University of Macau

Faculty of Science and Technology



Design of Efficient Mobile Edge Computing in Wireless Network  
Design of Efficient Mobile Edge Computing in Wireless Network

by

LI KIN TAK, Student No: D-B4-2640-6Dxxxxxx

Final Project Report submitted in partial fulfillment  
of the requirements of the Degree of   
Bachelor of Science in Computer Science

Project Supervisor

Prof. Yuan WU

04 June 2020

# 

DECLARATION

I sincerely declare that:

1. I and my teammates are the sole authors of this report,
2. All the information contained in this report is certain and correct to the best of my knowledge,
3. I declare that the thesis here submitted is original except for the source materials explicitly acknowledged and that this thesis or parts of this thesis have not been previously submitted for the same degree or for a different degree, and
4. I also acknowledge that I am aware of the Rules on Handling Student Academic Dishonesty and the Regulations of the Student Discipline of the University of Macau.

Signature : 

Name : LI KIN TAK

Student No. : D-B4-2640-6

Date : 04 June 2020

ACKNOWLEDGEMENTS

The years of the undergraduate university have passed by in a hurry. There are bumps, confusion, struggle, and joy on the way. All the people who have accompanied me and taught me over the years, thank you! In this project, my mentor, Professor Wu Yuan, taught me carefully, guiding me when I was confused, encouraging me to forge ahead when I was lost, and sharing with me when I was happy. His rigorous learning, responsibility for teaching, and focus on research will be my permanent role model and learning goal. His teaching taught me how to treat learning and how to learn step by step. With his help, my research planning and research direction became more clearly, and it also allowed me to see a larger academic world. Thanks to the other teachers, they gave me great help when I focused on this project, so that my progress in other aspects has not stopped. Finally, I would like to thank my parents. Years of upbringing have enabled me to grow up healthily, establish the correct social concept, and give me the opportunity to learn and see such a valuable academic world. I sincerely thank you for making me the person I am today.

**ABSTRACT**

With the fast development of 5G mobile network and internet of things (IoT), mobile edge computing (MEC)became a popular topic as the key technology to solve problems of current network with high delay, high energy consumption and security vulnerabilities. MEC aims to migrate the computing tasks of mobile users from the local devices to the servers located at the edge of the wireless network for computing. The theme of this thesis is to simultaneously transfer the workload of multiple computing tasks of multiple users (multiple terminal devices) to the edge servers of wireless network and find the solution that minimizes the energy consumption required by the mobile terminal to complete the tasks. This thesis decomposes the problem into two parts: the first part is to optimize the computing workload and NOMA transmission time of the terminal service (TS), this part aims to calculate the minimum energy consumption required by the mobile user to transmit multiple tasks to different servers under a give matching scheme. In the second part, we will use the simulated annealing algorithm (SA) to find a matching scheme that minimizes the total energy consumption by swapping the server matching order. Finally, we use python to simulate the algorithm and compare the results with the full arrangement results, to verify the feasibility of the algorithm. According the results, the multi-user multi-task multi-access offloading scheme base on NOMA in this thesis has lower latency than the traditional OMA-based transmission scheme.

TABLE OF CONTENTS

[CHAPTER 1. INTRODUCTION 9](#_Toc41777894)

[1.1 Background 9](#_Toc41777895)

[1.2 Advantage of MEC 9](#_Toc41777896)

[1.2.1 Delay 9](#_Toc41777897)

[1.2.2 Energy consumption 9](#_Toc41777898)

[1.2.3 Context-aware 10](#_Toc41777899)

[1.2.4 Security 10](#_Toc41777900)

[1.3 Application Scenarios 10](#_Toc41777901)

[1.3.1 Video analysis 10](#_Toc41777902)

[1.3.2 VR / AR 10](#_Toc41777903)

[1.3.3 Internet of Things / Internet of Vehicles 10](#_Toc41777904)

[1.3.4 Environmental monitoring 11](#_Toc41777905)

[1.4 Research Status 11](#_Toc41777906)

[1.5 Typical MEC-Related Models 11](#_Toc41777907)

[1.5.1 MEC calculation model 11](#_Toc41777908)

[1.5.2 Communication model 12](#_Toc41777909)

[1.6 Related Work 12](#_Toc41777910)

[1.7 Paper Motivation and Outline 13](#_Toc41777911)

[CHAPTER 2. PROBLEM MODELING 14](#_Toc41777912)

[2.1 Algorithm decomposition 16](#_Toc41777913)

[2.2 Algorithm details 18](#_Toc41777914)

[2.3 Specific steps of the algorithm 20](#_Toc41777915)

[CHAPTER 3. ALGORITHM IMPLEMENTATION 23](#_Toc41777916)

[3.1 The EEO algorithm 23](#_Toc41777917)

[3.2 The Simulated Annealing algorithm 25](#_Toc41777918)

[3.3 Algorithm summary 26](#_Toc41777919)

[CHAPTER 4. TOOL AND METHOD 27](#_Toc41777920)

[4.1 Implementation environment 27](#_Toc41777921)

[4.2 File structure 27](#_Toc41777922)

[4.2.1 Program file structure 27](#_Toc41777923)

[4.2.2 Data file structure 27](#_Toc41777924)

[4.3 Tools 28](#_Toc41777925)

[4.3.1 CVX toolbox for EEO algorithm part 28](#_Toc41777926)

[4.3.2 Arithmetic steps and Data processing 28](#_Toc41777927)

[4.3.3 Interface 28](#_Toc41777928)

[CHAPTER 5. Ethics and Professional Conduct 29](#_Toc41777929)

[5.1 Data of this thesis 29](#_Toc41777930)

[5.2 Coding 29](#_Toc41777931)

[5.3 Miscellaneous 29](#_Toc41777932)

[CHAPTER 6. CONCLUSION 30](#_Toc41777933)

[6.1 Thesis Summary 30](#_Toc41777934)

[6.2 Challenge 30](#_Toc41777935)

[6.2.1 Mathematical models 30](#_Toc41777936)

[6.2.2 Solving combinatorial optimization problem 30](#_Toc41777937)

[6.2.3 Programming 30](#_Toc41777938)

[6.3 Future Work 31](#_Toc41777939)

[CHAPTER 7. PROJECT EXCUTION SCHEDULE 32](#_Toc41777940)

[7.1 Project schedule 32](#_Toc41777941)

[7.2 Member contribution 32](#_Toc41777942)

[CHAPTER 8. REFERENCES 33](#_Toc41777943)

LIST OF FIGURES

[Figure I: System model 15](#_Toc41778155)

[Figure II: Assignment of tasks to Edge Servers 16](#_Toc41778156)

[Figure III: Idea for finding matching scheme 21](#_Toc41778157)

[Figure IV: Matching scheme of task-ES 22](#_Toc41778158)

[Figure V: Algorithm Steps 23](#_Toc41778159)

[Figure VI: Linear changing t for searching Vt 25](#_Toc41778160)

[Figure VII: Result from three different scenario 25](#_Toc41778161)

[Figure VIII Gantt table of the project 34](#_Toc41778162)

LIST OF TABLES

[Table 1: Comparison between three scenarios and calculate locally 26](#_Toc41778163)

[Table 2:Results comparison between SA algorithm and full arrangement 27](#_Toc41778164)

# INTRODUCTION

## Background

With the fast development of the Internet of Things, the number of IoT devices has increased rapidly. Cisco predicts that approximately 50 billion IoT devices will be added to the Internet in 2020 [1]. Most of these devices have limited communication and storage resources and cannot complete all tasks through its own resources, you need to rely on cloud or edge devices to enhance its functions. And because of the huge amount of data, it is impossible for the cloud computing center to handle all tasks alone. With the advent of the 5G era, the limitations of Mobile Cloud Computing (MCC), that is, the propagation distance from the end user to a remote cloud center is very long, and the network bandwidth and computing resources are limited, making cloud computing transmission and calculation take a certain time, causing it has caused a huge delay, which has led to the inability to meet the needs of more and more emerging applications. MEC can solve this problem. MEC deployed cloud computing capabilities to mobile networks to provide IT service environments at the edge of mobile networks and cloud computing resources, offloading computing tasks from mobile devices, making cloud computing tasks as close to the data producer as possible, and making haptic application experience possible. Migrating computing users to edge servers can not only reduce latency, but also reduce overall energy consumption due to factors such as short transmission distances [2].

## Advantage of MEC

Compared with MCC, MEC has lower latency, lower energy consumption, and also supports context awareness and stronger security.

## Delay

Mobile service delay is generally composed of transmission delay, calculation delay and communication delay. MEC's ​​information transmission distance is generally the distance from a small cellular network or device to the edge device. Compared to the distance between the MCC from the end user to the cloud computing center of the core network, the MEC has a lower distance. Transmission delay. For computing latency, although the edge device is not enough to compare with the huge computing power of the cloud computing center, since the number of users facing a single edge device is correspondingly reduced, users can enjoy more computing resources, which reduces the calculation delay. And because the communication distance and the number of passing nodes are reduced, data will only be transmitted between the data origin and the edge server, reducing its communication delay.

### Energy consumption

Energy consumption is more important than edge devices with limited energy. The number of IoT devices is extremely large. It is impossible to frequently charge or replace the battery for each device. Through effective calculation and shunting, the computing tasks are diverted from IoT. When migrating to an edge server from a device, the energy consumption of IoT will be greatly reduced compared to the local computing tasks.

### Context-aware

In MEC, the distance between the edge device and the end user is short, so that edge computing can obtain the user's real-time information, such as its location, behavior, and environment, and then use this information to provide various services.

### Security

In MCC, user information is concentrated in the cloud computing data center, which easily leads to information security issues such as data leakage and loss. However, the scale of MEC is small, the stored data is small, and the attack value of edge servers is relatively higher than that of cloud computing centers. Small, thereby reducing the possibility of being attacked. MEC's ​​distributed deployment also improves its security.

### Application Scenarios

The advantages of edge computing enable edge computing to play a huge role in 5G systems [3]. It has been applied to many application scenarios, such as VR, AR applications, cloud gaming platforms, ultra-high-definition video streaming, etc., and it can also be applied to many emerging scenarios. Such as video analysis, real-time environmental monitoring and so on.

### Video analysis

In order to create a smart city, a large number of cameras are deployed in the city for vehicle license plate recognition, face recognition, and security monitoring. The transmission and analysis of video streams require a lot of computing and network resources, and most of the cameras are not sufficient. Computing power cannot perform video analysis locally, and edge servers that are close and have sufficient computing resources can just meet their needs. Edge computing can reduce its delay and reduce the energy consumption of sensor devices.

### VR / AR

VR and AR are very latency-sensitive applications that require extremely low latency to satisfy the user experience. The average VR / AR mobile device does not provide the computing power that satisfies the application experience. Offload computing tasks from the device to the edge. The server can greatly reduce its latency and improve the user experience.

### Internet of Things / Internet of Vehicles

With the development of the Internet of Things, machine learning and other technologies, vehicles can also be used as smart devices to access the Internet. The low latency and context awareness of edge computing enable smart vehicles to respond quickly to vehicle behavior, status, and surrounding environment, and to automatically Techniques such as driving, abnormal condition judgment, and even road condition analysis all help a lot.

### Environmental monitoring

Edge computing has location awareness capabilities. Edge computing can collect and process data at nearby geographic locations without transmitting to a cloud computing center. This feature can be applied to public management and scientific research, such as deploying sensors in various places to obtain information about the surrounding environment. To help scientists monitor geographic conditions and understand the climate change in the area. Data transmission in sensors and cloud computing centers will generate huge energy consumption and delays, affecting the real-time nature of information. Mobile edge computing allows people to make reaction.

## Research Status

The huge advantages of edge computing make people work hard on research and development in the field of edge computing. Edge computing originated in the 1990s. Akamai proposed a content distribution network. A storage transmission node was set up near the end user to store static content and allow users to get the content they need nearby. In 2006, Amazon released the EC2 service, announcing the advent of the cloud computing era. With the development of cloud computing, in order to reduce the delay, reduce the computing load of the cloud computing center, and solve the problem of insufficient network transmission bandwidth, in 2009, Satyanarayanan and others proposed the concept of Cloudlet [5] , which deploys a trusted and resource-rich host on the network Like the cloud computing center, the edge provides users with IT and cloud computing services. In 2012, Cisco proposed the concept of "fog computing". In 2013, "edge computing" proposed by American scholars received great attention in academia and industry. Since then, relevant international conferences on edge computing have continued to develop and move to edge computing. Technology has entered a period of rapid growth. In 2016, Professor Shi Weisong's team gave a formal definition of edge computing and published a paper entitled "Edge Computing: Vision and Challenges", which was widely cited. In 2018, edge computing was introduced to the general public, and more people began to become familiar with this technology, which has increased the number of participants in this field and has become a hot topic in academia and industry.

## Typical MEC-Related Models

This section will introduce some typical MEC calculation, communication and other models. Analysis based on these models can provide ideas for research and promote theoretical analysis.

### MEC calculation model

Due to multiple factors such as delay, bandwidth, and utilization, it is very complicated to accurately model computing tasks, but there are also some simple models that are reasonable and mathematically easier to handle. The following two computing models are commonly used in the MCC and MEC literature, namely binary and partial computing migration, respectively.

#### Binary migration task model

For highly integrated or relatively simple tasks, you can choose to complete all locally on the mobile device or uninstall the whole to the MEC server.

#### Partial migration task model

For a task consisting of multiple processes, the task can be divided into two parts, one is executed locally on the mobile device, and the other is migrated to the edge server for simultaneous execution.

### Communication model

#### Wireless bit pipe model

Different from the mobile pipe and cloud computing server in the MCC literature, the communication pipe is given as a bit pipe model with a constant rate and a random rate. MEC needs to consider the requirements of edge cloud and delay sensitive applications, so Consider some key attributes of wireless transmission, such as multi-path fading of wireless channels caused by scattering of objects in the environment, interference between wireless signals, and spectrum shortage.

#### Wireless channel fading model

In the task migration of edge computing, the influence of channel gain on delay must be considered, and the migration must be performed on channels of good quality, so the model needs to be considered in conjunction with migration and wireless transmission.

## Related Work

MEC helps terminal devices with limited resources to offload computing tasks to the edge server to complete the problem of insufficient intelligent IoT computing resources, which has aroused widespread concern in academia and industry [6]. In order to make MEC more efficient, the multi-access MEC paradigm was conceived [7]. In a multi-access MEC, a terminal device can offload single or multiple computing tasks to different edge servers at the same time, thereby better utilizing edge server resources. In this paper, starting from the task migration method of edge computing, this paper studies the design algorithm for the multi-access MEC energy efficiency optimization problem based on non-orthogonal multiple-access transmission, and finds the best task migration solution for single-user multi-tasking and multi-server. The academic achievements related to the research in this article are reviewed as follows:

Chen et al. First studied the multi-user joint offloading decision and channel allocation problem, and then further studied the multi-tasking scheme to simultaneously optimize the migration decision and the allocation of computing and communication resources in order to reduce the total energy and delay of all users in edge computing cost [8]. Huang et al. In "Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-Edge Computing Networks, "proposed a solution for wireless MEC online computing offload [9]. Guo et al. Proposed a greedy migration solution based on the multi-user ultra-dense MEC server [10].

In a large number of researches on edge computing, multiple-access edge computing based on non-orthogonal multiple-access technology (NOMA) can further reduce the delay, thereby meeting more requirements. Different from traditional frequency division, time division, etc., NOMA allows multiple mobile users to use the same resource block at the same time and provide services to multiple users at the same time on the same subchannel. In order to use NOMA to improve throughput and energy efficiency, many studies envision NOMA as a potential solution for multiple-access MEC, and use NOMA to simultaneously migrate computing tasks to multiple edge servers, further reducing transmission delays and energy consumption [11][12].

The relevant achievements of NOMA are reviewed as follows:

In order to improve the energy efficiency of non-orthogonal multiple-access transmission, scholars such as Zhang and Fang proposed various resource allocation schemes [13][14]. Ding et al. Studied the migration delay minimization problem of NOMA-MEC [15]. Wu et al. Studied the joint power allocation and service scheduling of NOMA relay networks [16]. Recent studies by some scholars have shown that NOMA can reduce the delay and energy consumption of MEC [17].

Mobile edge computing based on non-orthogonal multiple access transmission allows users to divide computing tasks into multiple parts and simultaneously migrate to different servers to complete, shortening the transmission distance, reducing latency and energy consumption, and greatly improving efficiency and application experience.

## Paper Motivation and Outline

Edge computing's low latency, low energy consumption, high security, and support for context awareness make edge computing have considerable potential. Based on curiosity in this field, after reading the papers of seniors and learning related background knowledge, I chose the topic of designing mobile edge computing energy efficiency solutions for orthogonal multiple access transmission is arranged as follows:

Chapter 2 is the mean modeling in our project.

Chapter 3 is the implementation and of the algorithm in this thesis.

Chapter 4 describes the tools and methods used in this thesis.

Chapter 5 is the ethics and professional conduct of this thesis.

Chapter 6, we summarized this research and described the difficulties of this work and the direction of future work.

Chapter 7, we show the member contributions and project timeline.

And the last chapter is the reference.

# PROBLEM MODELING

In the non-orthogonal multiple access transmission mode, {, , ...., } is simultaneously migrated to each edge server, and the power consumption will be affected by the channel power gain factor. We use to indicate the user to the edge server Channel power gain of i. We assume that the order of the edge servers is based on:

(1)

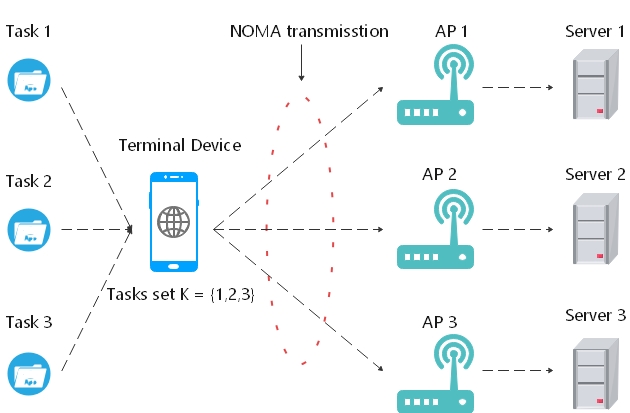


Figure I: System model

Assuming the transmission duration is t, according to this formula, we can get the minimum total power for the user to send to the edge server I:

(2)

where W is the channel bandwidth, parameter represents the power density of the background noise, and is set to 0. That is, the total power of the user using NOMA transmission tasks is

(3)

Based on non-orthogonal multiple-access transmission, this paper studies the edge computing of single-user, multi-tasking, and multi-server. In order to find its energy-efficient optimization solution, that is, to minimize the total energy consumption of mobile users to complete computing tasks, we can do three aspects to get start:

1. How much migrates per task

2. Transfer time of each task migration to the edge server t

3. Which server should mobile user tasks be migrated to for calculation

If you want to solve problems 1 and 2, you need to optimize and transmission time to minimize its energy consumption. This article assumes that there are I computing tasks on a mobile device of a single mobile user, which are represented by , where the total task amount of each task is represented by, because Part of the task amount is migrated, so each task is divided into two parts, one part is the task amount that needs to be migrated to the edge server through non-orthogonal multiple access, and the other part is the local computing task amount .

(4)

For each edge server , there is a fixed calculation rate, expressed as , and is used to indicate the local calculation rate on the terminal device. In order to clearly indicate that we measure by and bits / s.

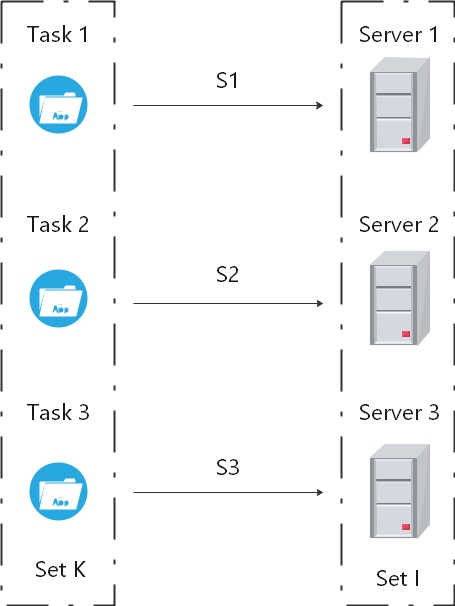


Figure II: Assignment of tasks to Edge Servers

Because the power consumption of the processor can be modeled as a cubic relationship with respect to its calculation rate. So, we can get the total energy consumption of mobile users to complete all remaining tasks locally:

(5)

where is used to represent the coefficient determined by the CPU chip architecture.

## Algorithm decomposition

(6)

This formula is constrained by

(7)

(8)

(9)

(10)

Variables in this formula:

According to (8) and the above conditions, we can derive more constraints:

(11)

This formula is used to indicate the minimum computing speed of the terminal device when processing tasks locally, then because of

(12)

We get,

(13)

After optimizing (6) by (11)(13), we will get (EEO-E):

(14)

Up to here, we can know that (14) is still a non-convex function about variables s and t, but through the constraints above, we can make (14) to be a convex optimization problem by fixing t, so we decompose the problem to two parts:

1. The first step is trying to optimize the workload that needs to be transmitted, so that the total energy consumed by the terminal device to complete the tasks under a given t is minimized.

(20)

Variable is

Constrained by

(21)

(22)

(23)

1. At the second step, we want to optimize t, we get the Vt that is minimized at a given t in the first step, so by adjusting the value of t, we can calculate the Vt that is minimized, means get the minimum energy consumption of a user complete all tasks under a fixed matching scheme.

(24)

The variable is .

The formula is constrained by

(25)

According to the above two-step calculation, we optimize under a given t at first step to find the minimum energy consumption in this case. Then due to the limitation of the transmission time t, we can use a small step size to calculate each t to minimized Vt, so as to find the minimum energy consumption for a single user to migrate computing tasks to the edge servers under a fixed matching scheme.

## Algorithm details

After the decomposition of EEO-E, we get the minimized Vt under a given matching scheme, so we need to find the best matching scheme to further minimize Vt.

Regard the problem of finding matching schemes as a combination optimization problem, the simplest method is to enumerate all matching schemes, then calculate the results of each scheme and compare them to find the minimized Vt. This method can be selected when the numbers of tasks and servers are small, but the amounts of matching schemes will become extremely large when the numbers of tasks and servers increases, the calculate workload and time will be large.

In order to enable each task to be transferred to different servers, we can present the tasks set with and present the servers set with , means the number of edges servers is greater than the numbers of tasks. In order to make the algorithm runs successfully and ensure that each server and task can be matched, we set  tasks with . We can express the matching relationship between server and the task as , means the task is transmitted to server . Conversely, we can use to indicate the offload portion of computing task on server. As the Figure II, III shows.

Therefore, the minimum energy consumption for the terminal device to transfer the workload to the server can be expressed as:

(26),

(27),

When , the calculation workload of server is the workload which offloaded by task . From (26)(27), the total latency for mobile user to complete all tasks is updated as:

(28)

The EEO problem translates into:

Constrained by

variables are

Base on above conditions, the idea of finding the best matching scheme is: Set up a set of matching relationships as the initial solution, use the EEO algorithm to optimize the offload task and transmission time, so as to obtain the minimum energy consumption under this matching relationship, then set it as the current optimal solution. According to the corresponding matching relationship and its energy consumption, set the new matching relationship, of the new result is smaller than ,then set the new result as the current optimal solution and repeat the above steps continuously until a matching scheme that minimizes the total energy consumption is obtained.

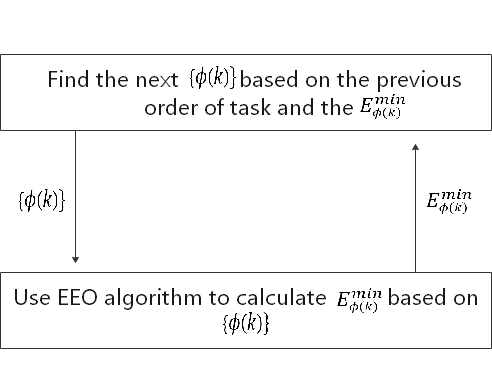


Figure III: Idea for finding matching scheme

Since the solution to this problem is not monotonous increasing or decreasing, if the initial matching relationship established is not good, the algorithm may only get the local optimal solution, so this thesis introduces the simulated annealing algorithm. The simulated annealing algorithm imitates the physical principle of solid annealing. When the solid is slowly heated to a sufficiently high temperature, the molecules inside the solid will become disordered as the temperature increases, and become stable as the temperature decreases. The simulated annealing algorithm simulates this principle. After setting up a sufficiently high initial temperature and initial solution, by continuously cooling down and calculating a new solution, when caught in the local optimal solution, the sudden characteristic of the simulated metal molecule jumps out of the local area with a certain probability Optimal solution and accept new solution. Through continuous cooling and calculation to update to obtain a more optimal solution, when the temperature reaches the lowest, the jump probability is set to 0, that is, the current optimal solution is output as the global optimal solution.

## Specific steps of the algorithm

The specific steps of the algorithm in this article are as follows:

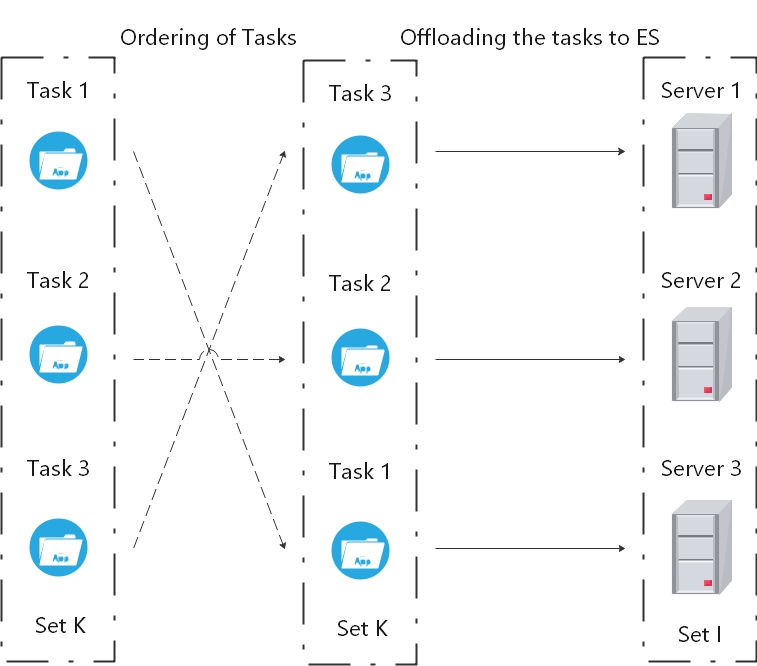
1. Set a sufficiently large initial temperature , the initial value of the number of iterations is , a random scheme of tasks and server matching as the initial solution and put into the above EEO algorithm to solve, get the minimum energy consumption under this matching scheme ;
2. 

Figure IV: Matching scheme of task-ES

1. Select two tasks , randomly from the matching scheme and change their positions in the peering relationship to generate new scheme , that is,,, and put as a new solution into EEO algorithm to get the new minimum energy consumption , the iterations number ;
2. Compare the results of two algorithm, if the energy consumption of the new solution is smaller than the previous one , means , then the new solution is regarded as the current optimal solution, so that ; Otherwise, if , the new solution is accepted with a probability which is decreasing with temperature. In this step, , reduce the temperature gradually as the number of iterations increases, repeat steps 2 and 3;
3. When the temperature approaches 0 or multiple consecutive solutions are not accepted or the optimal solution does not change after multiple iterations, the optimal solution is output as the final result and the algorithm is stopped.

The algorithm flow is shown in Figure V.

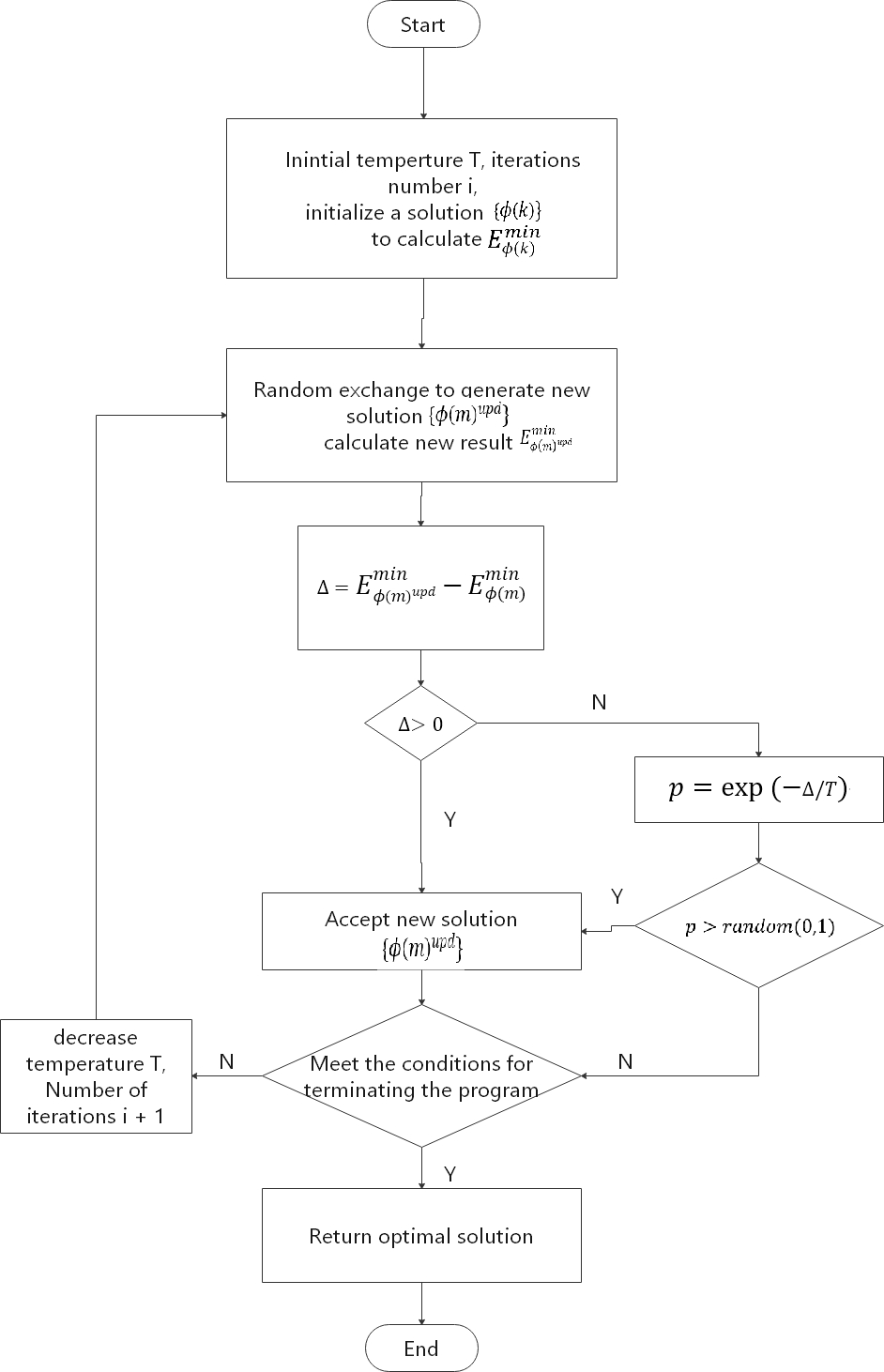


Figure V: Algorithm Steps

# ALGORITHM IMPLEMENTATION

## The EEO algorithm

In order to verify the EEO algorithm, set up a single user multi-task multi-base station virtual environment: there are three base stations in the environment located at (0,500), (500,0) and (-500,0). Randomly generated the channel power gain , the channel bandwidth , the background noise density . The calculation speed of every single edge server . In addition, the mobile user’s power budget .5Joul/s, the total task amount , the maximum transmission time is set to , the local calculation speed of the mobile terminal device is . All simulations are performed on the PC of Inter(R) Core™ i7-7820HK [CPU@2.90GHz](mailto:CPU@2.90GHz).

Based on the first step of the algorithm, optimize at a given t to solve . Given , the minimum energy consumption , the .

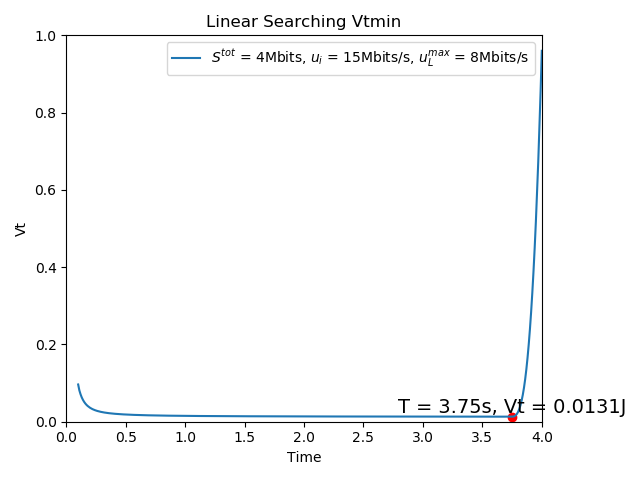
The second step: optimize with a smaller step size and solve under the given matching scheme. At , get the minimum value of , The offloading workload at this time is .The results are shown in the figure.

Figure VI: Linear changing t for searching Vt

In order to better verify the effectiveness of the algorithm, this thesis sets up three scenarios with different total task workload. Set for above scene. When the total task workload,, the offloading workload . When the total task workload,, the offloading workload .

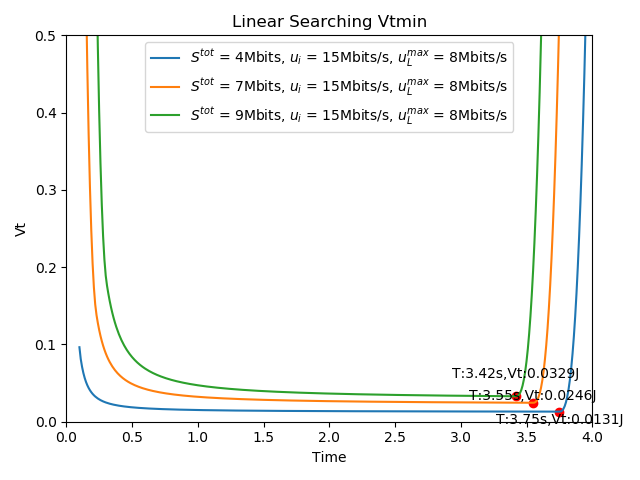


Figure VII: Result from three different scenario

According to the figure VII and results, increases as the total workload increases, but the corresponding optimal decreases. This is because the increase in the total workload makes each task’s offloading workload also increase, so when the server’s calculation speed and maximum acceptable delay reach a condition, the optimal transmission time can be deduced from formula (13).This means when the computing speed of the edge server is large enough and the delay is sufficient, the local computing energy consumption can be reduced by sacrificing some transmission energy consumption , thereby making the total energy consumption smaller.

The comparison between the above three scenarios and the situation of local calculation is shown in the following table.

Table 1: Comparison between three scenarios and calculate locally

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Total Workload  (Mbits) | Transmission energy consumption  (Joul) | Local computing energy consumption  (Joul) | Total energy consumption of mobile user  (Joul) | Total energy consumption of calculate locally  (Joul) | Optimization ratio |  |
|  | 0.0116 | 0.0015 | 0.0131 | 0.9600 | 98.6% | 0.5181 |
|  | 0.0227 | 0.0019 | 0.0246 | 5.1450 | 99.5% | 0.9428 |
|  | 0.0298 | 0.0031 | 0.0329 | 10.9350 | 96.9% | 0.8966 |

Base on the table, the total energy consumption is reduced when user offloads some parts of tasks to the edge servers than the user completes all tasks locally, the result shows the effectiveness of edge computing and the advantages of low energy consumption.

## The Simulated Annealing algorithm

The second part of the algorithm in this paper needs to find the corresponding matching scheme to minimize Vt, means the total energy consumption of mobile users is minimum. In order to verify the effectiveness of the algorithm, this thesis will set up multiple scenarios and compare the results of SA algorithm with the results generated by arranging all matching schemes. If the two results are the same, it proves that the SA algorithm in this paper is effective.

Set up scenarios that need to complete 3,4 and 5 tasks respectively and record each matching scheme and Vt under the scheme. For each scenario, the initial temperature is 300 and the initial number of iterations . In the scenario of 3 tasks, the workload of each task is , the maximum acceptable delay is , the channel gain , the calculation speed of each edge servers is . In the scenario of 4 tasks, the workload of each task is , the maximum acceptable delay is , the channel gain , the calculation speed of each edge servers is . In the scenario of 5 tasks, the workload of each task is , the maximum acceptable delay is , the channel gain , the calculation speed of each edge servers is .

The comparison between the running results of the algorithm and the results generated by full arrangement is shown in the following table:

Table 2:Results comparison between SA algorithm and full arrangement

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Result | 3-tasks | 4-tasks | 5-tasks |
| SA algorithm | Best scheme |  | [7 6 5 8] | [8 7 6 5 9] |
| Minimum energy consumption | 0.0166J | 0.0299J | 0.0455J |
| Full arrangement | Best scheme |  |  |  |
| Minimum energy consumption | 0.0166J | 0.0299J | 0.0455J |

When the matching scheme calculation of 3 tasks is completed, the SA algorithm iterates 28 times and the running time is s. When the matching scheme calculation of 5 tasks is completed, the SA algorithm iterates 40 times and the running time is s. When the matching scheme calculation of 5 tasks is completed, the SA algorithm iterates 81 times and the running time is s.

It can be seen from the above table and the operation results that the SA algorithm results in this thesis are consistent with the results obtained by the full arrangement, means the global optimal solution can be obtained through the SA algorithm, which proves the effectiveness of the algorithm in this thesis.

## Algorithm summary

This chapter shows the environment configuration, parameter settings and execution results of the two algorithms under python simulation. Through figures VI and VII, it is verified the minimum energy consumption of the user to complete the task under the fixed matching scheme can be obtained by optimizing the transmission time and offloading workload through the EEO algorithm. Then, the comparison of Table 1 proves that the EEO algorithm has more energy consumption advantages than the user completing all tasks locally. Next, the comparison between the results of the full permutation and the SA algorithm proves that the SA algorithm can obtain the optimal solution, thereby verifying the effectiveness of the algorithm in this paper.

# TOOL AND METHOD

## Implementation environment

All the programs in this thesis are based on Python 3.7 on the PC of Inter (R) Core ™ i7-7820HK [CPU@2.90GHz](mailto:CPU@2.90GHz).

## File structure

### Program file structure

There are 7 programs in this project.

CVX.py -- The program to handle EEO algorithm part.

SA.py -- The program to handle SA algorithm part.

Allposible.py -- The program to produce results of permutations.

plotCVX.py -- The program to plot the CVX figure.

Interface.py -- The program to show the algorithm process dynamically.

Setparam.py -- The program to let user custom parameters.

Verify.py -- The program to verify the result from CVX.

### Data file structure

There are some csv data files in this project.

. /CSV/CVX/ -- The path to save each result of EEO algorithm, every single file records the time and Vt of each step.

. /CSV/SA/ -- The path to save each result of SA algorithm, every single file records the matching scheme and Vt of each step.

. /CSV/PARAM/ -- The path to save param which generated by Setparam.py.

. /CSV/GLOBAL/ -- The path to save global iteration results.

. /CSV/ FULLARRANGEMENT/ -- The path to save the permutation results which generated by Allpossible.py

. /CSV/LOG/ -- The path to save program log form SA.

## Tools

### CVX toolbox for EEO algorithm part

To solve the convex optimization problem in EEO algorithm, this thesis uses the CVX toolbox. CVX is a modelling system for convex optimization, CVX allowing constraints and objectives to be specified using standard MATLAB expression syntax.

### Arithmetic steps and Data processing

In order to process the calculation steps of the algorithm in the algorithm and store and read data, this thesis use the NumPy and Pandas package from python.

### Interface

For showing the process and results of algorithm, this thesis uses two drawing tools: Matplotlib and Plotly.

Plotly can be well linked with the dash package and dynamically display the data. This project attempts to use PyQt to design the UI interface, but due to the limitations of the implementation environment, the linkage effect of the Plotly package and Dash package is better than the design with the PyQt package.

For the interface process package, this project chooses PyQt package and Dash package. The PyQt package is used to display the interface of Setparam.py and receive input from user. Dash package is used to link with Plotly and dynamically display the algorithm process interface.

# Ethics and Professional Conduct

## Data of this thesis

The data used in this thesis and the user input received by the program are non-sensitive information and do not contain personal information. The algorithm test data in this thesis are all randomly generated from theories and formulas.

## Coding

All the code of this project is written by myself, there is no plagiarism code from others.

All the software used in this thesis for coding, generating images is genuine authorized / open source software.

## Miscellaneous

This project does not have any anti-social and human views, and aims to contribute to the field of scientific research.

# CONCLUSION

## Thesis Summary

This thesis mainly studies the mobile edge computing algorithm for non-orthogonal multiple access transmission. Edge computing has become a key technology to solve many problems of traditional cloud computing, it can solve the problem of security, latency, bandwidth and energy consumption of current cloud computing security do not meet the needs of new generation applications. Mobile edge computing has obvious advantages in such as public safety, environmental monitoring, smart city virtual reality and other emerging applications. This thesis base on non-orthogonal multiple access transmission, communication and calculation models of edge computing used EEO algorithm to optimize energy consumption in multi-task multi-server multi-access mobile edge computing. The algorithm in this thesis is mainly divided into two parts. The first step is optimizing the offloading workload and do a linear searching by a small step size of time to get the minimum energy consumption when user complete tasks under a fixed matching scheme. The second part is to exchange the order of tasks to generate new matching schemes and substitute the EEO algorithm to find the minimum energy consumption under the scheme, and finally iterate through the SA algorithm to find the global optimal solution, means the best matching scheme and its energy consumption.

## Challenge

### Mathematical models

The algorithm model in this thesis is derived based on the NOMA transmission model and the calculation and communication model of edge computing, the generated model is a non-convex optimization problem and cannot directly find the global optimal solution. After the guidance of Professor Wu and the research, the model is transformed into a convex optimization model through the decomposition algorithm and the global optimal solution is obtained.

### Solving combinatorial optimization problem

Searching the best matching scheme can be regarded as a combinatorial optimization problem. Since there is no obvious logical relationship between each combination and the total number of schemes is too large, it is hard to find a loss function to solve the best matching scheme using algorithm such as A3C. After studies, this thesis refers to the concept of the A3C algorithm, optimizes the SA algorithm and adds multiple calculation processes (SA worker) and statistic processes (Count worker) on the basis of SA algorithm to accelerate the calculation speed to find the best match program

### Programming

In terms of writing algorithm programs, due to the complexity of the model and the steep learning of tools such CVX toolbox, it is a challenge in the early stages of writing algorithms, especially because of the addition of virtual servers and tasks. In order to speed up the calculation, this thesis attempts to add multi-process to the algorithm. When coding multi-process logic, due to the existence of Global Interpreter Lock (GIL) in Python, the program will easily enter the stuck state. Finally, this thesis solves the problem that the process cannot be synchronized and main thread stuck by using Queue, Multiprocessing modules.

When coding the interface, due to the compatibility problem between PyQT and matplotlib, when the amount of data is large and the refresh speed of interface is fast, the interface is easy to crash, so the thesis chose to use Plotly and Dash to design a more lightweight and simpler web UI interface.

## Future Work

This thesis implements the method of single-user offloading transmission, in the field of edge computing, more application scenarios are based on multi-users, so more in-depth research can be conducted on the multi-user situation.

The second part of the algorithm in this thesis is based on the multi-threaded SA algorithm. Finding a suitable loss function and substituting deep learning and other algorithms will also be a good topic. For example, combining the migration algorithm with the Pointer-network.

For the algorithm, there are many areas that can be improved, such as adding other factors that affect transmission as parameters of the algorithm, such as the stability of network transmission. This thesis optimizes the energy consumption of completing tasks, there are many good research directions in edge computing such as latency, security.

# PROJECT EXCUTION SCHEDULE

## Project schedule

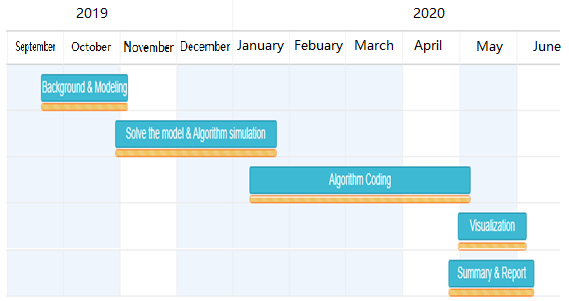


Figure VIII Gantt table of the project

## Member contribution

This project is completed by LI KIN TAK under the guidance of Professor Wu.

# REFERENCES

*[1] Y. Mao, C. You, J. Zhang, K. Huang and K. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective", IEEE Communications Surveys & Tutorials, vol. 19, no. 4, pp. 2322-2358, 2017.*

*[2] W. Shi, J. Cao, Q. Zhang, Y. Li and L. Xu, "Edge Computing: Vision and Challenges", IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637-646, 2016.*

*[3] Y. Tao, L. Liu, S. Liu and Z. Zhang, "A survey: Several technologies of non-orthogonal transmission for 5G", China Communications, vol. 12, no. 10, pp. 1-15, 2015.*

*[4] T. Yu, Y. Zhu and X. Wang, "Autoencoder neural network-based abnormal data detection in edge computing enabled large-scale IoT systems", Chinese Journal on Internet of Things, vol. 2, no. 4, 2018.*

*[5] T. Beck, M. Werner, S. Feld and S. Schimper, "Mobile Edge Computing: A Taxonomy", in The Sixth International Conference on Advances in Future Internet, 2014.*

*[6] W. Shi and X. Zhang, "Edge Computing: State-of-the-Art and Future Directions", Computer Research and Development, vol. 56, pp. 69-89, 2019*

*[7] F. Giust, X. Costa-Perez and A. Reznik, "Multi-Access Edge Computing: An Overview of ETSI MEC ISG", IEEE 5G Tech Focus, vol. 1, no. 4, 2017.*

*[8] 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, 2016.*

*[9] L. Huang, S. Bi and Y. Zhang, "Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-Edge Computing Networks", IEEE Transactions on Mobile Computing, pp. 1-1, 2019.*

*[10] J. Liu, G. Shou, Y. Liu, Y. Hu and Z. Guo, "Performance Evaluation of Integrated Multi-Access Edge Computing and Fiber-Wireless Access Networks", IEEE Access, vol. 6, pp. 30269-30279, 2018*

*[11] Z. Ding, Y. Liu, J. Choi, Q. Sun, M. Elkashlan, I. Chih-Lin and H. Poor, "Application of Non-Orthogonal Multiple Access in LTE and 5G Networks", IEEE Communications Magazine, vol. 55, no. 2, pp. 185-191, 2017.*

*[12] H. Huang, J. Xiong, J. Yang, G. Gui and H. Sari, "Rate region analysis in a full-duplex-aided cooperative nonorthogonal multiple-access system", IEEE Access, vol. 5, pp. 17869-17880, 2017.*

*[13] Y. Zhang, H. Wang, T. Zheng and Q. Yang, "Energy-efficient transmission design in non-orthogonal multiple access", IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, vol. 66, no. 3, pp. 2582-2857, 2017.*

*[14] F. Fang, H. Zhang and J. Cheng, "Energy-Efficient Resource Allocation for Downlink Non-Orthogonal Multiple Access (NOMA) Network", IEEE Transactions on Communications, vol. 64, no. 9, pp. 3722-3723, 2016.*

*[15] Z. Ding, D. Ng, R. Schober and H. Poor, "Delay Minimization for NOMA-MEC Offloading", IEEE Signal Processing Letters, vol. 25, no. 12, pp. 1875-1879, 2018.*

*[16] Y. Wu, L. Qian, H. Mao, X. Yang, H. Zhou and X. Shen, "Optimal Power Allocation and Scheduling for Non-Orthogonal Multiple Access Relay-Assisted Networks", IEEE Transactions on Mobile Computing, vol. 17, no. 11, pp. 2591-2606, 2018.*

*[17] Z. Ding, P. Fan and H. Poor, "Impact of Non-Orthogonal Multiple Access on the Offloading of Mobile Edge Computing", IEEE Transactions on Communications, vol. 67, no. 1, pp. 375-390, 2019.*