PROFILING AND DYNAMIC PARTITIONING FOR OFFLOADING IN MOBILE EDGE CLOUD COMPUTING

A PROJECT REPORT

Submitted by

B.BALAJI 2015115009

M.MOHAMMED HASIF 2015115029

K.SARGURU 2015115049

submitted to the Faculty of

INFORMATION AND COMMUNICATION ENGINEERING

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY

COLLEGE OF ENGINEERING, GUINDY

ANNA UNIVERSITY

CHENNAI 600 025

OCTOBER 2018

ANNA UNIVERSITY

CHENNAI - 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled PROFILING AND PARTITIONING FOR MOBILE EDGE CLOUD COMPUTING is the bonafide work of B.BALAJI, M.MOHAMMED HASIF and K.SARGURU who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

PLACE:CHENNAI
DATE: 25.10.2018

Dr.M.VIJAYALAKSHMI
ASSISTANT PROFESSOR (Sr.Gr)
PROJECT GUIDE
DEPARTMENT OF IST, CEG
ANNA UNIVERSITY
CHENNAI 600025

COUNTERSIGNED

Dr.S.SASWATI MUKHERJEE
HEAD OF THE DEPARTMENT
DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600025

ABSTRACT IN TAMIL - 1 page

ABSTRACT

Offloading is necessary due to hardware limitations of a computer system handling a particular task on its own. Computational Offloading is a solution to augment system's capabilities by migrating computation to more resourceful computers (i.e.,servers). Offloading decision depends on many parameters such as the network bandwidth, server speed, edge speed, available memory, server load and the amount of data exchanged between mobile systems, mobile edge and server. Computational offloading refers to the transfer of certain computation tasks to an external platform, such as a mobile edge or a cloud. Mobile edge Cloud Computing (MECC) has becoming an attractive solution for augmenting the computing and storage capacity of Mobile Devices (MDs) by exploiting the available resources at the network edge thereby eliminating the intensive tasks to be offloaded to a fog node or a remote cloud server. The project work presents the formation of mobile edge cloud computing setup by integrating raspberry pi as fog node and using Digital Ocean as cloud server. The profiler consists of three components namely device profiler, application profiler and network profiler which are necessary in making decision for offloading. Device profiler gives the details about battery level, number of cpu cores, cpu core utilisation and cpu frequency. A program profiler collects characteristics of applications, e.g., the execution time, hit ratio, etc. A network profiler collects information about wireless connection status and available bandwidth. Application partitioning is a technique of splitting the application into separate tasks, while preserving the semantics of the original application. Method based partitioning is performed here.

ACKNOWLEDGEMENT

We wish to record our deep sense of gratitude and profound thanks to our project supervisor **Dr. M. Vijayalakshmi**, Assistant Professor (Sr.Gr.), Department of Information Science and Technology, College of Engineering, Guindy for her keen interest, inspiring guidance, constant encouragement with our work during all stages, to bring this thesis into fruition.

We are extremely indebted to **Dr. Saswati Mukherjee**, Professor and Head of the Department of Information Science and Technology, Anna University, Chennai, for extending the facilities of the Department towards our project and for her unstinting support.

We express our sincere thanks to the course instructors **Dr. P. Yogesh**, Associate Professor, **Mr. B. Yuvaraj**, Teaching Fellow, **Ms. Amala**, Research Scholar and **Ms. Nancy**, Research Scholar for helping us in completion of this project.

We express our sincere thanks to the review committee panel members **Dr. K. Vani**, Associate Professor, **Dr. P. Yogesh**, Associate Professor, **Dr. S. Bama**, Assistant Professor, **Dr. N. Thangaraj**, Assistant Professor and **Ms. B. Yuvaraj**, Teaching Fellow for their critical reviews throughout the course of our project.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

BALAJI B **MOHAMMED HASIF M** **SARGURU K**

Contents

ABSTRACT IN TAMIL iii
ABSTRACT iv
ACKNOWLEDGEMENT v
LIST OF FIGURES 1

1 INTRODUCTION **2**
1.1 NEED FOR MOBILE EDGE CLOUD COMPUTING 2
1.2 FORMATION OF MOBILE EDGE CLOUD COMPUTING SETUP 2

2 LITERATURE SURVEY **4**
2.1 EXISTING SYSTEM 4
2.2 OBSERVATION AND MOTIVATION 5

3 DESIGN **6**
3.1 ARCHITECTURE DIAGRAM 6
3.2 PROFILER 7
3.2.1 Device Profiler 8
3.2.2 Application Profiler 8
3.2.3 Network Profiler 8
3.3 PARTITIONING 9

4 IMPLEMENTATION AND RESULTS **10**
4.1 PROFILING ALGORITHM 10
4.2 PARTITIONING ALGORITHM 12

5 CONCLUSION AND FUTURE WORK **14**
5.1 CONCLUSION 14
5.2 FUTURE WORK 14

REFERENCES 15

List of Figures

FIGURE NO.	TITLE	PAGE NO.
1.1	Mobile Edge Cloud Computing	3
3.1	Overall Architecture Diagram for Multitask Computation Offloading and Scheduling in MECC	6
3.2	Profiler for Mobile Edge Cloud Computing	7
4.1	Profiler Information	11
4.2	Dependency graph created by partioning algorithm	13

Chapter 1

INTRODUCTION

1.1 NEED FOR MOBILE EDGE CLOUD COMPUTING

Nowadays, Mobile Devices (MDs) such as smartphones, tablet, computers, wearable devices and etc., are playing more and more important role in our daily life. Moreover, mobile applications running on MDs, such as immersive gaming, human-computer interaction, often demand stringent delay and processing requirements. Consequently, despite the tremendous improvement of MDs' processing capacity, executing applications solely on a MD is still incapable of providing satisfactory Quality of Experience (QoE) to users. Besides, MDs are energy constrained, which further increases the tension between resource limited MDs and computation intensive mobile applications. Fortunately, with the advanced networking and multitask technology, offloading part of the work load of an application (or simply put it as a task) from MDs to a remote cloud or a nearby mobile edge cloud (also known as cloudlet) provides a promising alternative to reduce the task execution time as well as prolong the lifetime of MDs. A mobile edge cloud consists of either one edge server or a set of devices that serve mobile users cooperatively. Though the mobile edge cloud is less powerful compared with a remote cloud, as it is located at the edge of the network the transmission latency between a MD and the mobile edge cloud is much lower than that of the remote cloud. Besides, a mobile edge cloud can explore the idle computing and storage resources at the network edge efficiently. In this work, we mainly focus on profiling and partitioning for multitask computation offloading in Mobile Edge Cloud Computing.

1.2 FORMATION OF MOBILE EDGE CLOUD COMPUTING SETUP

Mobile Edge Cloud Computation system setup is formed by integrating Raspberry Pi as mobile edge and the Digital Ocean as cloud server. The setup for Mobile Edge Cloud Computation system is depicted in figure 1.1.

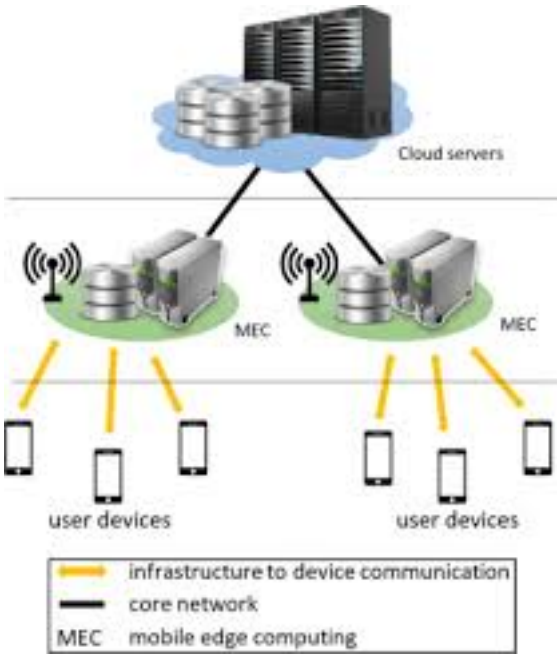

Figure 1.1: Mobile Edge Cloud Computing

The device is connected to Raspberry Pi through wireless medium and the Raspberry Pi is connected to the cloud server. The end user device is connected to cloud server through Raspberry Pi. Any data that is uploaded sent to cloud from user device will reach into the mobile edge (Raspberry Pi), the mobile edge will transfer that data to the cloud server.

Chapter 2

LITERATURE SURVEY

2.1 EXISTING SYSTEM

Weiwei Chen discussed multi-user multi-task offloading scheduling schemes in a renewable mobile edge cloud system. As the mobile edge cloud is composed of a set of WDs with relatively low processing ability (the processing ability is not as high as that of a server), scheduling schemes needs to map the workload from a MD to multiple WDs. Moreover, scheduling schemes also needs to handle uncertain energy supply at each WD properly so as to make the best of the mobile edge cloud system. In detail, the scheduling algorithms determine the energy harvesting strategy, a set of offloading requests to be admitted, and a sub-set of WDs to compute the workload for the admitted offloading request so as to maximize the overall system utility. We show that it is NP-hard to compute a centralized optimal solution, and hence adopt a game theoretic approach for achieving efficient computation offloading in a distributed manner. Xu Chen shown that multiuser computation offloading is NP-hard to compute a centralized optimal solution, and hence he adopted a game theoretic approach for achieving efficient computation offloading in a distributed manner.He formulated the distributed computation offloading decision making problem among mobile device users as a multi-user computation offloading game. Hamed Shah-Mansouri addressed the issues in selection of tasks to be offloaded to cloud servers and the optimal price of cloud services. He revealed that the scheduler effectively balances the tradeoff between the energy consumption and delay.

2.2 OBSERVATION AND MOTIVATION

A literature survey was done by surveying three research papers. The limitations and the knowledge gained from the papers will help us to create a better system. To exploit the computing capacity at the green mobile edge cloud thoroughly, the scheduling algorithms should match the offloading energy consumption at the mobile edge cloud to its harvestable energy. It is observed that faster the computation and lower the delay better the user experience. Mobile Cloud Systems can be efficiently utilised by reducing energy consumption, exploring the idle devices and storage resources at the network edge which leads to increasing the revenue based on the offloading of the task to the edge or cloud.

Chapter 3

DESIGN

3.1 ARCHITECTURE DIAGRAM

The overall architecture which comprises of the application, profiler, partitioner, mobile edge and cloud. The profilers consists the device profiler, network profiler and the program profiler.

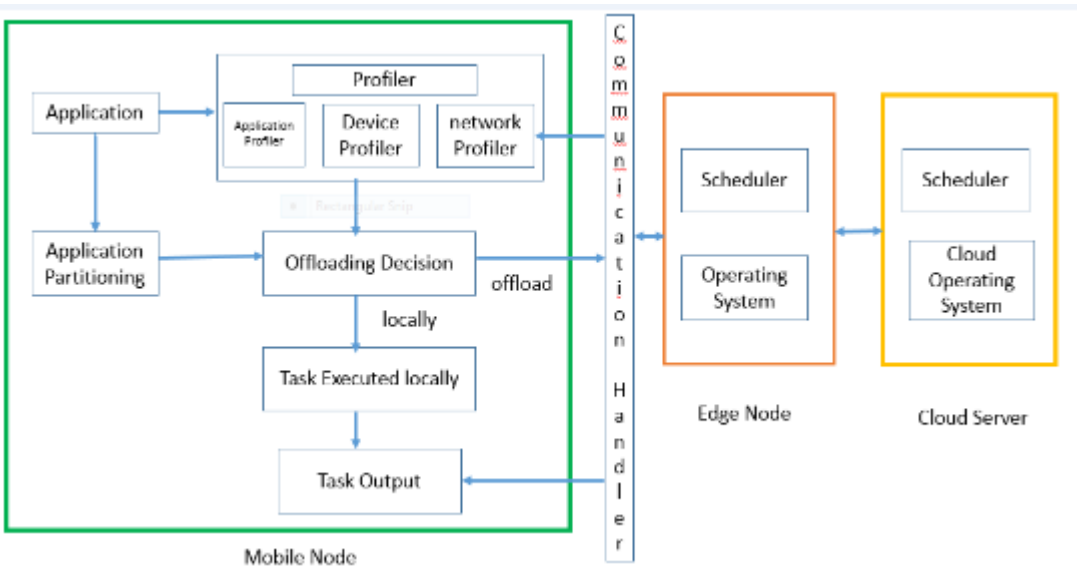

Figure 3.1: Overall Architecture Diagram for Multitask Computation Offloading and Scheduling in MECC

The profilers send the various parameters which are required for the offloading decision maker. After the application is profiled, partitioning takes place by methods partitioning which can be offloaded. Offloading decision algorithm is applied and the list of methods to be executed in mobile edge, cloud and mobile device are obtained. The methods which are to be executed in the mobile edge or cloud are sent to the communication handler. The communication handler transfers the method from the mobile device to the mobile edge or

cloud. The methods are executed in the mobile edge or cloud and the output is integrated in the mobile side.

3.2 PROFILER

The Profiler consists of three profilers namely Device Profiler, Application Profiler and Network profiler which are necessary in making decision for offloading and scheduling. The details of the Profiler is shown in figure 3.2.

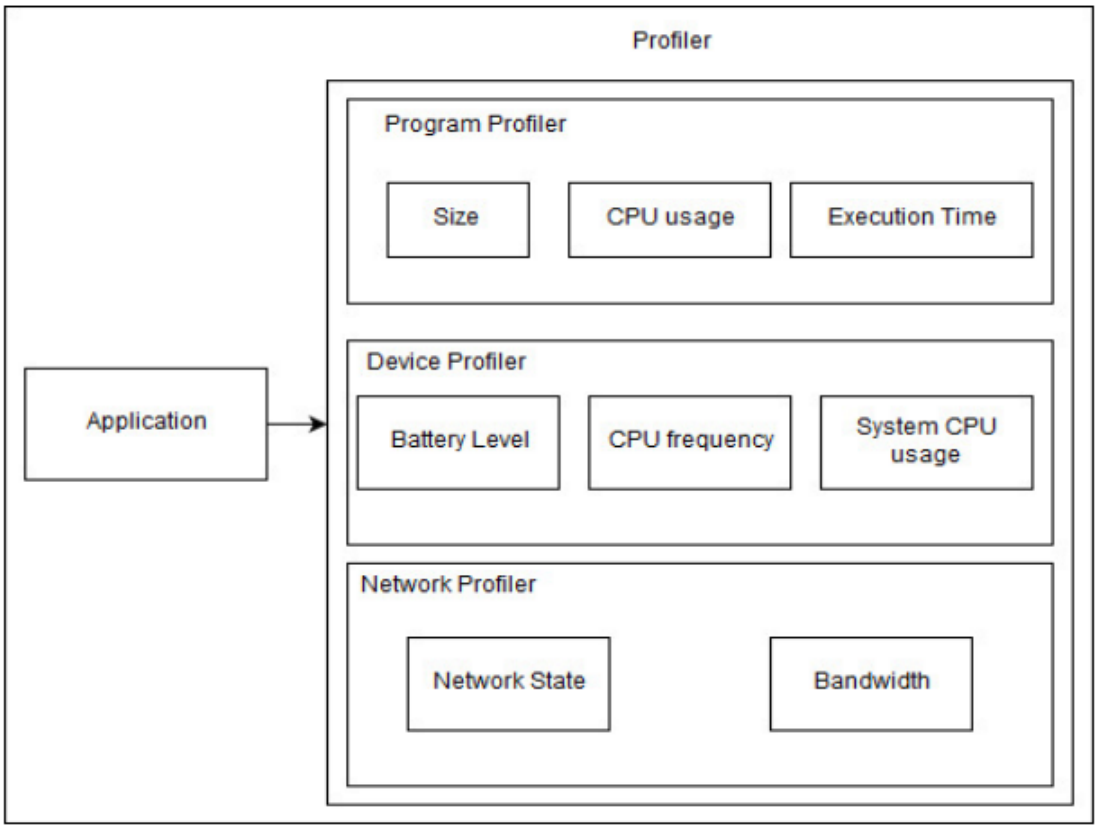

Figure 3.2: Profiler for Mobile Edge Cloud Computing

3.2.1 Device Profiler

Device profiler contains information about battery level, number of CPU Cores, CPU core utilisation and the CPU frequency.It also tells whether the battery is recharging or not, total number of cores available in local device and fog node. It also provides the information on average utilisation in fog node.

3.2.2 Application Profiler

A program profiler collects characteristics of applications, e.g., the execution time, the memory usage and the size of data. It gives information on hit ratio, percentage of time taken for each methods. It also provides the dependency between methods. It provides these information in graph based structure by representing methods as vertices and dependency between methods as edges.

3.2.3 Network Profiler

A network profiler collects information about wireless connection status and available bandwidth. The profiler tracks several parameters for the WiFi and mobile data interfaces including the receiving and transmitting data rate.These measurements enable better estimation of the current network. It provides information such as uplink and downlink speed from local device to mobile edge by establishing connection to mobile edge. It also tells us response time from mobile edge to cloud server by pinging the cloud. performance being achieved.

3.3 PARTITIONING

Static analysis obtains the control flow graph of an application by analyzing the byte code with nodes representing methods and edges representing relations between objects. The methods and the relations between them are obtained by traversing the graph. Application partitioning is a technique of splitting the application into separate tasks, while preserving the semantics of the original application. Method based partitioning is performed here. Each method in the application is considered as a separate partition. Certain methods that cannot be offloaded to the cloud as they can be only done in local device. e.g. method requiring input from sensors in the devices.

Chapter 4

IMPLEMENTATION AND RESULTS

4.1 PROFILING ALGORITHM

Input: None

Output: Bandwidth, Network type, battery, CPU usage, No of cores, CPU frequency

```
1: psutil.sensors_battery();
2: battery.power_plugged;
3: str(battery.percent);
4: battery.secsleft;
5: psutil.cpu_times();
6: psutil.cpu_count();
7: psutil.cpu_percent(interval=5,percpu=True);
8: for i=0 to 2 do
9: last_up_down = up_down;
10: upload=psutil.net_io_counters(pernic=True)['wlan0'].bytes_sent;
11: download=psutil.net_io_counters(pernic=True)['wlan0'].bytes_recv ;
12: up_down = (upload,download);
13: end
```

Figure 4.1: Profiler Information

Figure 3.1 shows the various profilers that are added to the application in order to provide the required input in the deciding making of partition offloading. The program profiler collect the characteristics of application by analyzing the control flow graph.Network profiler collects information about wireless connection and the available bandwidth. Device profiler contains information about battery level and the CPU frequency.

4.2 PARTITIONING ALGORITHM

Input: File(s)

Output: Graph G(V,E)

```
1: f=open(li,'r')
2: l=f.readlines()
3: s='@profile'
4: for j,line in enumerate(l)
5: if(line.startswith('def'))
6: line=s+line
7: del l[j]
8: l.insert(j,line)
9: end
10: f.close()
11: f=open(li[i],'w')
12: for line in l
13: f.write(line)
14: end
15: f.close()
```

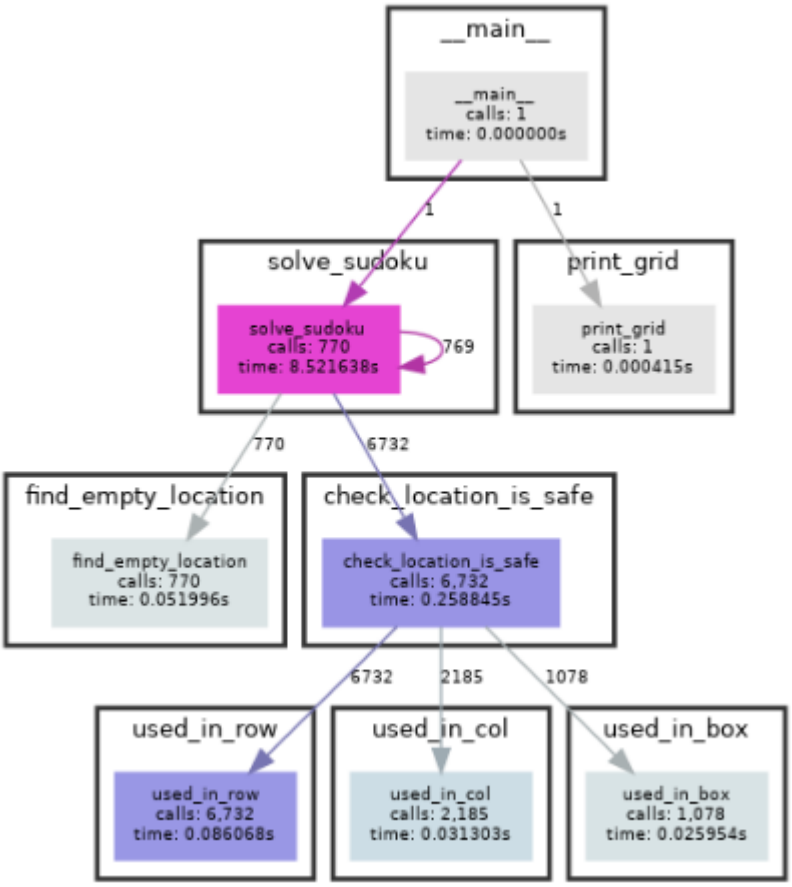

Figure 4.2: Dependency graph created by partioning algorithm

Figure 3.2 shows the visualization on dependency graph created by the partitioning algorithm for a application.

The partitioning provides information on dependency by give graph structure representing each methods as vertices and dependency between each methods as edges.It also gives the number of times the tasks are needed to executed along with estimated time.

Chapter 5

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

Thus Mobile Edge Cloud Computing System setup is formed by integrating Raspberry pi as Mobile edge and Digital Ocean as cloud server. Two modules for multitask computation offloading in mobile edge cloud computing has been implemented. i.e Profiling and Partitioning modules are implemented. Profiling is implemented by the importing the default package PSUTIL. Using the default package, the battery information, CPU Information and network status has been collected which will be helpful in making decision for offloading and scheduling in mobile edge cloud computing.Partitioning takes one or more applications as input and provides a graph based structure along with cost. It uses method based partitioning. It also tells us the time complexity of the application. It even also tells the time complexity for individual methods. It calculates the hit percentage of individual line. And finally it generates the graph. It also provides dependency graph as output in form of png.

5.2 FUTURE WORK

Using the profiler we can make decision whether the tasks to be executed in local node or mobile edge or cloud server. Further works include making offloading and scheduling decision. Offloading decision tells which are the tasks to be offloaded into mobile node and cloud server and the tasks to be executed locally. It makes decision based on the latency between local node, mobile edge (fog node) and cloud server. i.e. it makes decision based on the information given by profiler.

REFERENCES

1. Weiwei Chen, Member IEEE, Dong Wang and Keqin Li, Fellow IEEE, ”Multi-user Multi-task Computation Offloading in Green Mobile Edge Cloud Computing”, IEEE Transactions on Services Computing, DOI 10.1109/TSC.2018.2826544.

2. Hamed Shah-Mansouri, Member, IEEE, Vincent W. S. Wong, Fellow, IEEE, and Robert Schober, Fellow, IEEE ”Joint Optimal Pricing and Task Scheduling in Mobile Cloud Computing Systems”, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, VOL. 16, NO. 8, AUGUST 2017.

3. Xu Chen, Member, IEEE, Lei Jiao, Member, IEEE, Wenzhong Li, Member, IEEE, ACM, and Xiaoming Fu, Senior Member, IEEE ”Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing”, IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 24, NO. 5, OCTOBER 2016.