Université de Montréal

# Mobile Data and Computation Offloading in Mobile Cloud Computing

par

**Dongqing Liu**

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée à la Faculté des études supérieures et postdoctorales
en vue de l'obtention du grade de
Philosophiæ Doctor (Ph.D.)
en Discipline

juillet 2019

# Résumé

Le trafic mobile augmente considérablement en raison de la popularité des appareils mobiles et des applications mobiles. Le déchargement de données mobiles est une solution permettant de réduire la congestion du réseau cellulaire. Le déchargement de calcul mobile peut déplacer les tâches de calcul d'appareils mobiles vers le cloud. Dans cette thèse, nous étudions d'abord le problème du déchargement de données mobiles dans l'architecture du cloud computing mobile. Afin de minimiser les coûts de transmission des données, nous formulons le processus de déchargement des données sous la forme d'un processus de décision de Markov à horizon fini. Nous proposons deux algorithmes de déchargement des données pour un coût minimal. Ensuite, nous considérons un marché sur lequel un opérateur de réseau mobile peut vendre de la bande passante à des utilisateurs mobiles. Nous formulons ce problème sous la forme d'une enchère comportant plusieurs éléments afin de maximiser les bénéfices de l'opérateur de réseau mobile. Nous proposons un algorithme d'optimisation robuste et deux algorithmes itératifs pour résoudre ce problème. Enfin, nous nous concentrons sur les problèmes d'équilibrage de charge afin de minimiser la latence du déchargement des calculs. Nous formulons ce problème comme un jeu de population. Nous proposons deux algorithmes d'équilibrage de la charge de travail basés sur la dynamique évolutive et des protocoles de révision. Les résultats de la simulation montrent l'efficacité et la robustesse des méthodes proposées.

**Mots clés:** Informatique mobile, Informatique dans les nuages, Théorie des jeux, Processus de Markov

# Abstract

Global mobile traffic is increasing dramatically due to the popularity of smart mobile devices and data hungry mobile applications. Mobile data offloading is considered as a promising solution to alleviate congestion in cellular network. Mobile computation offloading can move computation intensive tasks and large data storage from mobile devices to cloud. In this thesis, we first study mobile data offloading problem under the architecture of mobile cloud computing. In order to minimize the overall cost for data delivery, we formulate the data offloading process, as a finite horizon Markov decision process, and we propose two data offloading algorithms to achieve minimal communication cost. Then, we consider a mobile data offloading market where mobile network operator can sell bandwidth to mobile users. We formulate this problem as a multi-item auction in order to maximize the profit of mobile network operator. We propose one robust optimization algorithm and two iterative algorithms to solve this problem. Finally, we investigate computation offloading problem in mobile edge computing. We focus on workload balancing problems to minimize the transmission latency and computation latency of computation offloading. We formulate this problem as a population game, in order to analyze the aggregate offloading decisions, and we propose two workload balancing algorithms based on evolutionary dynamics and revision protocols. Simulation results show the efficiency and robustness of our proposed methods.

**Keywords:** Mobile computing, Cloud computing, Game theory, Markov process

# Table des matières

# Liste des tableaux

# Liste des figures

# List of Acronyms

| | |
|---|---|
| **AP** | **A**ccess **P**oint |
| **CC** | **C**loud **C**omputing |
| **D2D** | **D**evice to **D**evice |
| **FHMDP** | **F**inite **H**orizon **M**arkov **D**ecision **P**rocess |
| **IoT** | **I**nternet **o**f **T**hings |
| **IoTD** | **I**nternet **o**f **T**hings **D**evice |
| **LTE** | **L**ong-**T**erm **E**volution |
| **MCC** | **M**obile **C**loud **C**omputing |
| **MCO** | **M**obile **C**omputation **O**ffloading |
| **MCS** | **M**obile **C**rowd**s**ourcing |
| **MDO** | **M**obile **D**ata **O**ffloading |
| **MDP** | **M**arkov **D**ecision **P**rocess |
| **MEC** | **M**obile **E**dge **C**omputing |
| **MIA** | **M**ulti- **I**tem **A**uction |
| **MNO** | **M**obile **N**etwork **O**perator |
| **MH** | **M**obile **H**elper |
| **MU** | **M**obile **U**ser |
| **MS** | **M**obile **S**ubscriber |
| **QoE** | **Q**uality **o**f **E**xperience |
| **QoS** | **Q**uality **o**f **S**ervice |
| **VCG** | **V**ickrey-**C**larke-**G**roves |

# Dedication

To my Mother, Father
Girlfriend, and Son.

# Acknowledgments

I would like to thank all people who have helped me to achieve my goal. This dissertation would not have been possible without their help.

My deep appreciation goes to Prof. Abdelhakim Hafid and Prof. Lyes Khoukhi for giving me valuable advices in my research work and helping me a lot in my daily life. Their strict supervision, great support and constructive suggestions have helped me become a better researcher.

I would like also to thank all my colleagues at LRC and ERA Laboratories for the beneficial discussions, research collaboration, and continuous exchange of knowledge. I have a great time with these nice and brilliant colleagues.

In addition, I would like to thank all my friends in Montreal, Canada and in Troyes, France, for their continuous support and encouragements.

Last but not least, I would like to thank my parents for their never ending support. They suggested me to pursue my doctoral degree.

# Chapitre 1

---

# Introduction

We begin this chapter by introducing the concept of mobile data offloading, mobile computation offloading, mobile cloud computing and mobile edge computing in Section 1.1. We then present the motivations and contributions in Section 1.2 and Section 1.3, respectively. We finally introduce the thesis organization in Section 1.4.

## 1.1. Research Background

In this section, we present the research background of two mobile techniques (i.e., mobile data offloading and mobile computation offloading), and two computing paradigms (i.e., mobile cloud computing and mobile edge computing).

Section 1.1.1 presents two data offloading techniques: WiFi offloading and device-to-device (D2D) offloading. Section 1.1.2 presents two computation offloading techniques: mobile cloud computing and mobile edge computing. These two computation offloading techniques lead to two emerging computing paradigms, namely, mobile cloud computing and mobile edge computing, We present the details of these two paradigms in Sections 1.1.3 and 1.1.4 respectively.

### 1.1.1. Mobile Data Offloading

Data traffic in cellular networks has seen an exponential rise, due to the explosion of mobile devices and mobile applications. The rapid growth of mobile data traffic raises big challenges to cellular networks. Global mobile data traffic grew 63 percent and reached 7.2 exabytes per month in 2016, which is 18-fold over the past 5 years [1]. The huge amount of mobile data traffic exceeds the capacity of cellular networks and reduces quality of service (QoS) of the network. [2]. To address such challenges, one simple solution is to increase the capacity of cellular networks, which is inefficient and expensive due to the corresponding expensive investments in radio access networks and the core infrastructure. One promising solution, namely mobile data offloading (MDO), is to offload cellular traffic to other kinds of networks, e.g. WiFi access points and D2D communication; this can solve the cellular traffic overload problem.

Mobile data offloading refers to the use of complementary network technologies and innovative techniques for delivery of data originally targeted for cellular networkss in order to alleviate congestion and make better use of available network resources. The objective is to maintain QoS for customers, while also to reduce the cost and impact of carrying capacity-hungry services on the wireless network [3]. It is expected that mobile data offloading will become a key industry segment in the near future as data traffic on mobile networks continues to increase rapidly [1].

There are two types of data offloading, WiFi offloading and D2D offloading, as shown in Figs. 1.1 and 1.2, respectively. WiFi offloading uses WiFi hot spots to transfer data originally targeted to cellular networks, while D2D offloading uses nearby mobile helper (MH) to transfer data to mobile subscriber (MS).

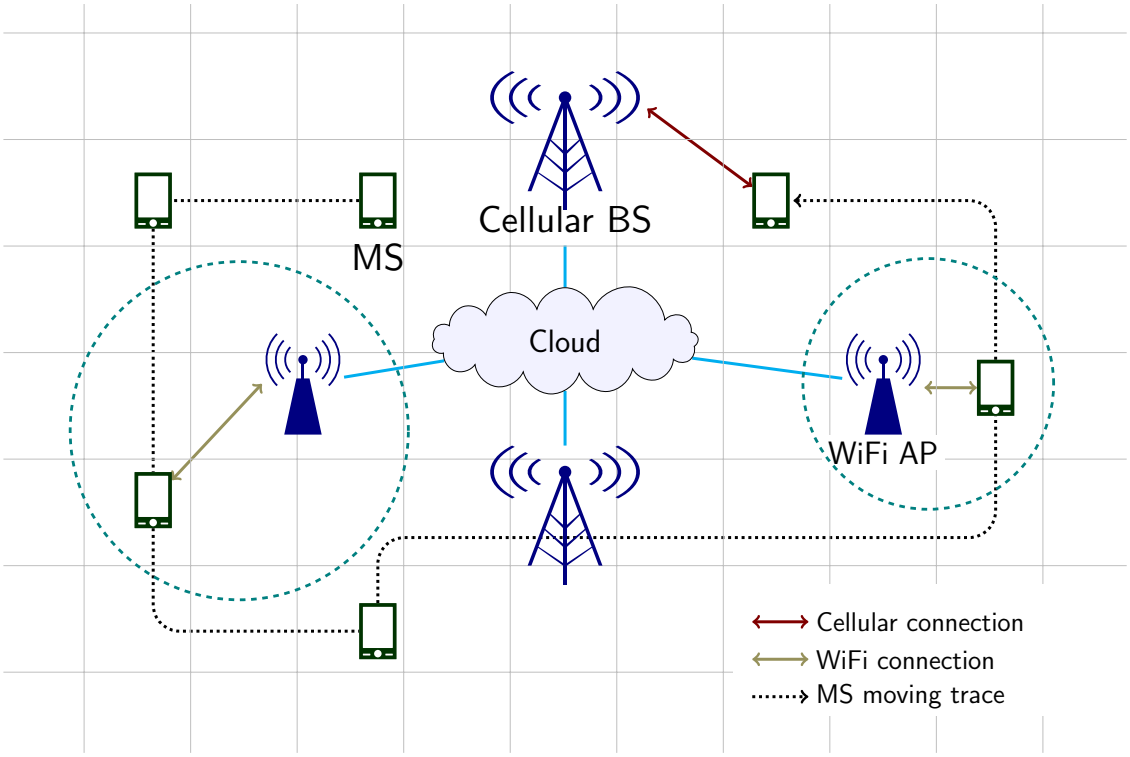



**Fig. 1.1.** WiFi offloading.

#### 1.1.1.1. *WiFi offloading*

WiFi offloading is considered as a promising solution to reduce mobile data traffic in cellular networks. WiFi Access Points (APs) can efficiently reduce cellular traffic. It is shown that about 65% of cellular traffic can be offloaded through WiFi APs[4]. Although WiFi APs can provide better data rate than cellular networks, their coverage area is much smaller than cellular networks [5].

#### 1.1.1.2. *D2D offloading*

D2D offloading (opportunistic offloading) is based on D2D communication [6]. D2D offloading uses the store-carry-forward strategy, where some mobile users can

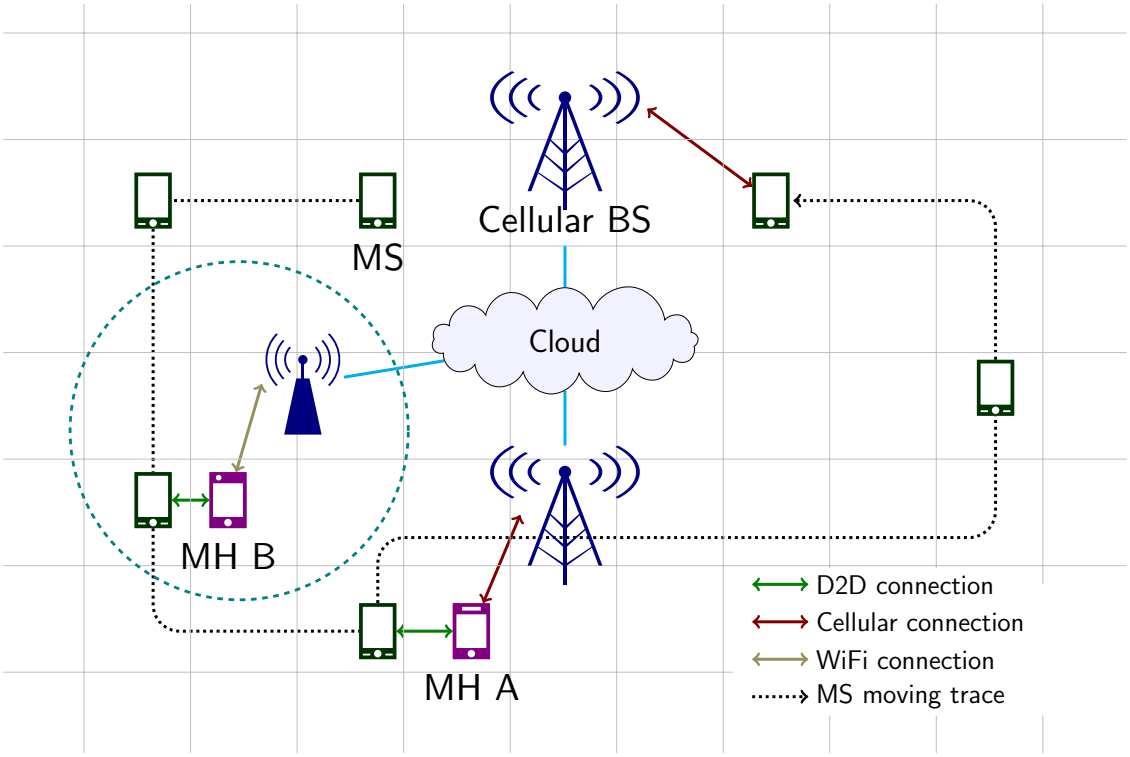



**Fig. 1.2.** D2D offloading.

store data in the buffer (called MHs), carry the data when they are moving, and forward the data to other mobile users (called MSs) [7]. When mobile network operator (MNO) wants to deliver data to MSs, it can first send the data to MHs. Then, MHs will transmit data to MSs using opportunistic connections. With more than half a billion mobile devices and connections added in 2015 [1], D2D communication is becoming an important data delivery scheme. However, the data rate of D2D communication is low and the mobility patterns of MHs or MSs are difficult to predict. The comparison between WiFi offloading and D2D offloading is shown in Table 1.1.

**Tab. 1.1.** Comparison of WiFi offloading and D2D offloading

| | **Coverage Area** | **Delay** | **Mobility** | **Data Size** | **Security** |
|---|---|---|---|---|---|
| **WiFi offloading** | Medium | Low | No | Large | Medium |
| **D2D offloading** | Small | Medium | Yes | Small | Low |

### 1.1.2. Mobile Computation Offloading

Mobile devices are widely used as the most convenient communication tools in our life. Mobile users can acquire large amounts of various services from mobile applications running on local devices and/or on remote servers via wireless networks. However, mobile devices are facing many challenges in their resources (e.g., battery life, storage, and bandwidth) and communications (e.g., mobility, availability and heterogeneity). These challenges may reduce the QoS that can be provided to mobile users. One feasible solution, mobile computation offloading (MCO), is to offload part of the computation tasks from mobile devices to remote cloud servers or local edge servers. MCO can improve the performance of mobile devices by taking advantage of computation offloading. Moreover, MCO can reduce energy consumption in mobile devices and/or implement sophisticated applications by offloading computation intensive tasks to cloud or edge servers with higher computation and storage capabilities. We first introduce the computation offloading decision problem and then discuss the computation offloading process.

There are mainly three kinds of computation offloading decisions.

- Non-offloading: mobile computation is executed fully in mobile devices;
- Full offloading: the whole computation is offloaded from mobile devices to cloud servers or edge servers;

- Partial offloading: only a part of computation is executed in mobile devices while the rest is executed in servers.

The objective of MCO is to minimize the processing delay of mobile applications, to minimize energy consumption of mobile devices while satisfying delay constraints or to trade off between processing delay and energy consumption. There are two types of computing paradigms, namely, mobile cloud computing introduced in Section 1.1.3 and mobile edge computing introduced in Section 1.1.4.

### 1.1.3. Mobile Cloud Computing

Mobile devices are capable of supporting large numbers of mobile applications, some of which demand an ever increasing computational power. This poses a challenge because mobile devices are resource constrained devices with limited computation power, memory, storage, and energy. Fortunately, cloud computing technology offers virtually unlimited dynamic resources for computation, storage, and service provision. Therefore, researchers did envision extending cloud computing services to mobile devices to overcome the constraints of mobile devices. The challenge in doing so is that traditional mobile application models do not support the development of applications that can incorporate cloud computing features; they require specialized mobile cloud application models. In order to solve this problem, mobile cloud computing (MCC) is proposed and defined as follows [8].

*Mobile cloud computing at its simplest, refers to an infrastructure where both the data storage and data processing happen outside of the mobile device. Mobile cloud applications move the computing power and data storage away from mobile phones and into cloud, bringing applications and mobile computing to not just mobile users but a much broader range of mobile subscribers.*

MCC is an extension of cloud computing (CC). It integrates CC into the mobile environment. CC is widely recognized as the next generation computing infrastructure. Cloud providers provide computing, storage, services and applications as services to cloud users. CC enables users to utilize cloud resources in on-demand and pay-as-you-go model. Moreover, it helps reducing capital cost, decouples services from the underlying technology, and provides flexibility in terms of resource provisioning. MCC can take advantage of the benefits of CC to improve the QoS of mobile services. With the explosive growth of mobile applications, MCC has been proposed as a potential technology for mobile services. MCC integrates CC into mobile devices to overcomes obstacles related to the performance (e.g., battery life, storage, and bandwidth), environment (e.g., heterogeneity, scalability, and availability), and security (e.g., reliability and privacy).

1.1.3.1. *Advantages of MCC*

MCC is an extension of cloud computing. It can overcome several obstacles in mobile computing. The following are the advantages of introducing cloud computing into mobile environment.

- MCC can extend battery lifetime of mobile devices. In [9], the authors show that cloud computing can save energy for mobile systems. Since the battery for mobile devices is limited, offloading computation intensive tasks to cloud computing can save computation time and energy for mobile devices.
- MCC can store and process data in cloud side. It can extend the storage capacity for mobile devices. MCC is developed to enable mobile users to store/access large amounts of data on cloud through wireless networks.
- MCC can reduce the running cost for compute-intensive applications that consume large amount of computing resources. It can help run applications that cannot be executed on the limited-resources devices.

- MCC can improve reliability. Storing and processing data on cloud side is an effective way to improve the reliability thanks to the back-up technology that is used in cloud servers. This reduces the chance of data loss/damage on mobile devices.

#### 1.1.3.2. *Challenges of MCC*

Although it has many advantages for cloud providers and mobile users, MCC has to face many issues in computation side and mobile communication side. For computation side, the optimal program partition for offloading is difficult to find. Also, it is difficult to obtain the accurate execution time of computations because the time varies in different instances of the computations, and the inaccurate information results in inefficient offloading performance. For communication side, bandwidth is the most important issue for MCC because the radio resource in wireless networks is scarce. This issue is even worse with the increase of the number of smart mobile devices and data heavy mobile applications, such as video streaming and cloud backup.

Many researchers proposed optimal and efficient solutions for bandwidth allocation. New technologies (e.g. 5G network) are being developed to increase significantly bandwidth for wireless communication. However, bandwidth limitation is still a big concern because the number of mobile devices is dramatically increasing.

- **Low bandwidth**. Bandwidth is one of the big issues in MCC because the radio resource for wireless networks is much scarce as compared with traditional wired networks.
- **Availability**. Service availability becomes a more important issue in MCC than that in CC with wired networks. Mobile users may not be able to connect to cloud to obtain a service due to traffic congestion, network failures, and out-of-coverage.

- **Heterogeneity**. Mobile cloud computing will be used in highly heterogeneous networks in terms of wireless network interfaces. Different mobile nodes access cloud through different wireless networks, such as cellular and WiFi networks. This raises the problem of which wireless interface to use while satisfying satisfying MCC's requirements (e.g., always-on connectivity, on-demand scalability of wireless connectivity, and energy efficiency of mobile devices).
- **Enhancing the efficiency of data access**. Handling data storage on cloud is not easy because of the low bandwidth, mobility, and the limitation of resource capacity of mobile devices.

Mobile users need to access to servers located in cloud when requesting services and resources in cloud. However, mobile users may face some problems such as congestion due to the limitation of wireless bandwidth, network disconnection, and signal attenuation caused by mobile users' mobility. This may cause delays when users want to communicate with cloud degrading significantly QoS. To reduce network delay, mobile edge computing has been proposed.

### 1.1.4. Mobile Edge Computing

MCC introduces significant processing delay of computation offloading, consisting of uploading computation-related data (e.g., programming codes and input data) to cloud, code execution in cloud and getting back the computation result. Especially, the delay incurred between mobile users and cloud makes computation offloading unsuitable for many real-time applications. Moreover, transmitted data may not reach cloud and computation results may be lost while being returned to mobile devices. Instead of offloading tasks to remote cloud directly, mobile devices can offload tasks to nearby servers.

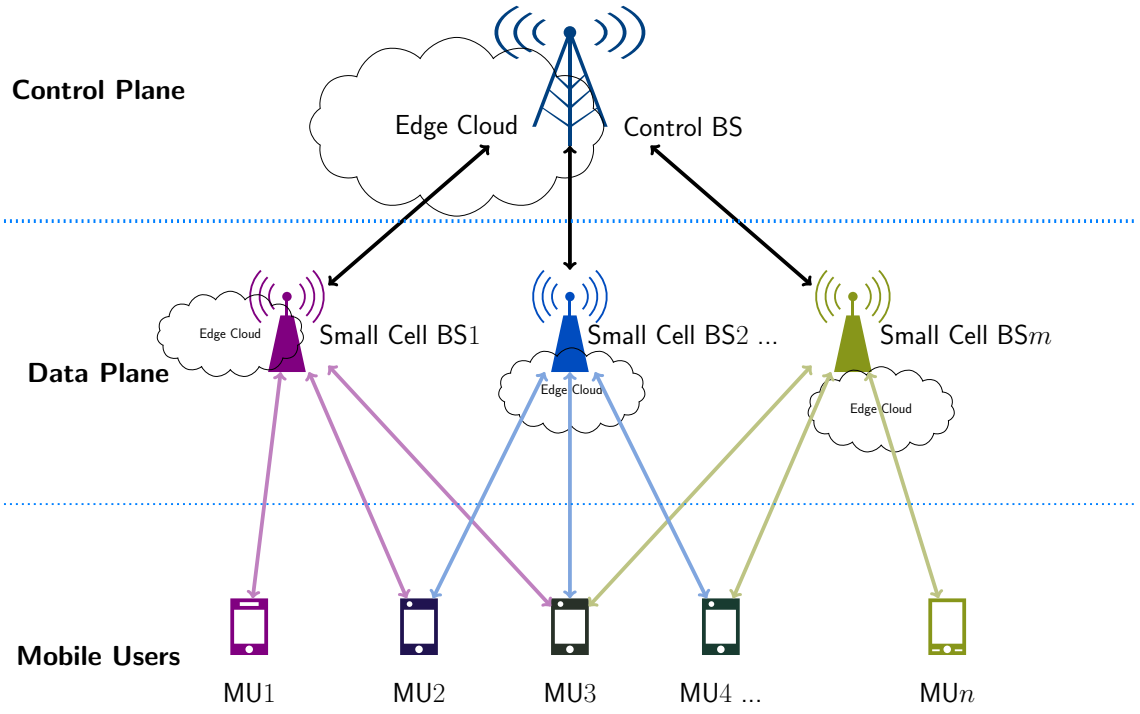



**Fig. 1.3.** Mobile edge computing in software defined networks.
BS denotes Base Station and MU represents Mobile User.

To cope with the delay problem introduced by MCC, another computing paradigm, mobile edge computing (MEC) is proposed. The main idea of MEC is to bring computation and storage resources to the edge of mobile networks while meeting strict delay constraints with short data transmission distance [10]. Bringing computation close to mobile users is the key concept of MEC. Fig. 1.3 illustrates MEC in software defined networks. Edge cloud can enhance small cells, e.g., microcells, picocells or femtocells, with augmented computation capability and storage capability. Since a large number of small cells will be deployed in the future, MEC is considered to be a promising destination of mobile computation offloading, especially for mobile tasks or applications having stringent latency constraints. The control BS is used to implement dynamic and elastic resource management of small cells and

have the knowledge of information about mobile users (e.g., user location and mobile tasks), information about edge clouds (e.g., computation resources and storage resources) and information about small cell BSs (e.g., network condition and signal interference). Based on this information, control BS is in charge of the computation offloading process to optimize the system performance (e.g., energy consumption minimization or delay minimization). A comparison between MEC and MCC is shown in Table 1.2.

**Tab. 1.2.** Comparison of MCC and MEC

| | **Capability** | **Latency** | **Scalability** | **Architecture** | **Location** | **Security** |
|---|---|---|---|---|---|---|
| **MCC** | Strong | High | Low | Centralized | Far | High |
| **MEC** | Medium | Low | High | Decentralized | Close | Low |

Another similar concept related to edge cloud, the so-called cloudlet has been proposed. A cloudlet is a trusted, resource-rich computer or cluster of computers which is well-connected to the Internet and available for use by nearby mobile devices [8]. Cloudlet is an important complement to the device-cloud hierarchy. It refers to a layer connecting mobile devices and cloud servers in MCC; it plays a mediator role focusing on the business logic [11]. It is a self-management mechanism that is used to strengthen communications between mobile devices and cloud servers by reducing latency [12].

## 1.2. Motivations

Although mobile data offloading can significantly reduce cellular traffic, the task of developing a comprehensive and reliable mobile data offloading system remains

challenging. A key challenge is how to achieve an efficient data offloading coordination among multiple mobile devices. By opportunistic utilization of lower cost access points, mobile subscribers will have better wireless access service with lower cost. In contrast, MNOs who have deployed these access points want to maximize the revenue by selling bandwidth. Thus, how to effectively allocate this bandwidth to mobile devices effectively becomes a key problem to be solved.

Given the limited bandwidth of APs deployed in a mobile data offloading market, when demands of mobile devices exceed supply, MNO needs to allocate the bandwidth to mobile devices and decide the price for allocated bandwidth in order to achieve the highest revenue. Auction mechanism is considered as an economically efficient approach towards the allocation of APs' bandwidth, and assigns bandwidth to mobile users who value it the most [13–16]. In a real-world data offloading market, the bidding prices of mobile users are private information unavailable for MNO. However, MNO may use historical information to identify the numerical characteristics of the bidding prices. Consequently, it is natural to consider how to model the bidding prices based on historical information. Here, uncertainty set is used to model the possibility of bidding prices. MNO assumes that all bidding prices belong to the uncertainty set derived from historical information. Then, MNO makes an offloading mechanism based on the uncertainty set instead of some fixed bidding prices.

The challenges for mobile devices are due to the characteristic of mobility (or wireless). Because of mobility, mobile devices do not have continuous power supply; this is one of the key problems for these devices. At the same time, mobile devices lack stable/continuous network connection due to wireless networks. Although having great improvement in recent years, mobile devices still have limited computing and storage resources.

These restrictions make many energy-consumption applications not suitable running in mobile devices. This is because, for example, these applications usually

consume too much power resources and generate lots of heat causing bad user experience. Furthermore, many sophisticated applications are not suitable to execute in mobile devices with restricted computation, memory and storage capacity.

To make MCC a reality, a number of problems need to be solved. These problems include: (1) intermittent connection caused by mobility and wireless environment, and (2) limited energy supply in mobile devices. Even though there has been a lot improvements in mobile devices (e.g., computing resources, storage ability and battery life) and wireless networks (e.g., LTE), there is still a need to address these problems:

- Limited battery life has been found as the biggest complaint for smartphones [17]. Two main factors contribute to the energy problem. One is the limited capacity of batteries. The other is the increasing demand for energy-hungry applications (such as video streaming and online gaming).
- Limited bandwidth and large network latency has a great impact on MCC. Also, intermittent network connection may cause problems for MCC. Indeed, even though the network condition may greatly improve in the future, with the development of high speed wireless network technologies (such as 5G technology [18]), it is not sufficient to solve the problem.
- Costly network access: Network access cost has a big impact on MCC users, since cellular networks service (e.g., 4G LTE) is more expensive than traditional wired Internet access or WiFi service.
- Problems in service integration: Using services in MCC involves both mobile service provider (MSP) and cloud service provider (CSP). However, MSPs and CSPs have different services management, customers management, methods of payment, and prices.

IoT is proposed to equip everyday objects with electronics, software, sensors, and network connectivity, and bring the vision of a connected world into reality [19].

However, computation-intensive applications, such as e-health, automatic driving, and industrial automation, consume large amounts of computing and storage capabilities of IoT devices. These sophisticated applications have stringent requirements of computation resources and processing delay on IoT Devices (IoTDs). However, IoTDs are resource-constrained and have limited computational capacities and battery life. Running computation intensive applications on IoTDs would result in high energy consumption and long processing delay [20]. The tension between computation intensive applications and resource constrained IoTDs brings a significant challenge for future mobile development. MEC is envisioned to be a promising solution to address this challenge, with the objective to provide cloud computing capabilities to IoTDs through radio access network [21]. By offloading computation intensive tasks to edge cloud (or MEC server) in proximity, local energy consumption on IoTDs can be reduced and local processing delay may be shortened [22].

To offload computation intensive tasks to edge cloud, task-related data should be transferred between IoTDs and edge cloud through base station (BS). If BS is congested by large amounts of IoTDs choosing to offload tasks simultaneously, the quality of experience and QoS of IoTDs will not be guaranteed [23, 24]. Moreover, facing the rapid increase of IoTDs and massive offloading tasks, the resource bottleneck of edge cloud becomes significant, since edge cloud has relatively limited resources compared to cloud [25]. Thus, lack of proper offloading coordination among large amounts of self-interested IoTDs may lead to large communication and computation latencies due to insufficient resources and severe interferences [26, 27]. As a result, how to design an energy-efficient offloading mechanism while satisfying the processing delay requirements becomes a challenging problem, especially when large amounts of IoTDs compete for limited resources.

## 1.3. Contributions

In this this thesis, we produced three contributions. Each contribution, coming from one journal paper, is made up of specific mobile technique, computing paradigm and mathematical tool. The characteristics of these contributions are shown in Table 1.3.

**Tab. 1.3.** Classification of contributions

| Contributions | Mobile Technique | Computing Paradigm |
|---|---|---|
| **Chapter 3** | Mobile Data Offloading | Mobile Cloud Computing |
| | Markov Decision Process, Paper [1] | |
| **Chapter 4** | Mobile Data Offloading | Heterogeneous Network |
| | Multi-Item Auction, Robust Optimization, Paper [2] | |
| **Chapter 5** | Mobile Computation Offloading | Mobile Edge Computing |
| | Population Game, Potential Game, Paper [3] | |

In the first contribution, we propose two mobile data offloading schemes based on Finite Horizon Markov Decision Process (FHMDP). Our objective is to minimize the communication cost for delivering mobile data with different delay sensitivities through multiple wireless networks, i.e., cellular networks, WiFi network and D2D communication. More specifically, in this contribution:

- We propose a hybrid offloading model, where multiple wireless networks are used to transfer mobile data. MNO can minimize the total communication cost by selecting different networks.

- We formulate the data offloading problem in hybrid wireless networks as an FHMDP model, and propose an offloading algorithm that can support different delay requirements (i.e., loose and tight delay tolerant).
- We prove that there exit threshold structures in the optimal policy and propose a monotone offloading algorithm for generating monotone policy with lower computational complexity.
- The simulation results demonstrate that our proposed schemes achieve the lowest communication cost as compared with three existing schemes.

In the second contribution, we focus on designing an efficient auction mechanism for allocating APs' bandwidth among multiple MSs; this is considered as a multi-item auction problem. MNO which owns the network infrastructure acts as the auctioneer and sells bandwidth to mobile devices through an auction. We formulate the auction problem based on robust optimization which models the desirable properties (budget feasibility, incentive compatibility, and individual rationality) of optimal auctions enabling the auctioneer to use historical data or prior knowledge of valuations. The uncertainty of item valuations is modeled as an uncertainty set, which is constructed based on limit theorems of probability theory. The optimal auction mechanism with reservation price has the structure of a Vickrey-Clarke-Groves (VCG) mechanism [28]. In this contribution:

- We characterize the interaction among MNO and MSs in a multi-item auction aiming at maximizing the MNO's revenue and the amount of offloaded traffic from mobile subscribers (MSs). Our proposed multi-item auction calculates reservation prices based on the uncertainty set and the MSs' budgets; this can prevent market manipulation. Our proposed auction is implemented by robust optimization. Instead of requiring the full knowledge of MSs' valuations, robust optimization uses few information of MSs' valuations and can obtain a global $\epsilon$-optimal solution.

- Since the optimal multi-item auction problem is difficult to solve, we propose two greedy auctions that can solve the offloading market problem in polynomial time, while preserving the properties of budget feasibility, incentive compatibility, and individual rationality. These two greedy auctions outperforms each other in different network scenarios.
- We perform numerical analysis and comparative evaluation of the proposed optimal and greedy auctions, considering realistic network scenarios. We further illustrate that the proposed offloading mechanisms can improve cellular data offloading performance and has higher robustness compared to Myerson auction.

In the third contribution, we propose a population game based approach to investigate workload balancing problem for MEC in IoT. Population game is envisioned as a powerful tool to model strategic interactions among large amounts of agents [29, 30]. Specifically, we model the offloading decision making problem among large amounts of competing IoTDs as a population game, wherein IoTDs are self-interested agents and make offloading decisions individually. In this contribution:

- We formulate MEC workload balancing problem as a population game and propose an IoT Device classification model. We design an inference affected queueing model that can capture the inference among IoTDs. We use $\alpha-$ utility function to implement different kinds of workload balancing.
- We calculate NE dynamically, i.e., IoTDs can change their offloading decisions through some learning mechanism. The learning mechanism is defined as a revision protocol that allows IoTDs to adjust their offloading decisions based on decisions of other IoTDs in proximity. The evolutionary process of IoTDs's offloading strategies can be modeled by evolutionary game dynamics (i.e., a differential equation). The evolutionary game dynamics describes the variation of IoTDs's offloading decisions until an NE is obtained.

- We propose two workload balancing algorithms, namely centralized workload balancing algorithm and decentralized workload balancing algorithm, based on the concept of evolutionary dynamics and revision protocols, respectively. We show that these algorithms can achieve an Nash Equilibrium (NE). Simulation results illustrate the evolutionary dynamics and show that the proposed algorithms can achieve efficient workload balancing in BSs and edge clouds.

## 1.4. Organization

The rest of this thesis is organized as follows. Chapter 2 presents related works. Chapter 3 presents our Markov decision process based mobile data offloading. Chapter 4 presents our multi-item auction based mobile data offloading. Chapter 5 presents our population game based workload balancing in mobile edge computing. Chapter 6 concludes the thesis, presents potential future research directions and lists the journal papers and conference papers produced during this thesis.

# Chapitre 2

---

# Review of the Literature

This chapter provides the related work for mobile data offloading and mobile computation offloading in mobile cloud computing and mobile edge computing.

## 2.1. Review on Mobile Data Offloading

### 2.1.1. Mobile Data Offloading From Implemention Perspective

In the following, we present a survey of prior work aiming to offload cellular traffic to other mobile networks, including WiFi network and D2D communication, to reduce the network congestion.

Several contributions have shown the benefits of offloading mobile data from cellular network to WiFi network. Song et al. [31] investigated offloading schemes for cellular and WLAN integrated networks. They considered the WLAN-first resource allocation scheme where WLAN connection is used whenever possible, in order to benefit from low cost and large bandwidth of WLAN. Siris et al. [32] investigated the methods for enhancing mobile data offloading from mobile networks to WiFi APs by using mobility prediction and prefetching techniques. They evaluated these methods in terms of offloading ratio, data transmission time and cache size when using prefetching. Cheng et al. [33] presented an analytical framework for offloading

cellular traffic to WiFi network using queuing theory. They evaluated the offloading performance in terms of average service delay. Mehmeti et al. [34] evaluated the performance of on-the-spot mobile data offloading. They analyzed the performance improvement by WiFi-based offloading using queuing theory. Jung et al. [35] proposed a network-assisted user-centric WiFi-offloading model in a heterogeneous network; the objective was to maximize throughput for each MS by utilizing network information.

Other contributions have shown the possibility of offloading mobile data from cellular network to D2D network. The main idea is to transmit mobile data using opportunistic communication among MSs; this has been shown to provide significant wireless capacity gains. Vinicius et al. [36] proposed a multi-criteria decision-making framework for data offloading from 3G network to D2D network. The framework avoids changes in the infrastructure by employing only user knowledge to select MHs. It shows that delay tolerant applications can offload six-fold mobile data compared to delay sensitive applications. Sciancalepore et al. [37] considered data offloading in D2D network with heterogeneous node mobility patterns. They used an optimization method to minimize cellular network traffic while satisfying the applications' constraints. Filippo et al. [38] proposed a method, called DROiD, to control popular data distribution in D2D network; the aim was to minimize the usage of infrastructure resources. They did show that the proposed method can offload a significant amount of data from cellular network to D2D network under tight delivery delay constraints. Andreev et al. [39] investigated the offloading method from cellular network to D2D network. They demonstrated that assisted offloading of cellular user sessions into D2D links improves the degree of spatial reuse and reduces the impact of interference.

Since the coverage area of D2D network is flexible with the movement of MHs, it can help offload data when WiFi connections are not available, especially for

transmitting small size data, due to the short connection time and low data rate. However, since the data rate of WiFi network is higher than that of D2D network and WiFi network is more stable than D2D network, WiFi based offloading generally outperforms D2D based offloading from MS's perspective [40]. Notice that D2D based offloading can offload significant mobile data from MNO's perspective, since the number of MHs can be quite large. In the simple case, WiFi APs can be considered as a special kind of MHs. Compared with MHs, WiFi APs are installed at some fixed locations and have more bandwidth.

We conclude that most existing contributions are based on WiFi offloading or opportunistic networks, without considering the combination of different mobile networks. In this thesis, we consider a hybrid offloading model, where mobile data can be offloaded through WiFi offloading and D2D communication. Our objective is to minimize the overall cost for data delivery while satisfying delay requirements of different user types.

### 2.1.2. Mobile Data Offloading From Decision Making Perspective

To cope with the growth of cellular traffic, some existing contributions have studied efficient data offloading methods from the perceptive of data offloading decision making. Cheung et al. [41] proposed a Markov decision process based network selection algorithm for delay-tolerant applications under the setting of a single MS. Barbarossa et al. [42] proposed a centralized scheduling algorithm to jointly optimize the communication and computation resource allocations among multiple users with latency requirements. Kang et al. [43] studied the offloading problem from MNO's perspective and proposed a usage-based charging model to maximize MNO's revenues. Wu et al. [44] studied optimal resource allocation for data offloading via dual-connectivity, while taking into account the trade-off between optimal bandwidth allocation for base stations and optimal power allocation for mobile users.

Other contributions have investigated data offloading problems based on auction theory or game theory. Chen et al. [45] studied the scenario where multiple users can access the same wireless base station, and designed a decentralized offloading mechanism that ensures the scalability of the proposed mechanism with the number of mobile users. Cheng et al. [46] took into consideration users' mobility information and proposed an auction based offloading mechanism to maximize MSs' social welfare and improve MNO's revenues. Lee et al. [47] proposed a two-stage sequential game to model the interaction between MNO and MSs, and demonstrated, via simulations, that WiFi offloading is economically beneficial for both MNO and MSs. Paris et al. [48] proposed a reverse auction based offloading algorithm leasing WiFi access points, owned by third parties, to allocate bandwidth to multiple mobile users. However, all these contributions assume that all players are rational and will take the truthful bidding. Different from existing contributions, we consider to implement worst case optimality as long as the bid values belong to the uncertainty set constructed by historical bidding information.

### 2.1.3. Mobile Data Offloading From Market Perspective

Most existing studies on multi-item mechanisms aim to maximize MNO's revenue or incentivize the participation of MSs. Zhao et al. [49] proposed an online auction method to maximize the value of services in mobile crowdsourcing (MCS), and to incentivize the participation of MSs in MCS applications. Gan et al. [50] proposed a reverse auction method to incentivize the participation of MSs in MCS applications. Wang et al. [51] designed a truthful, individual rational, budget feasible and quality-aware algorithm for task allocation in MCS. However, these contributions only considered the budget feasibility of MNO. This is because that, in MCS, MSs consume their own resources such as computational resources and computing power to help MNO solve a complex problem. MNO needs to pay MSs in return. In our

model, MSs request bandwidth resources of MNO, while in MCS, MNO request services from MSs. Thus, we need to consider the budget feasibility of all MSs, which is more complex than MCS.

Other contributions consider the budget constraints of MSs. Bhattacharya et al. [52] proposed an approximation algorithm to solve the multi-item auction problem. Wang et al. [53] studied distributed truthful auction mechanism for task allocation in MCC. They proposed an auction model considering computational efficiency, individual rationality, truthfulness guarantee of the bidders, and budget balance. Jin et al. [54] investigated the resource sharing problem for cloudlets in MCC. They proposed an incentive mechanism to charge MSs and reward cloudlets. Although these contributions considered the budget constraints of MSs, they do not use the historical bidding information. In this thesis, we design an optimal multi-item auction mechanism based on the historical bidding information, while taking into consideration MSs' budget constraints.

Compared with the above mechanisms, the auction problem designed in Section 4 is rather challenging, and has the following differences: (1) we take full advantage of historical bidding information and prevent abnormal auction to destroy the multi-item auction; (2) we consider the worst case optimization problem; thus, our proposed method has strong robustness compared to other optimal auction mechanisms; and (3) our optimal auction considers reservation prices that are functions of the uncertainty set and the budgets, thus can potentially protect the MNO's revenue.

## 2.2. Mobile Cloud Computing

Mobile computation offloading among multiple MUs has been studied in the context of two different approaches, namely centralized and decentralized computation offloading. Centralized offloading considers the scenarios where MUs don't negotiate with each other. MUs first send offloading requests to MNO with computational

meta data (e.g., size of transmission data and demanding computational resources). Then, MNO can perform centralized offloading algorithm in order to implement optimal allocation [55–58]. Cao et al. [56] proposed an optimal radio resource allocation method to minimize the overall execution time. Yang et al. [57] considered an an energy optimization problem for computation offloading in order to minimize the overall energy consumption. Chen et al. [58] proposed integer optimization based method aiming to minimize the time delay while saving the battery life of mobile devices. All these contributions investigated the minimization of time delay and energy consumption separately. They do not jointly consider these two problems.

Decentralized offloading investigates the interaction among multiple MUs. The offloading decision made by each MU is affected by other MUs's decisions. Meskar et al. [59] modeled the computation offloading problem as a competitive game wherein each MU aims to minimize his energy consumption. All these contributions considered a small number of MUs, since more MUs increase the overhead of system control. As a result, the computational overhead of these offloading schemes are sensitive to the number of MUs. In this thesis, we model offloading problem as a competitive population game that is insensitive to the number of MUs. Thus, large amounts of competing MUs would not increase the computational overhead. Moreover, we study the scenario where each MU aims to minimize his overall cost, i.e., the combination of the time delay minimization problem and energy consumption minimization problem.

Although many excellent work has been proposed to investigate mobile cloud computing and mobile data offloading, these two important areas have traditionally been addressed separately in the literature. To the best of our knowledge, the joint study of mobile cloud computing and mobile data offloading for next generation cellular networks has not been addressed in previous work. In Section 3, we study the network selection problem in mobile cloud computing with the objective of optimizing

the transmission cost of mobile network operators. Despite the potential benefits brought by mobile data offloading, one of the major challenges is that the mobility of mobile users is inaccurate due to the randomness of MUs. We take a Markov decision process, which has well developed mechanisms to predict the mobilities of mobile users. An optimal policy can be found based on the particular structure of the monotone policies.

Mobile devices can be used to form a mobile computing grid due to the increase of their computation and communication capabilities. However, it is challenging to organize the heterogeneous computation and communication capabilities of mobile devices in proximity. Viswanathan et al. [60] investigated the inherent uncertainty (e.g., network connectivity and device availability) of mobile computing grid in order to implement autonomic capabilities (i.e., self-organization, self-optimization, and self-healing) among mobile devices. Chen et al. [61] studied the dynamic nature of mobile computing grid, such as frequent topology changes due to device availability and mobility. They proposed an energy-efficient data storage and processing approach while considering the fault-tolerant problem. The nature of these contributions is to offload computation to peer mobile devices. The management cost of dynamic mobile computing grid is non-trivial; the reliability of mobile computing grid is hard to be guaranteed.

## 2.3. Recent Advances in Mobile Edge Computing

As an emerging computing paradigm, MEC is considered as a key enabler for future networks, and it can improve computing and storage capacities at network edge.

### 2.3.1. Cooperation with MHs

You et al. [62] considered a co-computing system where MS offloads computation to MHs. They formulated the energy-efficient co-computing problem as two subproblems: the slave problem and the master problem. They aimed at minimizing the energy consumption for computation offloading by considering the deadline and buffer constraints. He et al. [63] studied the cooperation of D2D communications and MEC to improve the computational capacity of cellular networks, where an MU's task can be offloaded to edge cloud or nearby MU. They aimed to maximize the number of MUs that can be supported by cellular networks under the constraints of limited communication and computation resources. They formulated the computation offloading problem as a mixed integer non-linear problem and solved it with two subproblems.

Although the above studies have demonstrated the help of D2D communications in improving the computation performance of wireless networks, the limited resources at MHs are not adequate to support all mobile applications. Furthermore, it is challenging to implement efficiently distributed management among MSs and MHs.

### 2.3.2. Cooperation with Data Caching

Zhang et al. [64] studied delay-optimal edge caching in wireless networks, where the content placement and content size are optimized based on the information of network topology, mobile traffic, channel quality, and content popularity. Wang et al. [65] jointly considered computation offloading decisions and content caching strategies in MEC. They formulated the computation offloading decisions and content caching strategies as an optimization problem, while considering the total revenue of the wireless network. Liu et al. [66] proposed a blockchain-based framework with adaptive block sizes for mobile video streaming in MEC. They designed an incentive

mechanism to facilitate collaboration among content creators, video transcoders, and content consumers. They formulated the resource allocation, offloading decisions, and adaptive block sizes as an optimization problem.

However, the cached contents change frequently with the variations of MUs' requests or network conditions, which may lead to network congestion and delay in computation offloading.

### 2.3.3. Cooperation with Energy-Harvesting Devices

MEC and wireless power transfer (WPT) are considered as promising techniques to provide mobile devices with enhanced computation capability and sustainable energy supply. Wang et al. [67] designed an MEC-WPT framework, where an AP can broadcast wireless power to charge multiple mobile devices and execute tasks offloaded from mobile devices. They jointly optimized energy transmit beamforming and computation offloading at AP. Hu et al. [68] studied a scenario where mobile devices are energized by WPT from an AP and they can offload mobile tasks to the AP connected with an edge cloud. They aimed at minimizing AP's total transmit energy under the constraints of mobile tasks. Bi et al. [69] investigated the combination of WPT and MEC to achieve sustainable device operation and enhanced computational capability. They considered a WPT enhanced MEC system, where MUs follow a binary computation offloading policy. They aimed at maximizing the total computation rate of all MUs by jointly optimizing the computing mode selection and transmission time allocation.

The integration of WPT and MEC technologies can potentially tackle the two fundamental performance limitations (battery and computation) in mobile devices. Meanwhile, it brings new challenges to the management of wireless networks, e.g., WPT and computation offloading need to share the limited wireless resources.

### 2.3.4. Dynamic Offloading Decision Model

Most researchers consider computation offloading decision making in a quasi-static scenario. In order to implement dynamic offloading decision making, two kinds of approaches are proposed. The first one is online approach and the second one is based on queueing models.

Lyu et al. [70] proposed an online approach to enable cooperations of $N$ selfish MUs, where selfish behaviors are discouraged by a tit-for-tat mechanism. They achieved asymptotic optimality in a fully distributed scenario. Neto et al. [71] proposed an User-Level Online Offloading Framework (ULOOF) for mobile computation offloading. Thet aimed to minimize remote execution overhead of computation offloading. Mao et al. [72] developed an online algorithm in MEC to jointly manage the radio and computational resources . They aimed at minimizing the long-term average weighted power consumption of MUs and edge clouds.

Sarikaya et al. [73] studied the stability and dynamic control of MEC. They proposed a centralized flow control and a scheduling algorithm to stabilize the queues of mobile devices. You et al. [74] studied the energy-efficient resource-management for asynchronous MEC systems. They assumed that mobile devices have heterogeneous input-data arrival time and computation deadlines. However, the inference among large amounts of mobile devices proposes a great challenge in these models.

We conclude that tiered clouds or hierarchy clouds, i.e., clouds at multiple distances (local or remote) can improve the performance and scalability of mobile applications. Thus, the combination of different computing paradigms, e.g., mobile edge computing and mobile cloud computing, can overcome the drawbacks of these computing paradigms and further improve QoS for mobile users. Moreover, the combination of computing paradigms and different network architectures (e.g., SDN) or different mobile devices (e.g., energy-harvesting devices) are drawing attention to

many researchers. Finally, different kinds of powerful mathematical tools (e.g., game theory, auction theory, optimization method and queueing model) are still widely used by many researchers.

# Chapitre 3

---

# Markov Decision Process Based Mobile Data Offloading

## 3.1. Abstract

Cellular network is facing severe traffic overload problem caused by phenomenal growth of mobile data. Offloading part of the mobile data traffic from cellular network to alternative networks is a promising solution. In this paper, we study mobile data offloading problem under the architecture of mobile cloud computing (MCC), where mobile data can be delivered by WiFi network and device-to-device (D2D) communication. In order to minimize the overall cost for data delivery task, it is crucial to reduce cellular network usage while satisfying delay requirements. In our proposed model, we formulate the data offloading task as a finite horizon Markov Decision Process. We first propose a hybrid offloading algorithm for mobile data with different delay requirements. Moreover, we establish the sufficient conditions for the existence of threshold policy. Then, we propose a monotone offloading algorithm based on threshold policy in order to reduce the computational complexity. The simulation results show that the proposed offloading approach can achieve minimal communication cost compared with other three offloading schemes.

**Keywords:** Mobile data offloading, device-to-device communication, mobile cloud computing, Markov decision process.

**Status:** This journal paper is published. Liu, Dongqing, Lyes Khoukhi, and Abdelhakim Hafid. Prediction-Based Mobile Data Offloading in Mobile Cloud Computing. IEEE Transactions on Wireless Communications 17.7 (2018): 4660-4673.

## 3.2. Introcution

With the increase of the number of smart mobile devices and data heavy mobile applications, such as video streaming and cloud backup, global mobile data traffic has been growing dramatically in recent years. The global mobile traffic grew 74% in 2015, while mobile network (cellular) connection speeds only grew 20% [1]. The growing speed of mobile traffic will push the current cellular network to the limit. The Quality of Experience (QoE) of mobile services will not be guaranteed without the high-speed and stable network connections. However, it is impractical to keep extending the current cellular network infrastructure to improve QoE, given the corresponding expensive investment. In order to cope with this problem, mobile data offloading technology can be an alternative solution. Mobile data offloading can opportunistically use alternative networks (e.g., WiFi network and D2D communnication) to reduce the network congestion.

Compared with data offloading, applications involving computation offloading usually are more delay-sensitive. This is because computation offloading includes two data delivery processes, i.e., uploading computation data and downloading computation results. In many cases, data offloading can improve these two processes by using alternative networks with higher data rates than cellular network, e.g., WiFi network. Thus, data offloading can be used in MCC to improve the performance of computation offloading.

WiFi offloading is considered as a promising solution to reduce mobile data traffic in cellular network. WiFi Access Points (APs) can efficiently reduce cellular traffic [47, 75]. It is shown that about 65% of cellular traffic can be offloaded through WiFi APs[4]. Although WiFi APs can provide better data rate than cellular network, their coverage area is much smaller than cellular network [5, 40].

Another mobile offloading method, called opportunistic offloading, is based on D2D communication [6]. Opportunistic offloading uses the store-carry-forward strategy, where some mobile users can store data in the buffer (called mobile helpers, MHs), carry the data when they are moving, and forward the data to other mobile users (called mobile subscribers, MSs) [7, 76]. When mobile network operator (MNO) wants to deliver data to MSs, it can first send the data to MHs. Then, MHs will transmit data to MSs using opportunistic connections. With more than half a billion mobile devices and connections added in 2015 [1], D2D communication is becoming an important data delivery scheme. However, the data rate of D2D communication is low and the mobility patterns of MHs or MSs are difficult to predict.

In this paper, we propose two mobile data offloading schemes based on Finite Horizon Markov Decision Process. Our objective is to minimize the communication cost for delivering mobile data with different delay sensitivities through multiple wireless networks, i.e., cellular network, WiFi network and D2D communication. The main contributions of this paper can be summarized as follows:

- We propose a hybrid offloading model, where multiple wireless networks are used to transfer mobile data. MNO can minimize the total communication cost by selecting different networks.
- We formulate the data offloading problem in hybrid wireless networks as an FHMDP model, and propose an offloading algorithm that can support different delay requirements (i.e., loose and tight delay tolerant).

- We prove that there exit threshold structures in the optimal policy and propose a monotone offloading algorithm for generating monotone policy with lower computational complexity.
- The simulation results demonstrate that our proposed schemes achieve the lowest communication cost as compared with three offloading schemes.

The rest of this paper is organized as follows. Section 3.3 describes the system model. Section 3.4 formulates the mobile data offloading problem as an FHMDP model. Section 3.5 proposes a hybrid offloading algorithm. Section 3.6 establishes the sufficient conditions for the existence of threshold policy and proposes a monotone offloading algorithm based on threshold policy. Section 3.7 evaluates the performance of the proposed offloading algorithms. Finally, Section 3.8 concludes the paper.

## 3.3. System Model

In this section, we present our system model to enhance data offloading in mobile cloud. In our model, we consider that mobile devices can access cloud services through multiple wireless networks, as shown in Fig. 3.1: (1) WiFi network. WiFi APs provide opportunistic WiFi communication (e.g., WLAN) for MS within its working coverage, and connect to distant cloud infrastructure through wired network; (2) Cellular network. Cellular base stations provide seamless cellular communication (e.g., 4G) for MS, and connect to cloud through wired network; (3) D2D network. MHs (e.g., MH A and MH B in Fig. 3.1) provide opportunistic D2D communication for MS, and connect to nearby WiFi APs or Cellular BSs through WiFi or cellular communication.

In our data offloading system, mobile helpers are chosen to work as data providers for mobile subscribers. Incentives for MHs to participate in data offloading can be provided by using some micro-payment scheme, or MNO can offer participants a reduced cost for the service. In this paper, we choose the micro-payment scheme [40,

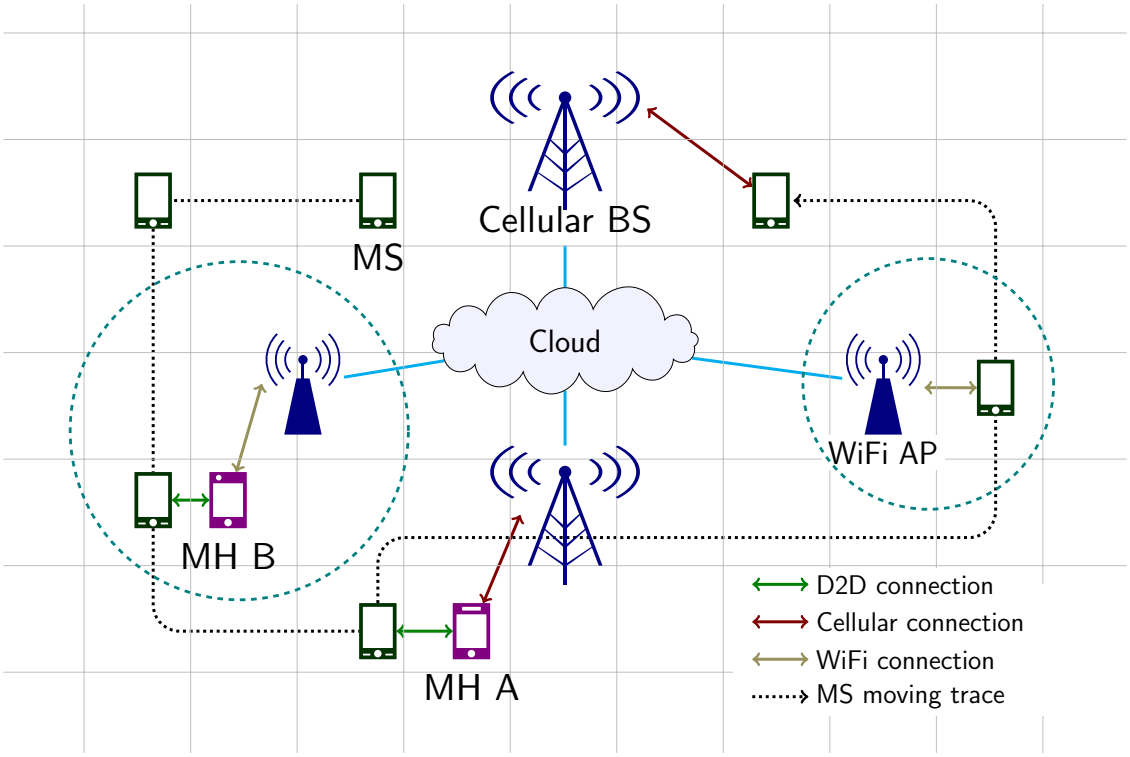



**Fig. 3.1.** The system model of mobile data offloading.

76], where MHs can get rewards by participating in data offloading. The price for transmitting a data unit (i.e., $\chi_4$) is set by MNO. MNO first announces the price to mobile users and then chooses MHs from those users who accept the price and are willing to participate in data offloading. It is worth noting that a full analysis of such process is not the focus of this paper.

The mobile data being received from cloud to MS is divided into a sequence of data units. The data units are predetermined by MNO. Delivering data means transmitting data of size $K$ to MS before deadline $D$. $K$ is the number of total data units and $D$ is the maximum available time for data transmission. The data delivery is completed when non-transmitted data size $k$ (i.e., $k$ is the size of data that has

not been received by MS) is zero before $D$. Conventionally, without WiFi APs and MHs, MS receives all mobile data through cellular communication. However, in our model, MS has an option to receive parts of the data through nearby WiFi or D2D communication, which may offer higher data rates and lower communication cost.

Upon arrival of a data delivery request from MS, MNO decides whether to transmit data by cellular network or offload it to WiFi and D2D networks according to data characteristics and network performance. The possibility of offloading depends on the delay characteristic (i.e. delay tolerant or not) of mobile data. If data is delay tolerant, MNO can defer data transmission to increase the possibility of offloading. Otherwise (i.e., data is delay sensitive), MNO will have less opportunities to offload mobile data from cellular network. Moreover, if data rates of WiFi and D2D networks are higher than that of cellular network, MNO can shorten the delivery time by cellular data offloading.

The main idea of data offloading is to use delay tolerance of mobile data and mobility of mobile users to seek opportunities to use WiFi and D2D networks.

We assume that a time slot $T$ is long enough for MS to receive at least one data unit from cellular, WiFi or D2D network. An offloading decision (i.e., selecting a network) is made at the beginning of each time slot. The time when offloading decision is determined is denoted by $d$; it is called decision epoch. Thus, a network is selected at each decision epoch and will be the working network during the time slot.

At each decision epoch, MNO observes the current system states, i.e., the location of MS, the non-transmitted data size and the locations of available MHs. Based on the observed system state, MNO computes the communication cost for available networks. Then, MNO makes an offloading decision of either transmitting data using cellular network or offloading data to other network (i.e., WiFi or D2D network).

In this paper, we propose a Finite Horizon Markov Decision Process to formulate this problem, with the aim to minimize communication costs and satisfy delay constraints by offloading mobile data as much as possible with WiFi network and D2D communication. Markov decision process is a useful model for sequential decision making, where MNO needs to take a sequence of actions (wireless network selection). FHMDP is a Markov decision process with a finite number of decision epochs [77]. Since every data delivery task should be finished before a given deadline, FHMDP will plan data offloading decisions at each decision epoch. FHMDP planning phase can be implemented in remote cloud and ease the heavy burden of complex data offloading management by MS.

It is worth noting that the locations of WiFi APs and the base station are stationary, while MHs are moving around in the coverage area of base station. MHs can be considered as supplementary to WiFi APs because of their mobility.

## 3.4. Problem Formulation

In this section, we formulate the mobile data offloading problem as an FHMDP problem. Table 3.1 shows the notations used in the rest of this paper. In our model, mobile data is initially delivered to one or more MSs through cellular and WiFi networks. Additionally, any MH who carries a copy of the data can opportunistically transmit it to MSs using D2D communication. For each MS, data of size $K$ needs to be transmitted before deadline $D$. MNO will select a wireless network for MS, at each decision epoch $d \in \mathcal{D} = \{1, \cdots, D\}$, based on the system state at that time.

### 3.4.1. System State and Action Space

The system state for multiple MSs and multiple MHs is defined as $\boldsymbol{s} = (\mathcal{M}, \mathcal{H})$, where $\mathcal{M}$ and $\mathcal{H}$ are the sets of states for MSs and MHs, respectively. More specifically, $\mathcal{M} = \{\boldsymbol{m}_i, i \in \{1, \cdots, M\}\}$ includes all the states of MSs, where

**Tab. 3.1.** Notations for Mobile Data Offloading

| | |
|---|---|
| $K$ | Total size of mobile data to be transmitted. |
| $D$ | Total length of time for data transmission. |
| $k$ | Size of mobile data that is not transmitted. |
| $d$ | Decision epoch: time for making offloading decision. |
| $\mathcal{A}$ | Set of transmission actions. |
| $\mathcal{U}$ | Set of mobile user types. |
| $a$ | Transmission action. |
| $u$ | Mobile user type. |
| $L$ | Total number of grids. |
| $T$ | Length of one time slot. |
| $\mu$ | Stable factor: probability of MS staying at the same location in two sequential decision epochs. |
| $\nu_a^l$ | Data rate for action $a$ at grid $l$. |
| $\chi_a$ | Unit price for transmitting data by action $a$. |

$\boldsymbol{m}_i = (l_i, u_i, k_i)$ denotes the possible state of MS $i$; $l_i$ denotes the location of MS $i$, $u_i$ denotes the user type, and $k_i$ is the size of data to be transmitted. $\mathcal{H} = \{l_j, j \in \{1, \cdots, N\}\}$ is a set that includes all the locations of MHs, where $l_j$ is the location of MH $j$.

In the following, we omit the subscripts $i$ and $j$ of state parameters $l$, $u$ and $k$ for simplification. The state parameter $l \in \mathcal{L} = \{1, \cdots, L\}$ denotes the index of grid (or location), where $L$ is the number of possible grids that MSs may reach before $D$. We assume that cellular network can provide seamless coverage to all grids. All grids are classified into four disjoint categories at decision epoch $d$ based on available WiFi or D2D connections. $\mathcal{L}_d^1$ denotes the grids covered by only cellular network,

$\mathcal{L}_d^2$ contains the grids covered by both cellular and WiFi networks, $\mathcal{L}_d^3$ represents the grids covered by both cellular network and D2D communication, and $\mathcal{L}_d^4$ denotes the grids covered by cellular, WiFi networks and D2D communication. Due to the mobility of MHs, $\mathcal{L}_d^2$, $\mathcal{L}_d^3$ and $\mathcal{L}_d^4$ change over decision epoch $d$.

The state parameter $u \in \mathcal{U} = \{1,2,\cdots,U\}$ represents the mobile user type (e.g. loose delay or tight delay), where $U$ represents the number of different user types. We consider that different user types have different delay requirements resulting in different deadlines. To simplify the model, we consider two sets of user types, each of which has different QoS requirements. More specifically, user types that are delay-sensitive are in set $\mathcal{U}^1$; the other types (e.g. software update) are in set $\mathcal{U}^0$. Thus, $\mathcal{U} = \mathcal{U}^0 \cup \mathcal{U}^1$.

We divide the data, to be transmitted, into $K$ equal portions; the state parameter $k \in \mathcal{K} = \{0,1,\cdots,K\}$ represents the number of data portions still to be transmitted. If $k = 0$ when $d \leq D$, the data delivery process is completed.

After defining the system state of FHMDP, we next introduce the action space of MNO in mobile data offloading system. At each decision epoch, MNO selects one of the offloading actions for data transmission. There are four actions in the action space corresponding to four offloading decisions. Formally, action $a \in \mathcal{A} = \{1,2,3,4\}$: (1) $a = 1$ (waiting action): MS will wait for a chance to receive data from WiFi or D2D network; (2) $a = 2$ (cellular action); (3) $a = 3$ (WiFi action); and (4) $a = 4$ (D2D action): MS can receive data from cellular network,WiFi network and D2D connection, respectively.

We observe that WiFi action is available when MS is in WiFi coverage and D2D action is available when MS can access a nearby MH. Thus, the available actions depend on the state parameter $l$. We also notice that the mobile user type $u$ impacts the available actions, i.e., D2D action is not available for delay sensitive data due to the low data rate. $\mathcal{A}(l,u) \subseteq \mathcal{A}$ representing the set of available actions at grid $l$ for

data with type $u$, is defined as follows:

$$\mathcal{A}(l,u) = \begin{cases} \{1,2\}, & l \in \mathcal{L}_d^1,\ u \in \mathcal{U}, \\ \{1,2,3\}, & l \in \mathcal{L}_d^2,\ u \in \mathcal{U}, \\ \{1,2,4\}, & l \in \mathcal{L}_d^3,\ u \in \mathcal{U}^0, \\ \{1,2,3,4\}, & l \in \mathcal{L}_d^4,\ u \in \mathcal{U}^0. \end{cases} \tag{3.4.1}$$

### 3.4.2. Transition Cost and Transition Probabilities

The transition cost for MS $i$, $c_d(\boldsymbol{m}_i,a)$, is independent from current state $\boldsymbol{m}_i$ and decision epoch $d$. It is equal to the action cost function $cost(a)$, which is defined as follows:

$$c_d(\boldsymbol{m}_i,a_i) = cost(a_i) = \nu_{a_i}^{l_i} \cdot T \cdot \chi_{a_i}, \tag{3.4.2}$$

where $\nu_a^l$ is the network data rate at grid $l$ with action $a$, $T$ is the period of time between two consecutive decision epochs, and $\chi_a$ is the cost to transmit a data unit by action $a$, i.e., $\chi_2$, $\chi_3$ and $\chi_4$ are incurred by the usage of cellular, WiFi and D2D actions, respectively. The benefit of mobile data offloading is based on the fact that $\chi_3 < \chi_2$ and $\chi_4 < \chi_2$. This means that the cost to send data using cellular network is higher than that of using WiFi network and D2D communication. The total cost of transmitting data of size $K$ is the sum of costs incurred at each period during the total transmission process.

There may be some data transmission tasks that cannot be completed before the deadline. For failed data transmissions (i.e. $k > 0$ when $d > D$), the penalty cost function is defined as follows:

$$c_{D+1}(\boldsymbol{m}_i) = penalty(u,k) = k^{(u+1)}, \tag{3.4.3}$$

where $u$ and $k$ finish the state parameters of $\boldsymbol{m}_i$. $k^{(u+1)}$ is an increasing function of $k$ and $u$ and reflects the fact that a larger remaining data size $k$ and a tighter delay sensitivity $u$ lead to a larger penalty.

In the following, we derive the transition probability between system states, which is the probability that current state $\boldsymbol{s}$ changes into $\boldsymbol{s}'$ in the next decision epoch by taking action $\boldsymbol{a}$. $\boldsymbol{a} = \{a_1, \cdots, a_M\}$ is the set of actions for all MSs. Since each MS or MH changes its state independently, we obtain the following state transition function.

$$\mathbb{P}(\boldsymbol{s}'|\boldsymbol{s},\boldsymbol{a}) = \prod_{i\in\mathcal{M}} \mathbb{P}(\boldsymbol{m}_i'|\boldsymbol{m}_i, a_i) \cdot \prod_{j\in\mathcal{N}} \mathbb{P}(l_j'|l_j), \tag{3.4.4}$$

where

$$\begin{aligned} \mathbb{P}(\boldsymbol{m}_i'|\boldsymbol{m}_i, a_i) &= \mathbb{P}(l_i', u_i', k_i'|l_i, u_i, k_i, a_i) \\ &= \mathbb{P}(l_i'|l_i) \cdot \mathbb{P}(k_i'|l_i, u_i, k_i, a_i). \end{aligned} \tag{3.4.5}$$

For MS $i$, the next grid $l_i'$ depends only on the current grid $l_i$ and the user type $u_i$ does not change during the offloading process, i.e., $u_i' = u_i$. The remaining data size $k_i'$ depends on current location $l_i$, user type $u_i$, data size $k_i$, and selected action $a_i$.

$\mathbb{P}(l_i'|l_i)$ is the probability that MS will move from grid $l_i$ to grid $l_i'$. We consider a two dimensions memoryless mobility pattern of MS as follows:

$$\mathbb{P}(l_i'|l_i) = \begin{cases} \mu, & \textit{if } l_i' = l_i, \\ \rho_j, & \textit{otherwise}, \end{cases} \tag{3.4.6}$$

where $\mu$ is the stable factor that denotes the probability that MS $i$ stays at the same grid in two sequential decision epochs. Alternatively, MS can move randomly to an adjacent location with probability $\rho_j, j \in \{1,2,3,4\}$, where $j$ represents one of four possible moving directions (i.e., north, south, east and west). $\mu$ and $\rho_j$ satisfy the relation $\mu + \sum_{j\in\{1,2,3,4\}} \rho_j = 1$.

$\mathbb{P}(k_i'|l_i, u_i, k_i, a_i)$ is the probability describing the change of remaining data size $k_i$ and is defined as follows:

$$\mathbb{P}(k_i'|l_i, u_i, k_i, a_i) = \begin{cases} 1, & if\, k_i' = \max\{k_i - \nu_{a_i}^{l_i}, 0\} \\ & \quad and\ a_i \in \mathcal{A}(l_i,u_i), \\ 0, & otherwise, \end{cases} \tag{3.4.7}$$

where the availability of $a_i$ depends on grid $l_i$ and user type $u_i$. Fig. 3.2 illustrates MS state transition graph considering parameters $l$ and $k$; the terminal states are those with $k = 0$.

## 3.5. Hybrid Offloading Algorithm

In this section, we propose an algorithm, called hybrid offloading algorithm, to compute the optimal offloading policy. A policy in FHMDP is denoted by $\boldsymbol{\pi} = \{\pi_i\}_{i\in\mathcal{M}}$, where $\pi_i : \boldsymbol{M}_i \times \mathcal{D} \to \mathcal{A}$ is the policy for MS $i$, which can decide an action based on $\boldsymbol{m}_i$ and the decision epoch $d$. The feasible domain for $\boldsymbol{\pi}$ is denoted by $\boldsymbol{\Pi}$. The objective function is defined as follows:

$$\min_{\boldsymbol{\pi}\in\boldsymbol{\Pi}} \sum_{i\in\mathcal{M}} E^{\pi_i}_{\boldsymbol{m}_{i,1}} \left[ \sum_{d=1}^{D} c_d\big(\boldsymbol{m}^{\pi_i}_{i,d}, \pi_i(\boldsymbol{m}^{\pi_i}_{i,d}, d)\big) + c_{D+1}(\boldsymbol{m}^{\pi_i}_{i,D+1}) \right]. \tag{3.5.1}$$

The objective function aims to minimize the expected cost to deliver data of size $K$ for all MSs. Before presenting our offloading algorithm, we define the value function as follows.

$$V_d^*(\boldsymbol{m}_i) = \min_{a_i\in\mathcal{A}(l_i,u_i)} Q_d(\boldsymbol{m}_i, a_i), \tag{3.5.2}$$

where $Q_d(\boldsymbol{m}_i,a_i)$

$$
\begin{aligned}
&= \sum_{\boldsymbol{m}'_i \in \boldsymbol{M}_i} \mathbb{P}(\boldsymbol{m}'_i|\boldsymbol{m}_i,a_i) \cdot \big(c_d(\boldsymbol{m}_i,a_i) + V^*_{d+1}(\boldsymbol{m}'_i)\big) \\
&= c_d(\boldsymbol{m}_i,a_i) + \sum_{\boldsymbol{m}'_i \in \boldsymbol{M}} \mathbb{P}(\boldsymbol{m}'_i|\boldsymbol{m}_i,a_i) \cdot V^*_{d+1}(\boldsymbol{m}'_i) \\
&= \nu_{a_i}^{l_i} \cdot T \cdot \chi_{a_i} + \\
&\quad \sum_{l'_i \in \mathcal{L}} \sum_{k'_i \in \mathcal{K}} \mathbb{P}(l'_i|l_i) \cdot \mathbb{P}(k'_i|l_i,u_i,k_i,a_i) \cdot V^*_{d+1}(l'_i,u_i,k'_i) \\
&= \nu_{a_i}^{l_i} \cdot T \cdot \chi_{a_i} + \sum_{l'_i \in \mathcal{L}} \mathbb{P}(l'_i|l_i) \cdot V^*_{d+1}(l'_i,u_i,(k_i - \nu_{a_i}^{l_i} T)).
\end{aligned}
\tag{3.5.3}
$$

The value function $V^*_d(\boldsymbol{m}_i)$ denotes the minimal expected cost for MS $i$ in state $\boldsymbol{m}_i$ in decision epoch $d$ to finish the data delivery process. $Q_d(\boldsymbol{m}_i,a_i)$ is a one step forward function that calculates the minimal expected cost if MS $i$ selects action $a_i$; it is the sum of current cost $c_d(\boldsymbol{m}_i,a_i)$ and expected future cost. Eq (3.5.3) is derived from Eqs. (3.4.2), (3.4.5) and (3.4.7). The optimal policy is defined as

$$
\pi^*_d(\boldsymbol{m}_i) = \underset{a_i \in \mathcal{A}(l_i,u_i)}{\operatorname{argmin}} \; Q_d(\boldsymbol{m}_i,a_i). \tag{3.5.4}
$$

Due to the mobility of MHs, we are interested in the expected number of MHs in grid $l$ at decision epoch $d$, denoted by $\mathbb{N}(d,l)$.

$$
\mathbb{N}(d,l) = \begin{cases} \sum_{j \in \mathcal{N}} \delta(l_j^1, l), & \text{if } d = 1, \\ \sum_{l' \in \mathcal{L}} \mathbb{P}(l|l') \cdot \mathbb{N}(d-1,l'), & \text{if } d = 2, \cdots, D, \end{cases} \tag{3.5.5}
$$

where $l_j^1$ is the initial location of MH $j$. The function $\delta(l_j^1, l)$ returns 1, if $l_j^1 = l$; otherwise, it returns 0.

Our hybrid offloading algorithm, illustrated in Algorithm 1, consists of three phases: initialization phase (steps 1-3), planning phase (steps 4-12) and offloading phase (steps 13-22). In the initialization phase, we calculate the expected number

**Algorithm 1** Hybrid Offloading Algorithm

**for** $d \leftarrow \{1, \cdots, D\}$ and $l \in \mathcal{L}$ **do**
    Compute $\mathbb{N}(d,l)$ using Eq. (3.5.5)
**end for**
**for** $l \in \mathcal{L}$, $u \in \mathcal{U}$ and $k \in \mathcal{K}$ **do**
    $V^*_{D+1}(l,u,k) \leftarrow c_{D+1}(l,u,k)$
**end for**
**for** $u \in \mathcal{U}$ **do**
    **for** $d \leftarrow \{D, \cdots, 1\}$, $k \leftarrow \{0, \cdots, K\}$ and $l \in \mathcal{L}$ **do**
        Compute $V^*_d(l,u,k)$ using Eq. (3.5.2)
        Compute $\pi^*_u(l,u,k,d)$ using Eq. (3.5.4)
    **end for**
**end for**
**for** $i \in \mathcal{M}$ **do**
    Set $d \leftarrow 1$ and $k \leftarrow K$
    **while** $d < D + 1$ and $k > 0$ **do**
        Get current location $l^d_i$ of MS
        Get current idle $\mathbb{N}(d,l^d_i)$
        Set $a \leftarrow \pi^*_{u_i}(l^d_i,u_i,k,d)$
        **if** $k - (D - d) \cdot \bar{\nu}_2 \cdot \kappa(u) > 0$ **then**
            Set $a \leftarrow \text{argmax}_{a \in \mathcal{A}(l^d_i,u_i)} \nu^l_a$
        **end if**
        Set $k \leftarrow k - \nu^l_a \cdot T$
        Set $d \leftarrow d + 1$
    **end while**
**end for**

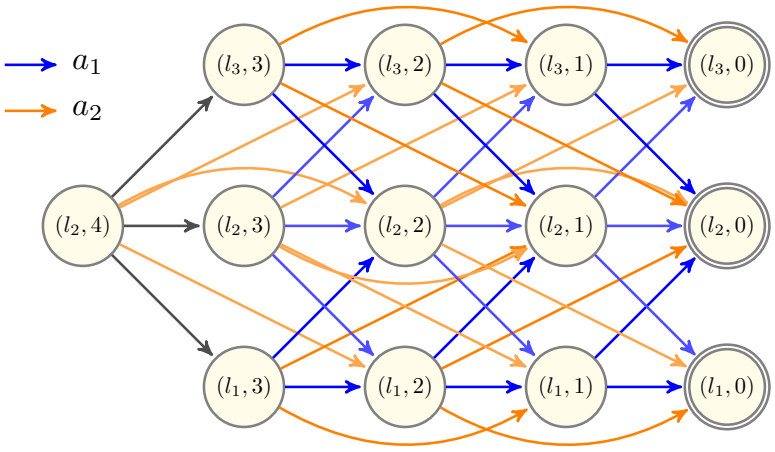



**Fig. 3.2.** A sample reduced state transition for MS, where the first component is the location of MS $l$, and the second component is data size $k$. Here, $\mathcal{L} = \{l_1, l_2, l_3\}$ and $K = 4$. Action $a_1$ can transfer 1 data unit each time, while $a_2$ can transfer 2 data units each time. The state with double circle is the terminal state, i.e., state $(l_i,0)$, $i \in \mathcal{L}$.

of MHs in different locations and decision epochs using Eq. (3.5.5). $\mathbb{N}(d,l)$ is used to indicate the availability of D2D action in planning phase. We consider that MSs with the same user type have the same offloading policy. Thus, we generate the optimal policy based on the user type. Since the state transition graph, illustrated in Fig. 3.2, is an acyclic graph, we use the backward induction method to obtain the optimal offloading policy. In the planning phase, we first calculate the value function in $D+1$ (steps 4-6). Then, we calculate the value function $V_d^*(l,u,k)$ and optimal policy $\pi_u^*(l,u,k,d)$ from decision epoch $D$ to 1 (steps 7-12).

In the offloading phase, MNO determines the offloading action at each decision epoch. Step 16 gets the location of MS $i$ at decision epoch $d$, denoted by $l_i^d$. Step 17 obtains the idle MHs (MHs that are not serving other MSs) in proximity to MS $i$. If

$\mathbb{N}(d,l_i^d) \geq 1$, then D2D action is available. Step 18 sets $a$ to the action provided by optimal policy. In order to counteract the prediction error caused by the mobility of MSs and MHs, steps 19-21 first check whether data of size $k$ can be transmitted using cellular network before deadline. If the response is yes, MNO will take action according to the optimal policy. Otherwise, the network with highest data rate will be selected (step 20). $\bar{\nu}_2$ is the average data rate of cellular network. Notice that the function $\kappa(u)$ (step 19) is used to control the delay sensitivity of different user types. Generally, user type accepting D2D communication results in larger prediction error, leading to a higher value of $\kappa(u)$. We set $\kappa(u)$ to 1.2 and 1 for $u \in \mathcal{U}^0$ and $u \in \mathcal{U}^1$, respectively [78]. However, these two values can be adjusted according to different situations. The main computational complexity of Algorithm 1 is associated with the planning phase, that is $\mathcal{O}(UDLK)$.

## 3.6. Monotone Policy and Offloading Algorithm

Since the state space becomes extremely large with the increase of deadline $D$, data size $K$ and the number of grids $L$, Algorithm 1 will take more time and resources to solve the problem. In order to reduce the computational complexity for generating the optimal policy, we provide sufficient conditions under which the offloading policy is monotone (non-decreasing or non-increasing) in terms of data size $k$ and decision epoch $d$, called monotone policy. The monotone policy enables efficient computation due to the existence of threshold structure in optimal policy. Threshold structure has several boundaries between different offloading decisions according to $k$ and $d$, as discussed in Section 3.7.1. Thus, instead of generating the optimal actions for all the system state, we only need to determine the threshold states, which can greatly reduce the computational complexity.

The rest of this section is organized as follows. Subsection 3.6.1 presents our assumptions. Subsection 3.6.2 discusses the properties of optimal policy. Subsection

3.6.3 shows the special case where the monotone policy degrades into a single action $a^*$ that does not change with $k$ and $d$. Subsection 3.6.4 shows the general case where the monotone policy has threshold structures. Subsection 3.6.5 presents an algorithm for generating and executing monotone policy, called monotone offloading algorithm.

### 3.6.1. Assumptions

We make the following assumptions for deriving the monotone policy.

**Assumption 1.** *The unit costs of cellular, WiFi and D2D networks ($\chi_2$, $\chi_3$ and $\chi_4$) satisfy the relation $\chi_3 < \chi_4 < \chi_2$.*

Note that the benefit of cellular data offloading is based on the fact that $\chi_2 > \chi_3$ and $\chi_2 > \chi_4$ [79, 80]. We further assume that $\chi_3 < \chi_4$, since free WiFi can often be found in places such as homes, offices, or coffee shops [43, 47].

**Assumption 2.** *The data rates of cellular, WiFi and D2D networks ($\nu_2$, $\nu_3$ and $\nu_4$) are location independent [41].*

We underline that our model can be extended to location dependent rates by considering the same action with different data rates as different actions. For example, the data rate of action $a$ in grid $l$, denoted by $\nu_a^l$, is location dependent. We replace action $a$ with actions $(a_1, \cdots, a_W)$, each of which represents the WiFi action with a different data rate. Thus, the data rate of WiFi action becomes location independent, denoted by $\nu_{a_i}, i \in \{1, \cdots, W\}$, where $W$ denotes the number of different data rates that can be used in the case of WiFi action.

### 3.6.2. Properties of the optimal policy

We discuss some properties of the optimal policy under above assumptions.

**Lemma 1.** *The penalty function $c_{D+1}(l,u,k)$ satisfies the following relation:*

$$c_{D+1}(l,u,k) - c_{D+1}(l,u,(k - \nu_a T)) \geq c_d(l,u,k,a), \tag{3.6.1}$$

*for all* $d \in \mathcal{D}$, $l \in \mathcal{L}$, $u \in \mathcal{U}$, $k \geq \nu_a T$ *and* $a \in \mathcal{A}$.

PROOF. Note that the penalty cost (defined in Eq. (3.4.3)) is greater than the action cost (defined in Eq. (3.4.2)) for the same data size $\nu_a T$, as shown in *Assumption 1.*

Moreover, given $u$, the penalty cost is a power function with respect to data size $k$ (see Eq. (3.4.3)). Thus, we obtain that, $\forall k \geq \nu_a T$,

$$c_{D+1}(l,u,k) - c_{D+1}(l,u,(k-\nu_a T)) \geq c_{D+1}(l,u,\nu_a T) - c_{D+1}(l,u,0) = c_{D+1}(l,u,\nu_a T) \tag{3.6.2}$$

From Eq. (3.6.2) and the definition of action cost, we get Eq. (3.6.1). □

**Lemma 2.** *The value function* $V_d^*(l,u,k)$ *is non-decreasing in the remaining data size* $k$, $\forall l \in \mathcal{L}, u \in \mathcal{U}, d \in \mathcal{D}$.

PROOF. We prove the result using backward induction. Since the penalty function $c_{D+1}(l,u,k)$ is non-decreasing in $k$, the value function $V_{D+1}^*(l,u,k) = c_{D+1}(l,u,k)$ is non-decreasing in $k$. Assume that $V_{d'}^*(l,u,k)$ is non-decreasing in $k$ for $d' \in \{d+1, \cdots, D\}$. Based on Eq (3.5.3), we get

$$V_d^*(l,u,k) = Q_d(l,u,k,a^*) = \nu_{a^*} T \chi_{a^*} + \sum_{l' \in \mathcal{L}} \mathbb{P}(l'|l) \cdot V_{d+1}^*(l',u,(k-\nu_{a^*}T)). \tag{3.6.3}$$

By induction hypothesis, $V_{d+1}^*(l',u,(k-\nu_{a^*}T)$ is non-decreasing in $k$. Thus, $V_d^*(l,u,k)$ is non-decreasing in $k$. □

**Lemma 3.** *The value function* $V_d^*(l,u,k)$ *is non-decreasing in decision epoch* $d$, $\forall l \in \mathcal{L}, u \in \mathcal{U}, k \in \mathcal{K}$.

PROOF. We first show that $V_{D+1}^*(l,u,k) \geq V_D^*(l,u,k)$. From Eq (3.5.2), we obtain

$$\begin{aligned} V_D^*(l,u,k) &= c_D(l,u,k,a^*) + c_{D+1}(l',u,(k-\nu_{a^*}T)) \\ &\leq c_{D+1}(l,u,k) = V_{D+1}^*(l,u,k), \end{aligned} \tag{3.6.4}$$

where the inequality is based on *Assumption 1*. Next, we assume that $V_{d'}(l,u,k)$ is non-decreasing for $d' \in \{d+1, \cdots, D\}$. By induction hypothesis, $V^*_{d+1}(l',u,(k-\nu_{a^*}T)$ is non-decreasing in Eq. (3.6.3). Thus, $V^*_d(l,u,k)$ is non-decreasing in $d$. □

*Lemma 2* reflects the fact that the expected cost is higher when the non-transmitted data size $k$ is larger. *Lemma 3* shows that a larger decision epoch $d$ (i.e. the deadline is closer) results in higher expected cost.

### 3.6.3. Single action monotone policy

In this subsection, we show the special case where the monotone policy degrades into a dominant action policy, i.e., $\pi^*_d(l,u,k) = a^*_{(l,u)}$.

**Definition 1.** *Given $l \in \mathcal{L}$ and $u \in \mathcal{U}$, $a^*_{(l,u)} \in \mathcal{A}(l,u)$ is a dominant action if*

$$\pi^*_d(l,u,k) = \underset{a_{(l,u)} \in \mathcal{A}(l,u)}{\operatorname{argmin}} Q_d(l,u,k,a) = a^*_{(l,u)}, \tag{3.6.5}$$

*for all $d \in \mathcal{D}$ and $k \in \mathcal{K}^+$,*

$\mathcal{K}^+ = \mathcal{K} \setminus \{0\}$; $k = 0$ indicates that the data transmission process is finished and thus no action is chosen. Notice that $a^*_{(l,u)}$ is different from $a^*$. $a^*_{(l,u)}$ is the optimal action for all $d \in \mathcal{D}$ and $k \in \mathcal{K}^+$, while $a^*$ is the optimal action for some $k$ and $d$. Next, we establish conditions under which a dominant action exists.

**Theorem 1.** *Given $u \in \mathcal{U}$ and $l \in \mathcal{L}^2_d \cup \mathcal{L}^4_d$ where WiFi action is available, if $\nu_2 < \nu_3$ and $\nu_4 < \nu_3$, then $a^*_{(l,u)} = 3\,(WiFi)$ is a dominant action, for all $d \in \mathcal{D}$ and $k \in \mathcal{K}^+$. The optimal policy is*

$$\pi^*_d(l,u,k) = a^*_{(l,u)} = 3\,(WiFi). \tag{3.6.6}$$

Proof. We first prove that $Q_d(l,u,k,3) \leq Q_d(l,u,k,2)$. From Eq. (3.6.3), we get

$$Q_d(l,u,k,3) = \nu_3 T \chi_3 + \sum_{l' \in \mathcal{L}} P(l'|l) \cdot V^*_{d+1}(l',u,(k-\nu_3 T)), \tag{3.6.7}$$

$$Q_d(l,u,k,2) = \nu_2 T\chi_2 + \sum_{l' \in \mathcal{L}} P(l'|l) \cdot V^*_{d+1}(l',u,(k-\nu_2 T)). \tag{3.6.8}$$

Subtracting Eq. (3.6.7) from Eq. (3.6.8), we get Eq. (3.6.9a). Since $\nu_2 < \nu_3$, let $\nu_3 = \nu_2 + \delta$ and $\delta > 0$. Replacing $\nu_3$ with $\nu_2 + \delta$ in Eq. (3.6.9a), we get Eq. (3.6.9b).

$$\begin{aligned}
&Q_d(l,u,k,2) - Q_d(l,u,k,3) \\
&= \nu_2 T\chi_2 - \nu_3 T\chi_3 + \sum_{l' \in \mathcal{L}} P(l'|l) \cdot \Big( V^*_{d+1}\big(l',u,(k-\nu_2 T)\big) - V^*_{d+1}\big(l',u,(k-\nu_3 T)\big) \Big) && \text{(3.6.9a)} \\
&= \nu_2 T(\chi_2 - \chi_3) - \delta T\chi_3 + \sum_{l' \in \mathcal{L}} P(l'|l) \cdot \Big( V^*_{d+1}\big(l',u,(k-\nu_2 T)\big) - V^*_{d+1}\big(l',u,(k-\nu_2 T-\delta T)\big) \Big) && \text{(3.6.9b)} \\
&> \sum_{l' \in \mathcal{L}} P(l'|l) \cdot \Big( V^*_{d+1}\big(l',u,(k-\nu_2 T)\big) - V^*_{d+1}\big(l',u,(k-\nu_2 T-\delta T)\big) - \delta T\chi_3 \Big) && \text{(3.6.9c)} \\
&= \sum_{l' \in \mathcal{L}} P(l'|l) \cdot \Big( \sum_{a \in \mathcal{A}} p_a \delta T\chi_a - \delta T\chi_3 \Big) && \text{(3.6.9d)} \\
&\geq \sum_{a \in \mathcal{A}} p_a \delta T\chi_3 - \delta T\chi_3 && \text{(3.6.9e)} \\
&= 0. && \text{(3.6.9f)}
\end{aligned}$$

Note that: (1) Based on *Assumption 1*, where $\chi_2 > \chi_3$, we get $\nu_2 T(\chi_2 - \chi_3) > 0$ in Eq. (3.6.9b). By eliminating $\nu_2 T(\chi_2 - \chi_3) > 0$, we get Eq. (3.6.9c); (2) Eq. (3.6.9d) is obtained based on *Lemma 2*. Since $(k - \nu_2 T) \geq (k - \nu_2 T - \delta T)$, we get Eq. (3.6.10).

$$V^*_{d+1}\big(l',u,(k-\nu_2 T)\big) - V^*_{d+1}\big(l',u,(k-\nu_2 T-\delta T)\big) = \Delta \geq 0, \tag{3.6.10}$$

where $\Delta$ is the cost for transmitting data of size $\delta T$; it is defined in Eq. (3.6.11).

$$\Delta = \sum_{a \in \mathcal{A}} p_a \delta T \chi_a, \tag{3.6.11}$$

where $p_a$ is the percentage of data size $\delta$ transmitted by choosing action $a$. Except for the waiting action $a = 1$, where $p_1 = 0$, $p_a$ is unknown for other actions. From *Assumption 1*, $\chi_3$ is the minimum cost, we get Eq. (3.6.12).

$$\Delta = \delta T \sum_{a \in \mathcal{A} \setminus \{1\}} p_a \chi_a \geq \delta T \sum_{a \in \mathcal{A} \setminus \{1\}} p_a \chi_3 = \delta T \chi_3. \tag{3.6.12}$$

By Eq. (3.6.9), we have proved that $Q_d(l,u,k,3) \leq Q_d(l,u,k,2)$. Similarly, if $\nu_4 < \nu_3$, we can prove that $Q_d(l,u,k,3) \leq Q_d(l,u,k,4)$. Moreover, since $\nu_1 = 0 < \nu_3$, we obtain $Q_d(l,u,k,3) \leq Q_d(l,u,k,1)$. According to *Definition 1*, $a^*_{(l,u)} = 3\,(WiFi)$ is the dominant action.

□

Based on *Theorem 1*, where the waiting action, cellular action and D2D action are dominated by WiFi action, we obtain the following corollary.

**Corollary 1.** *(1) Given $l \in \mathcal{L}$ and $u \in \mathcal{U}$, for all $a^1_{(l,u)} \in \mathcal{A}(l,u)$ and $a^2_{(l,u)} \in \mathcal{A}(l,u)$, if $\chi_{a^1_{(l,u)}} > \chi_{a^2_{(l,u)}}$ and $\nu_{a^1_{(l,u)}} < \nu_{a^2_{(l,u)}}$, then $a^1_{(l,u)}$ is dominated by $a^2_{(l,u)}$. (2) Given $l^1, l^2 \in \mathcal{L}$ and $u \in \mathcal{U}$, for all $a^1_{(l^1,u)} \in \mathcal{A}(l^1,u)$ and $a^2_{(l^2,u)} \in \mathcal{A}(l^2,u)$, if $\chi_{a^1_{(l^1,u)}} > \chi_{a^2_{(l^2,u)}}$, $\nu_{a^1_{(l^1,u)}} < \nu_{a^2_{(l^2,u)}}$ and $a^2_{(l^2,u)} \in \mathcal{A}(l^2,u) \setminus \mathcal{A}(l^1,u)$, then $a^1_{(l^1,u)}$ is potentially dominated by $a^2_{(l^2,u)}$.*

This corollary includes two parts. The first part implies the dominant relationship between two actions in same grid, while the second part reveals the potentially dominant relationship between actions in different grids.

### 3.6.4. General monotone policy

In this subsection, we show the general case where the monotone policy exists in dimensions $k$ and $d$. We first introduce the basic definitions and properties of superadditive function and illustrate that $Q_d(l,u,k,a)$ is a superadditive function in $\mathcal{K} \times \mathcal{A}$ and $\mathcal{D} \times \mathcal{A}$. Then we derive the optimal monotone policy $\pi_d^*(l,u,k)$ in dimensions $k$ and $d$.

**Definition 2.** *A real valued function $f(m,a)$ is superadditive in $\mathcal{M} \times \mathcal{A}$, if*

$$f(m^+,a^+) - f(m^+,a^-) \geq f(m^-,a^+) - f(m^-,a^-), \tag{3.6.13}$$

*for $\forall m^+, m^- \in \mathcal{M}$ and $\forall a^+, a^- \in \mathcal{A}$, where $m^+ \geq m^-$ and $a^+ \geq a^-$.*

Given the definition of superadditive function, we next illustrate its properties summarized in *Lemmas 4* and *5* [81].

**Lemma 4.** *If $f_1(m,a)$ and $f_2(m,a)$ are superadditive functions in $\mathcal{M} \times \mathcal{A}$, then the function $h(m,a) = f_1(m,a) + f_2(m,a)$ is superadditive in $\mathcal{M} \times \mathcal{A}$.*

**Lemma 5.** *If $f(m,a)$ is a superadditive function in $\mathcal{M} \times \mathcal{A}$, then the function $g(a)$ defined below is monotone increasing in $m$.*

$$g(a) = \underset{a \in \mathcal{A}}{\operatorname{argmin}} f(m,a). \tag{3.6.14}$$

*Lemma 4* shows that the sum of two superadditive functions satisfies superadditive property. *Lemma 5* states that a superadditive function including two variables can be considered as a monotone increasing function having one variable, which implies the theoretical basic of monotone policy. In our model, we consider $Q_d(l,u,k,a)$ as a superadditive function in $\mathcal{K} \times \mathcal{A}$ and $\mathcal{D} \times \mathcal{A}$. The monotone policy in dimensions $k$ and $d$ is summarized as follows.

**Theorem 2.** *The optimal monotone policy $\Pi^* = \left\{\pi_d^*(l,u,k) = a^*, \forall l \in \mathcal{L}, u \in \mathcal{U}, k \in \mathcal{K}, d \in \mathcal{D}\right\}$ has threshold structure in both $k$ and $d$ as follows:*

***(a)*** *For location $l \in \mathcal{L}^1$ with only cellular network and $u \in \mathcal{U}$, we get $\mathcal{A}(l,u) = \{1,2\}$ by Eq.(3.4.1). There is one threshold for both $k$ and $d$. That is $\forall d \in \mathcal{D}$,*

$$\pi_d^*(l,u,k) = \begin{cases} 2\,(cellular), & if\ k \geq k^*(l,u,d), \\ 1\,(waiting), & otherwise, \end{cases} \tag{3.6.15}$$

*and $\forall k \in \mathcal{K}$,*

$$\pi_d^*(l,u,k) = \begin{cases} 2\,(cellular), & if\ d \geq d^*(l,u,k), \\ 1\,(waiting), & otherwise. \end{cases} \tag{3.6.16}$$

***(b)*** *For location $l \in \mathcal{L}^2$ with cellular and WiFi networks, and $u \in \mathcal{U}$, we get $\mathcal{A}(l,u) = \{1,2,3\}$ by Eq.(3.4.1). If $\nu_2 > \nu_3 > \nu_4$, there is one threshold for both $k$ and $d$. That is $\forall d \in \mathcal{D}$,*

$$\pi_d^*(l,u,k) = \begin{cases} 2\,(cellular), & if\ k \geq k^*(l,u,d), \\ 3\,(WiFi), & otherwise, \end{cases} \tag{3.6.17}$$

*and $\forall k \in \mathcal{K}$,*

$$\pi_d^*(l,u,k) = \begin{cases} 2\,(cellular), & if\ d \geq d^*(l,u,k), \\ 3\,(WiFi), & otherwise. \end{cases} \tag{3.6.18}$$

***(c)*** *For location $l \in \mathcal{L}^3$ with cellular network and D2D communication, and $u \in \mathcal{U}^0$, we get $\mathcal{A}(l,u) = \{1,2,4\}$ by Eq. (3.4.1). If $\nu_2 > \nu_4$, there are two thresholds for both $k$ and $d$. That is $\forall d \in \mathcal{D}$,*

$$\pi_d^*(l,u,k) = \begin{cases} 1\,(waiting), & if\ k \leq k_1^*(l,u,d), \\ 2\,(cellular), & if\ k \geq k_2^*(l,u,d), \\ 4\,(D2D), & otherwise, \end{cases} \tag{3.6.19}$$

*and* $\forall k \in \mathcal{K}$,

$$\pi_d^*(l,u,k) = \begin{cases} 1\,(waiting), & if\ d \le d_1^*(l,u,k), \\ 2\,(cellular), & if\ d \ge d_2^*(l,u,k), \\ 4\,(D2D), & otherwise. \end{cases} \tag{3.6.20}$$

***(d)*** *For location* $l \in \mathcal{L}^4$ *with cellular network, WiFi network and D2D communication, and* $u \in \mathcal{U}^0$*, we get* $\mathcal{A}(l,u) = \{1,2,3,4\}$ *by Eq.(3.4.1). If* $\nu_2 > \nu_4 > \nu_3$*, there are two thresholds for both* $k$ *and* $d$*. That is* $\forall d \in \mathcal{D}$,

$$\pi_d^*(l,u,k) = \begin{cases} 3\,(WiFi), & if\ k \le k_1^*(l,u,d), \\ 2\,(cellular), & if\ k \ge k_2^*(l,u,d), \\ 4\,(D2D), & otherwise, \end{cases} \tag{3.6.21}$$

*and* $\forall k \in \mathcal{K}$,

$$\pi_d^*(l,u,k) = \begin{cases} 3\,(WiFi), & if\ d \le d_1^*(l,u,k), \\ 2\,(cellular), & if\ d \ge d_2^*(l,u,k), \\ 4\,(D2D), & otherwise. \end{cases} \tag{3.6.22}$$

PROOF. We prove the result of *Theorem 2.a.* First, we show that the transition cost $c_d(s,a)$ defined in Eq. (3.6.1) is superadditive in $\mathcal{M} \times \mathcal{A}$. From Eq. (3.6.1), we get

$$c_d(m^+,a^+) - c_d(m^+,a^-) = \nu_{a^+} T \chi_{a^+} - \nu_{a^-} T \chi_{a^-}, \tag{3.6.23}$$

$$c_d(m^-,a^+) - c_d(m^-,a^-) = \nu_{a^+} T \chi_{a^+} - \nu_{a^-} T \chi_{a^-}. \tag{3.6.24}$$

From Eqs. (3.6.23) and (3.6.24), we get

$$c_d(m^+,a^+) - c_d(m^+,a^-) = c_d(m^-,a^+) - c_d(m^-,a^-). \tag{3.6.25}$$

Since Eq. (3.6.25) satisfies the definition of superadditive function (*Definition 2*), $c_d(s,a)$ is superadditive in $\mathcal{M} \times \mathcal{A}$.

Then, we show that $V^*_{d+1}(l',u,(k-\nu_a))$ is superadditive in $\mathcal{M}\times\{a^-,a^+\}$, where $a^-=1$ and $a^+=2$.

According to *Lemma 2*, we get Eq. (3.6.26). Thus, $V^*_{d+1}(l',u,(k-\nu_a T))$ is superadditive in $\mathcal{M}\times\{a^-,a^+\}$.

$$\begin{aligned} &V^*_{d+1}(l',u,(k^+-\nu_{a^+}T))-V^*_{d+1}(l',u,(k^+-\nu_{a^-}T)) \\ \geq\ &V^*_{d+1}(l',u,(k^--\nu_{a^+}T))-V^*_{d+1}(l',u,(k^--\nu_{a^-}T)). \end{aligned} \tag{3.6.26}$$

Based on *Lemma 4*, we obtain that $Q_d(l,u,k,a)$ defined in Eq. (3.6.27) is superadditive in $\mathcal{M}\times\{a^-,a^+\}$.

$$Q_d(l,u,k,a)=c_d(m,a)+\sum_{l'\in\mathcal{L}}P(l'|l)\cdot V^*_{d+1}(l',u,(k-\nu_a T)) \tag{3.6.27}$$

Based on *Lemma 5*, we obtain that the optimal policy $\pi^*_d(l,u,k)$ defined in Eq. (3.6.28) is monotone increasing with $s$.

$$\pi^*_d(l,u,k)=\underset{a\in\{a^-,a^+\}}{\operatorname{argmin}}\ Q_d(l,u,k,a) \tag{3.6.28}$$

Thus, $\pi^*_d(l,u,k)$ is a step function of the form Eq. (3.6.15). $k^*(l,u,d)$ is a state at which the optimal policy switches from $a^-=1$ to $a^+=2$, called threshold state. Similarly, we can prove Eq. (3.6.16) by showing that $Q_d(l,u,k,a)$ is superadditive in $\mathcal{D}\times\mathcal{A}$ by *Lemma 3*. □

We can derive *Theorem 2.(b)-(d)* by *Corollary 1* and then prove them the same way as *Theorem 2.a*. For example, considering *Theorem 2.b*, where $\chi_3<\chi_4<\chi_2$ (by *Assumption 1*) and $\nu_2>\nu_3>\nu_4$, action 4 is potentially dominated by action 3. This is because that $\chi_3<\chi_4$ and $\nu_3>\nu_4$ when $3\in\mathcal{A}(l,u)$ and $4\notin\mathcal{A}(l,u)$. Notice that the waiting action 1 is used to delay data transmission by seeking better offloading action. However, we don't need to delay now, since action 4 (the only action not in $\mathcal{A}(l,u)$) is potentially dominated by action 3. Moreover, action 2 and action 3 is

not dominated by each other. Thus, $\mathcal{A}(l,u) = \{2,3\}$ and there is one threshold in $l \in \mathcal{L}^2$. Since $\chi_3 < \chi_2$, we first choose action 3 ($WiFi$) which has lower cost. When exceeding the threshold $k^*(l,u,d)$ or $d^*(l,u,k)$, action 2 (*cellular*) which has higher data rate is used, as shown in Eqs. (3.6.17) and (3.6.18).

The monotone offloading algorithm will search the threshold states from the state space. In order to reduce the searching complexity, we make use of the following corollary.

**Corollary 2.** *In the monotone policy,* $\forall l \in \mathcal{L}, u \in \mathcal{U}$, $i \in \{1,2\}$ *is the index of thresholds,*

*(1)* $k_i^*(l,u,d) \geq k_i^*(l,u,d+1)$ *and* $k_2^*(l,u,d) \geq k_1^*(l,u,d)$, $\forall d \in \mathcal{D}$;

*(2)* $d_i^*(l,u,k) \geq d_i^*(l,u,k+1)$ *and* $d_2^*(l,u,k) \geq d_1^*(l,u,k)$, $\forall k \in \mathcal{K}$.

### 3.6.5. Monotone offloading algorithm

We propose monotone offloading algorithm to calculate the general monotone policy with a lower computational complexity, compared with hybrid offloading algorithm. By taking advantage of the threshold structure, monotone offloading algorithm only searches for the threshold states, instead of computing the optimal action for every system state. The threshold states are calculated in dimension $k$, denoted by $\Pi = \left\{k_i^*(l,u,d) = k, \forall\, l \in \mathcal{L}, u \in \mathcal{U}, d \in \mathcal{D}, i \in \{1,2\}\right\}$.

*Algorithm 2* consists of two phases: (1) planning phase (steps 3-5): *Algorithm 3* is used to calculate the threshold states; and (2) running phase (steps 7-14): The offloading action is decided by MS's grid $l$. For grid with WiFi coverage (step 10), the optimal action is WiFi action; For grid with only cellular coverage (step 11), the optimal action is determined by *Theorem 2.a*; For grid with D2D coverage (step 12), the optimal action is determined by *Theorem 2.c*.

*Algorithm 3* calculates the threshold states for $l \in \mathcal{L}_d^1 \cup \mathcal{L}_d^3$. Notice that there is one threshold in $l \in \mathcal{L}_d^1$ and two thresholds in $l \in \mathcal{L}_d^3$, as illustrated in Section

**Algorithm 2** Monotone Offloading Algorithm

1: *Planning Phase*
2: Initialize $V_{d+1}^*(s)$ with Eq. (3.4.3) and $\Pi \leftarrow \varnothing$
3: **for** $l \in \mathcal{L}$, $u \in \mathcal{U}$, $d \leftarrow \{D,...,1\}$ **do**
4: Call *Calculate Threshold States Algorithm*
5: **end for**
6: *Running Phase*
7: Set $d \leftarrow 1$ and $k \leftarrow K$
8: **while** $d < D+1$ and $k > 0$ **do**
9: Get the location of MS as $l$
10: **if** $l \in \mathcal{L}_d^2 \cup \mathcal{L}_d^4$, set $a \leftarrow 3$ by *Theorem 1*, **end if**
11: **if** $l \in \mathcal{L}_d^1$, choose action by *Theorem 2.a*, **end if**
12: **if** $l \in \mathcal{L}_d^3$, choose action by *Theorem 2.c*, **end if**
13: $k \leftarrow k - \nu_a$, $d \leftarrow d+1$
14: **end while**

**Algorithm 3** Calculate Threshold States

1: Calculate threshold states in dimension $k$
2: **if** $l \in \mathcal{L}_d^1$, set $numThreshold \leftarrow 1$ and $a_1 \leftarrow 2$ **endif**
**if** $l \in \mathcal{L}_d^3$, set $numThreshold \leftarrow 2$ , $a_1 \leftarrow 4$ and $a_2 \leftarrow 2$ **endif**
3: **for** $i \leftarrow \{1,...,numThreshold\}$ **do**
4: Set $k \leftarrow k_i^*(l,u,d+1)$ and $flag \leftarrow 0$
5: **while** $k \leq K$ and $flag == 0$ **do**
6: Calculate $Q_d(s,a)$, $\forall a \in \mathcal{A}(l,u)$ using Eq. (3.5.3)
7: Set $\pi_d^*(l,u,k) \leftarrow \text{argmin}_{a \in \mathcal{A}(l,u)} Q_d^*(s,a)$
8: Set $V^*(s,d) \leftarrow Q_d^*(s,\pi_d^*(l,u,k))$
9: **if** $\pi_d^*(l,u,k) == a_i$, set $\Pi \leftarrow \Pi \cup \left\{k_i^*(l,u,d) \leftarrow k\right\}$ and $flag \leftarrow 1$ **endif**
10: Set $k \leftarrow k+1$
11: **end while**
12: **end for**

3.7.1. *Algorithm 3* first determines the number of thresholds based on $l$ (step 1). Then it calculates the threshold states $k_i^*(l,u,d)$ in dimension $k$. Notice that we only consider $k$ that starts from $k_i^*(l,u,d+1)$ (step 3) instead of $\forall\, k \in \mathcal{K}$, due to the fact

that the threshold states cannot be found in $k < k_i^*(l,u,d+1)$ based on *Corollary 2*. *Algorithm 3* can find the threshold states within a constant number of loops. Indeed, the maximum number of outer loops (steps 2-10) is 2. The number of inner loops (steps 4-9) is determined by $\max_{d\in\mathcal{D}}\{k_i^*(l,u,d) - k_i^*(l,u,d+1)\}$, which is a constant number. Thus, the complexity of *Algorithm 3* is $\mathcal{O}(1)$. The complexity of *Algorithm 2* is mainly due to the planning phase, that is $\mathcal{O}(LUD)$.

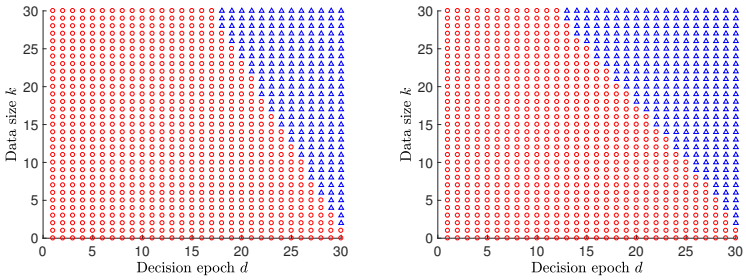
## 3.7. Performance Evaluation

This section evaluates the performance of our proposed approaches to implement efficient mobile data offloading. More specifically, we aim to evaluate the impact of mobile data size and delay tolerance on the performance of our offloading methods. First, we describe the parameters' settings used in our numerical analysis. Next, the threshold structures in monotone policy are illustrated. At last, we evaluate the performance metrics considered in our analysis.

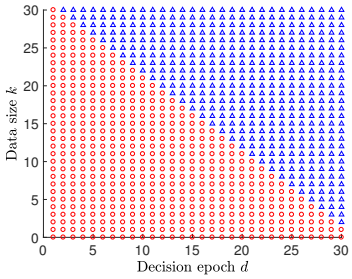
### 3.7.1. Threshold Structures in Monotone Policy

### 3.7.2. Experimental Setup

We have solved the optimal offloading policy and implemented the proposed offloading methods using MATLAB. For each choice of parameters' settings, we run the simulations 1000 times and show the average values. The number of grids $L$ equals to $20 \times 20$. MSs and MHs move in the grids according to the memoryless mobility pattern, where the stable factor $\mu = 0.6$ and $\rho_i = 0.1$, $\forall i \in \{1,2,3,4\}$ [7, 37, 41]. The number of WiFi APs and MHs are denoted as $n_w$ and $n_h$, respectively. The locations of WiFi APs and MHs are generated randomly. The data rates of WiFi, cellular and D2D actions follow normal distribution with means $\nu_2 = 16$ Mbps,




(a) $n_w = 33, n_h = 66$ (b) $n_w = 10, n_h = 66$

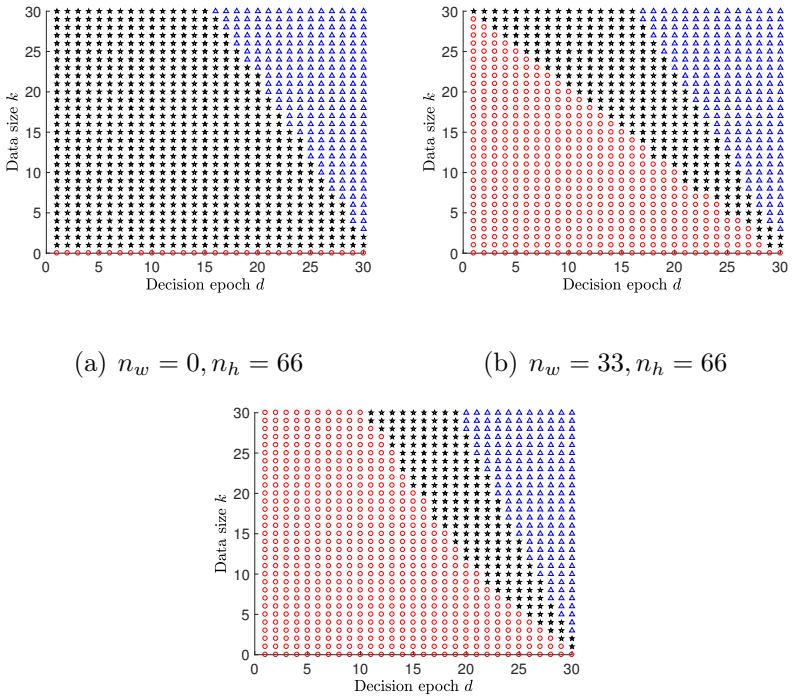



(c) $n_w = 0, n_h = 66$

**Fig. 3.3.** Monotone policy in $l \in \mathcal{L}^1$: $K = 30$ *Mbytes* and $D = 30$ *seconds*. The dots ($\circ$) and triangles ($\triangle$) represent the waiting and cellular action, respectively.

$\nu_3 = 24$ Mbps and $\nu_4 = 8$ Mbps, respectively, and standard deviations equal to 5 Mbps, which is a rational setting based on [47, 82].

Similar to [41], the cellular unit cost is set to $\chi_2 = 1$ serving as a baseline. We set the unit cost for WiFi network and D2D communication in terms of the reserve price (i.e., the price that MNO is willing to pay at most for offloading one data unit), where $\chi_3 = [0.05, 0.08]$ and $\chi_4 = 0.2$ [40]. The reserve price $\chi_4$ is set by MNO. If MNO sets a high $\chi_4$ for D2D communication, MHs will get high rewards and thus will be willing to help in D2D offloading. In our evaluation, we set $\chi_3 = 0.08$ and $\chi_4 = 0.2$. The length of time slot $T$ is 10 *seconds* unless stated otherwise.




(a) $n_w = 0, n_h = 66$ (b) $n_w = 33, n_h = 66$

(c) $n_w = 66, n_h = 66$

**Fig. 3.4.** Monotone policy in $l \in \mathcal{L}^3$: $K = 30$ *Mbytes* and $D = 30$ *seconds*. The dots ($\circ$), triangles ($\triangle$), and stars ($\star$) represent the waiting, cellular and D2D action, respectively.

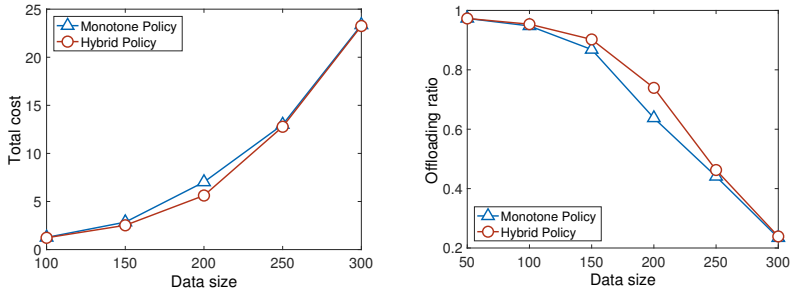
In our settings, WiFi action has the highest data rate and lowest cost. Thus, according to *Theorem 1*, WiFi action is the dominant action in $l \in \mathcal{L}_d^2 \cup \mathcal{L}_d^4$. The optimal policies having threshold structures in $\mathcal{L}_d^1$ and $\mathcal{L}_d^3$, based on *Theorem 2* $(a)$ *and* $(c)$, are shown in Figs. 3.3 and 3.4, respectively.

Fig. 3.3 illustrates the optimal policy for MSs in $l \in \mathcal{L}_d^1$, where $\mathcal{A}(l,u) = \{1,2\}$. Thus, MNO has two actions, i.e., waiting action and cellular action. As shown in Fig. 3.3, a single threshold exits in dimension $d$. It shows that cellular action is selected when $d$ is large enough. Otherwise, MNO chooses the waiting action. For

example, in Fig. 3(a), given $k = 10$, the optimal action changes from waiting action to cellular action when $d > 26$. The single threshold that exits in dimension $k$ has similar observation. Fig. 3.3 shows that our monotone policy will delay the usage of cellular network until the threshold state ($k^*(l,u,d)$ or $d^*(l,u,k)$). This is rational since MNO seeks to use WiFi network or D2D communication before deadline.

We observe that the threshold changes with the number of WiFi APs and MHs. With the belief that the number of WiFi APs in Fig. 3(a) is higher than that in Fig. 3(b), MS in Fig. 3(a) has larger WiFi connection probability, which implies higher offloading potential (the size of data can be transmitted using WiFi network or D2D communication before deadline). Thus, the waiting action area in Fig. 3(a) is larger than that in Fig. 3(b). Since Fig. 3(c) has the smallest offloading potential, the waiting action area is smaller than that in Figs. 3(a) and 3(b).

Fig. 4(a) shows that MNO chooses D2D action when $d$ or $k$ is small, instead of waiting action (see Fig. 3(c)), with the same knowledge of network setting (i.e., $n_w = 0$ and $n_h = 66$). Since $\chi_4 < \chi_2$, MNO chooses D2D action to minimize the transmission cost. However, when $d$ or $k$ is large, cellular action is selected to ensure that data transmission will be completed before deadline, due to the fact that $\nu_2 > \nu_4$. Figs. 4(b) and 4(c) show that two thresholds separate three actions (waiting action, cellular action and D2D action.) in dimensions $k$ and $d$, since $l \in \mathcal{L}_d^3$ and $\mathcal{A}(l,u) = \{1,2,4\}$. Compared with two actions (D2D and cellular actions) shown in Fig. 4(a), additional waiting action occurs in Figs. 4(b) and 4(c) with the knowledge of potential WiFi offloading. Both D2D and cellular action areas in Fig. 4(c) are smaller than those in Fig. 4(b), while the waiting action area in Fig. 4(c) is larger than that in Fig. 4(b); this can be explained by the fact that there are more WiFi APs in the case of Fig. 4(c).




(a) Total cost versus data size $K$ ($Mbytes$), the deadline $D$ is 200 seconds.

(b) Offloading ratio versus data size $K$ ($Mbytes$), the deadline $D$ is 200 seconds.

**Fig. 3.5.** Performance comparison of Hybrid and Monotone Policies.

### 3.7.3. Performance Comparisons among Different Schemes

In order to evaluate the performance of our proposed offloading methods, we consider following performance metrics: (1) *Total cost.* The total network cost for data transmission; (2) *Completion time.* The total time used for data transmission; (3) *Offloading ratio.* The percentage of cellular traffic that MNO transmits through WiFi or D2D networks; and (4) *Time usage percentage (TUP).* The ratio of completion time to the deadline. We compare our proposed scheme (we name $D4$) with

**Tab. 3.2.** Different offloading schemes

| Abbreviation | Schemes |
|---|---|
| D4 | Delayed optimal offloading with 4 actions |
| D3 | Delayed optimal offloading with 3 actions |
| ND4 | Non-Delayed offloading with 4 actions |
| ND3 | Non-Delayed offloading with 3 actions |
| NO | No Offloading |

four benchmark schemes (see Table 3.2). The abbreviation of each scheme includes a digit that indicates the available actions for a system state (i.e., 4 indicates that $\mathcal{A} = \{1,2,3,4\}$ and 3 indicates that $\mathcal{A} = \{1,2,3\}$). The benchmark schemes include: (1) optimal delayed WiFi offloading scheme ($D3$) [41]; (2) on-the-spot WiFi offloading scheme ($ND3$) [83]: data transmission is switched between WiFi and cellular networks. WiFi network is used whenever available; and (3) on-the-spot WiFi and D2D offloading scheme ($ND4$): WiFi network is used wherever available; D2D communication is used when MH is available and MS's state satisfies $k < \bar{\nu}_2 * (D - d)$. (4) no offloading scheme ($NO$): MS only uses cellular network. The results are averaged for a single MS.

3.7.3.1. *Performance Comparison of Hybrid and Monotone Policies*

Fig. 3.5 shows the performance comparison of our proposed hybrid offloading policy and monotone policy. We observe that hybrid policy has better performance in total cost (see Fig. 5(a)) and offloading ratio (see Fig. 5(b)). This is because that Algorithm 1 generates the hybrid policy based on location dependent data rates of different networks, while Algorithm 2 calculates the monotone policy based on the average data rates of different networks. Thus, hybrid policy achieves the better performance at the cost of higher computational complexity.

3.7.3.2. *Impact of Data Size*

We compare the performance metrics of different schemes with a given deadline $D$. Fig. 6(a) shows that the total cost increases with data size. We observe that $D4$ outperforms the other schemes by achieving the lowest total cost for any data size. Note that when $K < 150$ Mbytes, $D3$ outperforms $ND4$; it is not the case when $K > 150$ Mbytes. This can be explained by the fact that $ND4$ can use D2D action

to offload more data than $D3$ when $K > 150$ Mbytes, while $D3$ can use delayed WiFi offloading to offload more data than $ND4$ when $K < 150$ Mbytes.

Fig. 6(b) shows that TUP increases with data size, since it takes longer to transmit data of larger size. We observe that TUP of non-delayed schemes (i.e., $ND4$ and $ND3$) are smaller than that of delayed schemes (i.e., $D4$ and $D3$). This is because delayed schemes use additional time to wait for offloading opportunities. TUP of $D4$ is smaller than that of $D3$, since $D4$ can use D2D action to offload mobile data when $D3$ is waiting for another WiFi connection.

Fig. 6(c) shows that the offloading ratio decreases when data size increases except for $ND3$. This is because $ND3$ transmits data based on the available networks without considering current data size $k$ and decision epoch $d$. We observe that the offloading ratio of $D3$ drops rapidly with the increase of data size, while that of $D4$ and $ND4$ drop slowly. This is because $D4$ and $ND4$ use alternative D2D action to offload data. Notice that $D4$ has the highest offloading ratio. We also observe that the offloading ratios for delayed and non-delayed schemes are the same when $K = 400$ Mbytes. This can be explained by the fact that 400 Mbytes is the transmission limit under the setting used in our simulations. Since all offloading schemes try to complete data delivery before deadline, they use WiFi network wherever possible and cellular network when WiFi network is not available, which is the offloading policy used by $ND3$. It means that all other offloading polices (i.e., $D4$, $ND4$, and $D3$) degenerate to the policy $ND3$. Notice that, $ND4$ can offload more data than $D3$ when $K < 300$ Mbytes, while $D3$ can offload more data than $ND4$ when $K > 300$ Mbytes. This is because $D3$ uses delayed policy, while $ND4$ uses alternative D2D action.

In Figs. 6(d), 6(e) and 6(f), we investigate the impact of $n_w$ and $n_h$ on total cost, offloading ratio and completion time for $D4$. We observe that the total cost and completion time decreases, while the offloading ratio increases, with the increase of the number of WiFi APs and MHs. For example, when $n_w = 800$ and $n_h = 800$,

we observe that the lowest total cost and completion time is achieved since it has the largest number of WiFi APs and MHs. This shows that our proposed method performs well with the increase of the numbers of APs and MHs.

3.7.3.3. *Impact of Delay Tolerance*

We compare the performance metrics of different schemes with a given data size $K$. Fig. 7(a) shows that, for delayed schemes (i.e., $D4$ and $D3$) and $ND4$, the total cost decreases when the deadline increases; indeed, larger deadlines give more opportunities (i.e. more time) to look for WiFi and D2D actions to transmit data. We observe that the total cost of $D3$ is larger than that of $ND4$ when $D < 300$ Mbytes. However, the situation changes when $D > 300$ Mbytes. This is because $D3$ uses waiting based strategy for seeking WiFi offloading opportunities; larger deadlines imply more WiFi offloading opportunities. Note that $D4$ incurs the minimum total cost compared to other schemes.

Fig. 7(b) shows that the total transmission time increases with the deadline. We observe that, for $ND3$, the total time does not increase when $D > 200$ seconds. This is because $ND3$ uses on-the-spot strategy and cannot make use of the delay tolerance. The total time for delayed schemes (e.g., $D3$ and $D4$) increases almost linearly with the deadline; indeed $D3$ and $D4$ use delayed time to seek for offloading opportunities. Moreover, $D4$ uses slightly less total time than $D3$ when the deadline increases. However, $D4$ can offload more data than $D3$, as shown in Fig. 7(c). This is because D2D action can be used to transmit data when WiFi action is not available.

In Fig. 7(c), we observe that offloading ratio increases with the deadline for delayed schemes (i.e., $D3$ and $D4$). This is because delayed schemes can take advantage of the delay tolerance to seek offloading opportunities through WiFi and D2D actions. We also observe that offloading ratio for $ND3$ does not increase with the deadline, while offloading ratio for $ND4$ increases with the deadline. This is because,

for $ND4$, MS has more opportunities to use D2D action as the deadline increases. We conclude that $D4$ achieves the maximum offloading ratio while satisfying data transmission deadline.

3.7.3.4. *Offloading Component Analysis*

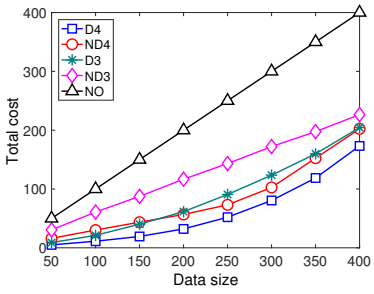
We evaluate the impact of delay tolerance on the transmission time and the amount of data transmitted by different networks, as shown in Figs. 3.8 and 3.9, respectively. Three scenarios with low, middle, and high delay tolerance are considered by setting $D$ to 200, 300 and 400 seconds, respectively. The data size $K$ is set to 400 Mbytes.

In Fig. 3.8, we observe that higher delay tolerance results in better data offloading performance for delayed schemes (i.e., $D4$ and $D3$) at the cost of longer completion time. To offload the same size of data, less cellular time is used in high delay tolerance scenario. We also observe that the completion time for delayed schemes is larger than non-delayed schemes in high delay tolerance scenario. This is because delayed schemes look for more offloading opportunities by extending the waiting time. Fig. 3.8 shows that the waiting time for $D4$ and $D3$ grows significantly when $D$ increases. Note that the completion time of $ND3$ is always smaller than that of $NO$, due to the fact that $\nu_3 > \nu_2$. This implies that using WiFi network can reduce the completion time. However, the completion time of $ND4$ is not always smaller than that of $NO$, as shown in Fig. 8(a). This can be explained by the fact that although WiFi network can reduce the completion time, the usage of D2D action may increase the completion time, due to the fact that $\nu_4 < \nu_2$. Fig. 3.8 shows that $ND4$ uses less cellular time than $D3$ when $D = 200$ and $D = 300$ seconds. However, $D3$ outperforms $ND4$ when $D = 400$ seconds. This is because $D3$ can offload more cellular traffic with higher delay tolerance. Furthermore, Fig. 3.8 shows that $D4$ uses the minimum cellular time in all situations.

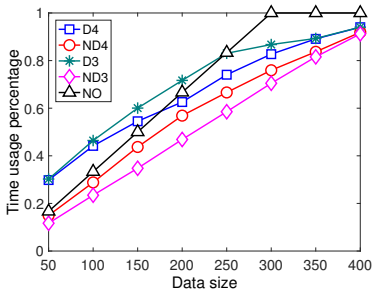
In Fig. 3.9, we evaluate the impact of delay tolerance on the amount of data transmitted by different networks. For each scheme, the total amount of data transmitted by different networks is equal to 400 Mbytes. We observe that the cellular data size of delayed schemes (e.g., $D4$ and $D3$) decreases as the deadline increases, due to the fact that a higher delay tolerance can provide more opportunities of delivering data using WiFi or D2D action. Figs. 9(a), 9(b) and 9(c) shows that for $D4$, with the increase of WiFi data size, cellular data size decreases accordingly. However, D2D data size first increases and then decreases with the increase of deadline $D$. The reason behind this phenomenon is that when higher delay tolerance is allocated, it is more likely to have more D2D connections or larger D2D data size. However, higher delay tolerance also increases the WiFi data size with lower cost $\chi_3 < \chi_4$. Thus, part of D2D offloading is replaced by WiFi offloading when more WiFi connections are possible. Moreover, compared to other schemes, the cellular data size of $D4$ decreases faster with higher delay tolerance, as shown in Figs. 9(a), 9(b) and 9(c). This demonstrates that the proposed $D4$ can offload more cellular data in practice.

## 3.8. Conclusion

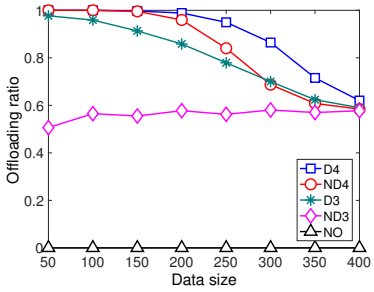
This paper proposed a hybrid data offloading model, where MNO can use WiFi network and D2D communication to offload mobile data of MSs. We formulated the mobile data offloading problem as an FHMDP and proposed a hybrid offloading algorithm for delay sensitive and delay tolerant applications. Moreover, we established sufficient conditions for the existence of thresholds in monotone policy and proposed a monotone offloading algorithm which can reduce the computational complexity caused by large data size and long deadline. The simulation results demonstrate that, compared to existing offloading schemes, our proposed schemes can achieve minimal data offloading cost and maximum offloading ratio.



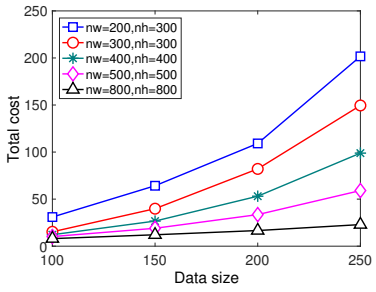

(a) Total cost versus data size $K$ for $D = 200$ seconds.



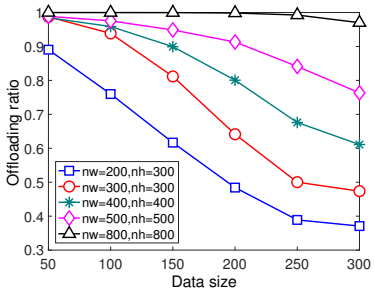

(b) Time usage percentage versus data size $K$ for $D = 140$ seconds.



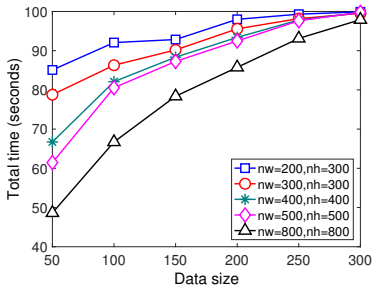

(c) Offloading ratio versus data size $K$ for $D = 140$ seconds.




(d) Total cost versus data size $K$ for $D = 100$ seconds.

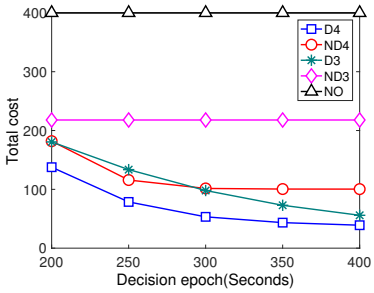



(e) Offloading ratio versus data size $K$ for $D = 100$ seconds.

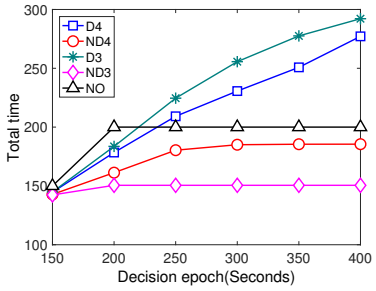



(f) Completion time versus data size $K$ for $D = 150$ seconds.

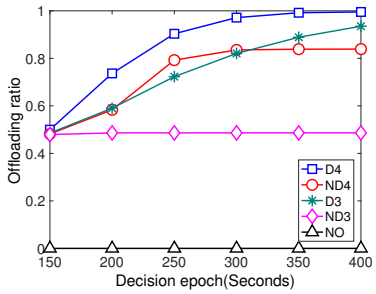
**Fig. 3.6.** Performance comparison versus data size $K$.




(a) Total cost versus deadline $D$ for $K = 400$ Mbytes.

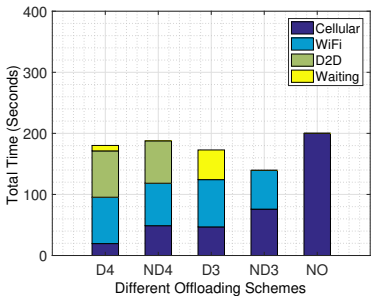



(b) Completion time versus deadline $D$ for $K = 400$ Mbytes.

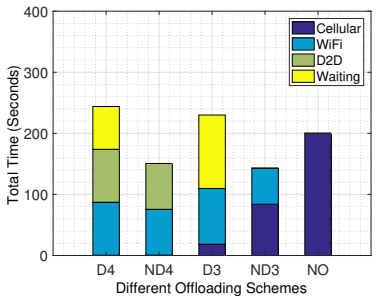



(c) Offloading ratio versus deadline $D$ for $K = 400$ Mbytes.

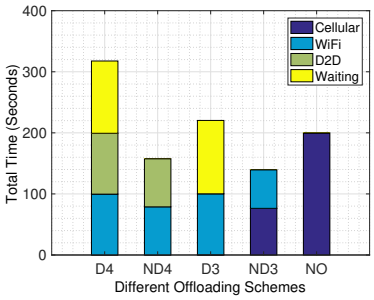
**Fig. 3.7.** Performance comparison versus deadline $D$.




(a) $K = 400$ Mbytes, and $D = 200$ seconds.

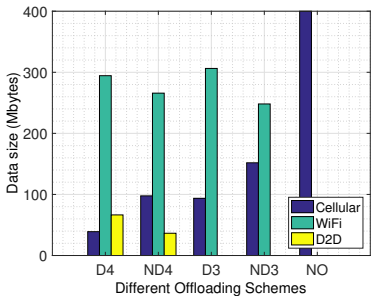



(b) $K = 400$ Mbytes, and $D = 300$ seconds.

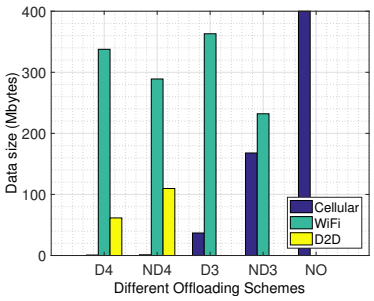



(c) $K = 400$ Mbytes, and $D = 400$ seconds.

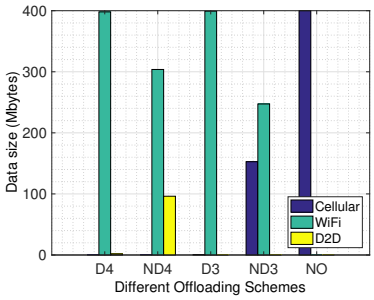
**Fig. 3.8.** Total completion time comparison for different schemes, with same data size $K = 400$ Mbytes and different deadline $D$.




(a) $K = 400$ Mbytes, and $D = 200$ seconds.




(b) $K = 400$ Mbytes, and $D = 300$ seconds.




(c) $K = 400$ Mbytes, and $D = 400$ seconds.

**Fig. 3.9.** Data transmitted comparison for different schemes, with same data size $K = 400$ Mbytes and different deadline $D$.

# Chapitre 4

---

# Multi-Item Auction Based Mechanism for Mobile Data Offloading

## 4.1. Abstract

The opportunistic utilization of access devices to offload mobile data from cellular network has been considered as a promising approach to cope with the explosive growth of cellular traffic. To foster this opportunistic utilization, we consider a mobile data offloading market where mobile network operator (MNO) can sell bandwidth made available by the access points (APs) to increase MNO's profit. We formulate the offloading problem as a multi-item auction and study MNO's profit maximization problem. We discuss the conditions to (i) offload the maximum amount of data traffic, (ii) foster the participation of mobile subscribers (MSs) (individual rationality), (iii) prevent market manipulation (incentive compatibility) and (iv) preserve budget feasibility of MSs. Then, we propose a robust optimization based method to implement multi-item auction mechanism. We further propose two iterative algorithms that efficiently solve the offloading problem. The simulation results show the efficiency and robustness of our proposed methods for cellular data offloading.

**Keywords:** Multi-item auction, mobile data offloading, heterogeneous networks, robust optimization.

**Status:** This journal paper is accepted. Liu, Dongqing, Abdelhakim Hafid, and Lyes Khoukhi. Multi-Item Auction Based Mechanism for Mobile Data Offloading: A Robust Optimization Approach. IEEE Transactions on Vehicular Technology. 2019.

## 4.2. Introduction

The rapid growth of mobile data traffic raises big challenges to cellular network. Global mobile data traffic grew 63 percent and reached 7.2 exabytes per month in 2016, which is 18-fold over the past 5 years [1]. The huge amount of mobile data traffic exceeds the capacity of cellular network and reduces the network quality [2]. To address such challenges, one simple solution is to increase the capacity of cellular network, which is inefficient and expensive due to the corresponding expensive investments in radio access networks and the core infrastructure. One promising solution, namely mobile data offloading, is to offload cellular traffic to other kinds of networks, e.g. WiFi APs and femtocells; this can solve the cellular traffic overload problem [84, 85].

Although mobile data offloading can significantly reduce cellular traffic, the task of developing a comprehensive and reliable mobile data offloading system remains challenging. A key challenge is how to achieve an efficient data offloading coordination among multiple mobile devices. By opportunistic utilization of lower cost APs, MSs will have better wireless access service with lower cost. In contrast, MNOs who have deployed these APs want to maximize the revenue by selling bandwidth. Thus, how to effectively allocate this bandwidth to mobile devices effectively becomes a key problem to be solved.

Given the limited bandwidth of APs deployed in a mobile data offloading market, when demands of mobile devices exceed supply, MNO needs to allocate the bandwidth to mobile devices and decide the price for allocated bandwidth in order to achieve the highest revenue. Auction mechanism is considered as an economically efficient approach towards the allocation of APs' bandwidth, and assigns bandwidth to MSs who value it the most [13–16, 22, 24, 86]. In a real-world data offloading market, the bidding prices of MSs are private information unavailable for MNO. However, MNO may use historical information to identify the numerical characteristics of the bidding prices. Consequently, it is natural to consider how to model the bidding prices based on historical information. We use uncertainty set to model the possibility of bidding prices. MNO assumes that all bidding prices belong to the uncertainty set derived from historical information. Then, MNO makes an offloading decision based on the uncertainty set instead of some fixed bidding prices.

In this paper, we focus on designing an efficient auction mechanism for allocating APs' bandwidth among multiple MSs; this is considered as a multi-item auction problem. MNO which owns the network infrastructure acts as the auctioneer and sells bandwidth to mobile devices through an auction. We formulated the auction problem based on robust optimization which models the desirable properties (budget feasibility, incentive compatibility, and individual rationality) of optimal auctions enabling the auctioneer to use historical data or prior knowledge of valuations. The uncertainty of item valuations is modeled as an uncertainty set, which is constructed based on limit theorems of probability theory. The optimal auction mechanism with reservation price has the structure of a VCG mechanism [28].

The main contributions of this paper can be summarized as follows:

- We characterize the interaction among MNO and MSs in a multi-item auction aiming at maximizing the MNO's revenue and the amount of offloaded traffic from mobile subscribers. Our proposed multi-item auction calculates

reservation prices based on the uncertainty set and the MSs' budgets; this can prevent market manipulation. Our proposed auction is implemented by robust optimization. Instead of requiring the full knowledge of MSs' valuations, robust optimization uses few information of MSs' valuations and can obtain a global $\epsilon$-optimal solution.

- Since the optimal multi-item auction problem is difficult to solve, we further propose two greedy auctions that can solve the offloading market problem in polynomial time, while preserving the properties of budget feasibility, incentive compatibility, and individual rationality. These two greedy auctions outperforms each other in different network scenarios.
- We perform numerical analysis and comparative evaluation of the proposed optimal and greedy auctions, considering realistic network scenarios. We further illustrate that the proposed offloading mechanisms can improve cellular data offloading performance and has higher robustness compared to Myerson auction.

The rest of the paper is organized as follows. Section 4.3 presents the system model. Section 4.4 formulates the multi-item auction as a robust optimization problem. Section 4.5 and Section 4.6 propose the optimal and greedy auction mechanisms, to solve the offloading market problem, respectively. Section 4.7 illustrates and analyzes the numerical results. Section 4.8 concludes the paper.

## 4.3. System Model

In this section, we present the economic definitions and network model that are considered in our multi-item auction mechanism; the objective of this mechanism is to implement efficient mobile data offloading. A scenario of data offloading among multiple APs and MSs is shown in Figure 4.1, where MSs, in the coverage area of APs, engage in an auction to acquire bandwidth (in WiFi network). We first model

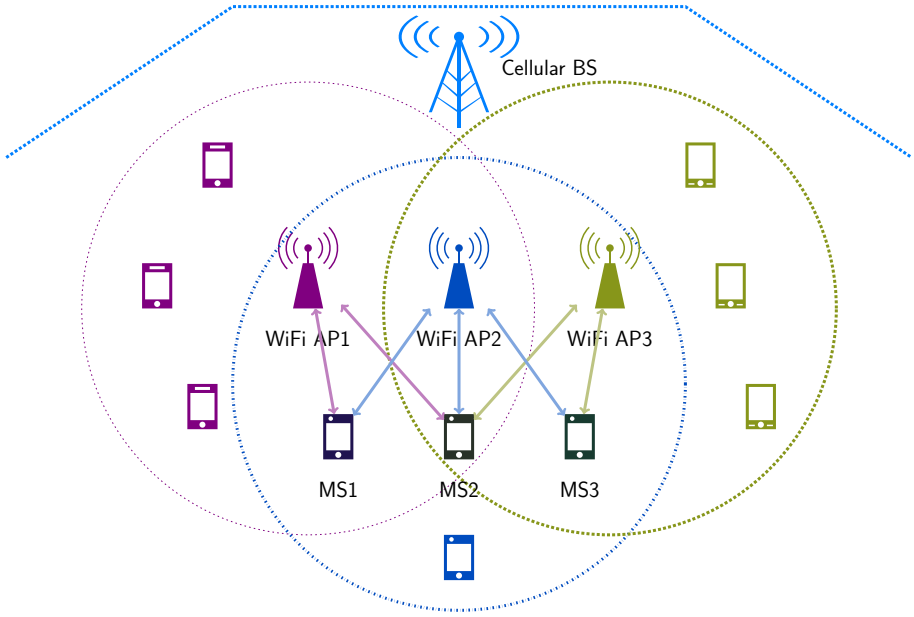



**Fig. 4.1.** An illustration of data offloading auction model. WiFi APs are managed by a single MNO that provides network access to its mobile subscribers (e.g., MS1). The network capacity of WiFi access points (e.g., AP1) is allocated to MSs for data traffic offloading. In this scenario, MS1, MS2 and MS3 bid for bandwidth (i.e., AP1, AP2 and AP3) with different valuations. Considering the coverage area of each AP, MS2 can bid for three APs, while MS1 and MS3 can bid for two APs. MNO who is the auctioneer allocates different APs' bandwidth to MSs. The winning MSs can use bandwidth determined by MNO.

the uncertainty of MNO's beliefs on MS's valuations using uncertainty set. Then, we introduce the general economical definitions for multi-item auction.

Let $\mathcal{N}$ denote the set of MSs, and $\mathcal{M}$ denote the set of APs owned by MNO, where $|\mathcal{N}| = n$ and $|\mathcal{M}| = m$. MS $i$ has a private valuation for the unit bandwidth usage associated with AP $j$, denoted by $v_{ij}$ which is unknown to MNO. Let $\boldsymbol{v} = \{v_{ij} | i \in \mathcal{N}, j \in \mathcal{M}\}$ denote the private valuation matrix. Thus, for AP $j$, $\boldsymbol{v}_j =$

$(v_{1j},\ldots,v_{nj})$ denotes the column vector of private valuation matrix $\mathcal{P}$. Moreover, MS $i$ is budget constrained and the available budget is denoted by $B_i, i \in \mathcal{N}$, while AP $j$ is bandwidth constrained and the available bandwidth is denoted by $C_j, j \in \mathcal{M}$. In this paper, we consider that the valuation information is private (only known to MS) and budget information is public (known to MNO). [1]

For AP $j$, since the private valuations of MSs are hidden from MNO, we model MNO's beliefs on the valuations of $n$ MSs using uncertainty set $\mathcal{U}_j$, where the valuation vector $\boldsymbol{v}_j \in \mathcal{U}_j$. MNO's belief on valuations for all APs is denoted as $\mathcal{U} = \{\mathcal{U}_j\}_{j\in\mathcal{M}}$.

We assume that the valuations for AP $j$ are independent and identically distributed, as well as the expectation and deviation of AP $j$ are $\mu_j$ and $\delta_j$ respectively. Based on the central limit theory, the distribution of

$$\frac{\sum_{i=1}^{n} v_{ij} - n\cdot\mu_j}{\sqrt{n}\cdot\delta_j}$$

is approximately a standard normal distribution when $n \to \infty$. Thus, the uncertainty set $\mathcal{U}_j$ can be constructed as follows.

$$\mathcal{U}_j = \left\{ (v_{1j},\ldots,v_{nj}) \;\middle|\; -\Gamma \le \frac{\sum_{i=1}^{n} v_{ij} - n\cdot\mu_j}{\sqrt{n}\cdot\delta_j} \le \Gamma \right\}, \tag{4.3.1}$$

where $\underline{F}_j$ and $\overline{F}_j$ are the lower bound and upper bound of the competition function $f_i(k)$, respectively. $\Gamma$ is a parameter that controls the conservativeness of the historical valuations. For example, under the central limit theorem, the probability that $(\hat{v}_{1j},\cdots,\hat{v}_{nj})$ belongs to

$$-\Gamma \le \frac{\sum_{i=1}^{n} v_{ij} - n\cdot\mu_j}{\sqrt{n}\cdot\delta_j} \le \Gamma$$

[1]The budget information can be extended to private situation by uncertainty set with extra computaional complexity.

can be calculated by

$$\mathbb{P}\big((\hat{v}_{1j}, \cdots, \hat{v}_{nj}) \in \mathcal{U}_j\big) = 2\Phi(\Gamma) - 1, \tag{4.3.2}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal. If we set $\Gamma$ to 1,2 and 3, then $\mathbb{P}((\hat{v}_{1j}, \cdots, \hat{v}_{nj}) \in \mathcal{U}_j)$ is 0.683, 0.955 and 0.997, respectively. A smaller $\Gamma$ makes MNO consider only those valuations with higher probability. A larger $\Gamma$ makes MNO consider a larger range of valuations, which increases the accuracy of auction at the cost of computational complexity. Thus, MNO needs to choose a proper $\Gamma$ to balance the accuracy and computational complexity of the auction.

## 4.4. Problem Statement

In this section, we formulate the multi-item auction based data offloading problem as a robust optimization problem. Our objective is to maximize the total revenue of MNO for all valuations in the uncertainty set $\mathcal{U}$. We first introduce the decision variables that represent the allocation rule and the payment rule. Then, we define the properties that the allocation and payment rules should satisfy in order to implement an efficient auction. The notations used in this paper are described in Table 4.1.

### 4.4.1. Allocation and Payment Rules

The decision variable $\boldsymbol{x}^{\boldsymbol{v}} = \{x_{ij}^{\boldsymbol{v}}\}_{i\in\mathcal{N}, j\in\mathcal{M}}$ describes APs' bandwidth allocation among multiple MSs based on the valuation matrix $\boldsymbol{v}$, that is, if the valuation matrix is $\boldsymbol{v}$, MNO will allocate $x_{ij}^{\boldsymbol{v}}$ bandwidth of AP $j$ to MS $i$. If MS $i$ is not in the coverage area of AP $j$, then $x_{ij}^{\boldsymbol{v}} = 0$. The decision variable $\boldsymbol{p}^{\boldsymbol{v}} = \{p_i^{\boldsymbol{v}}\}_{i\in\mathcal{N}}$ denotes the payment of MSs according to current valuation matrix $\boldsymbol{v}$, where $p_i^{\boldsymbol{v}}$ is the total payment of MS $i$ for using the bandwidth of APs. Thus, $p_i^{\boldsymbol{v}} \geq 0$.

**Tab. 4.1.** Notation used in the paper

| | |
|---|---|
| $\mathcal{N}$ | Set of Mobile Subscribers |
| $\mathcal{M}$ | Set of Access Points |
| $\mathcal{U}$ | Uncertainty set of $\boldsymbol{v}$ |
| $\boldsymbol{B} = \{B_i\}_{i\in\mathcal{N}}$ | MS budget constraints |
| $\boldsymbol{D} = \{D_i\}_{i\in\mathcal{N}}$ | MS bandwidth demand |
| $\boldsymbol{C} = \{C_i\}_{i\in\mathcal{M}}$ | AP bandwidth constraints |
| $\boldsymbol{v} = \{v_{ij}\}_{i\in\mathcal{N},j\in\mathcal{M}}$ | Bid matrix |
| $\boldsymbol{v}_k = \{v_{kj}\}_{j\in\mathcal{M}}$ | Bid vector of MS $k$ |
| $\boldsymbol{v}_{-k} = \{v_{ij}\}_{i\in\mathcal{N}\setminus\{k\},j\in\mathcal{M}}$ | Bid vector except for MS $k$ |
| $\boldsymbol{z} = \{z_{ij}\}_{i\in\mathcal{N},j\in\mathcal{M}}$ | Worst case bid vector |
| $\boldsymbol{x}^* = \{x^*_{ij}\}_{i\in\mathcal{N},j\in\mathcal{M}}$ | Nominal allocation in worst case |
| $\boldsymbol{r}^* = \{r^*_{ij}\}_{i\in\mathcal{N},j\in\mathcal{M}}$ | Reservation prices in worst case |
| $\boldsymbol{y}^{\boldsymbol{v}} = \{y^{\boldsymbol{v}}_{ij}\}_{i\in\mathcal{N},j\in\mathcal{M}}$ | Adapted allocation |
| $\boldsymbol{y}^{\boldsymbol{v}_{-k}} = \{y^{\boldsymbol{v}_{-k}}_{ij}\}_{i\in\mathcal{N}\setminus\{k\},j\in\mathcal{M}}$ | Adapted allocation without MS $k$ |
| $\boldsymbol{a}^{\boldsymbol{v}} = \{a^{\boldsymbol{v}}_{ij}\}_{i\in\mathcal{N},j\in\mathcal{M}}$ | Real allocation |
| $\boldsymbol{p}^{\boldsymbol{v}} = \{p^{\boldsymbol{v}}_i\}_{i\in\mathcal{N}}$ | Real payments |

Given the allocation variable $\boldsymbol{x}^{\boldsymbol{v}}$ and payment variable $\boldsymbol{p}^{\boldsymbol{v}}$, we can derive the utility (i.e., the difference of total valuation and payment) of MS $i$ as follows,

$$U_i^{\boldsymbol{v}} = \sum_{j\in\mathcal{M}} v_{ij}\cdot x^{\boldsymbol{v}}_{ij} - p^{\boldsymbol{v}}_i, \quad i\in\mathcal{N}, \quad \boldsymbol{v}\in\mathcal{U}. \tag{4.4.1}$$

The allocation and payment variables should satisfy the following properties in order to implement an efficient multi-item auction.

- Individual Rationality (IR). This property ensures nonnegative utilities (i.e., the payment of MS should be less than his obtained valuation) for MSs who bid truthfully. Formally,

$$p_i^{\boldsymbol{v}} \leq \sum_{j \in \mathcal{M}} v_{ij} \cdot x_{ij}^{\boldsymbol{v}}, \forall i \in \mathcal{N}, \forall \boldsymbol{v} \in \mathcal{U}. \tag{4.4.2}$$

- Budget Feasibility (BF). This property ensures the payment of each MS is within his budget constraint. Formally,

$$p_i^{\boldsymbol{v}} \leq B_i, \forall i \in \mathcal{N}, \forall \boldsymbol{v} \in \mathcal{U}, \tag{4.4.3}$$

  where $B_i$ is the limited budget of MS $i$.
- Incentive Compatibility (IC). This property ensures that MS cannot improve his utility by bidding untruthfully. Thus, the utility of MS under truthful bidding is higher than untruthful biddings; this allows avoiding market manipulation by MSs. Formally,

$$U_i^{(\boldsymbol{v}_i, \boldsymbol{v}_{-i})} \geq U_i^{(\boldsymbol{u}_i, \boldsymbol{v}_{-i})}, \forall i \in \mathcal{N}, \forall (\boldsymbol{v}_i, \boldsymbol{v}_{-i}) \in \mathcal{U}, \forall (\boldsymbol{u}_i, \boldsymbol{v}_{-i}) \in \mathcal{U}, \tag{4.4.4}$$

  where $\boldsymbol{v}_i = \{v_{ij}\}_{j \in \mathcal{M}}$ is the truthful valuation of MS $i$ and $\boldsymbol{u}_i = \{u_{ij}\}_{j \in \mathcal{M}}$ is a possible valuation of MS $i$. $\boldsymbol{v}_{-i} = \{v_{kj}\}_{k \in \mathcal{N} \setminus \{i\}, j \in \mathcal{M}}$ denotes the valuation matrix obtained by omitting the valuations from MS $i$. By substituting Eq. (4.4.1) into Eq. (4.4.4), we have

$$\begin{aligned} \sum_{j \in \mathcal{M}} v_{ij} \cdot x_{ij}^{(\boldsymbol{v}_i, \boldsymbol{v}_{-i})} - p_i^{(\boldsymbol{v}_i, \boldsymbol{v}_{-i})} \geq \sum_{j \in \mathcal{M}} v_{ij} \cdot x_{ij}^{(\boldsymbol{u}_i, \boldsymbol{v}_{-i})} - p_i^{(\boldsymbol{u}_i, \boldsymbol{v}_{-i})}, \\ \forall i \in \mathcal{N}, \forall (\boldsymbol{v}_i, \boldsymbol{v}_{-i}) \in \mathcal{U}, \forall (\boldsymbol{u}_i, \boldsymbol{v}_{-i}) \in \mathcal{U}, \end{aligned} \tag{4.4.5}$$

With some mathematical manipulation of Eq. (4.4.5), we obtain the following equation.

$$\sum_{j\in\mathcal{M}} v_{ij}\cdot\left(x_{ij}^{(\boldsymbol{u}_i,\boldsymbol{v}_{-i})}-x_{ij}^{(\boldsymbol{v}_i,\boldsymbol{v}_{-i})}\right)+p_i^{(\boldsymbol{u}_i,\boldsymbol{v}_{-i})}-p_i^{(\boldsymbol{v}_i,\boldsymbol{v}_{-i})}\geq 0, \\ \forall i\in\mathcal{N},\forall(\boldsymbol{v}_i,\boldsymbol{v}_{-i})\in\mathcal{U},\forall(\boldsymbol{u}_i,\boldsymbol{v}_{-i})\in\mathcal{U}. \tag{4.4.6}$$

### 4.4.2. Optimal auction problem

The optimal auction design problem, based on the above property constraints, is formulated as a robust optimization problem, with the objective to maximize the revenue of MNO for all the valuations in set $\mathcal{U}$. Since MNO's beliefs on MSs' valuations are modeled as an uncertainty set, we focus on maximizing the worst case revenue. The network constraints, including APs' bandwidth constraints and MSs'

demand constraints, are also formulated in the optimization problem.

$$\max_{\boldsymbol{x^v},\boldsymbol{p^v}} \quad W \tag{4.4.7a}$$

$$s.t. \quad W - \sum_{i\in\mathcal{N}} p_i^{\boldsymbol{v}} \leq 0, \forall \boldsymbol{v} \in \mathcal{U} \tag{4.4.7b}$$

$$p_i^{\boldsymbol{v}} \leq \sum_{j\in\mathcal{M}} v_{ij} \cdot x_{ij}^{\boldsymbol{v}}, \forall i \in \mathcal{N}, \forall \boldsymbol{v} \in \mathcal{U} \tag{4.4.7c}$$

$$p_i^{\boldsymbol{v}} \leq B_i, \forall i \in \mathcal{N}, \forall \boldsymbol{v} \in \mathcal{U} \tag{4.4.7d}$$

$$\sum_{j\in\mathcal{M}} v_{ij} \cdot \left( x_{ij}^{(\boldsymbol{u}_i,\boldsymbol{v}_{-i})} - x_{ij}^{(\boldsymbol{v}_i,\boldsymbol{v}_{-i})} \right) + p_i^{(\boldsymbol{u}_i,\boldsymbol{v}_{-i})} - p_i^{(\boldsymbol{v}_i,\boldsymbol{v}_{-i})} \geq 0, \forall i \in \mathcal{N}, \quad \forall (\boldsymbol{v}_i,\boldsymbol{v}_{-i}) \in \mathcal{U}, \quad \forall (\boldsymbol{u}_i,\boldsymbol{v}_{-i}) \in \mathcal{U} \tag{4.4.7e}$$

$$\sum_{i\in\mathcal{N}} x_{ij}^{\boldsymbol{v}} \leq C_j, \forall j \in \mathcal{M}, \forall \boldsymbol{v} \in \mathcal{U} \tag{4.4.7f}$$

$$\sum_{j\in\mathcal{M}} x_{ij}^{\boldsymbol{v}} \leq D_i, \forall i \in \mathcal{N}, \forall \boldsymbol{v} \in \mathcal{U} \tag{4.4.7g}$$

$$x_{ij}^{\boldsymbol{v}} \geq 0, \forall i \in \mathcal{N}, \forall j \in \mathcal{M}, \forall \boldsymbol{v} \in \mathcal{U} \tag{4.4.7h}$$

$$p_i^{\boldsymbol{v}} \geq 0, \forall i \in \mathcal{N}, \forall \boldsymbol{v} \in \mathcal{U}. \tag{4.4.7i}$$

Constraint (4.4.7b) ensures the maximization of worst case revenue considering all the possible valuations in the uncertainty set $\mathcal{U}$. Constraints (4.4.7c), (4.4.7d) and (4.4.7e) correspond to IR, BF and IC properties, respectively. Constraint (4.4.7f) ensures that the bandwidth allocation should not exceed the available bandwidth of an AP. Constraint (4.4.7g) guarantees that each MS cannot obtain over-demanding bandwidth. Note that the demand $D_i$ varies over time due to the stochastic nature of

MS traffic. We consider a quasi-static network scenario [87], and analyze the auction mechanism in a data offloading period (e.g., ten seconds), during which $D_i$ remains unchanged for all $i \in \mathcal{N}$. Finally, Constraint (4.4.7i) prevents negative allocation and payment for MSs. Note that $\boldsymbol{v} \in \mathcal{U}$ is defined as $\{\boldsymbol{v} = (\boldsymbol{v}_1, \cdots, \boldsymbol{v}_m) | \boldsymbol{v}_j \in \mathcal{U}_j, \forall j \in \mathcal{M}\}$. $\boldsymbol{v}_j \in \mathcal{U}_j$ is short for the following two constraints derived from Eq. (4.3.1).

$$\sum_{i=1}^{n} v_{ij} - n \cdot \mu_j \leq \Gamma\sqrt{n} \cdot \delta_j, \forall j \in \mathcal{M},$$
$$\sum_{i=1}^{n} v_{ij} - n \cdot \mu_j \geq -\Gamma\sqrt{n} \cdot \delta_j, \forall j \in \mathcal{M}.$$

For simplicity, we use $\boldsymbol{v} \in \mathcal{U}$ to stand for the above constraints in the rest of the paper.

## 4.5. Optimal auction mechanism

In this section, we propose the optimal auction mechanism to solve the optimization problem (4.4.7) in order to determine an optimal allocation and payment rules. That is, how APs' bandwidth is shared among multiple MSs, and how much MSs are charged for using allocated bandwidth. Our optimal auction mechanism illustrated in Algorithm 4, takes as input the uncertainty set $\mathcal{U}$, MS budget vector $\boldsymbol{B}$, AP constraint vector $\boldsymbol{C}$, MS demand vector $\boldsymbol{D}$ and bid matrix $\boldsymbol{v}$, and calculates as output the real allocation matrix $\boldsymbol{a^v}$ and the payment vector $\boldsymbol{p^v}$. We will refer to Algorithm 4 as Optimal Algorithm in the rest of paper. We first introduce the details of Optimal Algorithm. Then, we show the auction properties of Optimal Algorithm.

Fig. 4.2 shows the relationship among different optimization problems in Optimal Algorithm. Optimal Algorithm consists of two phases, the phase of nominal allocation (Steps $1-7$, left column of Fig. 4.2) and the phase of final allocation (Steps $8-19$, right column of Fig 4.2). The aim of nominal allocation is to calculate the

**Algorithm 4** Optimal offloading auction mechanism

**Input:** $\mathcal{U}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}, \boldsymbol{v}, \mathcal{M}, \mathcal{N}$

**Output:** $\boldsymbol{a}^{\boldsymbol{v}}$, $\boldsymbol{p}^{\boldsymbol{v}}$

$(\boldsymbol{z}, \boldsymbol{x}^*) \leftarrow$ solving problem (4.5.1)
$(\boldsymbol{\xi}^*, \boldsymbol{\eta}^*, \boldsymbol{\lambda}^*, \boldsymbol{\theta}^*) \leftarrow$ solving problem (4.5.4)
**for** $i \in \mathcal{N}$ **do**
    **for** $j \in \mathcal{M}$ **do**
        $r_{ij}^* = \xi_j^* + (\eta_i^* + \lambda_i^* + \theta_i^*) \cdot z_{ij}$
    **end for**
**end for**
$\boldsymbol{y}^{\boldsymbol{v}} \leftarrow$ solving problem (4.5.5)
**for** $k \in \mathcal{N}$ **do**
    $\boldsymbol{y}^{\boldsymbol{v}_{-k}} \leftarrow$ solving problem (4.5.6)
**end for**
**for** $i \in \mathcal{N}$ **do**
    **for** $j \in \mathcal{M}$ **do**
        $a_{ij}^{\boldsymbol{v}} = x_{ij}^* + y_{ij}^{\boldsymbol{v}}$
    **end for**
**end for**
**for** $k \in \mathcal{N}$ **do**
    Calculate $p_k^{\boldsymbol{v}}$ using Eq. (4.5.7)
**end for**

reservation price $\boldsymbol{r}^* = \{r_{ij}\}_{i\in\mathcal{N},j\in\mathcal{M}}$ and the nominal allocation $\boldsymbol{x}^* = \{x_{ij}\}_{i\in\mathcal{N},j\in\mathcal{M}}$. MS $i$ has to bid at least $r_{ij}$ in order to use the bandwidth provided by AP $j$. $\boldsymbol{x}^*$ represents the best allocation in worst case scenario, which is part of the final allocation calculated in the phase of final allocation. Reservation price is obtained by

calculating problems (10) and (13) sequentially. Final allocation calculates the real allocation $\boldsymbol{a}^{\boldsymbol{v}}$ and final payment $\boldsymbol{p}^{\boldsymbol{v}}$ based on a specific bid matrix $\boldsymbol{v}$. The final allocation $\boldsymbol{a}^{\boldsymbol{v}} = \boldsymbol{x}^* + \boldsymbol{y}^{\boldsymbol{v}}$, where $\boldsymbol{y}^{\boldsymbol{v}} = \{y_{ij}^{\boldsymbol{v}}\}_{i\in\mathcal{N},j\in\mathcal{M}}$, called adapted allocation, denotes the best allocation for a specific bid matrix $\boldsymbol{v}$. Final allocation is based on the results of optimization problems (14) and (15). Problems (14) and (15) can be calculated independently, since their inputs don't rely the results of each other.

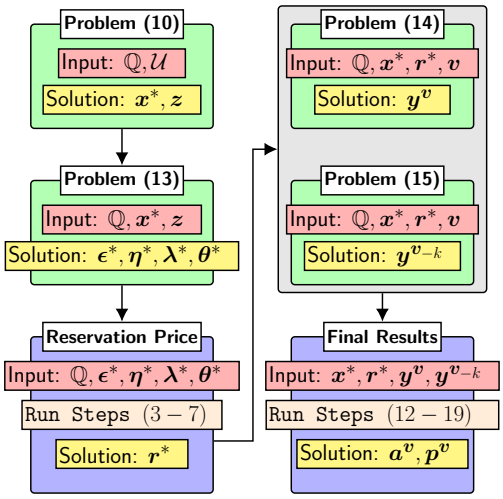



**Fig. 4.2.** Flow chart of the proposed Optimal Algorithm. The left column denotes the nominal allocation phase, while the right column denotes the final allocation phase. Note that set $\mathbb{Q} = \{\boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}, \mathcal{M}, \mathcal{N}\}$ contains the information of MSs' budgets and demands, as well as the capacity constraints of APs.

### 4.5.1. Phase of Nominal Allocation

In the phase of nominal allocation, Step 1 calculates the worst case bid matrix $\boldsymbol{z}$ and reservation price $\boldsymbol{r}^*$ by solving the bilinear optimization problem (4.5.1), where the constraints (4.5.1b), (4.5.1c) and (4.5.1d) are derived from constraints (4.4.7d), (4.4.7f) and (4.4.7g), respectively. Constraint (4.5.1e) that captures the IC and IR

properties of problem (4.4.7) is used to calculate the worst case bid matrix $\boldsymbol{z}$, under which the obtained payoff $\sum_{j\in\mathcal{M}} x_{ij}\cdot z_{ij}$ for MS $i$ is minimum. The nominal allocation $\boldsymbol{x}^*$ is a preallocation that corresponds to the worst case bid matrix $\boldsymbol{z}$.

$$\max_{\boldsymbol{x},\boldsymbol{v}} \quad \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} x_{ij}v_{ij} \tag{4.5.1a}$$

$$s.t. \quad \sum_{j\in\mathcal{M}} x_{ij}\cdot v_{ij} \le B_i, \forall i\in\mathcal{N}, \tag{4.5.1b}$$

$$\sum_{i\in\mathcal{N}} x_{ij} \le C_i, \forall j\in\mathcal{M}, \tag{4.5.1c}$$

$$\sum_{j\in\mathcal{M}} x_{ij} \le D_j, \forall i\in\mathcal{N}, \tag{4.5.1d}$$

$$\sum_{j\in\mathcal{M}} x_{ij}\cdot v_{ij} \le \sum_{j\in\mathcal{M}} x_{ij}\cdot u_{ij}, \forall \boldsymbol{u}\in\mathcal{U}, \forall i\in\mathcal{N}, \tag{4.5.1e}$$

$$\boldsymbol{x}\ge 0,\ \boldsymbol{v}\in\mathcal{U}. \tag{4.5.1f}$$

In order to obtain the reservation price $\boldsymbol{r}^*$, We first simplify the problem (4.5.1) as a linear programming problem with decision variable $\boldsymbol{x}$ by: 1) replacing variable $\boldsymbol{v}$ with constant $\boldsymbol{z}$ (obtained in Step 1); 2) replacing Constraint (4.5.1e) with Eq. (4.5.2).

$$\sum_{j\in\mathcal{M}} x_{ij}\cdot v_{ij} \le \sum_{j\in\mathcal{M}} x^*_{ij}\overline{u}^i_j, \quad \forall i\in\mathcal{N}, \tag{4.5.2}$$

where

$$\overline{\boldsymbol{u}}^i = arg\min_{\boldsymbol{u}\in\mathcal{U}} \sum_{j\in\mathcal{M}} x^*_{ij}\cdot u_{ij}, \quad \forall i\in\mathcal{N}. \tag{4.5.3}$$

Then, we can obtain the dual problem of simplified problem (4.5.1) as follows.

$$\min_{\boldsymbol{\xi},\boldsymbol{\eta},\boldsymbol{\lambda},\boldsymbol{\theta}} \quad \sum_{j\in\mathcal{M}} \xi_j C_i + \sum_{i\in\mathcal{N}} \left(\eta_i B_i + \lambda_i D_i + \theta_i \sum_{j\in\mathcal{M}} x_{ij}^* \overline{u}_j^i\right) \tag{4.5.4a}$$

$$s.t. \quad \xi_j + z_{ij}(\eta_i + \lambda_i + \theta_i) \geq z_{ij}, \forall i \in \mathcal{N}, j \in \mathcal{M}, \tag{4.5.4b}$$

$$\xi_j, \eta_i, \lambda_i, \theta_i \geq 0, \forall i \in \mathcal{N}, j \in \mathcal{M}. \tag{4.5.4c}$$

The decision variables $\boldsymbol{\xi}^* = \{\xi_j^*\}_{j\in\mathcal{M}}$, $\boldsymbol{\eta}^* = \{\eta_i^*\}_{i\in\mathcal{N}}$, $\boldsymbol{\lambda}^* = \{\lambda_i^*\}_{i\in\mathcal{N}}$ and $\boldsymbol{\theta}^* = \{\theta_i^*\}_{i\in\mathcal{N}}$ correspond to the constraints (4.5.1c), (4.5.1b), (4.5.1d) and (4.5.2), respectively. Step 2 calculates the solution of dual problem (4.5.4) used to obtain the reservation price $\boldsymbol{r}^*$ in Steps $3-7$, where $r_{ij}^*$ represents the minimum price that MS $i$ should bid in order to use bandwidth of AP $j$.

### 4.5.2. Phase of Final Allocation

In the phase of final allocation, We first calculates the adapted allocation $\boldsymbol{y^v}$ based on bid matrix $\boldsymbol{v}$ in Step 8. The adapted allocation $\boldsymbol{y^v}$ is obtained by solving the linear problem (4.5.5). The objective function (Eq. (4.5.5a)) of this problem maximizes the social welfare (i.e., the total valuations of all MSs) taking into consideration the reservation price $\boldsymbol{r}^*$. Thus, Constraints (4.5.5b), (4.5.5c) and (4.5.5d) are adjusted by considering the impact of nominal allocation $\boldsymbol{x}^*$ and reservation price $\boldsymbol{r}^*$ obtained

from the phase of nominal allocation.

$$\max_{\boldsymbol{y}^{\boldsymbol{v}}} \quad \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}} \cdot (v_{ij} - r_{ij}^*) \tag{4.5.5a}$$

$$s.t. \quad \sum_{i\in\mathcal{N}} y_{ij}^{\boldsymbol{v}} \leq C_i - \sum_{i\in\mathcal{N}} x_{ij}^*, \quad \forall j \in \mathcal{M}, \tag{4.5.5b}$$

$$\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}} \cdot v_{ij} \leq B_i - \sum_{j\in\mathcal{M}} x_{ij}^* \cdot r_{ij}^*, \quad \forall i \in \mathcal{N}, \tag{4.5.5c}$$

$$\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}} \leq D_j - \sum_{j\in\mathcal{M}} x_{ij}^*, \quad \forall i \in \mathcal{N}. \tag{4.5.5d}$$

Then we calculate the adapted allocation $\boldsymbol{y}^{\boldsymbol{v}_{-k}}$ without considering the auction participation of MS $k$ in Steps $9-11$. $\boldsymbol{y}^{\boldsymbol{v}_{-k}}$ is used to calculate the final payment of MS $k$ and is obtained by solving the linear problem (4.5.6), which is a reduced version of problem (4.5.5) by deleting the bidder $k$ from the set of bidders.

$$\max_{\boldsymbol{y}^{\boldsymbol{v}_{-k}}} \quad \sum_{i\in\mathcal{N}\setminus\{k\}}\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}} \cdot (v_{ij} - r_{ij}^*) \tag{4.5.6a}$$

$$s.t. \quad \sum_{i\in\mathcal{N}\setminus\{k\}} y_{ij}^{\boldsymbol{v}_{-k}} \leq C_i - \sum_{i\in\mathcal{N}\setminus\{k\}} x_{ij}^*, \quad \forall j \in \mathcal{M}, \tag{4.5.6b}$$

$$\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}} \cdot v_{ij} \leq B_i - \sum_{j\in\mathcal{M}} x_{ij}^* r_{ij}^*, \quad \forall i \in \mathcal{N}\setminus\{k\}, \tag{4.5.6c}$$

$$\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}} \leq D_j - \sum_{j\in\mathcal{M}} x_{ij}^*, \quad \forall i \in \mathcal{N}\setminus\{k\}. \tag{4.5.6d}$$

With $\boldsymbol{x}^*$ and $\boldsymbol{r}^*$ obtained in the phase of nominal allocation, as well as $\boldsymbol{y}^{\boldsymbol{v}}$ and $\boldsymbol{y}^{\boldsymbol{v}_{-k}}$ obtained in this phase, we can calculate the final allocation $\boldsymbol{a}^{\boldsymbol{v}}$ and the final payment $\boldsymbol{p}^{\boldsymbol{v}}$ for all $k \in \mathcal{N}$. Steps $12-16$ calculate the final allocation $\boldsymbol{a}^{\boldsymbol{v}}$ that is the sum of nominal allocation $\boldsymbol{x}^*$ and adapted allocation $\boldsymbol{y}^{\boldsymbol{v}}$. Steps $17-19$ calculate the final payment $\boldsymbol{p}^{\boldsymbol{v}}$ using Eq. (4.5.7), where $p_k^{\boldsymbol{v}}$ consists of the payment of using $\boldsymbol{a}_k^{\boldsymbol{v}}$ bandwidth and the difference between the optimal value of the objective function obtained with and without the participation of $k$. This payment scheme guarantees the

IR property of Optimal Algorithm. Furthermore, we show that Optimal Algorithm can implement an efficient auction according to Theorem 3.

$$p_k^{\boldsymbol{v}} = \sum_{j\in\mathcal{M}} y_{kj}^{\boldsymbol{v}} \cdot r_{kj}^* + \sum_{j\in\mathcal{M}} x_{kj}^* \cdot r_{kj}^* + \sum_{i\in\mathcal{N}\setminus\{k\}} \sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}}(v_{ij} - r_{ij}^*) - \sum_{i\in\mathcal{N}\setminus\{k\}} \sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}}(v_{ij} - r_{ij}^*),$$
$$\forall k \in \mathcal{N}. \tag{4.5.7}$$

**Theorem 3.** *The proposed auction mechanism illustrated in Optimal Algorithm has the properties of incentive compatibility, budget feasibility, individual rationality and worst case optimality.*

The proof of Theorem 3 is illustrated in Section 4.9.

### 4.5.3. Design Rational

We discuss the relationship of our proposed mechanism and VCG mechanism as follows.

(a) The allocation rule has a structure similar to that of VCG mechanism, where the bandwidth is allocated to a set of MUs in order to maximize a social welfare function. In Optimal Algorithm, the social welfare function is defined in Eq. (14a), which is parameterized by the reservation price $\boldsymbol{r}^*$.

(b) The payment rule, as defined in Eq. (16), is also similar to that of VCG mechanism. Each MU is charged with the opportunity cost, which is defined as the lowest amount that MU has to bid in order to win the allocation.

(c) Unlike VCG mechanism, we calculate the reservation price $\boldsymbol{r}^*$ in the worst case. Thus, $r_{ij}$ is defined as the lowest price that MNO would be willing to accept for allocating the corresponding bandwidth from AP $j$ to MS $i$. The reservation price is a threshold price; the bids less than the reservation price will not be accepted. The reservation price can accelerate the auction

process, since the set of prices that are lower than the reservation price can be discarded.

(d) Unlike VCG mechanism, we focus on the case where the payments of MUs provided by the optimal mechanism do not exceed their budget constraints. Standard mechanisms, such as VCG mechanism and its variants, are not applicable here [88, 89].

In summary, the well-known VCG mechanism is a dominant strategy mechanism, which can achieve ex-post incentive compatibility (truth-telling is a dominant strategy for every player in the game). However, VCG mechanism cannot implement the budget feasibility of the auction, which costs extra payment from MUs and decreases their payoffs. Thus, it cannot be properly used in the problem that we are solving in this paper. Compared with VCG mechanism, our proposed optimal mechanism is an incentive efficient mechanism that can maximize the expected total payoff of all MUs. Additionally, it achieves the budget feasibility of MUs. There is no extra cost paid in the auction when applying our optimal mechanism while VCG mechanism can not.

### 4.5.4. Solving Optimal Algorithm

Solving Optimal Algorithm needs to calculate one bilinear optimization problem (4.5.1) and three linear optimization problems (4.5.4), (4.5.5) and (4.5.6). The linear problems can be solved using simplex method [90]. The bilinear problem, which is the computation intensive step in the proposed mechanism, is NP-hard [91]. However, we can solve problem (10) in polynomial time to achieve global $\epsilon$-optimal solution. This is based on the observation that both inner and outer optimization problems of problem (10) are linear optimization problems. Thus, fixing the inner optimal solution, there always exists an extreme point solution to the outer problem and vice

versa. We can use Bender decomposition algorithm [92] to solve problem (10) by simply enumerating all the extreme points. Please refer to [92] for details.

## 4.6. Greedy Auction Mechanism

In this section, we turn to the concept of two-sided matching [93] to solve the data offloading problem in polynomial time. In our two-sided matching scenario, one matching partners are MSs and another matching partners are APs. Note that each AP can be matched to multiple MSs. We propose two greedy auction mechanisms: 1) *MatchingAP* scheme, i.e., it is AP which selects MSs that it will provide network connection to; 2) *MatchingMS* scheme, i.e., it is MS which selects appropriate AP for network connection. Then, we show that these two algorithms satisfy the properties of individual rationality and incentive capability.

### 4.6.1. MatchingAP Scheme

The greedy algorithm for MatchingAP scheme, illustrated in Algorithm 5, is composed of two phases, namely, allocation phase and payment phase. The allocation phase aims to select MSs for each AP that can offload mobile data traffic. The payment phase calculates the price paid by each winner by considering the maximum bid from un-winning MSs. This payment scheme is widely used in second price auction to derive a truthful bidding [94].

In Algorithm 5, Step 1 defines the allocation order for the set of APs. The sorted list $\boldsymbol{M}$ is obtained by sorting all APs participating in the auction in a non-decreasing order of bandwidth per number of covered MSs (i.e., the potential bidders for each AP). The allocation phase (Steps $3-13$) considers APs starting from the first AP in $\boldsymbol{M}$. In MatchingAP scheme, each AP can select MSs under its radio coverage area as potential bidders. Since one AP may have multiple bidders, we define an allocation rule for each AP, which states that the bidder who bids higher value has a higher

**Algorithm 5** Greedy MatchingAP Scheme

**Input:** $\boldsymbol{b}, \boldsymbol{d}, \mathcal{M}, \mathcal{N}, \boldsymbol{C}$

**Output:** $\boldsymbol{a}, \boldsymbol{p}$

1: $\boldsymbol{M} \leftarrow Sort(j \in \mathcal{M}, \frac{C_j}{|\mathcal{N}_j|}, \text{“}non - decreasing\text{”})$
2: $\boldsymbol{N} \leftarrow \mathcal{N}$
3: **while** $\boldsymbol{M} \neq \varnothing \wedge \boldsymbol{N} \neq \varnothing$ **do**
4: $\quad j \leftarrow Next(\boldsymbol{M})$, $\boldsymbol{M} \leftarrow \boldsymbol{M} \setminus \{j\}$
5: $\quad \boldsymbol{N}_j \leftarrow Sort(i \in \mathcal{N}_j, b_i, \text{“}non - decreasing\text{”})$
6: $\quad$ **while** $\sum_{i \in \mathcal{N}_j} a_{ij} \leq C_j \wedge \boldsymbol{N}_j \neq \varnothing$ **do**
7: $\quad\quad i \in Next(\boldsymbol{N}_j)$
8: $\quad\quad$ **if** $\sum_{j \in \mathcal{M}} a_{ij} = 0 \wedge d_i + \sum_{i \in \mathcal{N}_j} a_{ij} \leq C_j$ **then**
9: $\quad\quad\quad a_{ij} \leftarrow d_i$
10: $\quad\quad\quad \boldsymbol{N} \leftarrow \boldsymbol{N} \setminus \{i\}$
11: $\quad\quad$ **end if**
12: $\quad$ **end while**
13: **end while**
14: **for all** $j \in \mathcal{M}$ **do**
15: $\quad p_k \leftarrow \max_{\{i \in \mathcal{N}_j | a_{ij} = 0\}} b_i$
16: $\quad$ **for all** $i \in \mathcal{N}_j \wedge a_{ij} = d_i$ **do**
17: $\quad\quad p_i \leftarrow p_j \cdot d_i$
18: $\quad$ **end for**
19: **end for**

probability to be served, as shown in Step 5, where MSs under the coverage of AP $j$ are sorted in a non-decreasing order according to the bids submitted by MSs. The bandwidth allocation phase continues until AP $j$ has allocated all its bandwidth or it has no more MSs to be considered (Step 6). For each MS, if it is not allocated to

other APs (i.e., served by other APs) and the network demand does not exceed the bandwidth of AP $j$, it will be allocated to AP $j$ (Steps $8-9$). The payment phase (Steps $14-19$) defines the price paid by each winning MS as the maximum bid value of the set of un-winning MSs. The final payment of MS $i$ is calculated by the market clearing price $p_k$ (obtained in Step 15) and the network demand $D_i$ (Step 17).

### 4.6.2. MatchingMS Scheme

In the following, we present the greedy algorithm illustrated in Algorithm 6 for MatchingMS scheme; it has the same algorithm structure as MatchingAP scheme. It also includes allocation and payment phases. Particularly, MatchingMS scheme has same payment rule as MatchingAP scheme.

In Algorithm 6, Step 1 sorts the set of MSs by the maximum bid in a non-decreasing order. Since we aim to maximize the revenue of MNO, MSs are considered according to the allocation order obtained in $\boldsymbol{N}$. The allocation phase (Steps $3-13$) terminates until all MSs or APs are considered. In the inner loop, MS selects one AP that can provide network connection to it. APs that cover MS $i$ are sorted in the list $\boldsymbol{M}_i$ according to bandwidth (Step 5). The network selection phase continues until MS $i$ has selected one AP or it has no more APs to consider (Step 6). For each AP, if it has enough bandwidth to satisfy the demand of MS, it will be selected by MS (Steps $8-9$). The payment phase (Steps $14-19$) is the same as that in Algorithm 5.

These two algorithms satisfy the properties of individual rationality and incentive capability, since they adopted the similar auction structure used in [48]. The budget feasibility is satisfied by the fact that the payment of each MS will not be greater than its bid, i.e., if MS $i$ selects bid $b_i \leq \frac{B_i}{d_i}$, then its final payment satisfies $p_i \leq b_i$.

We next analyze the time complexity of MatchingAP and MatchingMS. We consider the time complexity of MatchingAP in three parts.

**Algorithm 6** Greedy MatchingMS Scheme

**Input:** $\boldsymbol{b}, \boldsymbol{d}, \mathcal{M}, \mathcal{N}, \boldsymbol{C}$

**Output:** $\boldsymbol{a}, \boldsymbol{p}$

1: $\boldsymbol{N} \leftarrow Sort(i \in \mathcal{N}, \max_{j \in \mathcal{M}} b_{ij}, \text{“}non - decreasing\text{”})$
2: $\boldsymbol{M} \leftarrow \mathcal{M}$
3: **while** $\boldsymbol{M} \neq \varnothing \wedge \boldsymbol{N} \neq \varnothing$ **do**
4: $\quad i \leftarrow Next(\boldsymbol{N})$, $\boldsymbol{N} \leftarrow \boldsymbol{N} \setminus \{i\}$
5: $\quad \boldsymbol{M}_i \leftarrow Sort(j \in \mathcal{M}_i, C_j, \text{“}non - decreasing\text{”})$
6: $\quad$ **while** $\sum_{j \in \mathcal{M}} a_{ij} < d_i \wedge \boldsymbol{M}_i \neq \varnothing$ **do**
7: $\quad\quad j \in Next(\boldsymbol{M}_i)$
8: $\quad\quad$ **if** $d_i + \sum_{i \in \mathcal{N}_j} a_{ij} \leq C_j$ **then**
9: $\quad\quad\quad a_{ij} \leftarrow d_i$
10: $\quad\quad\quad \boldsymbol{M} \leftarrow \boldsymbol{M} \setminus \{j\}$
11: $\quad\quad$ **end if**
12: $\quad$ **end while**
13: **end while**
14: **for all** $j \in \mathcal{M}$ **do**
15: $\quad p_k \leftarrow \max_{\{i \in \mathcal{N}_j | a_{ij} = 0\}} b_i$
16: $\quad$ **for all** $i \in \mathcal{N}_j \wedge a_{ij} = d_i$ **do**
17: $\quad\quad p_i \leftarrow p_j \cdot d_i$
18: $\quad$ **end for**
19: **end for**

- (Step 1) In MatchingAP algorithm, the construction of an AP preference list is the first step. Since there are $m$ APs, with an efficient sorting algorithm, we can get the AP preference list in time of $\mathcal{O}(m \log(m))$.

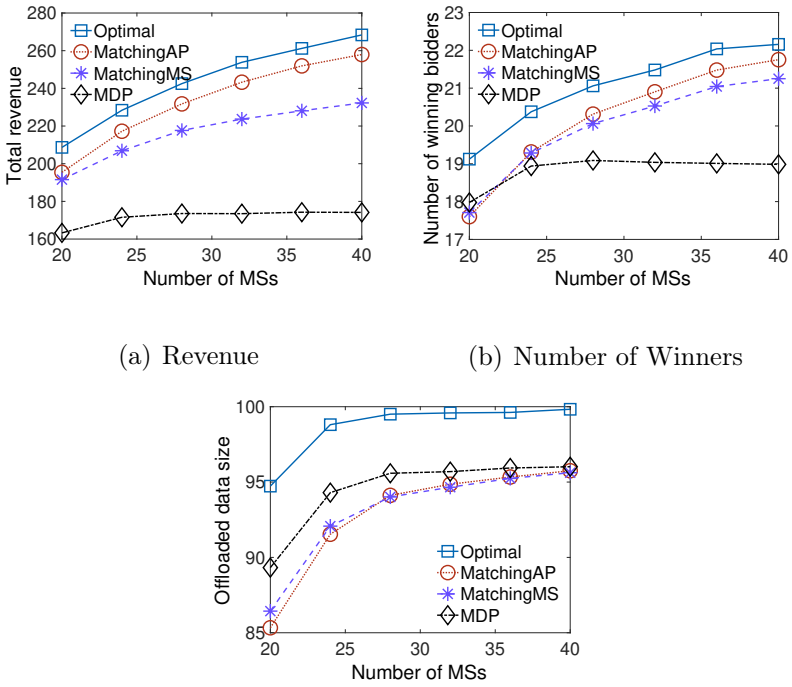
- (Steps $3-13$) We first consider the outer while loop of the algorithm. This loop will terminate when the set of APs or the set of MSs becomes empty. Thus, the maximum number of loops is $max\{m,n\}$. In step 5, we construct an MS preference list for each AP with the complexity of $\mathcal{O}(n\log(n))$. Then, we consider the inner while loop from step 6 to step 12. It is obvious that the maximum number of loops is $n$, which is the total attempts made by an AP. Indeed, assume that every summation has time complexity $\mathcal{O}(1)$. The total complexity of inner loop is $\mathcal{O}(n)$. Since step 5 has higher complexity than that of inner loop, the total complexity of outer loop is $\mathcal{O}(n\log(n)max\{m,n\})$.
- (Steps $14-19$ ) It is easy to see that the time complexity of these steps is $\mathcal{O}(mn)$.

Finally, the total complexity of the MatchingAP is $\mathcal{O}\big(max\{m\log(m),(n\log(n)max\{m,n\}),mn\}\big)$. In general cases, the number of MSs is larger than that of APs, i.e., $n>m$. Thus, the time complexity of MatchingAP is $\mathcal{O}(n^2\log(n))$, mainly due to steps $3-13$.

Following the similar analysis, we can obtain that the time complexity of MatchingMS is $\mathcal{O}\big(max\{n\log(n),(m\log(m)max\{m,n\}),mn\}\big)$. In general cases, the time complexity of MatchingMS is $\mathcal{O}\big((mn\log(m))\}\big)$.

## 4.7. Numerical Results

In this section, we evaluate the performance of the proposed auction mechanism for selling APs' bandwidth to MSs in proximity. More specifically, we aim to evaluate the impact of AP density (the number of APs), budget constraint and uncertainty set of valuation on the performance of the proposed mechanisms in order to implement an effective mobile data offloading marketplace. We first introduce the parameter settings, then we illustrate and discuss the numerical results achieved by the proposed offloading schemes.



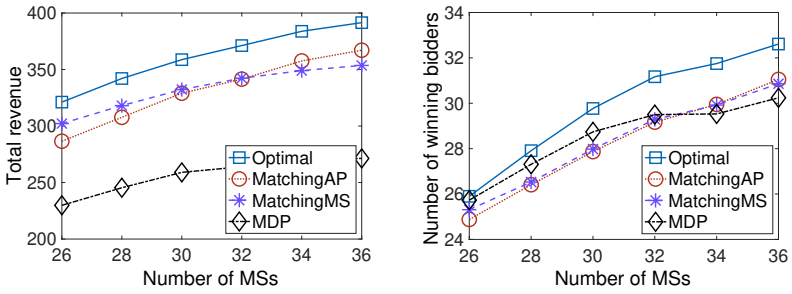

(a) Revenue (b) Number of Winners

(c) Offloaded Traffic

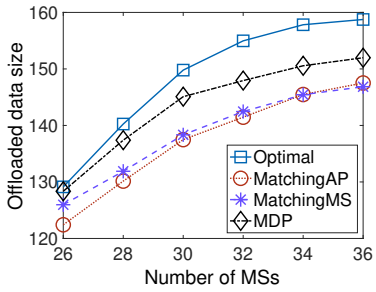
**Fig. 4.3.** Performance comparison with low AP density ($m = 5$).

We compare our proposed schemes, namely optimal scheme (Optimal Algorithm) and two greedy schemes (i.e., MatchingAP scheme and MatchingMS scheme), with the work in [95, 96], denoted as MDP scheme, since this work aims to maximize the amounts of offloaded data based on Markov Decision Process. The following performance metrics are considered in the evaluation.

- Total revenue: The total payoff of MNO.
- Offloaded traffic: The amount of traffic that can be offloaded.
- Winning MSs: The number of MSs that win the auction.




(a) Revenue (b) Number of Winners

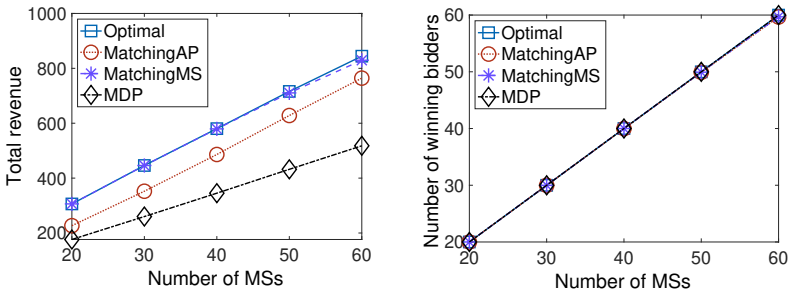



(c) Offloaded Traffic

**Fig. 4.4.** Performance comparison with medium AP density ($m = 10$).

### 4.7.1. Simulation Setup

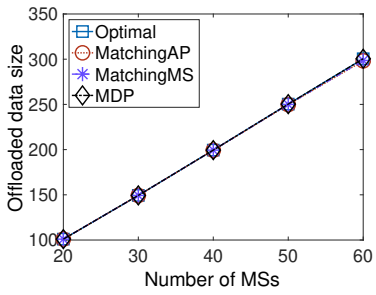
In our evaluation, we consider a measurement-based model [87], where there is an MNO represented by a macrocell BS. The number of APs and MSs, located in the coverage of BS, are chosen uniformly from the intervals [2,20] and [10,60], respectively. Unless stated otherwise, we use the information from [40, 87, 97] to set the parameters' values. Each MS submits a bid drawn from a normal distribution with mean value equal to $2\$/Mb$ and derivation equal to $1\$/Mb$. The maximum bandwidth of each AP is in the range of $[5Mbps, 40Mbps]$, while the traffic demand of each MS is in the range of $[2Mbps, 10Mbps]$. The budget of MS is selected from the range of $[10\$, 20\$]$. We use the historical bidding information, i.e., $\mu_j$ and $\delta_j$, to




(a) Revenue




(b) Number of Winners



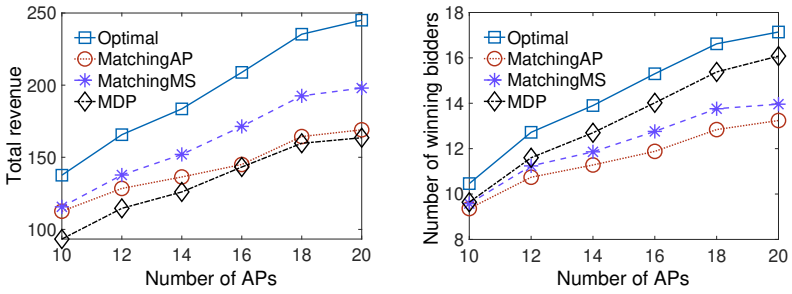

(c) Offloaded Traffic

**Fig. 4.5.** Performance comparison with high AP density ($m = 20$).

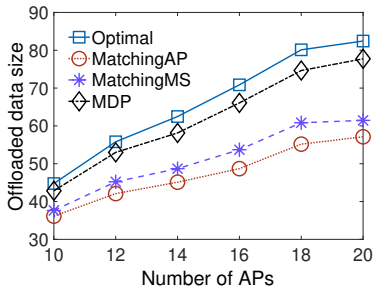
construct the uncertainty set $U_j$, $\forall j \in \mathcal{M}$. We assume that $\mu_j$ and $\delta_j$ are drawn randomly from the intervals $[1\$, 3\$]$ and $[1\$, 2\$]$, respectively. We consider the scenario with high conservativeness of historical valuations by setting $\Gamma$ to 1. To compare the performance of two greedy schemes, we define $I_m^n$ as $\frac{n}{m}$, which is the ratio between the number of MSs and the number of APs. $I_m^n$ measures the competition among MSs. Larger value of $I_m^n$ implies higher competition among MSs.

### 4.7.2. Impact of AP Density in Homogeneous Networks

In order to evaluate the impact of AP density on the performance of our proposed mechanisms, we consider three levels of AP density, i.e., $m$ is equal to 5, 10, and




(a) Revenue (b) Number of Winners

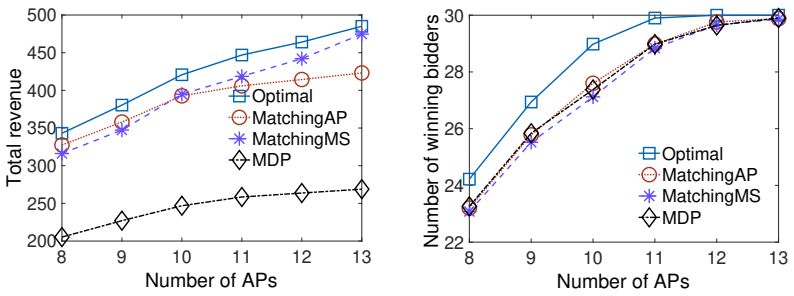



(c) Offloaded Traffic

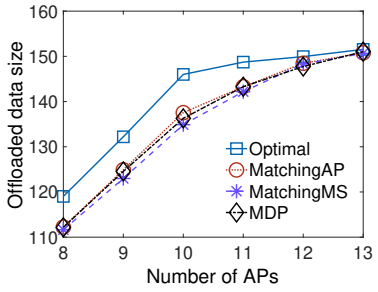
**Fig. 4.6.** Performance comparison with low capacity ($C_j = 5$ Mbps)

20, respectively. Each AP's bandwidth is set to $30Mbps$. The simulation results are shown in Figs. 4.3, 4.4 and 4.5.

We first evaluate MNO's revenue for three levels of AP density, as shown in Figs. 3(a), 4(a) and 5(a), respectively. We observe that MNO's revenue increases with the number of MSs. The larger number of MSs, the higher competition MSs may have, and consequently MNO can choose MSs with higher bid values. Moreover, optimal scheme outperforms two greedy schemes and MDP scheme in all scenarios. We further observe that MatchingAP outperforms MatchingMS in low AP density scenario (see in Fig. 3(a)), while MatchingMS outperforms MatchingAP in high AP density scenario ((see in Fig. 5(a))). This is because MatchingAP can take




(a) Revenue (b) Number of Winners

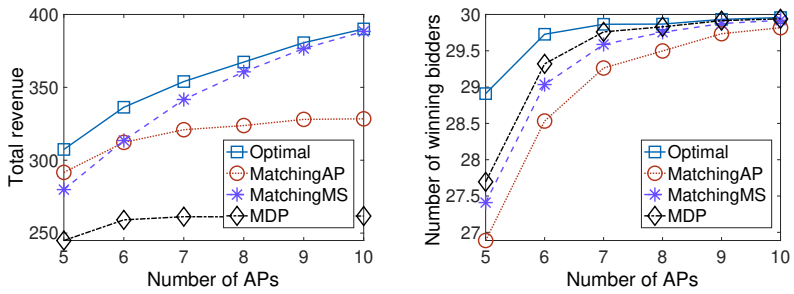



(c) Offloaded Traffic

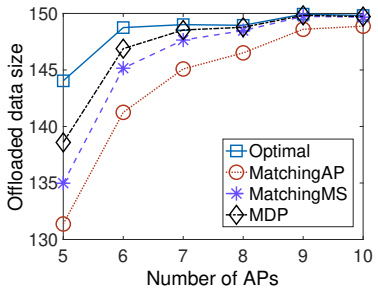
**Fig. 4.7.** Performance comparison with medium capacity ($C_j = 25$ Mbps)

advantage of the competition among MSs to obtain higher revenue. This observation can be further validated by Fig. 4(a), where MatchingAP achieves higher revenue than MatchingMS only when $n > 32$. Note that $m = 10$ in Fig. 4(a). Thus, we can obtain a threshold ratio when MatchingAP outperforms MatchingMS; that is $I_m^{n*} = 3.2$ in our settings. Since $I_m^n \geq 4 > I_m^{n*}$ in Fig. 3(a), MatchingAP achieves higher revenue than MatchingMS. In Fig. 5(a), MatchingAP achieves lower revenue than MatchingMS due to $I_m^n \leq 3 < I_m^{n*}$.

We then evaluate the number of winning bidders of different schemes. Figs. 3(b), 4(b) and 5(b) show that optimal scheme has the largest number of winning MSs; this indicates that optimal scheme can implement better fairness allocation among




(a) Revenue (b) Number of Winners




(c) Offloaded Traffic

**Fig. 4.8.** Performance comparison with high capacity ($C_j = 40$ Mbps)

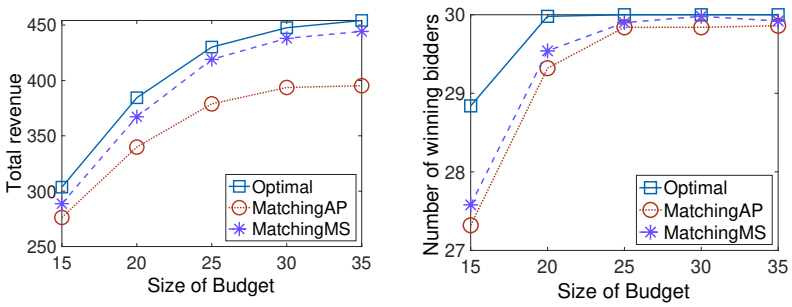
multiple MSs. Figure 5(b) illustrates that the number of winning MSs for all schemes increase linearly with the number of MSs. This is because high AP density implies enough bandwidth for traffic demand from MSs. However, it is not the same case for low AP density and medium AP density, where the total APs' bandwidth is not sufficient to support a large number of MSs. By comparing Figs. 5(a) and 5(b), we observe that MatchingMS achieves higher revenue than MatchingAP, even when two greedy schemes have the same number of winning MSs. This observation implies that two greedy schemes allocate bandwidth to different sets of MSs and MatchingMS can select the set of MSs with higher bid values in high AP density scenario.

We finally investigate how AP density affects the data offloading performance. We plot the offloaded traffic versus the number of MSs in Figs. 3(c), 4(c) and 5(c). We see that optimal scheme achieves the highest size of offloaded traffic and MDP scheme outperforms two greedy schemes, since MDP scheme aims to maximize the size of offloaded traffic. However, as illustrated in Figs. 4.3, 4.4 and 4.5, we observe that two greedy schemes achieve higher revenue than MDP scheme, even if MDP scheme can offload more data traffic. This is due to that MDP scheme does not take advantage of the competition of MSs to obtain revenue. Fig. 5(c) shows that all the schemes achieve the same size of offloaded traffic, since all traffic demands of MSs are satisfied (see Fig. 5(b)).

### 4.7.3. Impact of AP Bandwidth in Heterogeneous Networks

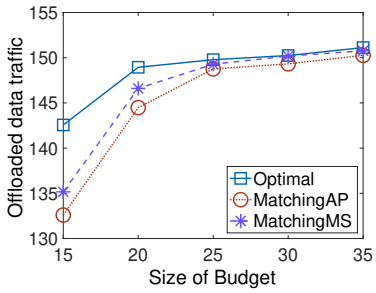
In order to evaluate the effect of AP bandwidth on the performance of our proposed schemes, we consider three levels of bandwidth $C$, namely low, medium and high, corresponding to 5, 25, and $40Mbps$, respectively. The number of MSs varies in the range of $[30, 40]$ and the demand of an MS varies in the range of [$3Mbps$,$10Mbps$]. The simulation results are shown in Figs. 4.6, 4.7 and 4.8.

Figs. 6(a), 7(a) and 8(a) show the variation of revenue with the number of APs. We observe that the revenues of all schemes increase with the number of APs. As more APs participate in the auction, MNO has more bandwidth provided to MSs, leading to higher revenue. We find that optimal scheme outperforms the other schemes in all scenarios. MatchingMS achieves higher revenue than MatchingAP when AP bandwidth is low, as shown in Figure 6(a). Note that with low bandwidth 5 Mbps, each AP can serve one MS at most, since the minimum demand of MS is 3 Mbps. In this scenario, the final payment of winning MS is the same as its bid value, since the only bidder is the winning MS itself. According to the sorting rule




(a) Revenue

(b) Number of Winners

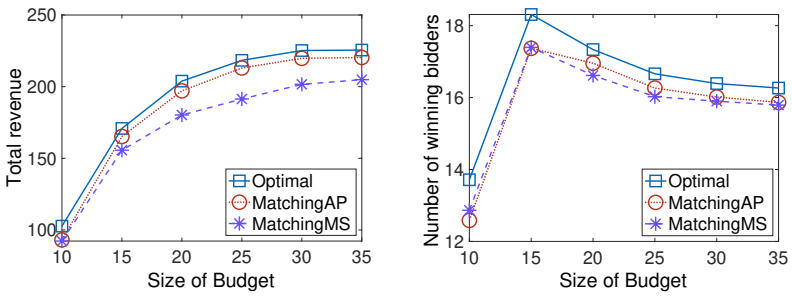



(c) Offloaded Traffic

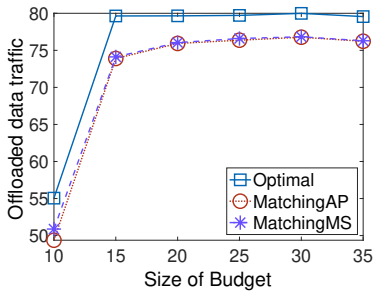
**Fig. 4.9.** Performance comparison with budget constraint ($n = 30, m = 10$)

of MatchingMS (see Algorithm 6), MS with a higher bid value has higher chance of winning the auction, resulting in a higher revenue.

However, the situation changes when AP bandwidth increases to 25 Mbps, as shown in Figure 7(a), where one AP can serve multiple MSs. In this scenario, MatchingAP achieves higher revenue than MatchingMS when $m < 10$. This is because MatchingAP selects AP based on its average bandwidth for each MS; larger AP bandwidth can serve more MSs and lead to higher competition among MSs, achieving higher revenue. While MatchingMS simply decides winning MSs based on bid values, without considering the introduction of more competition among MSs. Particularly, in the high bandwidth scenario, as shown in Fig. 8(a) where AP bandwidth




(a) Revenue (b) Number of Winners




(c) Offloaded Traffic

**Fig. 4.10.** Performance comparison with budget constraint ($n = 30, m = 5$).

is 40 Mbps, MatchingMS achieves higher revenue than MatchingAP when $m > 6$. This is because the benefit of competition among MSs is decreased with sufficient bandwidth provided by a large number of APs.

Figs. 6(b), 7(b) and 8(b) show that the number of winning MSs increase with the the number of APs, since large number of APs increases the potential of satisfying the demand of MSs. We observe that the optimal scheme has the highest number of winning MSs. The curves, as shown in Figs. 6(c), 7(c) and 8(c), follow similar trends as Figs. 6(b), 7(b) and 8(b), respectively, due to the fact that the offloaded traffic increases with the number of winning MSs.

We summarize that the optimal scheme outperforms all other schemes in all scenarios. MatchingMS outperforms MatchingAP in the following two scenarios:

- High AP density: In this case, choosing MS with higher value generates higher revenue, since its demand can always be satisfied;
- Low AP bandwidth: This leads to a special case of data offloading, where one AP is connected to at most one MS at a time.

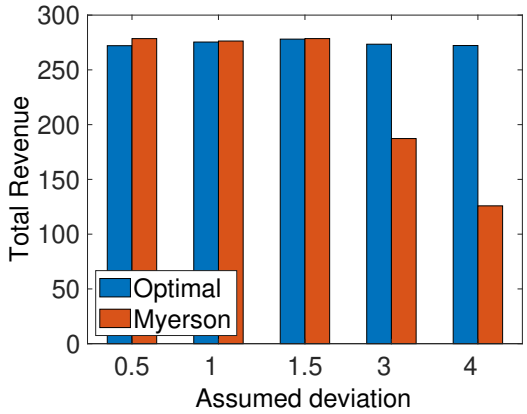



**Fig. 4.11.** Robustness comparison with different deviation

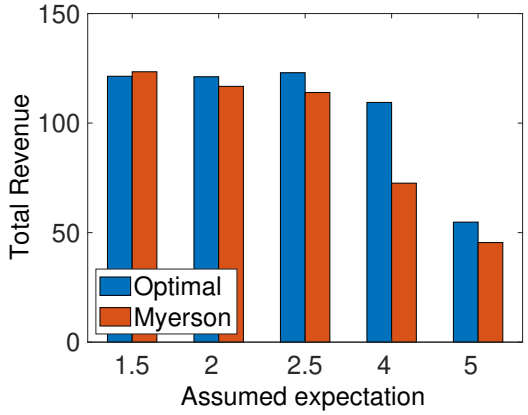



**Fig. 4.12.** Robustness comparison with different expectation

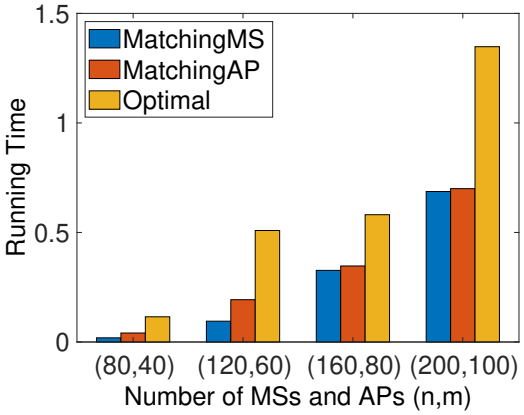



**Fig. 4.13.** Scalability comparison among different schemes

### 4.7.4. Impact of Budget Constraint

We evaluate the effect of budget constraint on the performance of our proposed schemes. We consider two scenarios based on whether the aggregate bandwidth of APs can satisfy the bandwidth demands of MSs or not. Fig. 4.9 shows the result when the aggregate bandwidth of APs is sufficient, i.e., $m = 10$, while Fig. 4.10 shows the result when the aggregate bandwidth of APs cannot satisfy all the demands from MSs, i.e., $m = 5$. We observe that the optimal method can obtain the highest revenue in all cases, as shown in Figs. 9(a) and 10(a). MatchingMS outperforms MatchingAP when the aggregate bandwidth of APs is sufficient, while MatchingAP outperforms MatchingMS when the aggregate bandwidth of APs is small.

In Fig. 4.9, we further observe that the total revenue increases with the value of budget. When $B_i \geq 25$, all MSs win the auction (see Fig. 9(b)) and bandwidth demands are satisfied (see Fig. 9(c)). Thus, the number of winning bidders and the offloaded traffic cannot increase with the value of budget when $B_i \geq 25$. However, the total revenue still increases when $B_i \geq 25$ (see Fig. 9(a)), since higher budget indicates higher valuation from MSs.

Fig. 4.10 shows the scenario where the total bandwidth demands of MSs is larger than the aggregate bandwidth of APs. As shown in Fig. 10(c), when $B_i \geq 15$, the offloaded traffic cannot increase the value of budget. This implies that all bandwidth of APs have been allocated. We observe that, when $B_i \geq 15$, the increase of budget leads to higher revenue (see Fig. 10(a)) and smaller number of winning bidders (see Fig. 10(b)). It is because higher budget increases the winning probability of MSs who have higher valuations and larger bandwidth demands. Thus, the total revenue increases while the number of winning MSs decreases when $B_i \geq 15$.

### 4.7.5. Robustness and Scalability Analysis

Now we illustrate the robustness and scalability of the proposed optimal offloading method. In order to show the robustness of the proposed method, we consider the scenario where the assumed distributions of MSs' valuations differ from the practical distributions, i.e., MNO's belief on the value of $\mu_j$ and $\delta_j$ is different from the realized value of $\mu_j^*$ and $\delta_j^*$. We compare optimal scheme with Myerson auction [98] that is an optimal auction with reservation price. Myerson auction calculates the reservation price by solving the following equation.

$$1 - F_j(v_j) = v_j * f_j(v_j), \tag{4.7.1}$$

where $F_j(.)$ and $f_j(.)$ are the cumulative distribution function and probability density function, respectively, of the probability distribution that the valuation $v_j$ is sampled from. Note that our method calculates the reservation price by solving the bilinear programming problem (4.5.1). Thus, the reservation prices obtained by Myerson auction are different from that calculated by our proposed method in most cases.

We consider a simple scenario where valuation $v_j$ follows the normal distribution with parameters $\mu_j^* = 3$ and $\delta_j^* = 2$, for all $j \in \mathcal{M}$, where $\mu_j^*$ and $\delta_j^*$ are the practical expectation and deviation of the normal distribution, respectively. The number of APs is 10 and the number of MSs is 30.

We first investigate the revenue achieved by MNO when the assumed deviation $\delta_j$ is different from the practical deviation $\delta_j^*$. To evaluate the impact of different deviations, we choose $\delta_j \in \{0.5, 1, 1.5, 3, 4\}$. Fig. 4.11 shows the total revenue obtained by Myerson auction and our optimal scheme. The larger value of $\delta_j$, the lower revenue that the Myerson auction can obtain. For example, when $\delta_j = 4$, optimal scheme outperforms Myerson auction by 56%. This is because the reservation price used in Myerson auction depends on the assumed distribution. Thus, a misspecified

(e.g., non-realistic) distribution reduces the performance of Myerson auction. Furthermore, our optimal scheme can achieve better performance due to its insensitivity to the assumed distribution.

We further evaluate how the assumed expectation $\mu_j$ affects the total revenue when using the Myerson auction and our optimal scheme. Fig. 4.12 shows the total revenue obtained by Myerson auction and optimal scheme, when the value of $\mu_j$ is chosen from $\{1.5, 2, 2.5, 4, 5\}$. We observe that both methods achieve good performance when $\mu_j < \mu_j^*$. However, the situation changes when $\mu_j > \mu_j^*$, e.g., $\mu_j = 5$, where both methods achieve lower revenue due to the misspecification of $\mu_j$.

We conclude that both Myerson auction and optimal scheme are sensitive to the misspecification of $\mu_j$. Furthermore, Myerson auction is sensitive to the misspecification of $\delta_j$, especially when $\delta_j > \delta_j^*$, while optimal scheme is insensitive to the misspecification of $\delta_j$. Thus, optimal scheme has stronger robustness than Myerson auction when the deviation of normal distribution is misspecified.

Lastly, we evaluate the running time of the proposed schemes on an Intel (R) Core(TM) i7-2620M CPU 2.70GHz processor with RAM of 16.00 GB and 64-bit Linux operating system. We measure the running time (seconds) of different schemes with different numbers of APs and MSs. In Fig. 4.13, we observe that MatchingAP achieves the lowest running time in all cases. The running time of optimal scheme increases faster than the two other schemes with the number of APs and MSs. Note that when the number of APs is 100 and the number of MSs is 200, the running time of optimal scheme is 1.34 seconds, which is a reasonable value, since the auction is executed every ten seconds.

## 4.8. Conclusion

This paper proposed a new trading marketplace where mobile operators can sell bandwidth made available by their own APs to offload data traffic of their MSs. The

offloading problem was formulated as a multi-item auction based robust optimization approach to guarantee individual rationality, incentive capability and budget feasibility for realistic scenarios in which only part of the valuation information of MSs is known to MNO. In order to solve efficiently (i.e., in polynomial time) the offloading problem for large-scale network scenarios, we also proposed two greedy algorithms. Numerical results show that the proposed schemes capture well the economical and networking essence of the problem, thus representing a promising solution to implement a trading marketplace for next-generation access networks composed of heterogeneous systems.

## 4.9. Proof of the properties of the proposed auction mechanism

In this section, we present the proof that our proposed auction mechanism has the following properties in sequence, i.e., incentive compatibility (see Lemma 7), budget feasibility (see Lemma 8), individual rationality (see Lemma 9) and worst case optimality (see Lemma 10).

**Lemma 6.** *If $\boldsymbol{z}$ and $\boldsymbol{x}^*$ are the optimal solution of problem* (4.5.1)*, then $\boldsymbol{z}$ and $\boldsymbol{x}^*$ satisfy the following conditions:*

$$\sum_{j\in\mathcal{M}} x^*_{ij}\cdot z_{ij} \le B_i, \quad \forall i\in\mathcal{N}, \tag{4.9.1}$$

$$\sum_{i\in\mathcal{N}} x^*_{ij} \le C_j, \quad \forall j\in\mathcal{M}, \tag{4.9.2}$$

$$\sum_{j\in\mathcal{M}} x^*_{ij} \ge D_i, \quad \forall i\in\mathcal{N}, \tag{4.9.3}$$

$$\sum_{j\in\mathcal{M}} x^*_{ij}\cdot z_{ij} \le \sum_{j\in\mathcal{M}} x^*_{ij}\cdot u_{ij}, \quad \forall \boldsymbol{u}\in\mathcal{U},\ \forall i\in\mathcal{N}. \tag{4.9.4}$$

$$\sum_{k\in\mathcal{N}} p^{\boldsymbol{z}}_k = \sum_{k\in\mathcal{N}}\sum_{j\in\mathcal{M}} x^*_{kj}\cdot z^*_{kj} \tag{4.9.5}$$

PROOF. *Lemma 6* can be proved by considering a reduced version of problem (4.5.1), where we set $\boldsymbol{v} = \boldsymbol{z}$. Thus, the original bilinear optimization problem (4.5.1) is reduced to a new linear optimization problem, since the only variable is $\boldsymbol{x}$. The relations (4.9.1), (4.9.2), (4.9.3) and (4.9.4) that $\boldsymbol{z}$ and $\boldsymbol{x}^*$ satisfy are derived directly from the constraints (4.5.1b), (4.5.1c), (4.5.1d) and (4.5.1e), respectively. Eq. (4.9.5) is derived from the objective function of problem (4.5.1). □

**Lemma 7.** *The proposed auction mechanism with final allocation matrix $\boldsymbol{a}^v$ and payment vector $\boldsymbol{p}^v$, satisfies the property of incentive compatibility. That is $U_k^{(\boldsymbol{v}_k,\boldsymbol{v}_{-k})} \geq U_k^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}$, which means that MS $k$ gets higher utility with truthful bidding $\boldsymbol{v}_k$.*

PROOF. We assume that the private valuation for MS $k$ is $\boldsymbol{v}_k \in \mathbb{R}^m$, and the private valuation for the rest $(n-1)$ MSs is $\boldsymbol{v}_{-k} \in \mathbb{R}^{(n-1)} \times \mathbb{R}^m$. Now if MS $k$ chooses to bid with valuation $\boldsymbol{u}_k \in \mathbb{R}^m$ instead of $\boldsymbol{v}_k$; using Eq. (4.4.1), where the utility $U_k^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}$ is the difference of payoff and payment, we obtain the utility of MS $k$ as follows:

$$U_k^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})} = \sum_{j\in\mathcal{M}} a_{kj}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})} \cdot v_{kj} - p_k^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}. \tag{4.9.6}$$

With the fact that $a_{ij}^{\boldsymbol{v}} = x_{ij}^* + y_{ij}^{\boldsymbol{v}}$ (Step 14 in Optimal Algorithm) and Eq. (4.5.7), Eq. (4.9.7) can be rewritten as

$$\begin{aligned} U_k^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})} = & \sum_{j\in\mathcal{M}} y_{kj}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})} \cdot v_{kj} + \sum_{j\in\mathcal{M}} x_{kj}^* \cdot v_{kj} \\ & - \sum_{j\in\mathcal{M}} y_{kj}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})} \cdot r_{kj}^* - \sum_{j\in\mathcal{M}} x_{kj}^* \cdot r_{kj}^* \\ & - \sum_{i\in\mathcal{N}\setminus\{k\}} \sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}} (v_{ij} - r_{ij}^*) \\ & + \sum_{i\in\mathcal{N}\setminus\{k\}} \sum_{j\in\mathcal{M}} y_{ij}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})} (v_{ij} - r_{ij}^*). \end{aligned} \tag{4.9.7}$$

By substituting the following identical equation

$$\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}(v_{ij}-r_{ij}^*) \equiv \sum_{j\in\mathcal{M}} y_{kj}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}\cdot v_{kj} - \sum_{j\in\mathcal{M}} y_{kj}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}\cdot r_{kj}^* + \sum_{i\in\mathcal{N}\setminus\{k\}}\sum_{j\in\mathcal{M}} y_{ij}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}(v_{ij}-r_{ij}^*), \tag{4.9.8}$$

into Eq. (4.9.7) and some mathematical manipulations, we have

$$\begin{aligned} U_k^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})} = &\sum_{j\in\mathcal{M}} x_{kj}^*\cdot v_{kj} - \sum_{j\in\mathcal{M}} x_{kj}^*\cdot r_{kj}^* \\ &- \sum_{i\in\mathcal{N}\setminus\{k\}}\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}}(v_{ij}-r_{ij}^*) \\ &+ \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}(v_{ij}-r_{ij}^*). \end{aligned} \tag{4.9.9}$$

Similarly, we get the utility $U_k^{(\boldsymbol{v}_k,\boldsymbol{v}_{-k})}$ when MS $k$ bid truthfully as follows:

$$\begin{aligned} U_k^{(\boldsymbol{v}_k,\boldsymbol{v}_{-k})} = &\sum_{j\in\mathcal{M}} x_{kj}^*\cdot v_{kj} - \sum_{j\in\mathcal{M}} x_{kj}^*\cdot r_{kj}^* \\ &- \sum_{i\in\mathcal{N}\setminus\{k\}}\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}}(v_{ij}-r_{ij}^*) \\ &+ \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}^{(\boldsymbol{v}_k,\boldsymbol{v}_{-k})}(v_{ij}-r_{ij}^*). \end{aligned} \tag{4.9.10}$$

By subtracting Eq. (4.9.9) from Eq. (4.9.10), we have

$$U_k^{(\boldsymbol{v}_k,\boldsymbol{v}_{-k})} - U_k^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})} = \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}^{(\boldsymbol{v}_k,\boldsymbol{v}_{-k})}(v_{ij}-r_{ij}^*) - \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}(v_{ij}-r_{ij}^*). \tag{4.9.11}$$

Note that $y_{ij}^{(\boldsymbol{v}_k,\boldsymbol{v}_{-k})}$ is the optimal solution of problem (4.5.5), while $y_{ij}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}$ is a feasible solution of problem (4.5.5). Thus, we obtain

$$\sum_{j\in\mathcal{M}} y_{ij}^{(\boldsymbol{v}_k,\boldsymbol{v}_{-k})}(v_{ij}-r_{ij}^*) \geq \sum_{j\in\mathcal{M}} y_{ij}^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}(v_{ij}-r_{ij}^*), \tag{4.9.12}$$

which demonstrates that $U_k^{(\boldsymbol{v}_k,\boldsymbol{v}_{-k})} \geq U_k^{(\boldsymbol{u}_k,\boldsymbol{v}_{-k})}$, due to Eq. (4.9.11). □

**Lemma 8.** *The proposed auction mechanism with final allocation matrix $\boldsymbol{a}^{\boldsymbol{v}}$ and payment vector $\boldsymbol{p}^{\boldsymbol{v}}$, satisfies the property of budget feasibility. That is $p_k^{\boldsymbol{v}} \leq B_k$, which implies that the payment of MS $k$ is smaller than its budget.*

Proof. We first construct an allocation matrix $\tilde{\boldsymbol{y}}^{\boldsymbol{v}} \in \mathbb{R}^{n\times m}$ based on $\boldsymbol{y}^{\boldsymbol{v}_{-k}} \in \mathbb{R}^{(n-1)\times m}$, where

$$\tilde{y}_{ij}^{\boldsymbol{v}} = \begin{cases} y_{ij}^{\boldsymbol{v}_{-k}}, & \forall i \in \mathcal{N}\setminus\{k\},\ \forall j \in \mathcal{M}, \\ 0, & i = k,\ \forall j \in \mathcal{M}. \end{cases} \tag{4.9.13}$$

Thus, we can obtain the following identical equation:

$$\sum_{i\in\mathcal{N}\setminus\{k\}}\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}}(v_{ij}-r_{ij}^*) \equiv \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} \tilde{y}_{ij}^{\boldsymbol{v}}(v_{ij}-r_{ij}^*). \tag{4.9.14}$$

Note that $\tilde{\boldsymbol{y}}^{\boldsymbol{v}}$ is a feasible solution to problem (4.5.6). That is, $\tilde{\boldsymbol{y}}^{\boldsymbol{v}}$ satisfies all the constraints of problem (4.5.6). From Eq. (4.5.6b), we obtain

$$\sum_{i\in\mathcal{N}} \tilde{y}_{ij}^{\boldsymbol{v}} \leq C_i - \sum_{i\in\mathcal{N}} x_{ij}^*, \quad \forall j \in \mathcal{M}. \tag{4.9.15}$$

From Eq. (4.5.6c), we obtain that $\forall i \in \mathcal{N}\setminus\{k\}$,

$$\sum_{j\in\mathcal{M}} \tilde{y}_{ij}^{\boldsymbol{v}} \cdot v_{ij} \leq B_i - \sum_{j\in\mathcal{M}} x_{ij}^* \cdot r_{ij}^*. \tag{4.9.16}$$

Note that

$$\sum_{j\in\mathcal{M}} \tilde{y}_{kj}^{\boldsymbol{v}} \cdot v_{kj} = 0 \leq B_i - \sum_{j\in\mathcal{M}} x_{ij}^* \cdot r_{ij}^*. \tag{4.9.17}$$

By combine Eqs. (4.9.16) and (4.9.17), we obtain

$$\sum_{j\in\mathcal{M}} \tilde{y}_{ij}^{\boldsymbol{v}} \cdot v_{ij} \leq B_i - \sum_{j\in\mathcal{M}} x_{ij}^* \cdot r_{ij}^*, \quad \forall i \in \mathcal{N}. \tag{4.9.18}$$

Similarly, we can obtain

$$\sum_{j\in\mathcal{M}} \tilde{y}_{ij}^{\boldsymbol{v}} \leq D_j - \sum_{j\in\mathcal{M}} x_{ij}^*, \quad \forall i \in \mathcal{N}. \tag{4.9.19}$$

From Eqs. (4.9.15), (4.9.18) and (4.9.19), we show that $\tilde{\boldsymbol{y}}^{\boldsymbol{v}}$ is a feasible solution to problem (4.5.5), since it satisfies the constraints (4.5.5b), (4.5.5c) and (4.5.5d). Note that $\boldsymbol{y}^{\boldsymbol{v}}$ is the optimal solution to problem (4.5.5), Thus, we obtain

$$\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} \tilde{y}_{ij}^{\boldsymbol{v}}(v_{ij} - r_{ij}^*) \leq \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}}(v_{ij} - r_{ij}^*). \tag{4.9.20}$$

By substituting Eq. (4.9.8) into Eq. (4.5.7), we have

$$p_k^{\boldsymbol{v}} = \sum_{j\in\mathcal{M}} y_{kj}^{\boldsymbol{v}} \cdot v_{kj} + \sum_{j\in\mathcal{M}} x_{kj}^* \cdot r_{kj}^* + \sum_{i\in\mathcal{N}\setminus\{k\}}\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}}(v_{ij} - r_{ij}^*) - \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}}(v_{ij} - r_{ij}^*). \tag{4.9.21}$$

By substituting Eq. (4.9.14) into Eq. (4.9.21), we have

$$p_k^{\boldsymbol{v}} = \sum_{j\in\mathcal{M}} y_{kj}^{\boldsymbol{v}} \cdot v_{kj} + \sum_{j\in\mathcal{M}} x_{kj}^* \cdot r_{kj}^* + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} \tilde{y}_{ij}^{\boldsymbol{v}}(v_{ij} - r_{ij}^*) - \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}}(v_{ij} - r_{ij}^*). \tag{4.9.22}$$

Due to Eq. (4.9.20), we have

$$p_k^{\boldsymbol{v}} \leq \sum_{j\in\mathcal{M}} y_{kj}^{\boldsymbol{v}} \cdot v_{kj} + \sum_{j\in\mathcal{M}} x_{kj}^* \cdot r_{kj}^*. \tag{4.9.23}$$

From Eq. (4.5.5c), we obtain that $p_k^{\boldsymbol{v}} \leq B_k$. □

**Lemma 9.** *The proposed auction mechanism with final allocation matrix $\boldsymbol{a}^{\boldsymbol{v}}$ and payment vector $\boldsymbol{p}^{\boldsymbol{v}}$, satisfy the property of individual rationality. That is, if MS $k$ bids truthfully with valuation vector $\boldsymbol{v}_k$, it will get a nonnegative utility $U_k^{(\boldsymbol{v}_k, \boldsymbol{v}_{-k})} \geq 0$.*

Proof. By substituting Eq. (4.9.14) into Eq. (4.9.20), we obtain

$$\sum_{i\in\mathcal{N}\setminus\{k\}}\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}}(v_{ij} - r_{ij}^*) \leq \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}}(v_{ij} - r_{ij}^*). \tag{4.9.24}$$

From Eqs. (4.9.10) and (4.9.24), we obtain

$$U_k^{(\boldsymbol{v}_k,\boldsymbol{v}_{-k})} \geq \sum_{j\in\mathcal{M}} x_{kj}^* \cdot v_{kj} - \sum_{j\in\mathcal{M}} x_{kj}^* \cdot r_{kj}^*. \tag{4.9.25}$$

From Lemma 6, we get

$$\sum_{j\in\mathcal{M}} x_{kj}^* \cdot v_{kj} \geq \sum_{j\in\mathcal{M}} x_{kj}^* \cdot z_{kj} = \sum_{j\in\mathcal{M}} x_{kj}^* \cdot r_{kj}^*. \tag{4.9.26}$$

This implies that $U_k^{(\boldsymbol{v}_k,\boldsymbol{v}_{-k})} \geq 0$. □

**Lemma 10.** *The proposed auction mechanism with final allocation matrix $\boldsymbol{a}^{\boldsymbol{z}}$ and payment vector $\boldsymbol{p}^{\boldsymbol{z}}$, satisfies the property of worst case optimality. That is $\sum_{k=1}^n \boldsymbol{p}_k^{\boldsymbol{z}} \leq \sum_{k=1}^n \boldsymbol{p}_k^{\boldsymbol{v}}$, which means that the revenue of MNO under valuation matrix $\boldsymbol{z}$ (obtained by Optimal Algorithm) is smaller than that under $\boldsymbol{v} \in \mathcal{U}$.*

Proof. We first construct an allocation matrix $\tilde{\boldsymbol{y}}^{\boldsymbol{v}} \in \mathbb{R}^{(n-1)\times m}$ based on $\boldsymbol{y}^{\boldsymbol{v}} \in \mathbb{R}^{n\times m}$, where

$$\tilde{y}_{ij}^{\boldsymbol{v}_{-k}} = y_{ij}^{\boldsymbol{v}}, \quad \forall i \in \mathcal{N}\setminus\{k\},\ \forall j \in \mathcal{M}. \tag{4.9.27}$$

Thus, we have the identical equation

$$\sum_{i\in\mathcal{N}\setminus\{k\}} \sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}}(v_{ij} - r_{ij}^*) \equiv \sum_{i\in\mathcal{N}\setminus\{k\}} \sum_{j\in\mathcal{M}} \tilde{y}_{ij}^{\boldsymbol{v}_{-k}}(v_{ij} - r_{ij}^*). \tag{4.9.28}$$

In the following, we show that $\tilde{\boldsymbol{y}}^{\boldsymbol{v}}$ is a feasible solution to problem (4.5.5). From Eq. (4.5.5b), we obtain that $\forall j \in \mathcal{M}$,

$$\sum_{i\in\mathcal{N}\setminus\{k\}} \tilde{y}_{ij}^{\boldsymbol{v}_{-k}} = \sum_{i\in\mathcal{N}\setminus\{k\}} y_{ij}^{\boldsymbol{v}} \leq C_i - \sum_{i\in\mathcal{N}\setminus\{k\}} x_{ij}^*. \tag{4.9.29}$$

From Eq. (4.5.5c), we obtain that $\forall i \in \mathcal{N}\setminus\{k\}$,

$$\sum_{j\in\mathcal{M}} \tilde{y}_{ij}^{\boldsymbol{v}_{-k}} \cdot v_{ij} = \sum_{j\in\mathcal{M}} y_{ij}^{\boldsymbol{v}} \cdot v_{ij} \leq B_i - \sum_{j\in\mathcal{M}} x_{ij}^* \cdot r_{ij}^*. \tag{4.9.30}$$

From Eq. (4.5.5d), we obtain that $\forall i \in \mathcal{N} \setminus \{k\}$,

$$\sum_{j \in \mathcal{M}} \tilde{y}_{ij}^{\boldsymbol{v}_{-k}} = \sum_{j \in \mathcal{M}} y_{ij}^{\boldsymbol{v}} \leq D_j - \sum_{j \in \mathcal{M}} x_{ij}^*. \tag{4.9.31}$$

From Eqs. (4.9.29), (4.9.30) and (4.9.31), we show that $\tilde{\boldsymbol{y}}^{\boldsymbol{v}}$ is a feasible solution to problem (4.5.6), since it satisfies constraints (4.5.6b), (4.5.6c) and (4.5.6d). Notice that $\boldsymbol{y}^{\boldsymbol{v}_{-k}}$ is the optimal solution to problem (4.5.6). Thus, we have

$$\sum_{i \in \mathcal{N} \setminus \{k\}} \sum_{j \in \mathcal{M}} \tilde{y}_{ij}^{\boldsymbol{v}_{-k}} (v_{ij} - r_{ij}^*) \leq \sum_{i \in \mathcal{N} \setminus \{k\}} \sum_{j \in \mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}} (v_{ij} - r_{ij}^*). \tag{4.9.32}$$

From Eqs. (4.9.28) and (4.9.32), we derive

$$\sum_{i \in \mathcal{N} \setminus \{k\}} \sum_{j \in \mathcal{M}} y_{ij}^{\boldsymbol{v}} (v_{ij} - r_{ij}^*) \leq \sum_{i \in \mathcal{N} \setminus \{k\}} \sum_{j \in \mathcal{M}} y_{ij}^{\boldsymbol{v}_{-k}} (v_{ij} - r_{ij}^*). \tag{4.9.33}$$

From Eqs. (4.5.7) and (4.9.33), we derive

$$p_k^{\boldsymbol{v}} \geq \sum_{j \in \mathcal{M}} y_{kj}^{\boldsymbol{v}} \cdot r_{kj}^* + \sum_{j \in \mathcal{M}} x_{kj}^* \cdot r_{kj}^*. \tag{4.9.34}$$

From Lemma 6, we know that

$$\sum_{k \in \mathcal{N}} p_k^{\boldsymbol{z}} = \sum_{k \in \mathcal{N}} \sum_{j \in \mathcal{M}} x_{kj}^* \cdot r_{kj}^*. \tag{4.9.35}$$

Finally, we get that $\sum_{k \in \mathcal{N}} p_k^{\boldsymbol{z}} \leq \sum_{k \in \mathcal{N}} p_k^{\boldsymbol{v}}$. □

With Lemmas $2 - 5$, we complete the proof of Theorem 3.

# Chapitre 5

---

# Population Game Based Workload Balancing in Mobile Edge Computing

## 5.1. Abstract

Mobile edge computing (MEC) is an emerging paradigm that provides radio access networks with augmented resources to meet the requirements of Internet of Things (IoT) services. MEC allows IoT devices to offload delay sensitive and computation intensive tasks to edge clouds deployed at base stations (BSs). Offloading tasks to edge clouds can alleviate the computing and battery limitations of IoT devices. However, task offloading in MEC for IoT may face serious transmission latency and computation latency problems with massive number of IoT devices. Moreover, some edge clouds can be overloaded due to the spatially inhomogeneous distributions of IoT tasks. To solve these problems, we investigate the workload balancing problems to minimize the transmission latency and computation latency in task offloading process while considering the limited bandwidth resources of BSs and computation resources in edge clouds. We formulate the workload balancing problem as a population game in order to analyze the aggregate offloading decisions. We analyze the aggregate offloading decisions of mobile users through evolutionary game dynamics

and show that the game always achieves a Nash equilibrium (NE). We further propose two workload balancing algorithms based on evolutionary dynamics and revision protocols. Simulation results show that our proposed workload balancing algorithms can achieve better performance than existing solutions.

**Keywords:** Task offloading, Population Game, Mobile edge computing, Internet of Things.

**Status:** Liu, Dongqing, Abdelhakim Hafid, and Lyes Khoukhi. Workload Balancing in Mobile Edge Computing for Internet of Things: A Population Game Approach. IEEE Internet of Things Journal. 2019. Submitted. This journal is based on the following conference papers:

[1] Liu, Dongqing, Abdelhakim Hafid, and Lyes Khoukhi. Population Game Based Energy and Time Aware Task Offloading for Large Amounts of Competing Users. 2018 IEEE Global Communications Conference (GLOBECOM). IEEE, 2018.

[2] Liu, Dongqing, Lyes Khoukhi, and Abdelhakim Hafid. Aggregate Offloading Decision Analysis for Mobile Edge Computing in Software Defined Network. 2019 IEEE International Conference on Communications (ICC). Accepted.

## 5.2. Introduction

IoT is proposed to equip everyday objects with electronics, software, sensors, and network connectivity, and bring the vision of a connected world into reality [19]. However, computation-intensive applications, such as e-health, automatic driving, and industrial automation, consume large amounts of computing and storage capabilities of IoT devices. These sophisticated applications have stringent requirements of computation resources and processing delay on IoT Devices (IoTDs). However, IoTDs are resource-constrained and have limited computational capacities and battery life. Running computation intensive applications on IoTDs would result in high

energy consumption and long processing delay [20]. The conflict between computation intensive applications and resource constrained IoTDs brings a significant challenge for future mobile development. MEC is envisioned to be a promising solution to address this challenge, with the objective to provide cloud computing capabilities to IoTDs through radio access network [21]. By offloading computation intensive tasks to edge cloud (or MEC server) in proximity, the local energy consumption on IoTDs can be reduced and the local processing delay may be shortened [24].

To offload computation intensive tasks to edge cloud, task-related data should be transferred between IoTDs and edge cloud through base station (BS). If BS is congested by large amounts of IoTDs choosing to offload tasks simultaneously, the quality of experience and quality of service of IoTDs will not be guaranteed [22, 23]. Moreover, facing the rapid increase of IoTDs and massive offloading tasks, the resource bottleneck of edge cloud becomes significant, since edge cloud has relatively limited resources compared to cloud computing [25]. Thus, lack of proper offloading coordination among large amounts of self-interested IoTDs may lead to serve interferences in wireless transmission and load unbalance in edge clouds. [26, 27]. As a result, design,ing an energy-efficient offloading mechanism while satisfying the processing delay requirements becomes a challenging problem, especially when large amounts of IoTDs compete for limited resources.

In this paper, we propose a population game based approach to investigate workload balancing problem for MEC in the context of IoT. Population game is envisioned as a powerful tool to model strategic interactions among large amounts of agents [29, 30]. Specifically, we model the offloading decision making problem among large amounts of competing IoTDs as a population game, wherein IoTDs are self-interested agents that make offloading decisions individually. The main contributions of this paper are summarized as follows:

- *Population game model formulation:* We formulate MEC workload balancing problem as a population game and propose an IoT Device classification model. We design an inference affected queueing model that can capture the inference among IoTDs. We use $\alpha-$ utility function to implement different kinds of workload balancing.
- *Evolutionary game dynamics analysis:* We calculate Nash Equilibrium (NE) dynamically, i.e., IoTDs can change their offloading decisions through some learning mechanism. The learning mechanism is defined as a revision protocol that allows IoTDs to adjust their offloading decisions based on decisions of other IoTDs in proximity. The evolutionary process of IoTDs' offloading strategies can be modeled by evolutionary game dynamics (i.e., a differential equation). The evolutionary game dynamics describes the variation of IoTDs' offloading decisions until an NE is obtained.
- *Workload balancing algorithms:* We propose two workload balancing algorithms, namely centralized workload balancing algorithm and decentralized workload balancing algorithm, based on the concept of evolutionary dynamics and revision protocols, respectively. We show that these algorithms can achieve an NE. Simulation results illustrate the evolutionary dynamics and show that the proposed algorithms can achieve efficient workload balancing in BSs and edge clouds.

The remainder of this paper is organized as follows. Section 5.3 introduces the system model of MEC workload balancing. Section 5.4 proposes $\alpha-$utility function based workload balancing model. Section 5.5 proposes two population game based workload balancing algorithms. Section 5.6 shows the evolutionary dynamics of three revision protocols and evaluate the performance of our proposed algorithms. Section 5.7 concludes the paper.

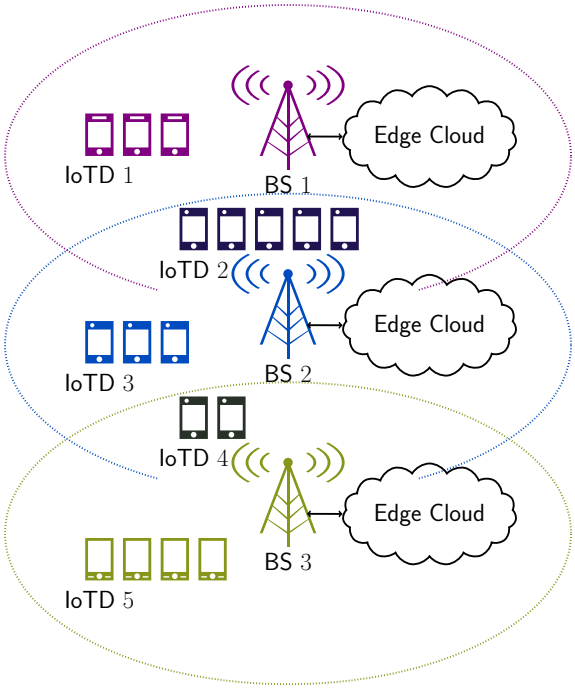



**Fig. 5.1.** MEC workload balancing model. $n$ IoTDs offload computation intensive tasks to $m$ edge clouds by BSs. The available offloading strategies depend on the location of IoTDs, e.g., IoTD1 can only offload tasks to BS1, while IoTD2 can offload tasks to BS1 or BS2, since IoTD can only access BSs in proximity.

## 5.3. System Model

We consider a cellular network consisting of a set of BSs, denoted by $\mathbb{B} = \{\mathbf{B}_m\}_{m\in\mathcal{M}}$, where $\mathcal{M} = \{1,2,\cdots,M\}$. We denote IoTDs within the coverage area of these BSs by a set $\mathbb{D} = \{\mathbf{D}_q\}_{q\in\mathcal{Q}}$, where $\mathcal{Q} = \{1,2,\cdots,Q\}$. A partition of set $\mathbb{D}$ is denoted by $\mathbb{C} = \{\mathbf{C}_n\}_{n\in\mathcal{N}}$, where $\mathcal{N} = \{1,2,\cdots,N\}$. $\mathbb{D}$ can be partitioned into $N$ subsets; each subset of $\mathbb{D}$, e.g., $\mathbf{C}_n$, is called a class in population game. IoTD makes the offloading decision (i.e., choosing optimal BS) based on network condition and task information. The aggregate offloading behaviors of IoTDs can be captured by class state and population state. We first define the class state as a distribution

of the number of IoTDs choosing different BSs, denoted by vector $\boldsymbol{a}^n = [a_m^n]_{m\in\mathcal{M}^n}$. Note that $a_m^n$ represents the number of IoTDs offloading tasks from $\mathbf{C}_n$ to $\mathbf{B}_m$. Then, we can represent the population state (i.e., the offloading decisions of all IoTDs) with the class states. The population state $\boldsymbol{a} = [\boldsymbol{a}^n]_{n\in\mathcal{N}}$ is a Cartesian product of all class states. Table 5.1 summarizes the basic notation used in the paper. We next consider the partition rule for $\mathbb{C}$.

### 5.3.1. IoT Device Classification

We classify IoTDs into different classes according to their locations and task information. The location of $\mathbf{D}_q$ is denoted by $L_q$, where $L_q$ belongs to Cartesian plane $\mathbb{R}^2$. Assume that the length of data flow from $\mathbf{D}_q$ follows an exponential distribution with average value $B_q$, the length of computation flow from $\mathbf{D}_q$ follows an exponential distribution with average value $E_q$ and the task generation rate from $\mathbf{D}_q$ follows a Poisson Point Process with rate $\lambda_q$ [99, 100]. We use $\mathbb{J}_q \triangleq (B_q, E_q, \lambda_q)$ to denote the task of $\mathbf{D}_q$. More specifically, $B_q$ represents size of data including computational input data and execution codes. $E_q$ denotes the required CPU cycles to execute task $\mathbb{J}_q$. Based on IoTD's location and task information, class $\mathbf{C}_n$ is defined as follows.

$$\mathbf{C}_n = \{q \in \mathcal{Q} \mid \frac{E_q}{B_q} = C^n, \lambda_q = \lambda^n, \text{ and } L_q = L^n\}. \tag{5.3.1}$$

IoTDs from class $\mathbf{C}_n$ should satisfy three conditions characterized by $C^n$, $\lambda^n$ and $L^n$. $C^n$ requires that IoTDs in $\mathbf{C}_n$ should have same computational density per unit size of data. $\lambda^n$ ensures that IoTDs in $\mathbf{C}_n$ have the same task generation rate. $L^n$ requires that IoTDs in $\mathbf{C}_n$ should be in the same location, since IoTDs in the same location face similar network environment (e.g., network traffic and link capacity). Note that IoTD can only offload mobile tasks to BSs in proximity. Let $\mathbb{B}^n = \{\mathbf{B}_m\}_{m\in\mathcal{M}^n}$ denote the available BSs that can execute tasks for $\mathbf{C}_n$. The

**Tab. 5.1.** Basic Notation

| | |
|---|---|
| **Elements** | |
| $\mathbf{B}_m$ | Base Station $m$ |
| $\mathbf{E}_m$ | Edge Cloud $m$, collocated with $\mathbf{B}_m$ |
| $\mathbf{D}_q$ | Internet of Thing Device $q$ |
| $\mathbf{C}_n$ | Class $n$, Set of Internet of Thing Devices |
| $\mathbf{D}_n^*$ | The IoTD with minimum data size in Class $n$ |
| $\mathbf{D}_n^m$ | An IoTD in Class $n$ choosing $\mathbf{B}_m$ |
| **Sets** | |
| $\mathbb{B} = \{\mathbf{B}_m\}_{m\in\mathcal{M}}$ | Set of Base Stations, $\lvert\mathcal{M}\rvert = M$ |
| $\mathbb{D} = \{\mathbf{D}_q\}_{q\in\mathcal{Q}}$ | Set of Internet of Thing Devices, $\lvert\mathcal{Q}\rvert = Q$ |
| $\mathbb{C} = \{\mathbf{C}_n\}_{n\in\mathcal{N}}$ | Set of Classes or Partitions of $\mathbb{D}$, $\lvert\mathcal{N}\rvert = N$ |
| $\mathbb{B}^n = \{\mathbf{B}_m\}_{m\in\mathcal{M}^n}$ | Subset of Base Stations that are available for $\mathbf{C}_n$ |
| **Parameters** | |
| $L_q \in \mathbb{R}^2$ | The location of $\mathbf{D}_q$ |
| $B_q$ | The length of data flow from $\mathbf{D}_q$ |
| $E_q$ | The length of computation flow from $\mathbf{D}_q$ |
| $\lambda_q$ | Task generation rate from $\mathbf{D}_q$ |
| $Z^n$ | The number of IoTDs in $\mathbf{C}_n$, $\lvert\mathbf{C}_n\rvert = Z^n$ |
| $\hat{Z}^n$ | The number of IoTDs in $\mathbf{C}_n$ after replacement |
| $C^n$ | Computational density per unit size of data in $\mathbf{C}_n$ |
| $L^n \in \mathbb{R}^2$ | The location of $\mathbf{C}_n$ |
| $B^n$ | The length of data flow from $\mathbf{D}_n^*$ |
| $E^n$ | The length of computation flow from $\mathbf{D}_n^*$ |
| $\lambda^n$ | Task generation rate from $\mathbf{D}_n^*$ |
| $\theta^n$ | Traffic generation density from $\mathbf{D}_n^*$ |
| $\eta^n$ | Computation generation density from $\mathbf{D}_n^*$ |
| **Variables** | |
| $a_m^n$ | The number of IoTDs in $\mathbf{C}_n$ choosing $\mathbf{B}_m$ |
| $\boldsymbol{a}^n$ | Class state vector $\boldsymbol{a}^n = [a_m^n]_{m\in\mathcal{M}^n}$ |
| $\boldsymbol{a}$ | Population state vector $\boldsymbol{a} = [\boldsymbol{a}^n]_{n\in\mathcal{N}}$ |

number of IoTDs in $\mathbf{C}_n$ is denoted by $Z^n$. $\sum_{m\in\mathcal{M}^n} a_m^n = Z^n$ and $\sum_{n\in\mathcal{N}} Z^n = Q$ ensures that the class size and population size remains stable. As illustrated in Fig. 5.2, all IoTDs in a same class fall in a same line; the slope of the line denotes the computational density of the class.

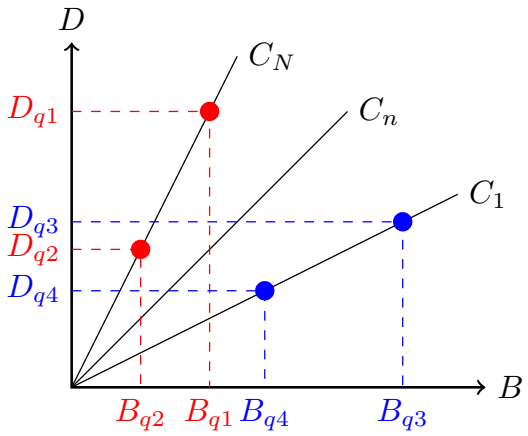



**Fig. 5.2.** Illustration of IoTDs classification model. We consider that four IoTDs $\mathbf{D}_{q_1}$, $\mathbf{D}_{q_2}$, $\mathbf{D}_{q_3}$ and $\mathbf{D}_{q_4}$ are located in the same place, i.e., $L_{q_1} = L_{q_2} = L_{q_3} = L_{q_4}$. Due to different computational density per size of data, e.g., $C^1 = \frac{E_{q_3}}{B_{q_3}} = \frac{E_{q_4}}{B_{q_4}}$ and $C^N = \frac{E_{q_1}}{B_{q_1}} = \frac{E_{q_2}}{B_{q_2}}$, IoTDs are classified into two classes. Note that each line can represent a class and the slope of the line denotes the computational density of the class. Thus, $q_1$,$q_2 \in \mathbf{C}_N$ and $q_3$,$q_4 \in \mathbf{C}_1$.

Population game requires that all IoTDs from the same class are homogeneous. Previous classification cannot preserve this property, since two IoTDs may have different data sizes even if they are in the same class. In order to solve this problem, we need to reconsider IoTDs' tasks and recalculate the class size. The basic idea is to divide the larger size data (and CPU cycles) into a number of minimum size data (and CPU cycles). We first select the IoTD with minimum data size (and corresponding minimum CPU cycles) in $\mathbf{C}_n$ as a benchmark, denoted as $\mathbf{D}_n^*$. Then,

we consider that all the other IoTDs are composed of multiple $\mathbf{D}_n^*$s. For example, if the data size of $\mathbf{D}'_n$ is 1.5 times of that $\mathbf{D}_n^*$, then $\mathbf{D}'_n$ can be replaced by 1.5 $\mathbf{D}_n^*$s. Since all IoTDs in a class are replaced by $\mathbf{D}_n^*$, the homogeneous property is preserved. We can recalculate the class size and population size as follows:

$$\hat{Z}^n = \sum_{q \in \mathbf{C}_n} \frac{B_q}{B^n}, \qquad \hat{Q} = \sum_{n \in \mathcal{N}} \hat{Z}^n, \tag{5.3.2}$$

where $B^n = min_{q \in \mathbf{C}_n} B_q$ is the data size of $\mathbf{D}_n^*$. $\hat{Z}^n$ and $\hat{Q}$ denote the class size and population size after replacement, respectively. Note that $\hat{Z}^n$ may not be integer while $Z^n$ is integer. Unless otherwise specified, we will use this new population model in the rest of the paper.

### 5.3.2. Task Execution Model

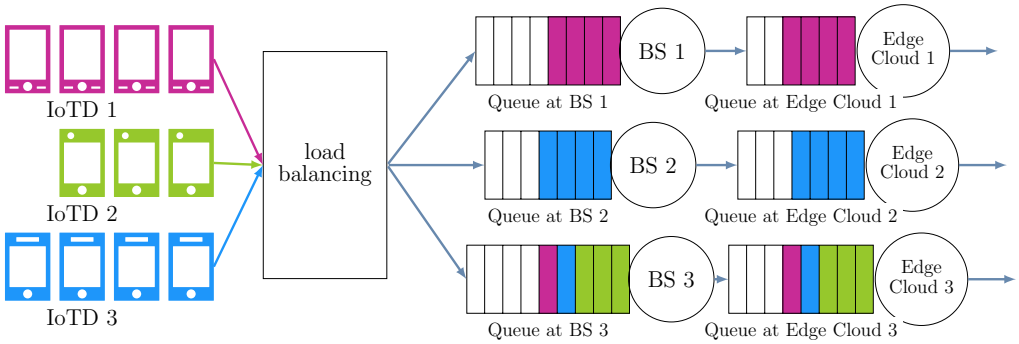



**Fig. 5.3.** Queueing model for MEC workload balancing. This figure illustrates that three classes of IoTDs offload tasks to three edge clouds. The processing delay consists of transmission delay in BS and computation delay in edge cloud. Load balancing mechanism can shorten the processing delay.

Tasks generated by IoTDs will be transferred to BSs and then executed in the corresponding edge cloud, as shown in Fig. 5.3. We assume that IoTDs in $\mathbf{C}_n$ generate tasks according to a Poisson Point Process with rate $\lambda^n$. We further assume

that the data size and CPU cycles of $\mathbf{C}_n$ follow the exponential distributions with average values of $B^n$ and $E^n$, respectively. Note that $B^n$ and $E^n$ correspond to the data size and CPU cycles of $\mathbf{D}_n^*$. We define the traffic generation density of $\mathbf{D}_n^*$ as $\theta^n = \lambda^n B^n$. Thus, the traffic generation density of $\mathbf{C}_n$ is $\hat{Z}^n \theta^n$, which is simply the multiplication of the number of IoTDs and the traffic generation density of $\mathbf{D}_n^*$. Similarly, we can define the computation generation density of $\mathbf{D}_n^*$ as $\eta^n = \lambda^n E^n$.

We first introduce the communication model between IoTDs in $\mathbf{C}_n$ and $\mathbf{B}_m$. We consider that IoTDs in the same class have the same data rate, while IoTDs in different classes can have different data rates. The data rate between $\mathbf{D}_n^*$ and $\mathbf{B}_m$ is defined as follows:

$$R_m^n(\boldsymbol{a}) = \frac{W_m}{a_m^n} \log_2 \left( 1 + \frac{a_m^n P_l^n H_m^n}{\delta + \sum_{k \in \mathcal{N}_m^n} a_m^k P_l^k H_m^k} \right), \tag{5.3.3}$$

where

$$\mathcal{N}_m^n = \left\{ k \in \mathcal{N} \setminus \{n\} : m \in \mathcal{M}^k \right\}, \quad a_m^n \neq 0.$$

$W_m$ denotes the total bandwidth of $\mathbf{B}_m$. $P_l^n$ and $P_l^k$ represent the average transmission power of IoTDs in $\mathbf{C}_n$ and $\mathbf{C}_k$, respectively. $H_m^n$ and $H_m^k$ are the average channel gain between $\mathbf{B}_m$ and IoTDs in $\mathbf{C}_n$ and $\mathbf{C}_k$, respectively. We use $\delta$ to denote the noise power. The interference from other classes is $\sum_{k \in \mathcal{N}_m^n} a_m^k P_l^k H_m^k$, where $\mathcal{N}_m^n$ denotes the set of classes whose available BSs include $\mathbf{B}_m$. We don't consider interference among IoTDs in a same class, since these IoTDs may come from a single IoTD before replacement. If no IoTD in $\mathbf{C}_n$ selects $\mathbf{B}_m$, i.e., $a_m^n = 0$, there is no need to calculate $R_m^n(\boldsymbol{a})$. Thus, we assume that $a_m^n \neq 0$ in Eq. (5.3.3).

The traffic load density of $\mathbf{B}_m$ serving an IoTD from $\mathbf{C}_n$ is defined as $\dot{T}_m^n(\boldsymbol{a}) = \frac{\theta^n}{R_m^n(\boldsymbol{a})}$, and denotes the time fraction of $\mathbf{B}_m$ serving $\mathbf{C}_n$. The utilization of $\mathbf{B}_m$ is the

aggregation of traffic load density of $\mathbf{B}_m$ serving $\mathbf{C}_n$, which is defined as

$$\begin{aligned}\dot{\rho}_m(\boldsymbol{a}) &= \sum_{n\in\mathcal{N}} \dot{T}_m^n(\boldsymbol{a}) a_m^n = \sum_{n\in\mathcal{N}} \frac{\theta^n a_m^n}{R_m^n(\boldsymbol{a})} \\ &= \sum_{n\in\mathcal{N}} \frac{\theta^n (a_m^n)^2}{W_m \log_2\left(1+\frac{a_m^n P_l^n H_m^n}{\delta+\sum_{k\in\mathcal{N}_m^n} a_m^k P_l^k H_m^k}\right)}.\end{aligned} \tag{5.3.4}$$

We then introduce the computation model when offloading tasks from $\mathbf{C}_n$ to $\mathbf{E}_m$. The computation load density of $\mathbf{E}_m$ (connected to $\mathbf{B}_m$ ) serving an IoTD from $\mathbf{C}_n$ is defined as $\ddot{T}_m^n = \frac{\eta^n}{F_m}$, where $F_m$ is the computational capability (in CPU cycles/second) of $\mathbf{E}_m$. $\ddot{T}_m^n$ represents the time fraction of $\mathbf{E}_m$ serving $\mathbf{C}_n$. The utilization of $\mathbf{E}_m$ is the aggregation of computation load density of $\mathbf{E}_m$ serving $\mathbf{C}_n$, which is defined as

$$\ddot{\rho}_m = \sum_{n\in\mathcal{N}} \ddot{T}_m^n a_m^n = \sum_{n\in\mathcal{N}} \frac{\eta^n a_m^n}{F_m}. \tag{5.3.5}$$

### 5.3.3. Workload Balancing Model

The utilization levels of $\mathbf{B}_m$ and $\mathbf{E}_m$ are described by $\dot{\rho}_m(\boldsymbol{a})$ and $\ddot{\rho}_m$, respectively. In order to implement different load balancing for BSs and edge clouds, we take advantage of $\alpha$-fair utility function [101] that we will maximize as follows:

$$\mathbf{T}(\alpha,\boldsymbol{\rho}) = \begin{cases} -\sum\limits_{m\in\mathcal{M}} \dfrac{(1-\rho_m)^{1-\alpha}-1}{\alpha-1}, & \alpha\neq 1, \\ \\ -\sum\limits_{m\in\mathcal{M}} \ln\left(\dfrac{1}{1-\rho_m}\right), & \alpha = 1, \end{cases} \tag{5.3.6}$$

where $\boldsymbol{\rho} = \{\rho_m\}_{m\in\mathcal{M}}$ denotes the utilization status of BSs (when $\rho_m = \dot{\rho}_m(\boldsymbol{a})$) or edge clouds (when $\rho_m = \ddot{\rho}_m$). The load balancing factor $\alpha$ can have four different values resulting in four load balancing policies. For example, if $\alpha = 0$, then $\mathbf{T}(\alpha,\boldsymbol{\rho}) = \sum_{m\in\mathcal{M}} \rho_m$. The offloading decision is only based on IoTDs' perspective and this

policy is called rate-optimal policy. If $\alpha = 2$, then $\mathbf{T}(\alpha, \boldsymbol{\rho}) = -\sum_{m\in\mathcal{M}} \frac{\rho_m}{1-\rho_m}$. Note that $\frac{\rho_m}{1-\rho_m}$ can represent the length of queue in $\mathbf{B}_m$ or $\mathbf{E}_m$. The negative sign is used to maximize $\alpha$-fair utility function, since we aim to minimize the total length of queue $\sum_{m\in\mathcal{M}} \frac{\rho_m}{1-\rho_m}$. When $\alpha = 2$, $\mathbf{T}(\alpha, \boldsymbol{\rho})$ is called delay-optimal policy. Moreover, $\alpha = 1$ and $\alpha = \infty$ denote throughput-optimal policy and equalizing-load policy, respectively [101]. The authors in [102] show that $\alpha \geq 0$ can take more values except for the above cases. Thus, we can implement many kinds of load balancing in BSs and edge clouds by using different values of $\alpha$.

## 5.4. Population Game Based Workload Balancing

In this section, we first propose a social welfare maximization problem that can implement efficient load balancing in BSs and edge clouds. Then, we define the payoff function of population game and introduce three basic evolutionary dynamics that can capture the evolution of population state.

### 5.4.1. Social Welfare Maximization

Our social welfare maximization aims to jointly implement load balancing in BSs and edge clouds and is defined as follows:

$$\begin{aligned}
\max \quad & \mathbf{T}(\dot{\alpha},\ddot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a}), \ddot{\boldsymbol{\rho}}) = && \mathbf{T}(\dot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a})) + \xi\mathbf{T}(\ddot{\alpha}, \ddot{\boldsymbol{\rho}}) && (5.4.1)\\
\text{s.t.} \quad & \dot{\rho}_m(\boldsymbol{a}) < 1 && \forall m \in \mathcal{M} && (5.4.2)\\
& \ddot{\rho}_m < 1 && \forall m \in \mathcal{M} && (5.4.3)\\
& \sum_{m\in\mathcal{M}^n} a_m^n = \hat{Z}^n && \forall n \in \mathcal{N}. && (5.4.4)
\end{aligned}$$

The $\dot{\alpha}$-fair utility function for BSs is denoted by $\mathbf{T}(\dot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a}))$. By replacing the input parameters of Eq. (5.3.6), we obtain

$$\mathbf{T}(\dot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a})) = \begin{cases} -\sum\limits_{m \in \mathcal{M}} \dfrac{(1-\dot{\rho}_m(\boldsymbol{a}))^{1-\dot{\alpha}} - 1}{\dot{\alpha} - 1}, & \dot{\alpha} \neq 1, \\ \\ -\sum\limits_{m \in \mathcal{M}} \ln\left(\dfrac{1}{1-\dot{\rho}_m(\boldsymbol{a})}\right), & \dot{\alpha} = 1. \end{cases} \tag{5.4.5}$$

Similarly, we can obtain the $\ddot{\alpha}$-fair utility function for edge clouds as follows:

$$\mathbf{T}(\ddot{\alpha}, \ddot{\boldsymbol{\rho}}) = \begin{cases} -\sum\limits_{m \in \mathcal{M}} \dfrac{(1-\ddot{\rho}_m)^{1-\ddot{\alpha}} - 1}{\ddot{\alpha} - 1}, & \ddot{\alpha} \neq 1, \\ \\ -\sum\limits_{m \in \mathcal{M}} \ln\left(\dfrac{1}{1-\ddot{\rho}_m}\right), & \ddot{\alpha} = 1. \end{cases} \tag{5.4.6}$$

Since we jointly optimize the load balancing for BSs and edge clouds, $\xi > 0$ is a trade-off between these two objectives. Larger $\xi$ implies higher priority in load balancing for edge clouds.

Constraints (5.4.2) and (5.4.3) indicate that the utilization of BSs and edge clouds can not exceed the maximum bandwidth and computational capability, respectively. Constraint (5.4.4) requires that the number of IoTDs in a class remains stable. Instead of solving this optimization problem directly, we propose a population game based method to solve the load balancing problem.

### 5.4.2. Population Game Formulation

In order to solve the optimization problem (5.4.1), we describe IoTDs' offloading decisions as a population state $\boldsymbol{a}$. The core of population game is the so-called payoff function. Payoff function defines IoTDs' payoffs based on a population state and is composed of a collection of marginal payoff functions, i.e., $\boldsymbol{F}(\boldsymbol{a}) = \{F_m^n(\boldsymbol{a}) : m \in \mathcal{M}^n, n \in \mathcal{N}\}$, $F_m^n(\boldsymbol{a})$ denotes the payoff of IoTD in $\mathbf{C}_n$ offloading task to $\mathbf{B}_m$ and is

defined as follows:

$$F_m^n(\boldsymbol{a}) = -\left[\frac{\eta^n \xi}{F_m(1-\ddot{\rho}_m)^{\ddot{\alpha}}} + \frac{\theta^n\Big(2 - g(\widehat{SINR}_m^n)\Big)}{R_m^n(\boldsymbol{a})(1-\dot{\rho}_m(\boldsymbol{a}))^{\dot{\alpha}}}\right], \tag{5.4.7}$$

where

$$g(\widehat{SINR}_m^n) = \frac{\widehat{SINR}_m^n}{(\widehat{SINR}_m^n + 1)\ln(\widehat{SINR}_m^n + 1)}, \tag{5.4.8}$$

$$\widehat{SINR}_m^n = \frac{a_m^n P_l^n H_m^n}{\delta + \sum_{k \in \mathcal{N}_m^n} a_m^k P_l^k H_m^k}. \tag{5.4.9}$$

$\widehat{SINR}_m^n$ denotes the Signal-to-Interference-plus-Noise Ratio (SINR) between $\mathbf{B}_m$ and $\mathbf{D}_m^n$. $g(\widehat{SINR}_m^n)$, called inference function, represents the effect of $\widehat{SINR}_m^n$ on load balancing among BSs. To better understand the definition of $F_m^n(\boldsymbol{a})$, we consider one simple case where $\dot{\alpha} = \ddot{\alpha} = 0$. In this case,

$$\hat{F}_m^n(\boldsymbol{a}) = -\left[\frac{\eta^n \xi}{F_m} + \frac{\theta^n\Big(2 - g(\widehat{SINR}_m^n)\Big)}{R_m^n(\boldsymbol{a})}\right].$$

Recall that $\dot{T}_m^n(\boldsymbol{a}) = \frac{\theta^n}{R_m^n(\boldsymbol{a})}$ denotes the time fraction of $\mathbf{B}_m$ serving $\mathbf{C}_n$ and $\ddot{T}_m^n = \frac{\eta^n}{F_m}$ is the time fraction of $\mathbf{E}_m$ serving $\mathbf{C}_n$ (see Section 5.3.2). We further obtain

$$\hat{F}_m^n(\boldsymbol{a}) = -\left[\xi \ddot{T}_m^n + \dot{T}_m^n(\boldsymbol{a})\Big(2 - g(\widehat{SINR}_m^n)\Big)\right],$$

which has a similar structure of Eq. (5.4.1) except that the effect of SINR is obvious now. $\hat{F}_m^n(\boldsymbol{a})$ is the payoff of IoTDs from $\mathbf{C}_n$ choosing $\mathbf{B}_m$ and $\mathbf{E}_m$ (or the payoff of $a_m^n$ ). We observe that the time for transmission is affected by SINR and the loads of BSs and edge clouds are not considered in this case. When $\dot{\alpha} > 0$ and $\ddot{\alpha} > 0$, the load of BSs and edge clouds will affect the payoff of IoTDs, since $\ddot{\rho}_m$ and $\dot{\rho}_m(\boldsymbol{a})$ will be reserved in payoff function. Moreover, we propose theorem 4 to analyze the effect of $\Big(2 - g(\widehat{SINR}_m^n)\Big)$ in payoff function.

**Theorem 4.** *Time fraction* $\hat{T}_m^n(\boldsymbol{a}) = \frac{\theta^n}{R_m^n(\boldsymbol{a})}\Big(2 - g(\widehat{SINR}_m^n)\Big)$ *increases from* $\frac{\theta^n}{R_m^n(\boldsymbol{a})}$ *to* $\frac{2\theta^n}{R_m^n(\boldsymbol{a})}$*, when* $\widehat{SINR}_m^n$ *increases from* 0 *to* $+\infty$*.*

Theorem 4 implies that when $\widehat{SINR}_m^n = 0$, $\hat{T}_m^n(\boldsymbol{a}) = \dot{T}_m^n(\boldsymbol{a})$. As $\widehat{SINR}_m^n$ increases, time fraction $\hat{T}_m^n(\boldsymbol{a})$ increases. This is because higher $\widehat{SINR}_m^n$ implies higher data rate from IoTDs in $\mathbf{C}_n$; thus resulting in higher bandwidth utilization of $\mathbf{B}_m$. However, $\hat{T}_m^n(\boldsymbol{a})$ should be less than $\frac{2\theta^n}{R_m^n(\boldsymbol{a})}$, even if the transmission power is much larger than inference and noise power. The proof of Theorem 1 is given in Appendix 5.8.

**Definition 3.** *A population game $\boldsymbol{F} : \mathbb{R}_+^{N\times M} \rightarrow \mathbb{R}^{N\times M}$ is a potential game if there exists a continuously differentiable function $\mathbf{T} : \mathbb{R}_+^{N\times M} \rightarrow \mathbb{R}$, called a potential function, satisfying $\nabla\mathbf{T}(\boldsymbol{a}) = \boldsymbol{F}(\boldsymbol{a})$ for all $\boldsymbol{a} \in \mathbb{R}_+^{N\times M}$, or $\frac{\partial\mathbf{T}}{\partial a_m^n}(\boldsymbol{a}) = F_m^n(\boldsymbol{a})$ for all $m \in \mathcal{M}$ and $n \in \mathcal{N}$.*

Definition 1 shows that the partial derivatives of the potential function are the payoff functions of the population game.

**Theorem 5.** *Our proposed population game $\boldsymbol{F}(\boldsymbol{a}) = \{F_m^n(\boldsymbol{a}) : m \in \mathcal{M}^n, n \in \mathcal{N}\}$ is a potential game. The potential function is $\mathbf{T}(\dot{\alpha},\ddot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a}), \ddot{\boldsymbol{\rho}})$.*

Potential game always reaches an NE and has the finite improvement property. The proof of Theorem 5 is given in Appendix 5.9.

### 5.4.3. Evolutionary Dynamics

NE is the solution concept of population game. By using the framework of evolutionary dynamics [103], we can analyze how population state evolves in time and converges to NE. The evolutionary dynamics is defined as follows:

$$\dot{a_m^n} = \sum_{k\in\mathcal{M}^n} a_k^n \rho_{km}^n(\boldsymbol{a}, \boldsymbol{F}(\boldsymbol{a})) - a_m^n \sum_{k\in\mathcal{M}^n} \rho_{mk}^n(\boldsymbol{a}, \boldsymbol{F}(\boldsymbol{a})), \tag{5.4.10}$$

where$\rho_{km}^n(\boldsymbol{a}, \boldsymbol{F}(\boldsymbol{a}))$, called revision protocol, represents the switching rate of IoTDs in $\mathbf{C}_n$ change offloading decision from $\mathbf{B}_k$ to $\mathbf{B}_m$ based on population state $\boldsymbol{a}$ and payoff function $\boldsymbol{F}(\boldsymbol{a})$. Larger value of $\rho_{km}^n(\boldsymbol{a}, \boldsymbol{F}(\boldsymbol{a}))$ implies higher probability

that IoTDs in $\mathbf{C}_n$ changing offloading decision from $\mathbf{B}_k$ to $\mathbf{B}_m$. The first term and second term of Eq. (5.4.10) denote the inflow rate of IoTDs choosing $\mathbf{B}_m$ and the outflow rate of IoTDs choosing any BS except $\mathbf{B}_m$, respectively. Thus, the difference of inflow rate and outflow rate describes the evolution of $a_m^n$.

We consider three types of revision protocols, namely Smith, Logit and BNN [30, 103]. For simplicity, we use $\mathbf{D}_m^n$ to represent an IoTD in $\mathbf{C}_n$ choosing $\mathbf{B}_m$. Thus, $a_m^n$ is the number of $\mathbf{D}_m^n$s. The function $[x]_+$ returns $x$ if $x \geq 0$. Otherwise, it returns 0. Smith protocol, defined in Eq. (5.4.11), describes that the switching rate of $\mathbf{D}_k^n$ changing current offloading decision from $\mathbf{B}_k$ to $\mathbf{B}_m$ is the payoff difference between $\mathbf{D}_m^n$ and $\mathbf{D}_k^n$. For example, if $\mathbf{D}_k^n$ knows that $\mathbf{D}_m^n$ has higher payoff, i.e., $\mathbf{B}_m$ is a better choice than $\mathbf{B}_k$ for $\mathbf{C}_n$, then $\mathbf{D}_k^n$ will change his offloading decision to $\mathbf{B}_m$ with switching rate $\rho_{km}^n(\boldsymbol{a}, \boldsymbol{F}(\boldsymbol{a}))$. $\mathbf{D}_k^n$ will not change his offloading decision if $\mathbf{D}_m^n$ has lower payoff.

$$\rho_{km}^n(\boldsymbol{a}, \boldsymbol{F}(\boldsymbol{a})) = \left[F_m^n(\boldsymbol{a}) - F_k^n(\boldsymbol{a})\right]_+. \tag{5.4.11}$$

Logit protocol is defined in Eq. (5.4.12), where $\omega > 0$ is the noise level. $\omega$ represents the rationality of IoTDs. For $\omega = 0$, IoTDs are completely rational and choose the best offloading decision. As $\omega$ increases, IoTDs become less rational and may choose non-optimal decision.

$$\rho_{km}^n(\boldsymbol{a}, \boldsymbol{F}(\boldsymbol{a})) = \frac{\exp\left(\omega^{-1} F_m^n(\boldsymbol{a})\right)}{\sum_{k \in \mathcal{M}^n} \exp\left(\omega^{-1} F_k^n(\boldsymbol{a})\right)}. \tag{5.4.12}$$

BNN protocol, defined in Eq. (5.4.13), describes that $\mathbf{D}_k^n$ compares $\mathbf{D}_m^n$'s payoff with the average payoff of $\mathbf{C}_n$. If $\mathbf{D}_m^n$'s payoff exceeds the average payoff, then $\mathbf{D}_k^n$ will change his offloading decision to $\mathbf{B}_m$ with switching rate $\rho_{km}^n(\boldsymbol{a}, \boldsymbol{F}(\boldsymbol{a}))$.

$$\rho_{km}^n(\boldsymbol{a}, \boldsymbol{F}(\boldsymbol{a})) = \left[F_m^n(\boldsymbol{a}) - \frac{1}{\hat{Z}^n} \sum_{l \in \mathcal{M}^n} a_l^n F_l^n(\boldsymbol{a})\right]_+. \tag{5.4.13}$$

Note that these protocols describe how IoTDs change their offloading decisions until an NE is reached. Smith uses less decision information compared to Logit and

BNN. Smith needs only the payoff of one BS, while Logit and BNN need the payoffs of all BSs in $\mathbb{B}^n$. In general, Smith has lower convergence speed than Logit and BNN. By substituting these revision protocols into Eq. (5.4.10), we can get Smith dynamics, Logit dynamics and BNN dynamics in Eqs. (5.4.14), (5.4.15) and (5.4.16), respectively.

$$\begin{aligned} \dot{a_m^n} = & \sum_{k \in \mathcal{M}^n} a_k^n \left[F_m^n(\boldsymbol{a}) - F_k^n(\boldsymbol{a})\right]_+ \\ & - a_m^n \sum_{k \in \mathcal{M}^n} \left[F_k^n(\boldsymbol{a}) - F_m^n(\boldsymbol{a})\right]_+ . \end{aligned} \tag{5.4.14}$$

$$\dot{a_m^n} = \frac{a_m^n \exp\left(\omega^{-1} F_m^n(\boldsymbol{a})\right)}{\sum_{k \in \mathcal{M}^n} a_k^n \exp\left(\omega^{-1} F_k^n(\boldsymbol{a})\right)} - a_m^n. \tag{5.4.15}$$

$$\dot{a_m^n} = \left[\hat{F}_m^n(\boldsymbol{a})\right]_+ - \frac{a_m^n}{\hat{Z}^n} \sum_{k \in \mathcal{M}^n} \left[\hat{F}_k^n(\boldsymbol{a})\right]_+, \tag{5.4.16}$$

$$\text{where } \hat{F}_m^n(\boldsymbol{a}) = F_m^n(\boldsymbol{a}) - \frac{1}{\hat{Z}^n} \sum_{l \in \mathcal{M}^n} a_l^n F_l^n(\boldsymbol{a}).$$

These evolutionary dynamics can generate an NE in iteration methods. We will propose a load balancing algorithm based on evolutionary dynamics in next section.

## 5.5. Workload Balancing Algorithms

In this section, we proposes two workload balancing algorithms, namely, CWB (Centralized Workload Balancing) algorithm and DWB (Decentralized Workload Balancing) algorithm. CWB is based on the evolutionary dynamics and DWB is based on Theorem 5.

### 5.5.1. Centralized Workload Balancing

Our CWB algorithm consists of two phases. The first phase (Steps $1-10$) is to calculate an NE based on evolutionary dynamics. Smith, Logit and BNN dynamics

can converge to an NE with different convergence speeds (as shown in Section 5.6.1). However, the resulting NE only shows the number of IoTDs in $\mathbf{C}_n$, that will choose $\mathbf{B}_m$, i.e., $a_m^n$, without specifying which IoTD in $\mathbf{C}_n$ will choose $\mathbf{B}_m$. Thus, we use the second phase (Steps $11-23$) to implement IoTDs' offloading decisions. The basic idea is to randomly select IoTDs from $\mathbf{C}_n$ for $\mathbf{B}_m$; the number of IoTDs should be no more than $a_m^n$. The randomness of selection can implement fair offloading decisions for IoTDs. Note that we need to replace $\mathbf{D}_n^*$ with original IoTD when calculating offloading decisions (Steps $18-20$).

### 5.5.2. Decentralized Workload Balancing

Our DWB algorithm is a distributed algorithm consisting of two parts of algorithms running in IoTDs and BSs separately. Each IoTD can change his current offloading decision whenever "stochastic update clock" rings. IoTD randomly chooses several candidate BSs and choose the BS with highest switching rate. IoTD only changes his offloading decision when the highest switching rate is positive, which implies higher payoff. With the property of potential game, any better update of offloading decision is guaranteed to reach an NE. Note that IoTD's algorithm (Part 1 in Algorithm 8) does not exactly follow the switching rates of Smith, Logit and BNN protocols. However, it captures the main features of these revision protocols: 1) randomness is ensured by Step 2, $K$ can be any value between 1 and $|\mathbb{B}^n|$, we choose $K = \left\lfloor \frac{|\mathbb{B}^n|}{2} \right\rfloor$; 2) higher switching rate implying higher probability of changing offloading decision is guaranteed by Steps $3-9$. BS's algorithm (Part 2 in Algorithm 8) is to first collect current IoTDs' offloading decisions and then broadcast the utilization level of BSs and edge clouds to IoTDs. IoTDs will use this information to improve their offloading decisions.

**Algorithm 7** CWB (Centralized Workload Balancing)

***Phase 1:*** *Calculate an NE.*

Initialize $\boldsymbol{a}$ with arbitrary value satisfying Constraint (5.4.4)
$\bar{\boldsymbol{a}} \leftarrow \mathbf{0}$
**while** $\bar{\boldsymbol{a}} \neq \boldsymbol{a}$ **do**
    $\bar{\boldsymbol{a}} \leftarrow \boldsymbol{a}$
    **for all** $n \in \mathcal{N}$ **do**
        **for all** $m \in \mathcal{M}^n$ **do**
            $a_m^n \leftarrow$ Update $\bar{a}_m^n$ with Eqs. (5.4.14), (5.4.15) or (5.4.16)
        **end for**
    **end for**
**end while**

***Phase 2:*** *Calculate offloading decisions.*

Initialize decision vector $\mathbf{A} \leftarrow \mathbf{0}$
**for all** $n \in \mathcal{N}$ **do**
    $\mathbf{C} \leftarrow \mathbf{C}^n$
    **for all** $m \in \mathcal{M}^n$ **do**
        $a \leftarrow a_m^n$
        **while** $\mathbf{C} \neq \varnothing \wedge a > 0$ **do**
            $i \leftarrow Next(\mathbf{C})$
            **if** $\frac{B_i}{B^n} \leq a$ **then**
                $a \leftarrow a - \frac{B_i}{B^n}$, $\mathbf{C} \leftarrow \mathbf{C} \setminus \{i\}$, $\mathbf{A}(i) \leftarrow m$
            **end if**
        **end while**
    **end for**
**end for**

**Algorithm 8** DWB (Decentralized Workload Balancing)

***Part 1:*** *For all* $\mathbf{D}_m^n$

**for all** "stochastic update clock" rings **do**
  $\mathcal{K} \leftarrow$ randomly choose $K$ BSs from $\mathbb{B}^n$
  **for all** $k \in \mathcal{K}$ **do**
    Calculate $\rho_{mk}^n(\hat{\boldsymbol{a}}, \hat{\boldsymbol{F}}(\boldsymbol{a}))$
      with Eqs. (5.4.11), (5.4.12) or (5.4.13)
  **end for**
  **if** $\max_{k \in \mathcal{K}} \rho_{km}^n(\hat{\boldsymbol{a}}, \hat{\boldsymbol{F}}(\boldsymbol{a})) > 0$ **then**
    $k^* = \arg\max_{k \in \mathcal{K}} \rho_{km}^n(\hat{\boldsymbol{a}}, \hat{\boldsymbol{F}}(\boldsymbol{a}))$
    Update the offloading decision of $\mathbf{D}_m^n$ with $k^*$
  **end if**
**end for**

***Part 2:*** *For all* $\mathbf{B}_m$

**for all** "stochastic update clock" rings **do**
  Collect current population state $\hat{\boldsymbol{a}}$
  Calculate $\hat{\hat{\rho}}_m$ and $\hat{\hat{\rho}}_m$ by Eqs. (5.3.4) and (5.3.5), respectively.
  Broadcast $\hat{\hat{\rho}}_m$ and $\hat{\hat{\rho}}_m$ to IoTDs within coverage of $\mathbf{B}_m$
**end for**

## 5.6. Performance Evaluation

In this section, we evaluate the performance of our proposed workload balancing algorithms by numerical studies. We compare our work with two related recent contributions: BRUTE [99] and TWB (Towards Workload Balancing) [100]. BRUTE uses $\alpha-$ fair function to implement energy-efficient traffic allocation among BSs. BRUTE considers energy consumption in BSs without considering the load in edge

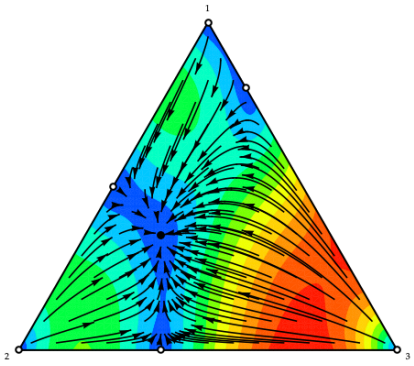



**Fig. 5.4.** Evolutionary dynamics of Smith protocol. The black dot denotes NE. The arrows describe the motions of different population states.

clouds. TWB considers the traffic load in BSs and computation load in edge clouds. However, none of them considers the effect of SINR in load balancing.

Without loss of generality, we randomly select the parameter's value from the normal distribution for different cases. The average value of these parameters are similar to the ones in [45, 59, 95, 104]. The data size $B^n$ is set to $800KB$ and the number of required CPU cycles $D^n$ is set to $1000Megacycles$. We set the allocated computational capability $F_m^n$ to $100GHz$ and the bandwidth of BS to $5MHz$. The channel gain between $\mathbf{D}_n$ and $\mathbf{B}_m$ is $H_m^n = (d_n^m)^{-\theta}$, where $\theta$ is the pass loss factor and $d_n^m$ is the distance between them. $\theta$ is set to 4 and $H_m^n$ is randomly selected from $[5m - 100m]$ [105]. The wireless transmission power $P_l^n$ is set to $100mWatts$.

### 5.6.1. Illustration of Evolutionary Dynamics: One Class Case

We first compare the evolutionary dynamics of Smith, Logit and BNN. We consider the scenario where a class of 400 IoTDs compete for the communication resources of 3 BSs and computation resources in 3 edge clouds. We observe that Smith

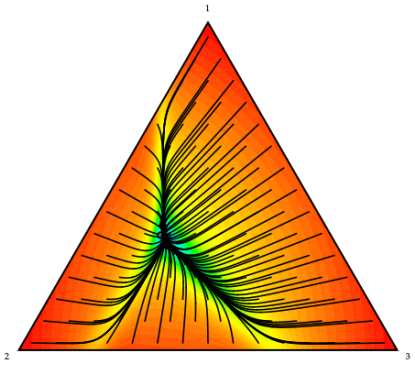



**Fig. 5.5.** Evolutionary dynamics of Logit protocol. The black dot denotes NE. The arrows describe the motions of different population states.

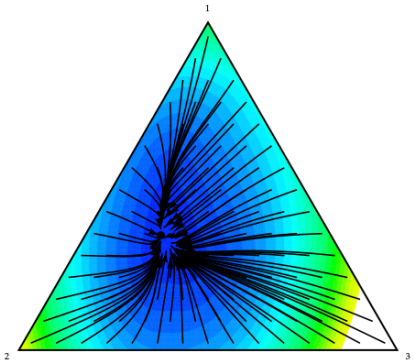



**Fig. 5.6.** Evolutionary dynamics of BNN dynamics. The black dot denotes NE. The arrows describe the motions of different population states.

converges more slowly than Logit and BNN, as shown in Figs. 5.4, 5.5 and 5.6. We also observe that the arrows of Smith approach NE in a less angular, more gradual fashion. This is because Smith changes its offloading decision based on the payoff of one BS, while BNN and Logit change the offloading decision based on all the

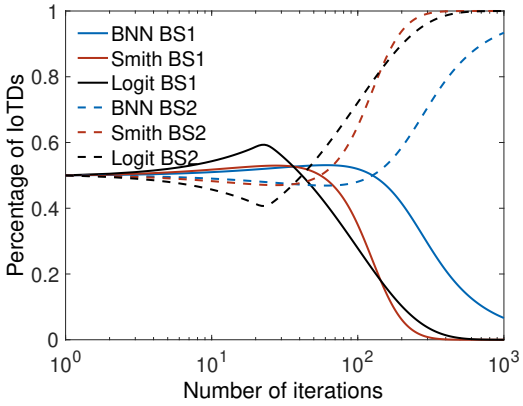



**Fig. 5.7.** Percentage of IoTDs in $\mathbf{C}_1$ choosing $\mathbf{B}_1$ and $\mathbf{B}_2$ with respect to the number of iterations. The initial class state $\boldsymbol{a}^1 = (0.5, 0.5)$. The solid line represents the percentage of IoTDs choosing $\mathbf{B}_1$, while the dashed line represents the percentage of IoTDs choosing $\mathbf{B}_2$.

payoff of BSs. Generally, using more payoff information to make offloading decisions can achieve better performance. Thus, BNN and Logit converge faster than Smith. Moreover, these three evolutionary dynamics can achieve the same NE.

Then, we investigate the resulting NE, i.e., the percentage of IoTDs choosing three BSs, denoted by black point in the figures. Note that the distance between black point and the vertex of triangle denotes the percentage of IoTDs choosing the corresponding BS; smaller distance represents higher percentage. NE is $(0.35, 0.45, 0.20)$ in our settings. We observe that NE is located close to $\mathbf{B}_2$ (i.e., the black point is close to vertex 2). This is because $\mathbf{B}_2$ and edge $\mathbf{E}_2$ have the highest communication and computation capabilities, respectively, in our settings. Consequently, more IoTDs prefer to offload tasks to $\mathbf{B}_2$.

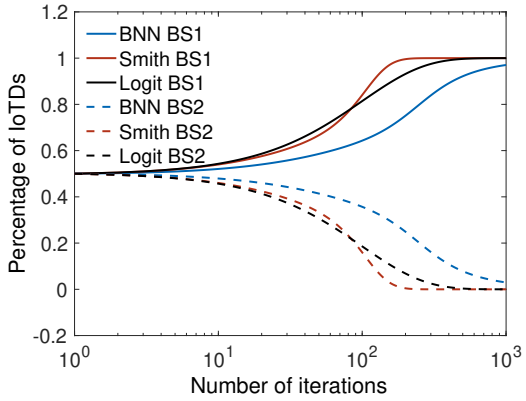



**Fig. 5.8.** Percentage of IoTDs in $\mathbf{C}_2$ choosing $\mathbf{B}_1$ and $\mathbf{B}_2$ with respect to the number of iterations. The initial class state $\boldsymbol{a}^2 = (0.5, 0.5)$.

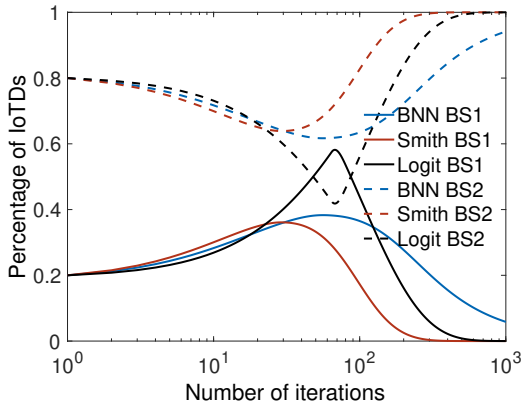



**Fig. 5.9.** Percentage of IoTDs in $\mathbf{C}_1$ choosing $\mathbf{B}_1$ and $\mathbf{B}_2$ with respect to the number of iterations. The initial class state $\boldsymbol{a}^1 = (0.2, 0.8)$.

### 5.6.2. Illustration of Evolutionary Dynamics: Two Classes Case

We consider the scenario where two classes of 400 IoTDs compete for the communication resources of 2 BSs and computation resources in 2 edge clouds. Figs. 5.7 and 5.8 show the evolutions of $\boldsymbol{a}^1$ and $\boldsymbol{a}^2$, respectively. The initial class states of $\boldsymbol{a}^1$ and $\boldsymbol{a}^2$ are both $(0.5, 0.5)$. NE for $\mathbf{C}_1$ is (0,1) and NE for $\mathbf{C}_2$ is (1,0); this shows

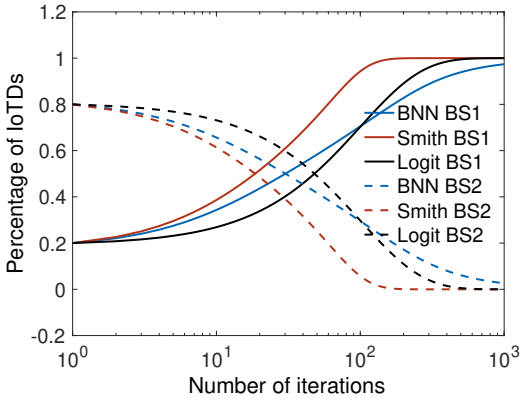



**Fig. 5.10.** Percentage of IoTDs in $\mathbf{C}_2$ choosing $\mathbf{B}_1$ and $\mathbf{B}_2$ with respect to the number of iterations. The initial class state $\boldsymbol{a}^2 = (0.2, 0.8)$.

that IoTDs in $\mathbf{C}_1$ will offload tasks to $\mathbf{B}_2$ and IoTDs in $\mathbf{C}_2$ will offload tasks to $\mathbf{B}_1$. We observe that three dynamics converge to NE with diverse convergence speeds. Smith has higher convergence speed than BNN and Logit. This is because Smith dynamic requires user to choose some BS with higher payoff. Since there are two BSs in this scenario, IoTDs using Smith dynamic have higher probability to choose optimal decisions. In BNN dynamic, IoTDs choose BS randomly and compare its payoff with average payoff in the same class; IoTDs have more chance to choose suboptimal decision compared to Smith. The same case happens to Logit dynamic where IoTDs change decisions according to the payoff ratios defined in Eq. (5.4.12). We state that the advantages of BNN and Logit increase with the number of BSs. The intuition is that IoTDs using Smith dynamic have lower probability to choose optimal decisions with the increase of the number of BSs, since Smith only ensures that the next payoff is better than current payoff, while BNN ensures that next payoff is better than average payoff and Logit uses the payoffs of all IoTDs in a class to make decisions.

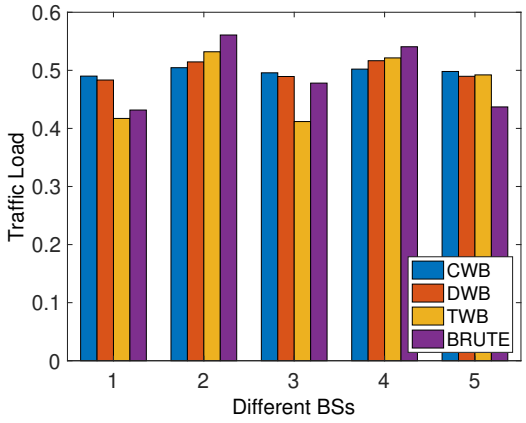



**Fig. 5.11.** Traffic load versus different BSs.

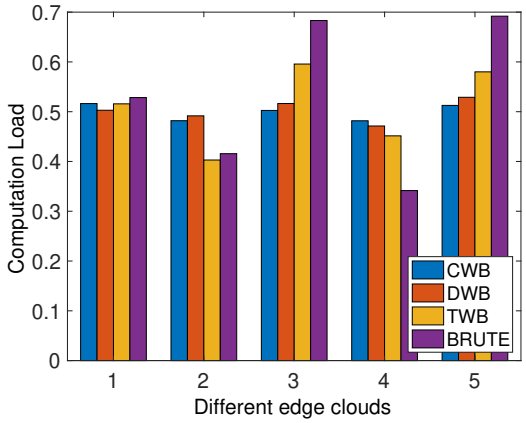



**Fig. 5.12.** Computation load versus different edge clouds.

Figs. 5.9 and 5.10 illustrate the evolutions of $\boldsymbol{a}^1$ and $\boldsymbol{a}^2$, respectively. We set the initial class states of $\boldsymbol{a}^1$ and $\boldsymbol{a}^2$ to $(0.2, 0.8)$, which is different from that in Figs. 5.7 and 5.8. We observe that three dynamics converge to NE. This demonstrates that diverse initial class states can reach the same NE.

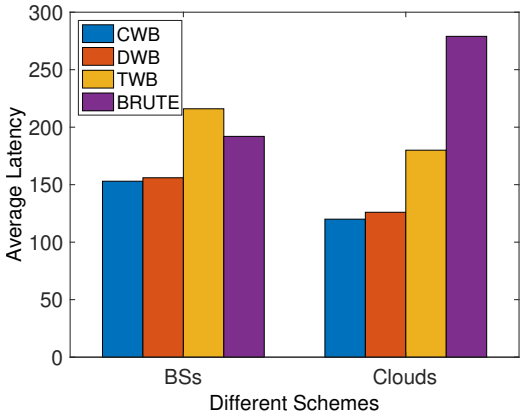



**Fig. 5.13.** Average latency versus different schemes.

### 5.6.3. Performance Comparison of Different Algorithms

To investigate the performance of workload balancing of different algorithms, we consider a scenario where 1000 IoTDs compete for communication and computation resources of 5 BS-edge-cloud pairs. Fig. 5.11 shows that CWB and DWB achieve better performance than TWB and BRUTE in traffic load balancing. We observe that the difference of traffic load among 5 BSs achieved by CWB and DWB are smaller than those achieved by TWB and BRUTE. This is because CWB and DWB use inference affected queueing model where SINR dynamically changes with population state, while TWB and BRUTE use static SINR model where SINR is estimated as location-dependent static value. We further observe that BRUTE achieves higher performance for traffic load balancing than TWB, since BRUTE does not consider the computation load balancing in edge clouds. Thus, BRUTE does not need to sacrifice the load balancing among BSs to implement load balancing among edge clouds. Fig. 5.12 shows the computation load among edge clouds. We observe that CWB and DWB achieve better performance than TWB and BRUTE in computation load balancing. This is because TWB and BRUTE do not consider the affect of dynamic

inference among IoTDs. Furthermore, BRUTE achieves worst performance without considering the computation load balancing in edge clouds.

Fig. 5.13 shows the average latency in BSs and edge clouds with the 4 schemes. We observe that CWB achieves lowest latency and DWB achieves quasi-optimal latency due to stochastic factor. Since BRUTE only focuses on communication latency among BSs, it has lower communication latency than TWB. In contrast, TWB achieves much lower computation latency than BRUTE, since TWB considers both communication latency and computation latency. By sacrificing slight communication latency, TWB achieves better performance than BRUTE. However, CWB and DWB outperform BRUTE and TWB, since BRUTE and TWB do not consider the impact of SINR.

## 5.7. Conclusion

In this paper, we analyze the workload balancing problem for MEC in the context of IoT. IoTDs can offload computation intensive tasks to nearby BSs. We propose a population game based approach to investigate this problem and show that the game always has an NE. We consider the impact of SINR in workload balancing problem and propose an inference function to analyze the impact of SINR. We use three kinds of revision protocols, namely, BNN, Smith and Logit to achieve NE. We design two workload balancing algorithms to iteratively calculate the optimal solution. We illustrate the evolutionary dynamics with two different scenarios. Numerical results show that our schemes outperform two existing schemes.

## 5.8. Proof of Theorem 1

PROOF. In order to proof Theorem 1, we need investigate the properties of inference function $g(\widehat{SINR}_m^n)$.

### 5.8.1. Inference Function is a Monotonic Function

We first calculate the derivative of $g(\widehat{SINR}_m^n)$ with respect to $\widehat{SINR}_m^n$ as follows:

$$g'(\widehat{SINR}_m^n) = \frac{(\widehat{SINR}_m^n + 1)\ln(\widehat{SINR}_m^n + 1)}{(\widehat{SINR}_m^n + 1)^2 \ln^2(\widehat{SINR}_m^n + 1)} - \frac{\widehat{SINR}_m^n(\ln(\widehat{SINR}_m^n + 1) + 1)}{(\widehat{SINR}_m^n + 1)^2 \ln^2(\widehat{SINR}_m^n + 1)} \tag{5.8.1}$$

$$= \frac{\ln(\widehat{SINR}_m^n + 1) - \widehat{SINR}_m^n}{(\widehat{SINR}_m^n + 1)^2 \ln^2(\widehat{SINR}_m^n + 1)}. \tag{5.8.2}$$

Note that the domain of $g(\widehat{SINR}_m^n)$ is $\{\widehat{SINR}_m^n \in \mathbb{R} | \widehat{SINR}_m^n > 0\}$, since $g(\widehat{SINR}_m^n)$ has no definition when $\widehat{SINR}_m^n = 0$. We will discuss the case where $\widehat{SINR}_m^n = 0$ later. Obviously, $\ln(\widehat{SINR}_m^n + 1) - \widehat{SINR}_m^n < 0$ for all $\widehat{SINR}_m^n > 0$. According to Eq. (5.8.2), we know that $g'(\widehat{SINR}_m^n) < 0$. Thus, $g(\widehat{SINR}_m^n)$ is a strictly decreasing function.

### 5.8.2. The Limit of Inference Function

We next consider the value of $\lim_{\widehat{SINR}_m^n \to 0} g(\widehat{SINR}_m^n)$.

$$\lim_{\widehat{SINR}_m^n \to 0} g(\widehat{SINR}_m^n) \tag{5.8.3}$$

$$= \lim_{\widehat{SINR}_m^n \to 0} \frac{\widehat{SINR}_m^n}{(\widehat{SINR}_m^n + 1)\ln(\widehat{SINR}_m^n + 1)} \tag{5.8.4}$$

$$= \lim_{\widehat{SINR}_m^n \to 0} \frac{\widehat{SINR}_m^n}{\ln(\widehat{SINR}_m^n + 1)} \lim_{\widehat{SINR}_m^n \to 0} \frac{1}{\widehat{SINR}_m^n + 1} \tag{5.8.5}$$

$$= \lim_{\widehat{SINR}_m^n \to 0} \frac{\widehat{SINR}_m^n}{\ln(\widehat{SINR}_m^n + 1)} \text{ (Applying l'Hopital's rule)} \tag{5.8.6}$$

$$= \lim_{\widehat{SINR}_m^n \to 0} \frac{\frac{d}{d\widehat{SINR}_m^n}(\widehat{SINR}_m^n)}{\frac{d}{d\widehat{SINR}_m^n}(\ln(\widehat{SINR}_m^n + 1))} \tag{5.8.7}$$

$$= \lim_{\widehat{SINR}_m^n \to 0} (\widehat{SINR}_m^n + 1) \tag{5.8.8}$$

$$= 1. \tag{5.8.9}$$

Without causing ambiguity, we say that $g(\widehat{SINR}_m^n) = 1$, when $\widehat{SINR}_m^n = 0$.

### 5.8.3. The Range of Inference Function

Since $g(\widehat{SINR}_m^n)$ is strictly decreasing and $g(0) = 1$, we know that

$$\max_{\widehat{SINR}_m^n \geq 0} g(\widehat{SINR}_m^n) = g(0) = 1. \tag{5.8.10}$$

Moreover, it is easy to verify that $g(\widehat{SINR}_m^n) > 0$ for all $\widehat{SINR}_m^n \geq 0$ and $\lim_{\widehat{SINR}_m^n \to +\infty} g(\widehat{SINR}_m^n) = 0$. Thus, the range of $g(\widehat{SINR}_m^n)$ is [0,1].

Based on the above discussions, we know that $g(\widehat{SINR}_m^n)$ decreases from 1 to 0 when $\widehat{SINR}_m^n$ increases from 0 to $+\infty$. Thus, $2 - g(\widehat{SINR}_m^n)$ increases from 1 to 2 when $\widehat{SINR}_m^n$ increases from 0 to $+\infty$. Finally, time fraction $\hat{T}_m^n(\boldsymbol{a}) =$

$\frac{\theta^n}{R_m^n(\boldsymbol{a})}\Big(2-g(\widehat{SINR_m^n})\Big)$ increases from $\frac{\theta^n}{R_m^n(\boldsymbol{a})}$ to $\frac{2\theta^n}{R_m^n(\boldsymbol{a})}$, when $\widehat{SINR_m^n}$ increases from 0 to $+\infty$. □

## 5.9. Proof of Theorem 2

In order to prove that $\mathbf{T}(\dot{\alpha},\ddot{\alpha},\dot{\boldsymbol{\rho}}(\boldsymbol{a}),\ddot{\boldsymbol{\rho}})$ is the potential function of population game $\boldsymbol{F}(\boldsymbol{a})$, we need to derive that

$$F_m^n(\boldsymbol{a})=\frac{\partial\mathbf{T}(\dot{\alpha},\ddot{\alpha},\dot{\boldsymbol{\rho}}(\boldsymbol{a}),\ddot{\boldsymbol{\rho}})}{\partial a_m^n}=\xi\cdot\frac{\partial\mathbf{T}(\ddot{\alpha},\ddot{\boldsymbol{\rho}})}{\partial a_m^n}+\frac{\partial\mathbf{T}(\dot{\alpha},\dot{\boldsymbol{\rho}}(\boldsymbol{a}))}{\partial a_m^n}. \tag{5.9.1}$$

### 5.9.1. Partial Derivatives of Load Balancing in Edge Clouds

We first calculate the partial derivatives of $\mathbf{T}(\ddot{\alpha},\ddot{\boldsymbol{\rho}})$ in two cases based on the value of $\ddot{\alpha}$.

Case 1: $\ddot{\alpha}\neq 1$,

$$\frac{\partial\mathbf{T}(\ddot{\alpha},\ddot{\boldsymbol{\rho}})}{\partial a_m^n}=\frac{\partial}{\partial a_m^n}\Big[-\sum_{m\in\mathcal{M}}\frac{(1-\ddot{\rho}_m)^{1-\ddot{\alpha}}-1}{\ddot{\alpha}-1}\Big] \tag{5.9.2}$$

$$=\frac{1-\ddot{\alpha}}{(\ddot{\alpha}-1)(1-\ddot{\rho}_m)^{\ddot{\alpha}}}\cdot\frac{\partial\ddot{\rho}_m}{\partial a_m^n} \tag{5.9.3}$$

$$=\frac{-\eta^n}{F_m(1-\ddot{\rho}_m)^{\ddot{\alpha}}}. \tag{5.9.4}$$

Based on Eq. (5.4.6), we have Eq. (5.9.2). According to Eq. (5.3.5), we can obtain that $\frac{\partial\ddot{\rho}_m}{\partial a_m^n}=\frac{\eta^n}{F_m}$. Thus, we can get Eq. (5.9.4) from Eq. (5.9.3). Similarly, we can calculate the partial derivatives of $\mathbf{T}(\ddot{\alpha},\ddot{\boldsymbol{\rho}})$ when $\ddot{\alpha}=1$.

Case 2: $\ddot{\alpha}=1$,

$$\frac{\partial\mathbf{T}(\ddot{\alpha},\ddot{\boldsymbol{\rho}})}{\partial a_m^n}=\frac{\partial}{\partial a_m^n}\Big[-\sum_{m\in\mathcal{M}}\ln\Big(\frac{1}{1-\ddot{\rho}_m}\Big)\Big] \tag{5.9.5}$$

$$=\frac{1-\ddot{\rho}_m}{(1-\ddot{\rho}_m)^2}\cdot\frac{-\partial\ddot{\rho}_m}{\partial a_m^n} \tag{5.9.6}$$

$$=\frac{-\eta^n}{F_m(1-\ddot{\rho}_m)}. \tag{5.9.7}$$

Note that Eq. (5.9.7) is equal to Eq. (5.9.4) when $\ddot{\alpha} = 1$. Thus, we can say that

$$\frac{\partial \mathbf{T}(\ddot{\alpha}, \ddot{\boldsymbol{\rho}})}{\partial a_m^n} = \frac{-\eta^n}{F_m(1-\ddot{\rho}_m)^{\ddot{\alpha}}}, \qquad \forall \ddot{\alpha} \geq 0. \tag{5.9.8}$$

### 5.9.2. Partial Derivatives of Load Balancing in BSs

We next calculate the partial derivatives of $\mathbf{T}(\dot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a}))$ based on different values of $\dot{\alpha}$.

$$\frac{\partial \dot{\rho}_m(\boldsymbol{a})}{\partial a_m^n}$$

$$= \frac{\partial}{\partial a_m^n}\left[\sum_{n\in\mathcal{N}} \frac{\theta^n (a_m^n)^2}{W_m \log_2\left(1+\frac{a_m^n P_l^n H_m^n}{\delta+\sum_{k\in\mathcal{N}_m^n} a_m^k P_l^k H_m^k}\right)}\right] = \frac{\theta^n \ln(2)}{W_m}\cdot\frac{\partial}{\partial a_m^n}\left[\frac{(a_m^n)^2}{\ln\left(1+\frac{a_m^n P_l^n H_m^n}{\delta+\sum_{k\in\mathcal{N}_m^n} a_m^k P_l^k H_m^k}\right)}\right] \tag{5.9.9}$$

$$= \frac{\theta^n \ln(2)\left(2a_m^n \ln\left(\frac{a_m^n P_l^n H_m^n}{\sum_{k\in\mathcal{N}_m^n} a_m^k P_l^k H_m^k+\delta}+1\right) - \frac{(a_m^n)^2 P_l^n H_m^n}{(\sum_{k\in\mathcal{N}_m^n} a_m^k P_l^k H_m^k+\delta)\left(\frac{a_m^n P_l^n H_m^n}{\sum_{k\in\mathcal{N}_m^n} a_m^k P_l^k H_m^k+\delta}+1\right)}\right)}{W_m \ln^2\left(\frac{a_m^n P_l^n H_m^n}{\sum_{k\in\mathcal{N}_m^n} a_m^k P_l^k H_m^k+\delta}+1\right)} \tag{5.9.10}$$

$$= \frac{\theta^n \left(2a_m^n \log_2\left(\frac{a_m^n P_l^n H_m^n}{\sum_{k\in\mathcal{N}_m^n} a_m^k P_l^k H_m^k+\delta}+1\right) - \frac{(a_m^n)^2 P_l^n H_m^n}{\ln(2)(\sum_{k\in\mathcal{N}_m^n} a_m^k P_l^k H_m^k+\delta)\left(\frac{a_m^n P_l^n H_m^n}{\sum_{k\in\mathcal{N}_m^n} a_m^k P_l^k H_m^k+\delta}+1\right)}\right)}{W_m \log_2^2\left(\frac{a_m^n P_l^n H_m^n}{\sum_{k\in\mathcal{N}_m^n} a_m^k P_l^k H_m^k+\delta}+1\right)}. \tag{5.9.11}$$

Case 1: $\dot{\alpha} \neq 1$,

$$\frac{\partial \mathbf{T}(\dot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a}))}{\partial a_m^n} = \frac{\partial}{\partial a_m^n}\Big[ - \sum_{m \in \mathcal{M}} \frac{(1 - \dot{\rho}_m(\boldsymbol{a}))^{1-\dot{\alpha}} - 1}{\dot{\alpha} - 1}\Big] \tag{5.9.12}$$

$$= \frac{1 - \dot{\alpha}}{(\dot{\alpha} - 1)(1 - \dot{\rho}_m(\boldsymbol{a}))^{\dot{\alpha}}} \cdot \frac{\partial \dot{\rho}_m(\boldsymbol{a})}{\partial a_m^n} \tag{5.9.13}$$

$$= \frac{-1}{(1 - \dot{\rho}_m(\boldsymbol{a}))^{\dot{\alpha}}} \cdot \frac{\partial \dot{\rho}_m(\boldsymbol{a})}{\partial a_m^n}. \tag{5.9.14}$$

Case 2: $\dot{\alpha} = 1$,

$$\frac{\partial \mathbf{T}(\dot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a}))}{\partial a_m^n} = \frac{\partial}{\partial a_m^n}\Big[ - \sum_{m \in \mathcal{M}} \ln\left(\frac{1}{1 - \dot{\rho}_m(\boldsymbol{a})}\right)\Big] \tag{5.9.15}$$

$$= \frac{1 - \dot{\rho}_m(\boldsymbol{a})}{(1 - \dot{\rho}_m(\boldsymbol{a}))^2} \cdot \frac{-\partial \dot{\rho}_m(\boldsymbol{a})}{\partial a_m^n} \tag{5.9.16}$$

$$= \frac{-1}{1 - \dot{\rho}_m(\boldsymbol{a})} \cdot \frac{\partial \dot{\rho}_m(\boldsymbol{a})}{\partial a_m^n}. \tag{5.9.17}$$

Note that Eqs. (5.9.14) and (5.9.17) are equivalent when $\dot{\alpha} = 1$. Thus, we obtain that

$$\frac{\partial \mathbf{T}(\dot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a}))}{\partial a_m^n} = \frac{-1}{(1 - \dot{\rho}_m(\boldsymbol{a}))^{\dot{\alpha}}} \cdot \frac{\partial \dot{\rho}_m(\boldsymbol{a})}{\partial a_m^n}, \qquad \forall \dot{\alpha} \geq 0. \tag{5.9.18}$$

### 5.9.3. Partial Derivatives of the Untilization of BSs

The partial derivatives $\frac{\partial \dot{\rho}_m(\boldsymbol{a})}{\partial a_m^n}$ are calculated as follows. Eq. (5.9.9) is the result of replacing $\dot{\rho}_m(\boldsymbol{a})$ with Eq. (5.3.4). By using derivative rules, we get Eq. (5.9.10) from Eq. (5.9.9). With some basic mathematical manipulation, we rewrite Eq. (5.9.10) as Eq. (5.9.11) in order to simplify the result. For the sake of clarity, we use $\widehat{IN_m^n} = \delta + \sum_{k \in \mathcal{N}_m^n} a_m^k P_l^k H_m^k$ to denote the sum of noise and inference from other classes. The Signal-to-Interference-plus-Noise Ratio $\frac{a_m^n P_l^n H_m^n}{\delta + \sum_{k \in \mathcal{N}_m^n} a_m^k P_l^k H_m^k}$ is denoted by $\widehat{SINR_m^n}$. With these two variables, Eq. (5.9.11) can be rewritten as Eq. (5.9.19). Note that $\frac{W_m}{R_m^n(\boldsymbol{a})}$ in Eq. (5.9.20) can be replaced by Eq. (5.3.3), thus resulting in Eq. (5.9.21). Since $\widehat{SINR_m^n} = \frac{a_m^n P_l^n H_m^n}{\widehat{IN_m^n}}$, we can obtain Eq. (5.9.22). By substituting the

function body of $g(\widehat{SINR}_m^n)$ (see Eq. (5.4.8)) in Eq. (5.9.22), we get Eq. (5.9.23).

$$\frac{\partial \dot{\rho}_m(\boldsymbol{a})}{\partial a_m^n}$$

$$= \frac{W_m \theta^n}{[a_m^n R_m^n(\boldsymbol{a})]^2} \left( \frac{2(a_m^n)^2 R_m^n(\boldsymbol{a})}{W_m} - \frac{(a_m^n)^2 P_l^n H_m^n}{ln(2)\widehat{IN}_m^n(\widehat{SINR}_m^n + 1)} \right) \tag{5.9.19}$$

$$= \frac{2\theta^n}{R_m^n(\boldsymbol{a})} - \frac{\theta^n W_m P_l^n H_m^n}{ln(2)[R_m^n(\boldsymbol{a})]^2 \widehat{IN}_m^n(\widehat{SINR}_m^n + 1)} \tag{5.9.20}$$

$$= \frac{2\theta^n}{R_m^n(\boldsymbol{a})} - \frac{\theta^n a_m^n P_l^n H_m^n}{ln(2) R_m^n(\boldsymbol{a}) \widehat{IN}_m^n(\widehat{SINR}_m^n + 1) \log_2(\widehat{SINR}_m^n + 1)} \tag{5.9.21}$$

$$= \frac{2\theta^n}{R_m^n(\boldsymbol{a})} - \frac{\theta^n \widehat{SINR}_m^n}{R_m^n(\boldsymbol{a})(\widehat{SINR}_m^n + 1) \ln(\widehat{SINR}_m^n + 1)} \tag{5.9.22}$$

$$= \frac{\theta^n \Big(2 - g(\widehat{SINR}_m^n)\Big)}{R_m^n(\boldsymbol{a})}. \tag{5.9.23}$$

Finally, we can obtain the partial derivatives of $\mathbf{T}(\dot{\alpha},\ddot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a}), \ddot{\boldsymbol{\rho}})$ as follows:

$$\frac{\partial \mathbf{T}(\dot{\alpha},\ddot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a}), \ddot{\boldsymbol{\rho}})}{\partial a_m^n} = \xi \cdot \frac{\partial \mathbf{T}(\ddot{\alpha}, \ddot{\boldsymbol{\rho}})}{\partial a_m^n} + \frac{\partial \mathbf{T}(\dot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a}))}{\partial a_m^n} \tag{5.9.24}$$

$$= \frac{-\eta^n \xi}{F_m(1 - \ddot{\rho}_m)^{\ddot{\alpha}}} + \frac{-1}{(1 - \dot{\rho}_m(\boldsymbol{a}))^{\dot{\alpha}}} \cdot \frac{\theta^n \Big(2 - g(\widehat{SINR}_m^n)\Big)}{R_m^n(\boldsymbol{a})} \tag{5.9.25}$$

$$= - \left[ \frac{\eta^n \xi}{F_m(1 - \ddot{\rho}_m)^{\ddot{\alpha}}} + \frac{\theta^n \Big(2 - g(\widehat{SINR}_m^n)\Big)}{R_m^n(\boldsymbol{a})(1 - \dot{\rho}_m(\boldsymbol{a}))^{\dot{\alpha}}} \right] \tag{5.9.26}$$

$$= F_m^n(\boldsymbol{a}). \tag{5.9.27}$$

According to Definition 1, The population game $\boldsymbol{F}(\boldsymbol{a}) = \{F_m^n(\boldsymbol{a}) : m \in \mathcal{M}^n, n \in \mathcal{N}\}$ is a potential game and the potential function is $\mathbf{T}(\dot{\alpha},\ddot{\alpha}, \dot{\boldsymbol{\rho}}(\boldsymbol{a}), \ddot{\boldsymbol{\rho}})$.

# Chapitre 6

## Conclusions and Future Work

### 6.1. Conclusions

In this thesis, we presented the research background, the state of the art and three contibutions for the doctoral thesis entitled *Mobile data and computation offloading in mobile cloud computing*. The first two contributions are related to mobile data offloading, while the third contribution is related to mobile computation offloading.

In the first contribution, we study mobile data offloading problem under the architecture of mobile cloud computing, where mobile data can be delivered by WiFi network and device-to-device communication. In order to minimize the overall cost for data delivery task, it is crucial to reduce cellular network usage while satisfying delay requirements. In our proposed model, we formulate the data offloading task as a finite horizon Markov Decision Process. We first propose a hybrid offloading algorithm for mobile data with different delay requirements. Moreover, we establish the sufficient conditions for the existence of threshold policy. Then, we propose a monotone offloading algorithm based on threshold policy in order to reduce the computational complexity. The simulation results show that the proposed offloading approach can achieve minimal communication cost compared with existing offloading schemes.

In the second contribution, we consider a mobile data offloading market where mobile network operator (MNO) can sell bandwidth made available by the access points to increase MNO's profit. We formulate the offloading problem as a multi-item auction and study MNO's profit maximization problem. We discuss the conditions to (i) offload the maximum amount of data traffic, (ii) foster the participation of mobile subscribers (MSs) (individual rationality), (iii) prevent market manipulation (incentive compatibility) and (iv) preserve budget feasibility of MSs. Then, we propose a robust optimization based method to implement multi-item auction mechanism. We further propose two iterative algorithms that efficiently solve the offloading problem. The simulation results show the efficiency and robustness of our proposed methods for cellular data offloading.

In the third contribution, we investigate the workload balancing problems to minimize the transmission latency and computation latency in task offloading process while considering the limited bandwidth resources of BSs and computation resources in edge clouds. We formulate the workload balancing problem as a population game in order to analyze the aggregate offloading decisions. We analyze the aggregate offloading decisions of mobile users through evolutionary game dynamics and show that the game always reaches a Nash equilibrium. We further propose two workload balancing algorithms based on evolutionary dynamics and revision protocols. Simulation results show that our proposed workload balancing algorithms can achieve better performance than existing contributions.

## 6.2. Future Work

### 6.2.1. Distributed Algorithms for Offloading Decision Making

Current computation offloading methods use lots of information, e.g., profiles of mobile devices, states of cloud servers and network conditions, to make offloading

decision. The more information they use, the better offloading decision they make. However, managing too much information may increase the overhead of offloading decision making, which may offset the benefits of offloading computation intensive tasks. To solve this problem, the overhead of these offloading decision methods needs to be analyzed. One of the possible method is to decrease the overhead by ignoring trivial information. To reduce the overhead of mobile devices, we can further offload the task of offloading decision making from mobile devices to trusted cloud servers or edge clouds. In the future, we will consider a distributed offloading model where offloading decision algorithms are implemented between mobile devices and cloud/edge servers.

### 6.2.2. Failure Recovery and Admission Control for Computation Offloading

We will consider the situation where the edge cloud can also decide whether to accept an offloading request or not. Indeed, one edge cloud may receive too many offloading requests from nearby mobile devices. It may overflow (i.e., becomes congested) if too many offloading requests are sent to it. One possible solution is to choose mobile tasks with higher priority. Some offloading requests may be rejected because of limited computation and storage resources in edge clouds; this is different from offloading to cloud servers, where there are virtually unlimited computing and storage resources. The resources and thus offloading requests will always be accepted. Due to the uncertain wireless environment, we will propose new methods to deal with offloading failure.

### 6.2.3. Programming Model for Mobile Cloud Computing

Programming model can remove the burden of distributed execution details from programmers by providing simple interfaces. Programmers can use these interfaces

to pay more attention to implement the main logic of an application without caring about the detailed implementation of distributed execution.

Although simple interfaces may result in some algorithms hard to implement, the programming model can greatly improve software productivity. It has been proved that programming models have played an important role in cloud computing. MapReduce proposed by Google and Dryad proposed by Microsoft are two famous programming models. More importantly, the open-source implementation of Hadoop plays an important role in cloud computing.

Since mobile cloud computing can augment the computational ability of mobile devices, mobile devices can execute sophisticated applications. From the perspective of mobile cloud computing, mobile devices only execute light tasks. Furthermore, the purpose of cloud computing is to promote the thin-client software deployment, which is suitable for mobile devices which are deployed with limited resources. Mobile cloud computing is a natural extension of cloud computing; it changes the execution mode of traditional mobile applications and proposes new challenges for cloud computing.

Basically, there are two kinds of mobile cloud computing application development methods. One is to develop mobile applications traditionally and execute them with new application models, such as CloneCloud. Developers know nothing about how applications are executed. They don't know whether the application is executed locally or remotely. The selection of where to run an application is based on the offloading decision maker. Even with the same application, the decision maker may calculate different execution plans according to different environments. Another one is to develop mobile applications in a distributed-aware platform. Developers know that mobile applications will run in a distributed environment and they can easily develop new applications with the help of the platform. Most of the applications in the future will run in distributed environment, so we think it is worth to design such a platform to improve the performance of mobile devices and cloud servers. Before

the implementation of a new platform, we need to design a new programming model for this platform.

Traditional programming models only execute tasks in cloud servers and do not support tasks running in mobile devices. This restricts programming models, such as MapReduce, to be implemented in mobile cloud directly. Many researchers adapted MapReduce from clusters to other architecture, such as GPUs and FPGA. This may result in some problems when we adopt programming models in cloud to mobile cloud due to the mobility of mobile devices.

We plan to design a new programming model for mobile cloud computing. The purpose of this work is to improve QoS in mobile cloud by extending the concept of programming model from cloud to mobile cloud. As a by-product, we can improve the overall performance of mobile devices and cloud servers. There are many problems to solve in order to achieve this goal.

## 6.3. Publications

### 6.3.1. Journal Papers


[1] Liu, Dongqing, Lyes Khoukhi, and Abdelhakim Hafid. Prediction-Based Mobile Data Offloading in Mobile Cloud Computing. IEEE Transactions on Wireless Communications 17.7 (2018): 4660-4673.

[2] Liu, Dongqing, Abdelhakim Hafid, and Lyes Khoukhi. Multi-Item Auction Based Mechanism for Mobile Data Offloading: A Robust Optimization Approach. IEEE Transactions on Vehicular Technology. 2019. Accepted.

[3] Liu, Dongqing, Abdelhakim Hafid, and Lyes Khoukhi. Workload Balancing in Mobile Edge Computing for Internet of Things: A Population Game Approach. IEEE Internet of Things Journal. 2019. Submitted.

### 6.3.2. Conference Papers

[1] Liu, Dongqing, Lyes Khoukhi, and Abdelhakim Hafid. Data offloading in mobile cloud computing: A Markov decision process approach. 2017 IEEE International Conference on Communications (ICC). IEEE, 2017.

[2] Liu, Dongqing, Lyes Khoukhi, and Abdelhakim Hafid. Decentralized data offloading for mobile cloud computing based on game theory. 2017 Second International Conference on Fog and Mobile Edge Computing (FMEC). IEEE, 2017.

[3] Liu, Dongqing, Abdelhakim Hafid, and Lyes Khoukhi. Population Game Based Energy and Time Aware Task Offloading for Large Amounts of Competing Users. 2018 IEEE Global Communications Conference (GLOBECOM). IEEE, 2018.

[4] Liu, Dongqing, Lyes Khoukhi, and Abdelhakim Hafid. Aggregate Offloading Decision Analysis for Mobile Edge Computing in Software Defined Network. 2019 IEEE International Conference on Communications (ICC). Accepted.

# Bibliography


[1] Cisco. *Cisco visual networking index: Global mobile data traffic forecast update, 2015-2020*. White Paper. Cisco, 2016.

[2] Filippo Rebecchi, Marcelo Dias De Amorim, Vania Conan, Andrea Passarella, Raffaele Bruno, and Marco Conti. "Data offloading techniques in cellular networks: A survey". In: *IEEE Communications Surveys & Tutorials* 17.2 (2015), pp. 580–603.

[3] Adnan Aijaz, Hamid Aghvami, and Mojdeh Amani. "A survey on mobile data offloading: technical and business perspectives". In: *IEEE Wireless Communications* 20.2 (2013), pp. 104–112.

[4] Ozlem Bilgir Yetim, Margaret Martonosi, Ozlem Bilgir Yetim, and Margaret Martonosi. "Adaptive usage of cellular and WiFi bandwidth: An optimal scheduling formulation". In: *Proceedings of the seventh ACM international workshop on Challenged networks*. ACM. 2012, pp. 69–72. ISBN: 9781450312844. DOI: 10.1145/2348616.2348631.

[5] George Iosifidis, Lin Gao, Jianwei Huang, and Leandros Tassiulas. "A double-auction mechanism for mobile data-offloading markets". In: *IEEE/ACM Transactions on Networking (TON)* 23.5 (2015), pp. 1634–1647.

[6] Kevin Fall. "A delay-tolerant network architecture for challenged internets". In: *Proceedings of the 2003 conference on Applications, technologies, architectures,*

*and protocols for computer communications*. ACM. 2003, pp. 27–34. ISBN: 1581137354. DOI: 10.1145/863955.863960. URL: http://dl.acm.org/citation.cfm?id=863955.863960.

[7] Ray-Guang Cheng, Nien-Sheng Chen, Yu-Feng Chou, and Zdenek Becvar. "Offloading multiple mobile data contents through opportunistic device-to-device communications". In: *Wireless Personal Communications* 84.3 (2015), pp. 1963–1979.

[8] Hoang T Dinh, Chonho Lee, Dusit Niyato, and Ping Wang. "A survey of mobile cloud computing: architecture, applications, and approaches". In: *Wireless communications and mobile computing* 13.18 (2013), pp. 1587–1611.

[9] Karthik Kumar and Yung-Hsiang Lu. "Cloud computing for mobile users: Can offloading computation save energy?" In: *Computer* 43.4 (2010), pp. 51–56.

[10] Pavel Mach and Zdenek Becvar. "Mobile edge computing: A survey on architecture and computation offloading". In: *IEEE Communications Surveys & Tutorials* 19.3 (2017), pp. 1628–1656.

[11] Keke Gai, Meikang Qiu, Hui Zhao, Lixin Tao, and Ziliang Zong. "Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing". In: *Journal of Network and Computer Applications* 59 (2016), pp. 46–54. ISSN: 10958592. DOI: 10.1016/j.jnca.2015.05.016. URL: http://dx.doi.org/10.1016/j.jnca.2015.05.016.

[12] Zhengyuan Pang, Lifeng Sun, Zhi Wang, Erfang Tian, and Shiqiang Yang. "A Survey of Cloudlet Based Mobile Computing". In: *2015 International Conference on Cloud Computing and Big Data (CCBD)*. IEEE. 2015, pp. 268–275.

[13] Michele Mangili, Fabio Martignon, Stefano Paris, and Antonio Capone. "Bandwidth and cache leasing in wireless information-centric networks: A game-theoretic study". In: *IEEE Transactions on Vehicular Technology* 66.1 (2017), pp. 679–695.

[14] Haoran Yu, George Iosifidis, Jianwei Huang, and Leandros Tassiulas. "Auction-based coopetition between LTE unlicensed and Wi-Fi". In: *IEEE Journal on Selected Areas in Communications* 35.1 (2017), pp. 79–90.

[15] Mohammad H Hajiesmaili, Lei Deng, Minghua Chen, and Zongpeng Li. "Incentivizing device-to-device load balancing for cellular networks: An online auction design". In: *IEEE Journal on Selected Areas in Communications* 35.2 (2017), pp. 265–279.

[16] Kehao Wang, Francis CM Lau, Lin Chen, and Robert Schober. "Pricing mobile data offloading: A distributed market framework". In: *IEEE Transactions on Wireless Communications* 15.2 (2016), pp. 913–927.

[17] Shiwen Xu, Yi and Mao, Yi Xu, and Shiwen Mao. "A survey of mobile cloud computing for rich media applications." In: *IEEE Wireless Commun.* 20.3 (2013), pp. 46–53.

[18] Sergio Barbarossa, Stefania Sardellitti, and Paolo Di Lorenzo. "Communicating while computing: Distributed mobile cloud computing over 5G heterogeneous networks". In: *IEEE Signal Processing Magazine* 31.6 (2014), pp. 45–55. ISSN: 10535888. DOI: 10.1109/MSP.2014.2334709. arXiv: arXiv:1105.3232v1.

[19] H. Shah-Mansouri and V. W. S. Wong. "Hierarchical Fog-Cloud Computing for IoT Systems: A Computation Offloading Game". In: *IEEE Internet of Things Journal* 5.4 (Aug. 2018), pp. 3246–3257. ISSN: 2327-4662. DOI: 10.1109/JIOT.2018.2838022.

[20] Hongzhi Guo, Jiajia Liu, and Jie Zhang. "Computation Offloading for Multi-Access Mobile Edge Computing in Ultra-Dense Networks". In: *IEEE Communications Magazine* 56.8 (2018), pp. 14–19.

[21] Hongzhi Guo, Jiajia Liu, and Huiling Qin. "Collaborative Mobile Edge Computation Offloading for IoT over Fiber-Wireless Networks". In: *IEEE Network* 32.1 (2018), pp. 66–71.

[22] D. Liu, A. Hafid, and L. Khoukhi. "Population Game Based Energy and Time Aware Task Offloading for Large Amounts of Competing Users". In: *2018 IEEE Global Communications Conference (GLOBECOM)*. Dec. 2018, pp. 1–6. DOI: 10.1109/GLOCOM.2018.8647522.

[23] D. Liu, L. Khoukhi, and A. Hafid. "Prediction-Based Mobile Data Offloading in Mobile Cloud Computing". In: *IEEE Transactions on Wireless Communications* 17.7 (July 2018), pp. 4660–4673. ISSN: 1536-1276. DOI: 10.1109/TWC.2018.2829513.

[24] Dongqing Liu, L. Khoukhi, and A. Hafid. "Decentralized data offloading for mobile cloud computing based on game theory". In: *2017 Second International Conference on Fog and Mobile Edge Computing (FMEC)*. May 2017, pp. 20–24. DOI: 10.1109/FMEC.2017.7946402.

[25] Jing Zhang, Weiwei Xia, Yueyue Zhang, Qian Zou, Bonan Huang, Feng Yan, and Lianfeng Shen. "Joint Offloading and Resource Allocation Optimization for Mobile Edge Computing". In: *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE. 2017, pp. 1–6.

[26] Chenmeng Wang, Chengchao Liang, F Richard Yu, Qianbin Chen, and Lun Tang. "Computation offloading and resource allocation in wireless cellular networks with mobile edge computing". In: *IEEE Transactions on Wireless Communications* 16.8 (2017), pp. 4924–4938.

[27] Ke Zhang, Supeng Leng, Yejun He, Sabita Maharjan, and Yan Zhang. "Mobile Edge Computing and Networking for Green and Low-Latency Internet of Things". In: *IEEE Communications Magazine* 56.5 (2018), pp. 39–45.

[28] Yiannis Giannakopoulos and Maria Kyropoulou. "The VCG Mechanism for Bayesian Scheduling". In: *ACM Transactions on Economics and Computation (TEAC)* 5.4 (2017), p. 19.

[29] Julian Barreiro-Gomez, Germán Obando, and Nicanor Quijano. "Distributed population dynamics: Optimization and control applications". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 47.2 (2017), pp. 304–314.

[30] Nicanor Quijano, Carlos Ocampo-Martinez, Julian Barreiro-Gomez, German Obando, Andres Pantoja, and Eduardo Mojica-Nava. "The role of population games and evolutionary dynamics in distributed control systems: The advantages of evolutionary game theory". In: *IEEE Control Systems* 37.1 (2017), pp. 70–97.

[31] Wei Song and Weihua Zhuang. *Interworking of wireless LANs and cellular networks*. Springer Science & Business Media, 2012.

[32] Vasilios A Siris and Dimitrios Kalyvas. "Enhancing mobile data offloading with mobility prediction and prefetching". In: *ACM SIGMOBILE Mobile Computing and Communications Review* 17.1 (2013), pp. 22–29. DOI: 10.1145/2348676.2348682. arXiv: arXiv:1306.3177v1.

[33] Nan Cheng, Ning Lu, Ning Zhang, Xuemin Sherman Shen, and Jon W Mark. "Opportunistic WiFi offloading in vehicular environment: A queueing analysis". In: *Global Communications Conference (GLOBECOM), 2014 IEEE*. IEEE. 2014, pp. 211–216. ISBN: 9781479935116. DOI: 10.1109/GLOCOM.2014.7036809.

[34] Fidan Mehmeti and Thrasyvoulos Spyropoulos. "Performance Analysis of Mobile Data Offloading in Heterogeneous Networks". In: *GLOBECOM - IEEE*

*Global Telecommunications Conference* (2013), pp. 1577–1583. DOI: 10.1109/GLOCOM.2013.6831298.

[35] Byoung Hoon Jung, Nah-Oak Song, and Dan Keun Sung. "A network-assisted user-centric WiFi-offloading model for maximizing per-user throughput in a heterogeneous network". In: *IEEE Transactions on Vehicular Technology* 63.4 (2014), pp. 1940–1945.

[36] Vinicius FS Mota, Daniel F Macedo, Yacine Ghamri-Doudanez, and Jose Nogueira. "Managing the decision-making process for opportunistic mobile data offloading". In: *Network Operations and Management Symposium (NOMS), 2014 IEEE*. IEEE. 2014, pp. 1–8.

[37] Vincenzo Sciancalepore, Domenico Giustiniano, Albert Banchs, Andreea Hossmann-Picu, Andreea Picu, and Andreea Hossmann-Picu. "Offloading cellular traffic through opportunistic communications: Analysis and optimization". In: *Selected Areas in Communications, IEEE Journal on* 34.1 (2016), pp. 122–137. arXiv: 1405.3548. URL: http://arxiv.org/abs/1405.3548.

[38] Filippo Rebecchi, Marcelo Dias De Amorim, and Vania Conan. "DROid: Adapting to individual mobility pays off in mobile data offloading". In: *2014 IFIP Networking Conference, IFIP Networking 2014* (2014). DOI: 10.1109/IFIPNetworking.2014.6857087.

[39] Sergey Andreev, Olga Galinina, Alexander Pyattaev, Kerstin Johnsson, and Yevgeni Koucheryavy. "Analyzing assisted offloading of cellular user sessions onto D2D links in unlicensed bands". In: *IEEE Journal on Selected Areas in Communications* 33.1 (2015), pp. 67–80. ISSN: 07338716. DOI: 10.1109/JSAC.2014.2369616.

[40] Xuejun Zhuo, Wei Gao, Guohong Cao, and Sha Hua. "An incentive framework for cellular traffic offloading". In: *Mobile Computing, IEEE Transactions on* 13.3 (2014), pp. 541–555.

[41] Man Hon Cheung and Jianwei Huang. "DAWN: Delay-Aware Wi-Fi offloading and network selection". In: *IEEE Journal on Selected Areas in Communications* 33.6 (2015), pp. 1214–1223. ISSN: 07338716. DOI: 10.1109/JSAC.2015.2416989. arXiv: arXiv:1502.07839v1.

[42] Sergio Barbarossa, Stefania Sardellitti, and Paolo Di Lorenzo. "Joint allocation of computation and communication resources in multiuser mobile cloud computing". In: *Signal Processing Advances in Wireless Communications (SPAWC), 2013 IEEE 14th Workshop on*. IEEE. 2013, pp. 26–30.

[43] Xin Kang, Yeow Khiang Chia, Sumei Sun, and Hon Fah Chong. "Mobile data offloading through a third-party WiFi access point: An operator's perspective". In: *IEEE Transactions on Wireless Communications* 13.10 (2014), pp. 5340–5351. ISSN: 15361276. DOI: 10.1109/TWC.2014.2353057. arXiv: arXiv:1408.5245v1.

[44] Yuan Wu, Yanfei He, Li Ping Qian, Jianwei Huang, and Xuemin Shen. "Optimal resource allocations for mobile data offloading via dual-connectivity". In: *IEEE Transactions on Mobile Computing* 17.10 (2018), pp. 2349–2365.

[45] Xu Chen. "Decentralized computation offloading game for mobile cloud computing". In: *IEEE Transactions on Parallel and Distributed Systems* 26.4 (2015), pp. 974–983.

[46] Nan Cheng, Ning Lu, Ning Zhang, Xiang Zhang, Xuemin Sherman Shen, and Jon W Mark. "Opportunistic WiFi offloading in vehicular environment: A game-theory approach". In: *IEEE Transactions on Intelligent Transportation Systems* 17.7 (2016), pp. 1944–1955.

[47] Joohyun Lee, Yung Yi, Song Chong, and Youngmi Jin. "Economics of WiFi offloading: Trading delay for cellular capacity". In: *IEEE Transactions on Wireless Communications* 13.3 (2014), pp. 1540–1554. ISSN: 15361276. DOI: 10.1109/TWC.2014.010214.130949. arXiv: 1207.6607.

[48] Stefano Paris, Fabio Martignon, Ilario Filippini, and Lin Chen. "An efficient auction-based mechanism for mobile data offloading". In: *IEEE Transactions on Mobile Computing* 14.8 (2015), pp. 1573–1586. ISSN: 15361233. DOI: 10.1109/TMC.2014.2361127.

[49] Dong Zhao, Xiang-Yang Li, and Huadong Ma. "Budget-feasible online incentive mechanisms for crowdsourcing tasks truthfully". In: *IEEE/ACM Transactions on Networking (TON)* 24.2 (2016), pp. 647–661.

[50] Xiaoying Gan, Xiong Wang, Wenhao Niu, Gai Hang, Xiaohua Tian, Xinbing Wang, and Jun Xu. "Incentivize multi-class crowd labeling under budget constraint". In: *IEEE Journal on Selected Areas in Communications* 35.4 (2017), pp. 893–905.

[51] Hongwei Wang, Song Guo, Jiannong Cao, and Minyi Guo. "MELODY: A long-term dynamic quality-aware incentive mechanism for crowdsourcing". In: *IEEE Transactions on Parallel and Distributed Systems* 29.4 (2018), pp. 901–914.

[52] Sayan Bhattacharya, Gagan Goel, Sreenivas Gollapudi, and Kamesh Munagala. "Budget constrained auctions with heterogeneous items". In: *Proceedings of the forty-second ACM symposium on Theory of computing*. ACM. 2010, pp. 379–388.

[53] Xiumin Wang, Yang Sui, Jianping Wang, Chau Yuen, and Weiwei Wu. "A distributed truthful auction mechanism for task allocation in mobile cloud computing". In: *IEEE Transactions on Services Computing* (2018).

[54] Jinfang Jiang, Guangjie Han, Hui Guo, Lei Shu, and Joel J P C Rodrigues. "Geographic multipath routing based on geospatial division in duty-cycled underwater wireless sensor networks". In: *Journal of Network and Computer Applications* 59 (2016), pp. 4–13. ISSN: 10958592. DOI: 10.1016/j.jnca.2015.01.005. URL: http://dx.doi.org/10.1016/j.jnca.2015.01.005.

[55] Yuyi Mao, Jun Zhang, and Khaled B Letaief. "Dynamic computation offloading for mobile-edge computing with energy harvesting devices". In: *IEEE Journal on Selected Areas in Communications* 34.12 (2016), pp. 3590–3605.

[56] Yang Cao, Tao Jiang, and Chonggang Wang. "Optimal radio resource allocation for mobile task offloading in cellular networks". In: *IEEE Network* 28.5 (2014), pp. 68–73.

[57] Lichao Yang, Heli Zhang, Ming Li, Jun Guo, and Hong Ji. "Mobile Edge Computing Empowered Energy Efficient Task Offloading in 5G". In: *IEEE Transactions on Vehicular Technology* (2018).

[58] Min Chen and Yixue Hao. "Task Offloading for Mobile Edge Computing in Software Defined Ultra-dense Network". In: *IEEE Journal on Selected Areas in Communications* (2018).

[59] Erfan Meskar, Terence D Todd, Dongmei Zhao, and George Karakostas. "Energy aware offloading for competing users on a shared communication channel". In: *IEEE Transactions on Mobile Computing* 16.1 (2017), pp. 87–96.

[60] H. Viswanathan, E. K. Lee, I. Rodero, and D. Pompili. "Uncertainty-Aware Autonomic Resource Provisioning for Mobile Cloud Computing". In: *IEEE Transactions on Parallel and Distributed Systems* 26.8 (Aug. 2015), pp. 2363–2372. ISSN: 1045-9219. DOI: 10.1109/TPDS.2014.2345057.

[61] C. Chen, M. Won, R. Stoleru, and G. G. Xie. "Energy-Efficient Fault-Tolerant Data Storage and Processing in Mobile Cloud". In: *IEEE Transactions on Cloud Computing* 3.1 (Jan. 2015), pp. 28–41. ISSN: 2168-7161. DOI: 10.1109/TCC.2014.2326169.

[62] C. You and K. Huang. "Exploiting Non-Causal CPU-State Information for Energy-Efficient Mobile Cooperative Computing". In: *IEEE Transactions on Wireless Communications* 17.6 (June 2018), pp. 4104–4117. ISSN: 1536-1276. DOI: 10.1109/TWC.2018.2820077.

[63] Y. He, J. Ren, G. Yu, and Y. Cai. "D2D Communications Meet Mobile Edge Computing for Enhanced Computation Capacity in Cellular Networks". In: *IEEE Transactions on Wireless Communications* 18.3 (Mar. 2019), pp. 1750–1763. ISSN: 1536-1276. DOI: 10.1109/TWC.2019.2896999.

[64] S. Zhang, P. He, K. Suto, P. Yang, L. Zhao, and X. Shen. "Cooperative Edge Caching in User-Centric Clustered Mobile Networks". In: *IEEE Transactions on Mobile Computing* 17.8 (Aug. 2018), pp. 1791–1805. ISSN: 1536-1233. DOI: 10.1109/TMC.2017.2780834.

[65] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang. "Computation Offloading and Resource Allocation in Wireless Cellular Networks With Mobile Edge Computing". In: *IEEE Transactions on Wireless Communications* 16.8 (Aug. 2017), pp. 4924–4938. ISSN: 1536-1276. DOI: 10.1109/TWC.2017.2703901.

[66] M. Liu, F. R. Yu, Y. Teng, V. C. M. Leung, and M. Song. "Distributed Resource Allocation in Blockchain-Based Video Streaming Systems With Mobile Edge Computing". In: *IEEE Transactions on Wireless Communications* 18.1 (Jan. 2019), pp. 695–708. ISSN: 1536-1276. DOI: 10.1109/TWC.2018.2885266.

[67] F. Wang, J. Xu, X. Wang, and S. Cui. "Joint Offloading and Computing Optimization in Wireless Powered Mobile-Edge Computing Systems". In: *IEEE Transactions on Wireless Communications* 17.3 (Mar. 2018), pp. 1784–1797. ISSN: 1536-1276. DOI: 10.1109/TWC.2017.2785305.

[68] X. Hu, K. Wong, and K. Yang. "Wireless Powered Cooperation-Assisted Mobile Edge Computing". In: *IEEE Transactions on Wireless Communications* 17.4 (Apr. 2018), pp. 2375–2388. ISSN: 1536-1276. DOI: 10.1109/TWC.2018.2794345.

[69] S. Bi and Y. J. Zhang. "Computation Rate Maximization for Wireless Powered Mobile-Edge Computing With Binary Computation Offloading". In: *IEEE*

*Transactions on Wireless Communications* 17.6 (June 2018), pp. 4177–4190. ISSN: 1536-1276. DOI: 10.1109/TWC.2018.2821664.

[70] X. Lyu, W. Ni, H. Tian, R. P. Liu, X. Wang, G. B. Giannakis, and A. Paulraj. "Distributed Online Optimization of Fog Computing for Selfish Devices With Out-of-Date Information". In: *IEEE Transactions on Wireless Communications* 17.11 (Nov. 2018), pp. 7704–7717. ISSN: 1536-1276. DOI: 10.1109/TWC.2018.2869764.

[71] J. L. D. Neto, S. Yu, D. F. Macedo, J. M. S. Nogueira, R. Langar, and S. Secci. "ULOOF: A User Level Online Offloading Framework for Mobile Edge Computing". In: *IEEE Transactions on Mobile Computing* 17.11 (Nov. 2018), pp. 2660–2674. ISSN: 1536-1233. DOI: 10.1109/TMC.2018.2815015.

[72] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief. "Stochastic Joint Radio and Computational Resource Management for Multi-User Mobile-Edge Computing Systems". In: *IEEE Transactions on Wireless Communications* 16.9 (Sept. 2017), pp. 5994–6009. ISSN: 1536-1276. DOI: 10.1109/TWC.2017.2717986.

[73] Y. Sarikaya, H. Inaltekin, T. Alpcan, and J. S. Evans. "Stability and Dynamic Control of Underlay Mobile Edge Networks". In: *IEEE Transactions on Mobile Computing* 17.9 (Sept. 2018), pp. 2195–2208. ISSN: 1536-1233. DOI: 10.1109/TMC.2017.2696939.

[74] C. You, Y. Zeng, R. Zhang, and K. Huang. "Asynchronous Mobile-Edge Computation Offloading: Energy-Efficient Resource Management". In: *IEEE Transactions on Wireless Communications* 17.11 (Nov. 2018), pp. 7590–7605. ISSN: 1536-1276. DOI: 10.1109/TWC.2018.2868710.

[75] Kyunghan Lee, Joohyun Lee, Yung Yi, Injong Rhee, and Song Chong. "Mobile data offloading: How much can WiFi deliver?" In: *IEEE/ACM Transactions on Networking (TON)* 21.2 (2013), pp. 536–550.

[76] Y. Li, M. Qian, D. Jin, P. Hui, Z. Wang, and S. Chen. "Multiple Mobile Data Offloading Through Disruption Tolerant Networks". In: *IEEE Transactions on Mobile Computing* 13.7 (July 2014), pp. 1579–1596. ISSN: 1536-1233. DOI: 10.1109/TMC.2013.61.

[77] Andrey Kolobov. "Planning with Markov decision processes: An AI perspective". In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 6.1 (2012), pp. 1–210.

[78] F. Calabrese, G. Di Lorenzo, and C. Ratti. "Human mobility prediction based on individual and collective geographical preferences". In: *13th International IEEE Conference on Intelligent Transportation Systems*. Sept. 2010, pp. 312–317. DOI: 10.1109/ITSC.2010.5625119.

[79] Filippo Rebecchi, Marcelo Dias de Amorim, Vania Conan, Andrea Passarella, Raffaele Bruno, and Marco Conti. "Data offloading techniques in cellular networks: a survey". In: *Communications Surveys & Tutorials, IEEE* 17.2 (2015), pp. 580–603.

[80] Guoju Gao, Mingjun Xiao, Jie Wu, Kai Han, Liusheng Huang, and Zhenhua Zhao. "Opportunistic Mobile Data Offloading with Deadline Constraints". In: *IEEE Transactions on Parallel and Distributed Systems* 28.12 (2017), pp. 3584–3599.

[81] Vikram Krishnamurthy. *Partially Observed Markov Decision Processes*. Cambridge University Press, 2016.

[82] Chia-Hao Hao Yu, Klaus Doppler, Cássio B Cassio B Ribeiro, and Olav Tirkkonen. "Resource sharing optimization for device-to-device communication underlaying cellular networks". In: *Wireless Communications, IEEE Transactions on* 10.8 (2011), pp. 2752–2763. ISSN: 15361276. DOI: 10.1109/TWC.2011.060811.102120.

[83] Fidan Mehmeti and Thrasyvoulos Spyropoulos. "Performance analysis of 'on-the-spot' mobile data offloading". In: *GLOBECOM - IEEE Global Telecommunications Conference* February 2016 (2013), pp. 1577–1583. DOI: 10.1109/GLOCOM.2013.6831298.

[84] A. Aijaz, H. Aghvami, and M. Amani. "A survey on mobile data offloading: technical and business perspectives". In: *IEEE Wireless Communications* 20.2 (2013), pp. 104–112. ISSN: 1536-1284. DOI: 10.1109/MWC.2013.6507401.

[85] Jeffrey G Andrews, Holger Claussen, Mischa Dohler, Sundeep Rangan, and Mark C Reed. "Femtocells: Past, present, and future". In: *IEEE Journal on Selected Areas in Communications* 30.3 (2012), pp. 497–508. ISSN: 07338716. DOI: 10.1109/JSAC.2012.120401.

[86] A. Bousia, E. Kartsakli, A. Antonopoulos, L. Alonso, and C. Verikoukis. "Multiobjective Auction-Based Switching-Off Scheme in Heterogeneous Networks: To Bid or Not to Bid?" In: *IEEE Transactions on Vehicular Technology* 65.11 (2016), pp. 9168–9180. ISSN: 0018-9545. DOI: 10.1109/TVT.2016.2517698.

[87] Hamed Shah-Mansouri, Vincent WS Wong, and Jianwei Huang. "An incentive framework for mobile data offloading market under price competition". In: *IEEE Transactions on Mobile Computing* 16.11 (2017), pp. 2983–2999.

[88] Yaron Singer. "Budget feasible mechanisms". In: *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE. 2010, pp. 765–774.

[89] Jun Deng, Rongqing Zhang, Lingyang Song, Zhu Han, and Bingli Jiao. "Truthful mechanisms for secure communication in wireless cooperative system". In: *IEEE Transactions on Wireless Communications* 12.9 (2013), pp. 4236–4245.

[90] Michael Holzhauser, Sven O Krumke, and Clemens Thielen. "A network simplex method for the budget-constrained minimum cost flow problem". In: *European Journal of Operational Research* 259.3 (2017), pp. 864–872.

[91] Alberto Del Pia, Santanu S Dey, and Marco Molinaro. "Mixed-integer quadratic programming is in NP". In: *Mathematical Programming* 162.1-2 (2017), pp. 225–240.

[92] Hadi Meshgi, Dongmei Zhao, and Rong Zheng. "Optimal resource allocation in multicast device-to-device communications underlaying LTE networks". In: *IEEE Transactions on Vehicular Technology* 66.9 (2017), pp. 8357–8371.

[93] Lin Gao, Lingjie Duan, and Jianwei Huang. "Two-sided matching based cooperative spectrum sharing". In: *IEEE Transactions on Mobile Computing* 16.2 (2017), pp. 538–551.

[94] Isabelle Brocas, Juan D Carrillo, and Manuel Castro. "Second-price common value auctions with uncertainty, private and public information: experimental evidence". In: *Journal of Behavioral and Experimental Economics* 67 (2017), pp. 28–40.

[95] D. Liu, L. Khoukhi, and A. Hafid. "Data offloading in mobile cloud computing: A Markov Decision Process approach". In: *2017 IEEE International Conference on Communications (ICC)*. IEEE. 2017, pp. 1–6. DOI: 10.1109/ICC.2017.7997070.

[96] Dongqing Liu, Lyes Khoukhi, and Abdelhakim Hafid. "Prediction based Mobile Data Offloading in Mobile Cloud Computing". In: *IEEE Transactions on Wireless Communications,(conditionally accepted; minor revision submitted)* (2017).

[97] Jinfeng Kou, Yang Xiao, and Dong Wang. "An economic user-centric WiFi offloading algorithm for heterogeneous network". In: *Mathematical Problems in Engineering* 2015 (2015). ISSN: 15635147. DOI: 10.1155/2015/341292.

[98] Pingzhong Tang and Zihe Wang. "Optimal mechanisms with simple menus". In: *Journal of Mathematical Economics* 69 (2017), pp. 54–70.

[99] Sangwoo Moon, Hongseok Kim, and Yung Yi. "Brute: Energy-efficient user association in cellular networks from population game perspective". In: *IEEE Transactions on Wireless Communications* 15.1 (2016), pp. 663–675.

[100] Qiang Fan and Nirwan Ansari. "Towards workload balancing in fog computing empowered IoT". In: *IEEE Transactions on Network Science and Engineering* (2018).

[101] Hongseok Kim, Gustavo De Veciana, Xiangying Yang, and Muthaiah Venkatachalam. "Distributed $\alpha$-optimal user association and cell load balancing in wireless networks". In: *IEEE/ACM Transactions on Networking (TON)* 20.1 (2012), pp. 177–190.

[102] Chongtao Guo, Yan Zhang, Min Sheng, Xijun Wang, and Yuzhou Li. "$\alpha$-Fair Power Allocation in Spectrum-Sharing Networks". In: *IEEE Transactions on Vehicular Technology* 65.5 (2016), pp. 3771–3777.

[103] William H Sandholm. "Population games and deterministic evolutionary dynamics". In: *Handbook of game theory with economic applications*. Vol. 4. Elsevier, 2015, pp. 703–778.

[104] Shuiguang Deng, Longtao Huang, Javid Taheri, and Albert Y. Zomaya. "Computation Offloading for Service Workflow in Mobile Cloud Computing". In: *IEEE Transactions on Parallel and Distributed Systems* 26.12 (2015), pp. 3317–3329. ISSN: 10459219. DOI: 10.1109/TPDS.2014.2381640.

[105] Xu Chen, Lei Jiao, Wenzhong Li, and Xiaoming Fu. "Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing". In: *IEEE/ACM Transactions on Networking* (2015), pp. 1–14. ISSN: 10636692. DOI: 10.1109/TNET.2015.2487344. arXiv: 1510.00888.