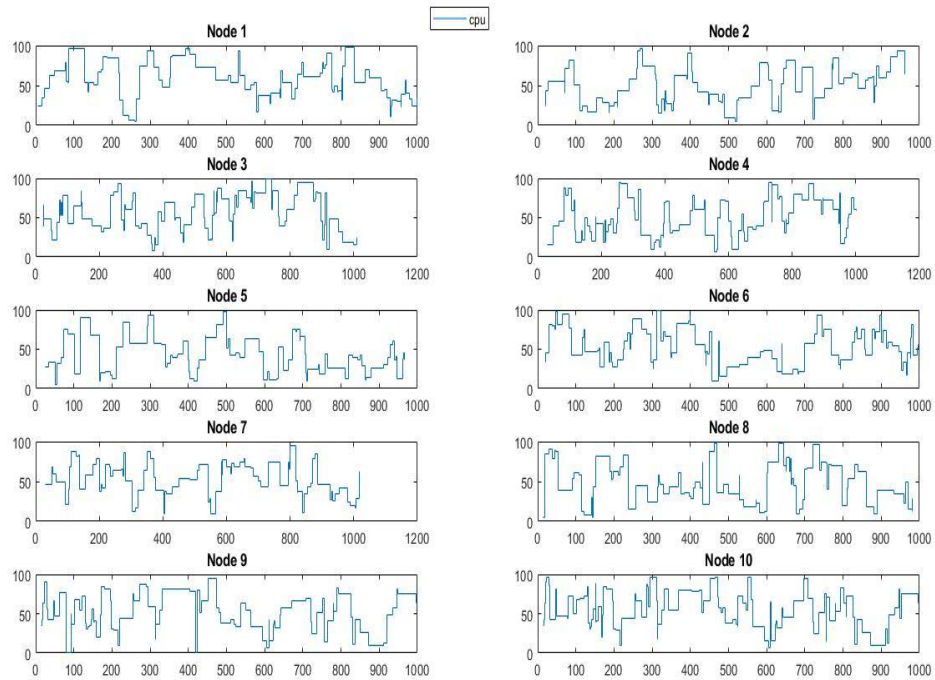


Figure 4.3 Peer-to-Peer subplot CPU utilization for 1000 tasks and 10 nodes

In Figure 4.3 and Figure 4.4, the CPU and RAM utilization, respectively, of each individual node is displayed as a separate panel. These illustrate the work of each node and show how to achieve the implementation of some tasks to 100% within the prescribed period of time.

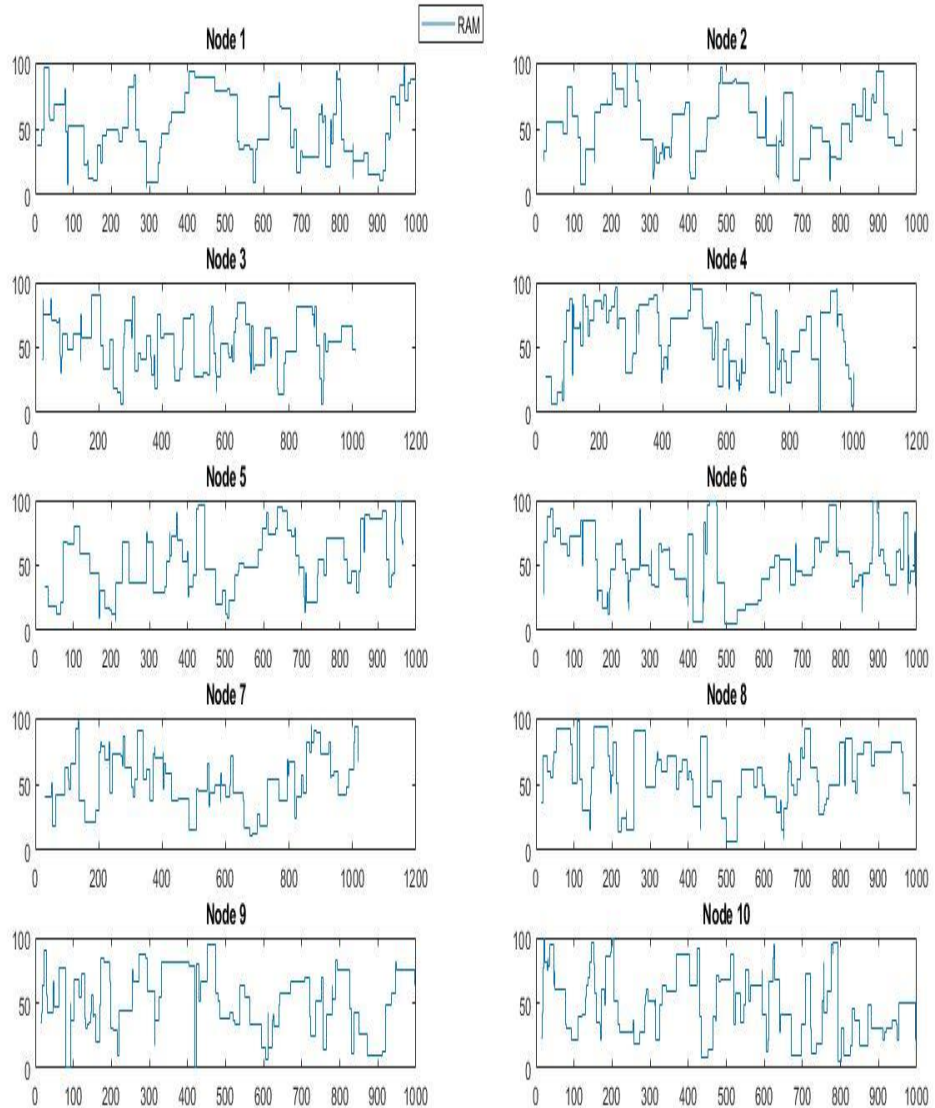


Figure 4.4 Subplot Peer to Peer RAM utilization for 1000 tasks and 10 nodes

In Table 4.1, the offloading performance of the P2P scheme can be investigated. As tasks arrive at the node, some of them are offloaded to other nodes when the original node is busy. Depending on the scenario, some nodes are less loaded by their original list of tasks, so, they are the natural candidates for tasks to be offloaded. It is clear that Node-10 has been the node which was available for task offloading for most of the times.

Table 4.1 P2P EXECUTION OF TASK FOR P2P

| Node | Original | Offloaded from other nodes | Total duration in Sec |
|---|----------|----------------------------|--------------------------|
| 1 | 93 | 0 | 998 Sec |
| 2 | 92 | 0 | 962 Sec |
| 3 | 94 | 6 | 1009 Sec |
| 4 | 93 | 11 | 1001 Sec |
| 5 | 92 | 1 | 966 Sec |
| 6 | 92 | 14 | 996 Sec |
| 7 | 95 | 0 | 1018 Sec |
| 8 | 91 | 6 | 981 Sec |
| 9 | 92 | 9 | 992 Sec |
| 10 | 89 | 30 | 996 Sec |
| Original + From other =1000 task | | | AVERAGE 991.9 Sec |

1. The number of tasks performed in the first node is 93, while the number of tasks not executed is 7 while the number of tasks assigned to it is 0.
2. The number of tasks executed in the second node is 92, while the number of non-executed is 8 while the number of tasks assigned to him and carried out is 0.
3. The number of tasks performed in the third node is 94, while the number of tasks not performed is 6, while the number of tasks assigned to him and executed is 6.
4. The number of tasks performed in the fourth node is 93, while the number of tasks not executed 7 while the number of tasks assigned to 11.
5. The number of tasks performed in the fifth node is 92, while the number of tasks not performed is 8 while the number of tasks assigned to it is 1.
6. The number of tasks performed in the sixth node is 92, while the number of tasks not executed is 8 while the number of tasks assigned to it is 14.
7. The number of tasks performed in the seventh node is 95, while the number of exemptions is 5 while the number of tasks assigned to it is 0.
8. The number of tasks carried out in the eighth node is 91, while the number of tasks not executed 9 while the number of tasks assigned to 6.

9. The number of tasks performed in the ninth node is 92, while the number of tasks not executed 8 while the number of tasks assigned to 9.

10. The number of tasks performed in the tenth node is 89, while the number of tasks not implemented 11, while the number of tasks assigned to him and carried out is 30 tasks.

The highest total duration is reached in the seventh node, which amounted to 1018 seconds, while the lowest total duration is reached in the second node, which reached 991.9 seconds. The average total duration reached the P2P system is 1002.1 seconds.

Table 4.2 P2P Average RAM and CPU utilization

| Nodes | Average CPU | Average RAM |
|----------------------|--------------------|--------------------|
| 1 | 54.76 | 51 |
| 2 | 47.3 | 50.4 |
| 3 | 54.5 | 50.5 |
| 4 | 49 | 56 |
| 5 | 41 | 52.5 |
| 6 | 54 | 52.18 |
| 7 | 51 | 56 |
| 8 | 48 | 53.19 |
| 9 | 48.45 | 48.45 |
| 10 | 54.70 | 51.12 |
| Average Total | 50.5 | 52.15 |

Table 4.2 displays the CPU and RAM utilization of all tasks in all nodes. The first node has recorded the highest utilization value that reached about 54.76 % but, the fifth node had the lowest CPU utilization rate of 41 %. On the other hand, the random-access memory (RAM) of the eighth node reached to the peak utilization value of 53.19 %, while the lowest utilization was 48.45% in the ninth node. In the

end, the average CPU and RAM utilization for all nodes was calculated as 61.8% and 59% respectively.

4.2 Master-Slave (M2S) results

In Figure 4.5 and Figure 4.6, the CPU and RAM utilization values for all nodes are given. It can be noticed that the time interval in calculating the rate of CPU usage exceeded 1100 seconds and that there are many nodes in which the utilization reached 100%. Similarly, the time interval in calculating the rate of RAM usage exceeded 1100 seconds and that there are many nodes in which the utilization reached 100%.

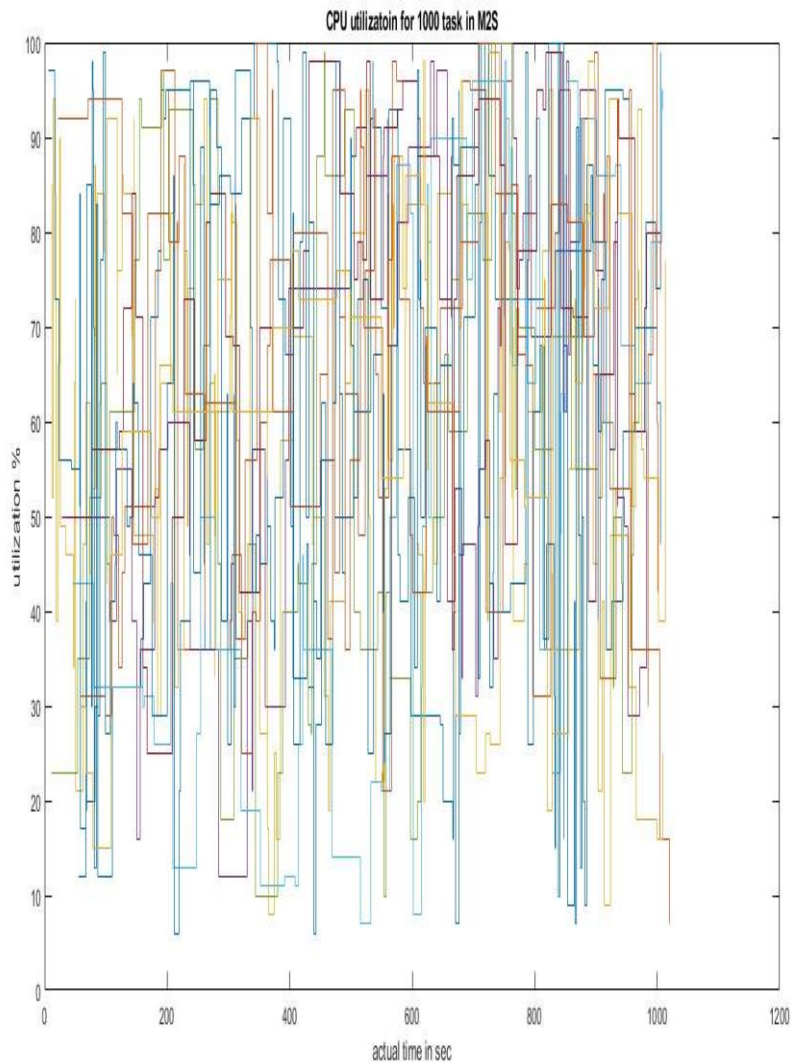


Figure 4.5 Master -Slave CPU utilization

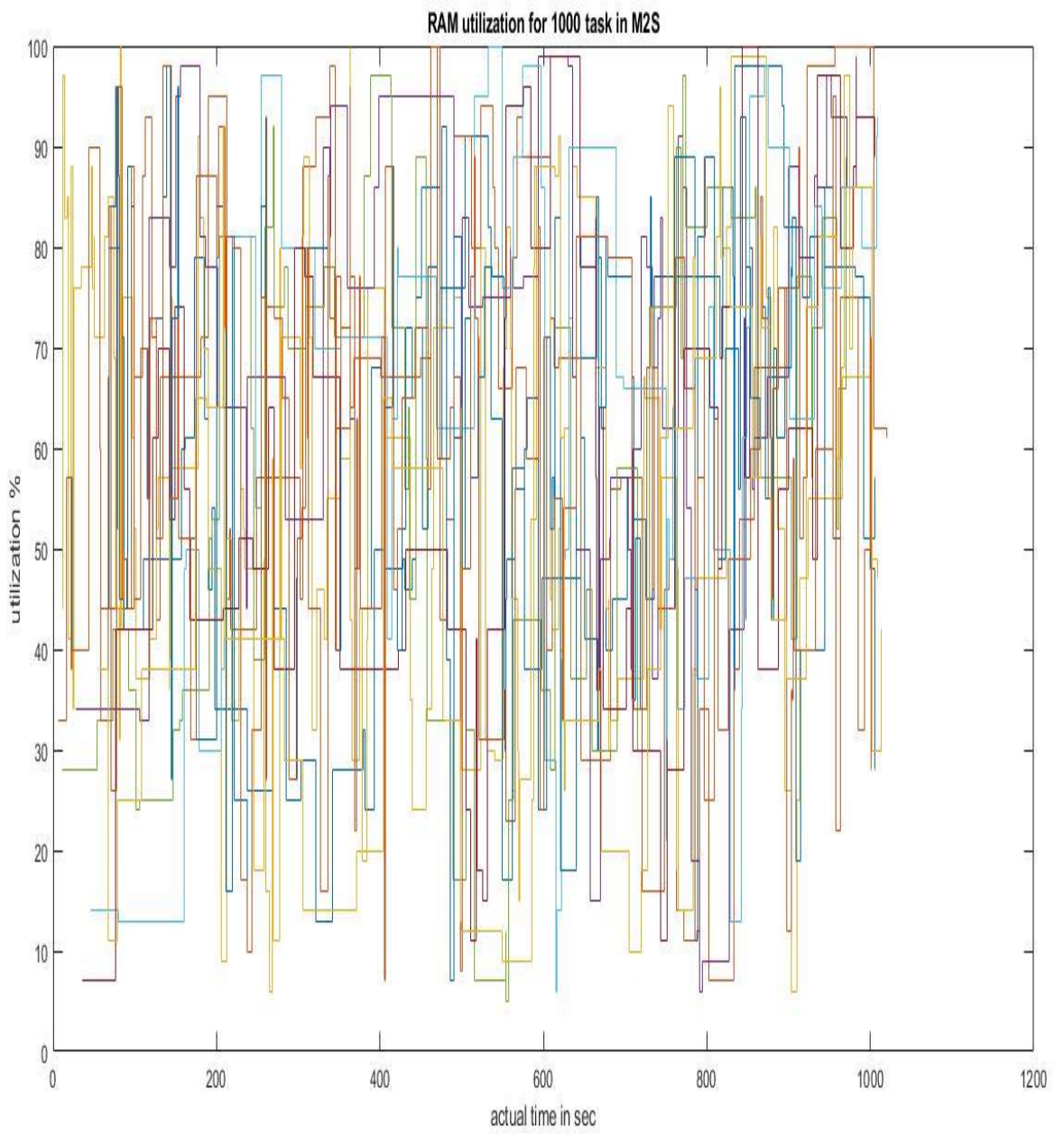


Figure 4.6 Master -Slave RAM utilization

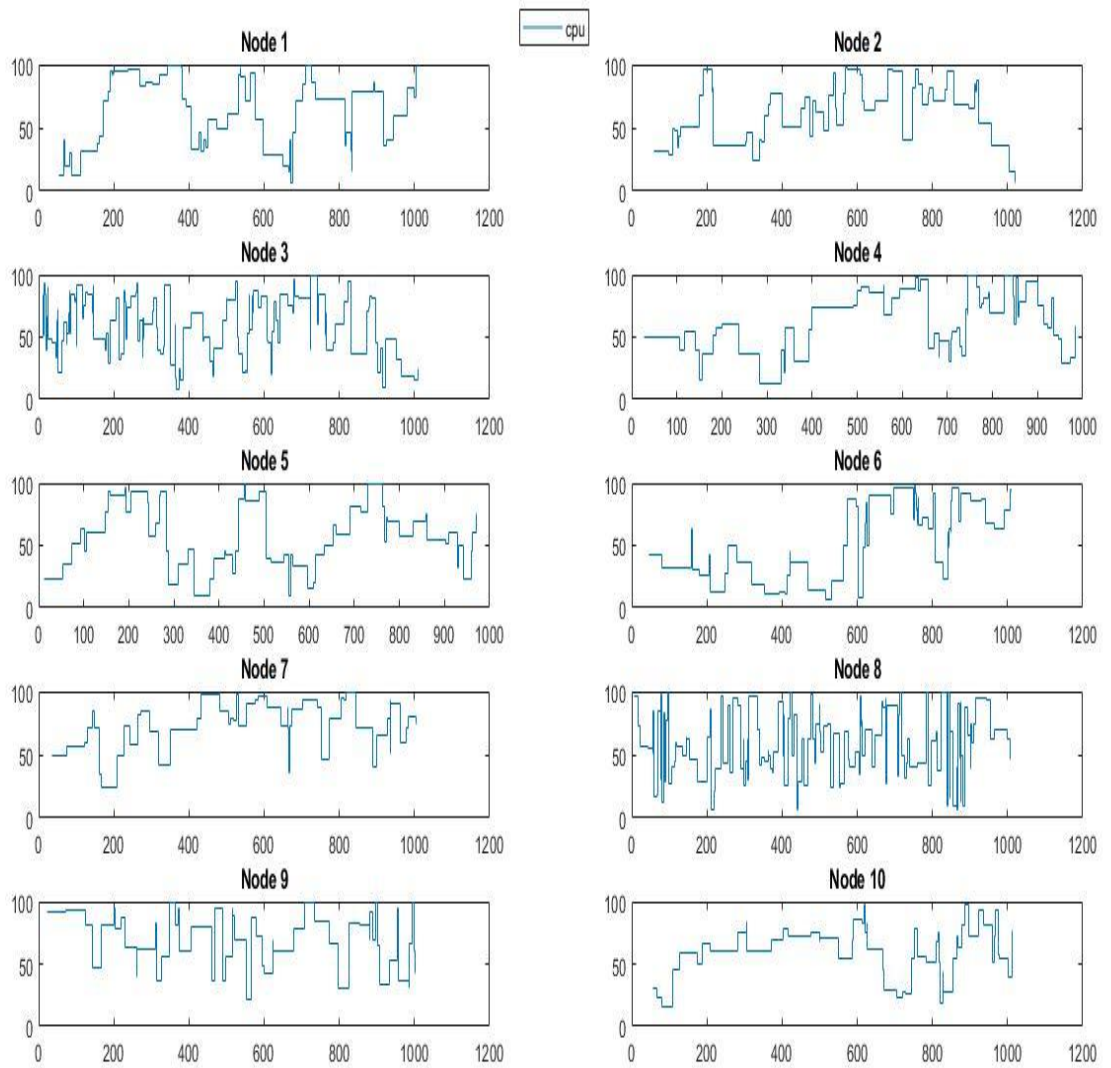


Figure 4.7 Master-Slave subplot CPU utilization

The utilization of CPU and RAM for each node can be seen in Figure 4.7 and Figure 4.8 respectively. Here, we have divided the original form into 10 nodes, these illustrate the work of each node and show how to achieve the implementation of tasks to 100% within a period of time.

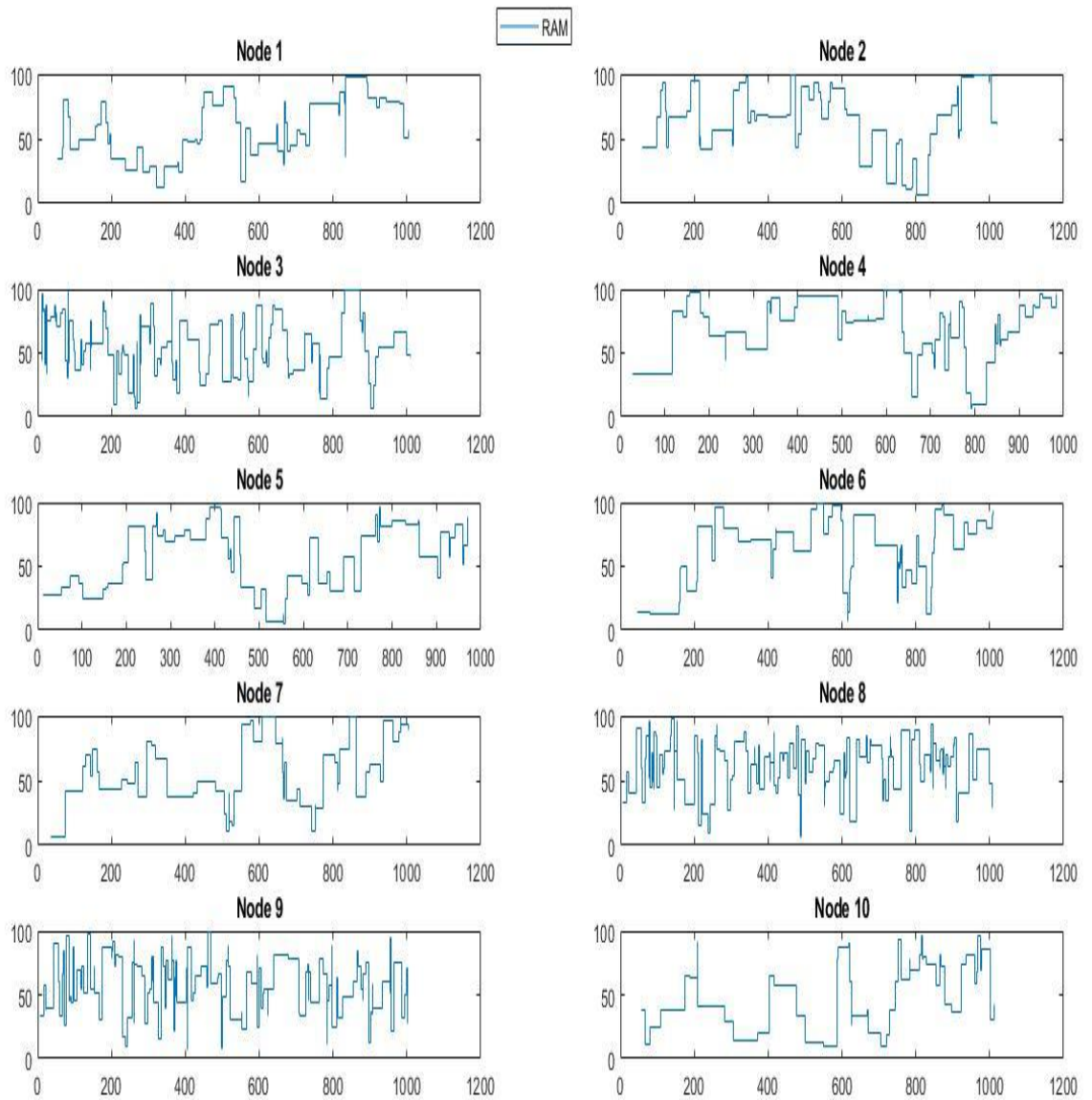


Figure 4.8 Master – Slave subplot RAM utilization

In Table 4.3 the utilization of all tasks in any node was calculated and the average values of use are displayed for the CPU and RAM. The seventh node has recorded the highest utilization value that reached about 74.3%, but, the fifth node had the lowest CPU utilization rate of 55.9%. On the other hand, the random-access memory (RAM) of the fourth node reached a maximum average utilization of 69%, while the lowest utilization has been reached as 53.1% in the tenth node. In the end, the average CPU and RAM utilization for all nodes was calculated as 61.8% and 59%.

Table 4.3 Master – Slave average CPU and RAM for each node

| Nodes | Average CPU utilization | Average RAM utilization |
|----------------|--------------------------------|--------------------------------|
| 1 | 59.9 % | 56.4 % |
| 2 | 63.7 % | 63.5 % |
| 3 | 58.7 % | 54 % |
| 4 | 62.2 % | 69 % |
| 5 | 55.9 % | 56.3 % |
| 6 | 56.4 % | 60.8 % |
| 7 | 74.3 % | 57.1 % |
| 8 | 56.9 % | 62.2 % |
| 9 | 68.9 % | 57.7 % |
| 10 | 60.8 % | 53.1 % |
| average | 61.8 % | 59 % |

In Table 4.4, the offloading performance of the Master-Slave scheme can be investigated. As tasks arrive at the node, some of them are offloaded to other nodes when the original node is busy. Depending on the scenario, some nodes are less loaded by their original list of tasks, so, they are the natural candidates for tasks to be offloaded. It is clear that Node-2 has been the node which was available for task offloading for most of the times.

Table 4.4 Master -Slave task execution

| Nodes | Original task executed | Task executed from other | Total duration in Sec |
|--------------|-------------------------------|---------------------------------|------------------------------|
| 1 | 63 | 0 | 1005 sec |
| 2 | 60 | 245 | 1020 sec |
| 3 | 94 | 83 | 1009 sec |
| 4 | 65 | 0 | 983 sec |
| 5 | 66 | 0 | 970 sec |
| 6 | 63 | 0 | 1009 sec |

| | | | |
|---|-----------|-----------|---------------------------|
| 7 | 60 | 0 | 1005 sec |
| 8 | 67 | 0 | 1006 sec |
| 9 | 59 | 0 | 1001 sec |
| 10 | 59 | 16 | 1013 sec |
| Original + From other =1000 task | | | Average=1002.1 sec |

1. The number of tasks performed in the first node is 63, while the number of tasks not executed is 37 while the number of tasks assigned to it is 0.
2. The number of tasks executed in the second node is 60, while the number of non-executed is 40 while the number of tasks assigned to him and carried out is 245 tasks.
3. The number of tasks performed in the third node is 94, while the number of tasks not performed is 6, while the number of tasks assigned to him and executed is 83.
4. The number of tasks performed in the fourth node is 65, while the number of tasks not executed 35 while the number of tasks assigned to 0.
5. The number of tasks performed in the fifth node is 66, while the number of tasks not performed is 34 while the number of tasks assigned to it is 0.
6. The number of tasks performed in the sixth node is 63, while the number of tasks not executed is 27 while the number of tasks assigned to it is 0.
7. The number of tasks performed in the seventh node is 60, while the number of exemptions is 40 while the number of tasks assigned to it is 0.
8. The number of tasks carried out in the eighth node is 67, while the number of tasks not executed 33 while the number of tasks assigned to 0.
9. The number of tasks performed in the ninth node is 59, while the number of tasks not executed 41 while the number of tasks assigned to it.
10. The number of tasks performed in the tenth node is 59, while the number of tasks not implemented 41, while the number of tasks assigned to him and carried out is 16 tasks.

The highest total duration has been reached in the second node, which amounted to 1,020 seconds, while the lowest total duration was reached in the fifth node, which reached 970 seconds. The average total duration for the master-slave system is 1002.1 seconds.

Table 4.5 Comparison between Peer-to-Peer and Master-Slave

| Topology | Average CPU | Average RAM | Average CPU utilization | Average RAM utilization | Task from other nodes | Average duration time (Sec) |
|--------------|-------------|-------------|-------------------------|-------------------------|-----------------------|-----------------------------|
| PEER TO PEER | 50.57 | 52.15 | 64 | 71 | 77 | 991.9 |
| MASTER-SLAVE | 61.81 | 59 | 81.7 | 77.3 | 344 | 1002.1 |

Table 4.5 shows the comparison between peer-to-peer topology and master-slave topology. It can be concluded that the master-slave topology exhibits a higher average duration time (1002.1s versus 991.9s) and also the number of tasks offloaded to other nodes is 344 versus 77 in peer-to-peer topology.

5. CONCLUSIONS

In this thesis, we considered the task offloading problem of busy resources in edge computing. The problem was evaluated based on the utilization rate and execution time for devices in the edge network. The research first considered the move from the centralized cloud computing architecture to the edge computing structure that would allow to easily access resources that are managed in the edge of the network. Master-Slave topology was worse than peer-to-peer topology. This was first shown in the number of tasks executed in the above two models. It has been noted in the topology of master-slave that the utilization and execution time are higher than the values in the topology of peer-to-peer and also number of tasks assigned to other nodes reached a much higher value in the master-slave indicating a more overloaded system. The system was modeled using the theoretical knowledge from the literature review. MATLAB simulations have been carried out for the scenarios where the same set of tasks has been applied for both the communication schemes.

6. REFERENCES

- Ahmed, E. and Rehmani, M. H., (2017). Mobile Edge Computing Opportunities, solutions, and challenges. *Future Generation Computer Systems*, pp. 59-63.
- Ben Issa, S. and Tu, Y., (2017). Integrated multi-resource planning and scheduling in an engineering project. *Journal of Project Management*, Volume 2, pp. 11-26.
- Benslimane N., (2019). Living on the Edge How Edge Computing is Redefining the Network Landscape. <https://www.idg.com/blog/living-on-the-edge-how-edge-computing-is-redefining-the-network-landscape>
- Brandon B., (2017). What is edge computing and how it's changing the network? (n.d.). Retrieved from <https://www.networkworld.com/article/3224893/what-is-edge-computing-and-how-it-s-changing-the-network.html>
- Buyya, R., and Dastjerdi, A., (2016). Internet of Things Principles and Paradigms. In *Internet of Things*, pp. 36-50
- Chauvet, F., Proth, J.-M. and Soumare, A., (2000). The Simple and Multiple Job Assignment Problems. *International Journal of Production Research*, 38(14), pp. 3165-3179.
- Dao, N.-N. et al., (2018). Pattern-Identified Online Task Scheduling in Multitier Edge Computing for Industrial IoT Services. *Mobile Information Systems*, pp. 9-19.
- De Filippi, P. and McCarthy, S., (2012). Cloud Computing Centralization and Data Sovereignty. *European Journal of Law and Technology*, 3(2), pp. 1-18.
- Dil Afroz, H. and Hossein, M. A., (2017). New Proposed Method for Solving Assignment Problem and Comparative Study with the Existing Methods. *Journal of Mathematics*, 13(2), pp. 84-88.
- Elijah, O., Rahman, T. A., Orikumhi, I. and Leow, C. Y., (2018). An Overview of Internet of Things (IoT) and DataAnalytics in Agriculture Benefits and Challenges. *IEEE Internet of Things Journal*, 5(5), pp. 3758-3773.

- Gawanmeh, A., Alomari, A. and April, A., (2017). Optimizing resource allocation scheduling in cloud computing services. *Journal of Theoretical and Applied Information Technology*, 95(1), pp. 31-39.
- Gezer, V., Um, J. and Ruskowski, M., (2018). An Introduction to Edge Computing and A Real-Time Capable Server Architecture. *International Journal of Intelligent Systems*, 11(1and2), pp. 105-114.
- Gusain, S. and Kumar, R., (2014). An Optimization in Cloud Computing for Job Forecast. *International Journal of Computer Science and Mobile Computing*, 3(5), pp. 304-309.
- Hassani, Z. I. M., El Barkany, A., Abdelouahhab, J. and Ikram, E. A., (2018). New Approach to Integrate Planning and Scheduling of Production System Heuristic Resolution. *International Journal of Engineering Research in Africa*, Volume 39, pp. 156-169.
- Hassan, Z., Ali, H. A. and Badawy, M. M., (2015). Internet of Things (IoT) Definitions, Challenges, and Recent Research Directions. *International Journal of Computer Applications*, 128(1), pp. 975-987.
- Hyari, K. H., El-Rayes, K. and Asce, M., (2006). Optimal Planning and Scheduling for Repetitive Construction Projects. *Journal of Management in Engineering*, 22(1), pp. 11-19.
- Jennings, B. and Stadler, R., (2015). Resource management in clouds Survey and research challenges. *Journal of Network and Systems Management*, 23(3), pp. 567-619.
- Jindal, F., Jamar, R. and Churi, P., (2018). Future and Challenges of Internet of Things. *International Journal of Computer Science and Information Technology*, 10(2), pp. 13-25.
- Josilo, S., (2018). Decentralized Algorithms for Resource Allocation in Mobile Cloud Computing Systems, Stockholm, Sweden KTH School of Electrical Engineering and Computer Science.
- Kabiru, S., Saidu, B. M., Abdul, A. Z. and Ali, U. A., (2017). An Optimal Assignment Schedule of Staff-Subject Allocation. *Journal of Mathematical Finance*, Volume 805-820, p. 7.

- Kang, Y.-M., Han, M.-R., Han, K.-S. and Kim, J.-B., (2015). A Study on the Internet of Things (IoT) Applications. *International Journal of Software Engineering and Its Applications*, 9(9), pp. 117-126.
- Karimi, K. and Atkinson, G. (2013). What the Internet of Things (IoT) Needs to Become a Reality. Technical report, Freescale
- Krogh, S., (2013). Cloud Computing A Social Relations Perspective. *Journal of Information Architecture*, 5(1-2), pp. 21-30.
- Liang, B., (2017). Mobile Edge Computing, Toronto, Canada University of Toronto.
- Liu, H. et al., (2017). Mobile Edge Cloud System Architectures, Challenges, and Approaches. *IEEE Systems Journal*, 12(3), pp. 2495 - 2508.
- More, P. and Kulkarni, J., (2017). Fog Computing. *International Research Journal of Engineering and Technology*, 4(2), pp. 1113-1116.
- Nzanywayingoma, F. and Yang, Y., (2017). Efficient Resource Management techniques in Cloud Computing Environment A Review and discussion. *Telkominika*, 15(4), pp. 1918-1928.
- Pardalos, P. M., 1996. Continuous Approaches to Discrete Optimization Problems. Boston, MA Nonlinear Optimization and Applications. Springer.
- Prabhu, L. G. K. S. L. N. and Rao J., V., (2015). Resource Allocation in Mobile Cloud Computing using Optimization Techniques. *International Journal of Wireless Communications and Networking Technologies*, 4(2), pp. 30-36.
- Prakash, P., Darshaun, K. G., Yaazhlene, P. and Ganesh, M. V., (2017). Fog Computing Issues, Challenges, and Future Directions. *International Journal of Electrical and Computer Engineering*, 7(6), pp. 3669-3673.
- Silva, C. A., Aquino Jr., G. S., Melo, S. R. M. and Egídio, D. J. B., 2019. A Fog Computing-Based Architecture for Medical Records Management. *Wireless Communications and Mobile Computing*, Issue 1968960, pp. 1-16.
- Sonkar, S. K. and Kharat, M. U., (2015). A Survey on Resource Management in Cloud Computing Environment. *International Journal of Advanced Trends in Computer Science and Engineering*, 4(4), pp. 48-51.
- Supian, S., Sri, W., Nahar, J. and Subiyanto, (2018). Optimization of Personnel Assignment Problem Based on Traveling Time by Using Hungarian Methods

Case Study on the Central Post Office Bandung. Medan Indonesia, The Indonesian Mathematical Society Section Aceh, and Sumatera Utara.

Weintraub, E. and Cohen, Y., (2017). Multi-Objective Optimization of Cloud Computing Services for Consumers. International Journal of Advanced Computer Science and Applications, 8(2), pp. 139-147.

Whitake B., (2017). Cloud Edge Computing Beyond the Data Center - OpenStack is open source software for creating private and public clouds.

<https://www.openstack.com>

Yetimler E., (2018). What is Edge Computing? What is the Difference from Cloud Computing and Fog Computing? <https://www.karel.com.tr/blog/edge-computing-nedir-cloud-computing-ve-fog-computingden-farki-nedir>.

CURRICULUM VITAE

Personal Information

Name Surname MOSTAFA ZIADOON IBRAHIM IBRAHIM
Place and Date of Birth IRAQ - 11\08\1990

Education

Undergraduate Education B. Sc. Computer Engineering
Graduate Education M. Sc. Computer Engineering
Foreign Language Skills Arabic, English

Work Experience

Name of Employer and Dates of Employment:

EARTHLINK Internet Service Company in IRAQ: From 2011 To 2016

Contact

Telephone 00905050947173
E-mail Address M.ziedoon@gmail.com