Northumbria University

Unmanned Aerial Vehicle-Enabled Mobile Edge Computing for 5G and Beyond

by

Liang Wang

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the
Department of Computer & Information Sciences
Northumbria University

December 2021

To my dearest parents and wife . . .

Declaration

I declare that the work contained in this thesis has not been submitted for any other award and that it is all my own work. I also confirm that this work fully acknowledges opinions, ideas and contributions from the work of others.

Any ethical clearance for the research presented in this commentary has been approved. Approval has been sought and granted through the Researcher's submission to Northumbria University's Ethics Online System on 29/08/2019.

I declare that the Word Count of this Thesis is 38003 words

Name: Liang Wang

Date: 11/10/2021

Abstract

The technological evolution of the fifth generation (5G) and beyond wireless networks not only enables the ubiquitous connectivity of massive user equipments (UEs), i.e., smartphones, laptops, tablets, but also boosts the development of various kinds of emerging applications, such as smart navigation, augmented reality (AR), virtual reality (VR) and online gaming. However, due to the limited battery capacity and computational capability such as central processing unit (CPU), storage, memory of UEs, running these computationally intensive applications is challenging for UEs in terms of latency and energy consumption. In order to realize the metrics of 5G, such as higher data rate and reliability, lower latency, energy reduction, etc, mobile edge computing (MEC) and unmanned aerial vehicles (UAVs) are developed as the key technologies of 5G. Essentially, the combination of MEC and UAV is becoming more and more important in current communication systems. Precisely, as the MEC server is deployed at the edge network, more and more applications can benefit from task offloading, which could save more energy and reduce round trip latency. Additionally, the implementation of UAV in 5G and beyond networks could play various roles, such as relaying, data collection, delivery, SWIFT, which can flexibly enhance the QoS of customers and reduce the load of network. In this regard, the main objective of this thesis is to investigate the UAV-enabled MEC system, and propose novel artificial intelligence (AI)-based algorithms for optimizing some challenging variables like the computation resource, the offloading strategy (user association) and UAVs' trajectory.

To achieve this, some of existing research challenges in UAV-enabled MEC can be tackled by some proposed AI or DRL based approaches in this thesis. First of all, a multi-UAV enabled MEC (UAVE) is studied, where several UAVs are deployed as flying MEC platform to provide computing resource to ground UEs. In this context, the user association between multiple UEs and UAVs, the resource allocation from UAVs to UEs are

optimized by the proposed reinforcement learning-based user association and resource allocation (RLAA) algorithm, which is based on the well-known Q-learning method and aims at minimizing the overall energy consumption of UEs. Note that in the architecture of Q-learning, a Q-table is implemented to restore the information of all state and action pairs, which will be kept updating until the convergence is obtained. The proposed RLAA algorithm is shown to achieve the optimal performance with comparison to the exhaustive search in small scale and have considerable performance gain over typical algorithms in large-scale cases.

Then, in order to tackle the more complicated problems in UAV-enabled MEC system, we first propose a convex optimization based trajectory control algorithm (CAT), which jointly optimizes the user association, resource allocation and trajectory of UAVs in the iterative way, aiming at minimizing the overall energy consumption of UEs. Considering the dynamics of communication environment, we further propose a deep reinforcement learning based trajectory control algorithm (RAT), which deploys deep neural network (DNN) and reinforcement learning (RL) techniques. Precisely, we apply DNN to optimize the UAV trajectory with continuous manner and optimize the user association and resource allocation based on matching algorithm. It performs more stable during the training procedure. The simulation results prove that the proposed CAT and RAT algorithms both achieve considerable performance and outperform other traditional benckmarks.

Next, another metric named geographical fairness in UAV-enabled MEC system is considered. In order to make the DRL-based approaches more practical and easy to be implemented in real world, we further consider the multi agent reinforcement learning system. To this end, a multi-agent deep reinforcement learning based trajectory control algorithm (MAT) is proposed to optimize the UAV trajectory, in which each of UAV is instructed by its dedicated agent. The experimental results prove that it has considerable performance benefits over other traditional algorithms and can flexibly adjusts according to the change of environment.

Finally, the integration of UAV in emergence situation is studied, where an UAV is deployed to support ground UEs for emergence communications. A deep Q network (DQN) based algorithm is proposed to optimize the UAV trajectory, the power control of each UE, while considering the number of UEs served, the fairness, and the overall

uplink data rate. The numerical simulations demonstrate that the proposed DQN based algorithm outperforms the existing benchmark algorithms.

Acknowledgements

First and foremost, I would like to greatly thank my supervisor, Dr. Kezhi Wang, for his persistent support on the success of my PhD study. In particular, I would like to thank Kezhi for his patience and trust during my PhD study, I could not finish this thesis successfully without him. It is my luck to be his first student. I also would like to thank my second supervisor, Prof. Nauman Aslam for providing me the opportunity for joining the department of CIS as a PGR student at Northumbria University. My sincere gratitude for Dr. Cunhua Pan and Prof. Wei Xu for their insightful guidance and constant encouragement, which has enriched my growth for being a competent researcher.

My deepest appreciation for my parents for bringing up me from a baby to a independent man. It is also my fortune to meet my dearest wife Yuqing Xia during my PhD study, who has scarified a lot for me.

Finally, I would like to thank myself for always keeping positive and never giving up in the darkest period of life.

Liang Wang December 2021

"Continuous effort — not strength or intelligence — is the key to unlocking our potential."

Winston Churchill

Contents

List of Figures

List of Tables

Abbreviations

5G	the fifth generation
UE	user equipment
AR	augmented reality
VR	virtual reality
CPU	central processing unit
MEC	mobile edge computing
UAV	unmanned aerial vehicle
RLAA	reinforcement learning-based user association and resource allocation
F-MEC	flying mobile edge computing
CAT	convex optimization based trajectory control
RAT	deep reinforcement learning based trajectory control
MAT	multi-agent deep reinforcement learning based trajectory control
DQN	deep Q network
1G	the first generation
4G	the forth generation
MCC	mobile cloud computing
QoS	quality of service
LoS	line-of-sight
AI	artifical intelligence
PER	prioritized experience replay
MINLP	mixed integer nonlinear programming
MADDPG	multi-agent deep deterministic policy gradient
AMPS	advanced mobile phone system
FDMA	frequency division multiple access
GSM	global system for mobile communications

TDMA	time division multiple access
CDMA	code division multiple access
SMS	short message service
MMS	multimedia service
GPRS	general packet radio service
EDGE	enhanced data rates in GSM
UMTS	universal mobile telecommunication system
CDMA 2000	code division multiple access 2000
TD-SCDMA	time division synchronous code division multiple access
HSPA	high speed packet access
HSPA+	evolved high speed packet access
LTE	long term evolution
WiMAX	mobile worldwide interoperability for microwave access
IP	internet protocol
DVB	digital video broadcasting
HD	high definition
IoT	internet of things
IoV	internet of vehicles
M2M	machine to machine
D2D	device to device
IMT2020	international mobile telecommunications 2020
eMBB	enhanced mobile broadband
URLLC	ultra reliable and low latency communication
mMTC	massive machine type communications
LPWAN	the low power wide area networks
eMTC	the enhanced machine type communication
NB-IoT	the narrowband internet-of-things
FAA	federal aviation administration
UAS	unmanned aircraft system
BV-LoS	beyond-visual-line-of-sight
ISM	the industrial scientific medical
MANET	mobile ad hoc network
VANET	vehicular ad hoc network

FANET	fying ad hoc network
WSN	wireless sensor network
BS	base station
3-D	3-dimensional
TSP	raveling salesman problem
PDP	pickup-and-deliver problem
BCD	block coordinates descent
SCA	successive convex approximation
SWAP	size, weight, and power
ETSI	european telecommunications standards institute
ISG	industry specication group
RAN	radio access network
MNO	mobile network operator
ASP	application service provider
CDN	content delivery network
IaaS	infrastructure-as-a service
V2V	vehicle to vehicle
DoS	disk operating system
GPS	global positioning system
ANN	artificial neural network
DNN	deep neural network
C-RAN	cloud radio Access Network
ES	exhaustive search
LE	local execution
RO	random offloading
GO	greedy offloading
OFDMA	orthogonal frequency division multiple access
H-MEC	heterogeneous MEC
DRL	deep reinforcement learning
MMKP	multiple-choice multi-dimensional 0-1 Knapsack problem
DDPG	deep deterministic policy gradient
RM	random moving
CM	cluster moving

OMA	orthogonal multiple access
NOMA	non-orthogonal multiple access
ET	energy transmitter
WPT	wireless power transfer
AP	access point
D-DQN	double DQN
TS	time slot
MDP	markov decision process
TD	temporal difference
ML	machine learning
HRL	hierarchical reinforcement learning
AV	autonomous vehicle

Chapter 1

Introduction

1.1 Motivation

In the last few decades, human have witnessed the tremendous evolution of wireless communications, which is initially designed to provide voice-based service in the first generation (1G) and eventually developed as an indispensable part with ubiquitous multimedia service. Driven by the launch of forth generation (4G), the forthcoming fifth generation (5G) is expected bring up to 10 Gbps data rate, 1 ms round trip latency, 100 % coverage, and 90 % energy reduction [2], for the beyond 5G, it will achieve 1 Tbps/second data rate, 3-5 times spectral efficiency. The 5G and beyond network not only boost the development of various kinds of user-oriented mobile applications, such as augmented reality (AR), virtual reality (VR), online gaming, but also cultivate myriad of industry-oriented applications in the domain of industry automation, e-healthcare, connected vehicles. These kinds of emerging applications are normally time consuming, energy consuming, high computational. In the early stage of 4G, the integration of mobile cloud computing (MCC) is envisioned as a suitable technique, which offers storage, computation resource for user equipments (UEs) or mobile devices in a centralized architecture. However, as the requirement of quality-of-service (QoS) of UEs is greatly enhanced, MCC is no longer a qualified choice and it has to face the issues in terms of latency, security, coverage, and data rate. Consequently, mobile edge computing (MEC) has became a significant technique, which is designed to tackle the issues for MCC and provide sufficient resource at the edge network.

Furthermore, in order to achieve the different QoS requirements in 5G, unmanned aerial vehicles (UAVs), which are also known as drones, remote piloted aircrafts, are envisioned as a vital role in wireless communications, due to their high mobility, low cost, fast and flexible deployment. Primarily, UAVs are used in the domain of military, which is deployed to reduce pilot losses. Due to the constant cost reduction of UAVs' production and maintenance, numerous UAV-based applications have emerged, such as weather monitoring, fire detection, traffic control, delivery, emergence search and rescue, communication, etc [3]. Essentially, the integration of UAVs in wireless communication systems is expected to provide connectivity for ground devices that are not under the coverage of communication infrastructures, due to the shadowing, terrain, damage caused by natural disasters [4]. In most wireless scenarios, the direct line-of-sight (LoS) communication links can be established between UAVs and ground devices, which significantly enhance the system performance. Additionally, considering the dynamic communication environment, the state of UAVs can be flexibly adjusted through appropriately control UAVs' mobility.

In order to further improve the communication performance and achieve better QoS, the research of combination of UAV and MEC is becoming more and more popular both in academia and industry. However, UAV-enabled MEC is still in its early state and it has huge number of research challenges, such as the deployment of UAVs (UAVs' trajectory control, path planning), resource allocation, security issues, user association (offloading strategy), energy efficiency of UAVs, channel modeling. Existing traditional algorithms are normally based on convex optimization, dynamic programming, and evolutionary computing, which are time consuming, near optimal and need plenty of iterations in order to achieving considerable performance. In addition, when the environment changes, like the UEs' distribution, the channel model, the algorithms may not work or even re-design, which are not practical.

Given the recent advances in machine learning or artificial intelligence [97], the combination of deep neural networks (DNNs) [98] and reinforcement learning (RL) [99], i.e., deep reinforcement learning (DRL) has become a hot research topic. In DRL, an agent is assumed to interact with the environment for learning the optimal policy with the aid of exploration. Compared to traditional RL, DRL facilitates more accurate convergence and approximation by exploiting the power of DNNs for estimating the associated functions in RL [100]. The great potential of DRL in solving complex control problems

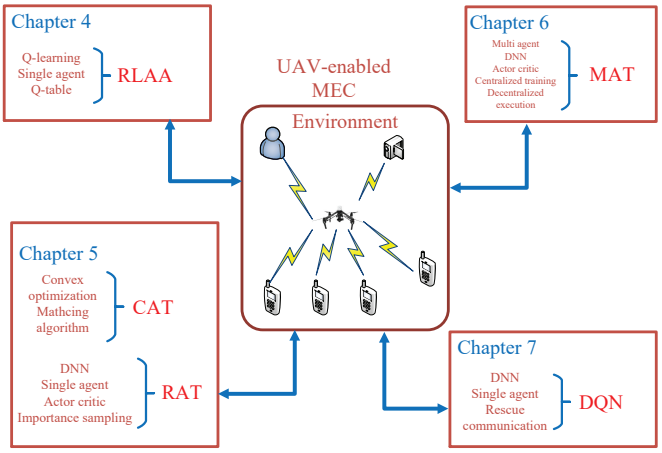


FIGURE 1.1: Thesis contribution

has also been demonstrated in [29, 31, 32, 101, 102]. In [29], Mnih *et al.* introduced the deep Q network (DQN) philosophy, which ignited the field of DRL. For instance, Wang *et al.*[101] systematically investigated the problem of distributed Q-learning aided heterogeneous network association in the content of energy-efficient Internet of things (IoT). In order to improve the training procedure, DQN relies on a pair of techniques namely, experience replay and target networks. For the sake of tackling the typical over-estimation problem of RL, a double DQN (D-DQN) was proposed by Van Hasselt *et al.* [32]. However, DQN may suffer from the curse of high-dimensional action spaces and cannot be readily applied to continuous domains. Thus, motivated by this, Lillicrap *et al.* [31] proposed a deep deterministic policy gradient (DDPG) technique based on the so-called actor-critic architecture, which can be readily applied for a range of challenging problems. A comprehensive survey of multi-agent RL, have also been provided by Bu *et al* [102].

Thus, the AI-based techniques can be applied to tackle the challenges in UAV-enabled MEC, which adapt the dynamics of the environment after training procedure, and obtain the optimal solutions in real-time. Therefore, the main goal of this thesis is to design novel AI-based algorithms for solving the key challenges in UAV-enabled MEC.

1.2 Thesis Contribution

We briefly demonstrate the thesis contribution in Fig. 1.1. Compared with existing traditional approaches, our proposed approaches can tackle some of challenges in UAV-enabled MEC sufficiently, especially considering the dynamics of the complicated environment. Precisely, the main contributions of this thesis are summarized as follows:

- We first study the multi-UAV-enabled MEC system, where several UAVs that fly in circles are deployed to serve UEs. We assume each UE has its computation-intensive task to be executed in each time slot, whose time duration could be flexibly changed according to the requirement of communication environment. In this case, the energy minimization of UEs for task execution is regarded as a mixed integer nonlinear programming (MINLP) problem, which involves the user association between UEs and UAVs, and the resource allocation from UAVs to UEs. Thus, we propose a reinforcement learning-based user association and resource allocation (RLAA) algorithm to solve the optimization problem efficiently. We also analyze the performance of proposed RLAA algorithm by comparing it with other traditional benckmarks. In small scale, RLAA can achieve the optimal performance with exhaustive search. While in large-scale cases, RLAA also has considerable performance gain over other typical algorithms, such as local execution and random offloading.
- Second, we consider a platform of flying mobile edge computing (F-MEC), which enables task offloading from ground UEs. The objective is to minimize the overall energy consumption of UEs while guaranteeing all tasks of UEs can be sufficiently executed through optimizing the UAVs' trajectory, user association between UAVs and UEs, the resource allocation from UAVs to UEs. The optimization problem involves both continuous variables and discrete variables, which is difficult to solve in general. To this end, we first propose a convex optimization based trajectory control algorithm (CAT), which solve the problem in an iterative way. Then, in order to make the real-time decision and consider the dynamics of the communication environment, we further propose a deep reinforcement learning based trajectory control algorithm (RAT), which apply the prioritized experience replay (PER) scheme to improve the convergence of the training procedure. Different from the proposed CAT algorithm that relies on the initial feasible solution and needs iterations, the proposed RAT

algorithm can obtain the optimal solutions in real-time and can be adapted to the dynamics of the environment. From the experimental results, both CAT and RAT have considerable performance and outperform other benckmark algorithms.

- Third, we study an UAV-enabled MEC framework, where a group of UAVs fly over the target area and support UEs on the ground. We consider the geographical fairness of UEs, the fairness of UE-load of each UAV, the energy consumption of UEs while optimizing the UAVs' trajectory, the offloading decision of UEs. To address this optimization problem, a multi-agent deep reinforcement learning (DRL) based trajectory control algorithm is proposed, in which a well known multi-agent deep deterministic policy gradient (MADDPG) method is applied. In the proposed MAT algorithm, a group of agent control their dedicated UAV and cooperate with each other to support UEs. The simulation results show that MAT can consistently outperform other traditional algorithms in testing process.
- Fourth, we extend the implementation of UAV in emergence communication system, where the terrestrial network infrastructure is severely destroyed by the natural disaster. In order to provide temporary communication and assist rescue, we assume an UAV is deployed to serve ground UEs without the help of terrestrial base station and access point. We introduce a DQN based algorithm to optimize the UAV trajectory with discrete manner and power control of UEs. We further aim at maximizing the overall uplink data rate from ground UEs to the UAV, while considering the fairness. The simulation results prove that our proposed approach outperforms other traditional benckmarks.

This thesis includes the following journal and conference papers:

1.2.1 Journal Papers

- **L. Wang**, K. Wang, C. Pan, W. Xu, N. Aslam and A. Nallanathan, "Deep Reinforcement Learning Based Dynamic Trajectory Control for UAV-assisted Mobile Edge Computing," in IEEE Transactions on Mobile Computing, doi: 10.1109/TMC.2021.3059691.
- **L. Wang**, K. Wang, C. Pan, W. Xu, N. Aslam and L. Hanzo, "Multi-Agent Deep Reinforcement Learning-Based Trajectory Planning for Multi-UAV Assisted Mobile Edge Computing," in IEEE Transactions on Cognitive Communications and Networking, vol. 7, no. 1, pp. 73-84, March 2021, doi: 10.1109/TCCN.2020.3027695.

- **L. Wang**, K. Wang, C. Pan, X. Chen and N. Aslam, "Deep Q-Network Based Dynamic Trajectory Design for UAV-Aided Emergency Communications," in Journal of Communications and Information Networks, vol. 5, no. 4, pp. 393-402, Dec. 2020, doi: 10.23919/JCIN.2020.9306013.

1.2.2 Conference Papers

- **L. Wang**, P. Huang, K. Wang, G. Zhang, L. Zhang, N. Aslam, K. Yang, "RL-Based User Association and Resource Allocation for Multi-UAV enabled MEC," 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), 2019, pp. 741-746, doi: 10.1109/IWCMC.2019.8766458.

1.2.3 Submitted Papaers

- **L. Wang**, K. Wang, C. Pan, N. Aslam. "Joint trajectory and passive beamforming design for intelligent reflecting surface-aided UAV communications: A deep reinforcement learning approach," in IEEE Transactions on Green Communications and Networking.
- M. Khalid, **L. Wang**, K. Wang, C. Pan, N. Aslam "Learning-Based Path Planning for Long-Range Autonomous Valet Parking," in IEEE Transactions on Intelligent Transportation Systems.
- K. Wang, **L. Wang**, C. Pan, R. Hong "Deep-Reinforcement-Learning-Based Dynamic Resource Management For Flexible Mobile-Edge-Computing," in IEEE Vehicular Technology Magazine.

1.3 Thesis Structure

We introduce the structure of this thesis as follows:

- **Chapter 2** introduces the relevant topics that are presented in this thesis. In the first part of this chapter, we present necessary background on 5G and beyond network. Also, the motivation of developing 5G is discussed in the following subsection. Then, we study the main use cases and technical metrics for 5G. In the second part, we introduce the essential information about UAV communication. In the third part, we

discuss some fundamental knowledge about MEC technique, such as the motivation, advantages, applications, and research challenges. Furthermore, the technique of combining UAV and MEC is introduced in the following part. Finally, in order to tackling the key challenges in UAV-enabled MEC, the machine learning technique is introduced, such as the basics of machine learning, artificial neural network.

- **Chapter 3** gives some of related work about UAV-enabled MEC system and deep reinforcement learning.
- **Chapter 4** presents an advanced RL-based user association and resource allocation scheme in multi-UAV enabled MEC. The proposed scheme enables multi-UAV flying in circles to serve a group of UEs and allows optimal user association and resource allocation from UAVs to UEs in real-time.
- **Chapter 5** introduces a flying-mobile edge computing (MEC) platform. In order to provide the optimal QoS to UEs, a convex optimization based trajectory control algorithm is proposed to optimize the UAV trajectory, user association, and resource allocation, which has considerable performance in terms of energy consumption of UEs. While considering the dynamics of environment, a DRL-based algorithm is further proposed, which can obtain the optimal solutions in real-time.
- **Chapter 6** further discusses the framework of UAV-enabled MEC and proposes a multi-agent reinforcement learning based algorithm to control a group of UAVs. Additionally, the proposed algorithm considers the fairness of UEs served by UAVs and fairness of UE-load of UAVs during the entire process.
- **Chapter 7** studies another application of UAV in emergence situation, where the terrestrial base station is destroyed by natural disaster. The UAV is deployed as a flying base station and a deep Q network based algorithm is proposed to optimize the UAV trajectory with discrete manner, aiming at maximizing the data rate from UEs to the UAV and guaranteeing the fairness of UEs.
- **Chapter 8**: summaries the proposed work presented in this thesis and gives some of future work within the scope of this thesis.

Chapter 2

Background

In this chapter, we give the general background of relevant topics related to this thesis. Particularly, we first introduce the background, the motivation, the use cases, and technical metrics for 5G and beyond network. Then, we discuss one of key techniques in 5G named UAV-assisted wireless communication. In this section, some key use cases, advantages and challenges of UAV in wireless communication are discussed. Furthermore, we also overview the essence of MEC, including the motivation, advantages, applications and research challenges. In addition, the technique of combining UAV and MEC, which is named UAV-enabled MEC, is discussed in the next section. Finally, some background information about machine learning, such as the categories of machine learning, artificial neural network are introduced.

In this chapter, we give the general background of relevant topics related to this thesis. In Section 2.1, we first introduce the relevant knowledge of 5G and beyond network, including the research background, motivation, use cases. Then, in Section 2.2, we also first introduce the background of UAV, the categories of UAV in wireless communication system, use cases, advantages and challenges. Followed by 2.2, we introduce MEC in 2.3. The motivation, advantages, applications, and challenges are briefly discussed in this section. Furthermore, in Section 2.4, we introduce the technique of combining UAV and MEC. Finally, in Section 2.5, we discuss machine learning, including the supervised learning, unsupervised learning, reinforcement learning, DQN, DDPG and artificial neural network.

2.1 5G and Beyond Network

In this section, we give the essential research background of 5G and beyond as follows.

2.1.1 Research Background

In the last few decades, human have witnessed the tremendous evolution of wireless communications. In the late 20th century, the first generation (1G) wireless system was envisioned as the symbol of changing human thinking. It is widely acknowledged that the well-known advanced mobile phone system (AMPS) was viewed as the 1G wireless system, which was based on frequency division multiple access (FDMA) and frequency modulation. Note that in the 1G era, AMPS only supported voice calls and only had up to 2.4 Kbps data rate [5], although it had forced massive scientists, researchers and engineers to work on data transportation, and encryption technique. In 1991, the second generation (2G) wireless communication was launched by global system for mobile communications (GSM) in Finland, which was based on time division multiple access (TDMA) and code division multiple access (CDMA). Different from 1G, 2G could support the short message service (SMS), international roaming, picture message, multimedia service (MMS). Besides, 2G also enabled up to 64 Kbps and digital signals for voice transmission [5, 6]. Based on GSM, the general packet radio service (GPRS) was implemented, which was viewed as the major step towards 2.5G. In GPRS, as the switching protocol was based on packet switching instead of circuit switching, the data rate can reach up to 144 Kbps. Shortly after, enhanced data rates in GSM (EDGE) is applied as the symbol of 2.75G. In the early 2000s, the third generation (3G) wireless communication was officially launched, which aims at providing up tp 2 Mbps data rate. To achieve this, three standards of 3G, such as universal mobile telecommunication system (UMTS), code division multiple access 2000 (CDMA2000), and time division-synchronous code division multiple access (TD-SCDMA) were developed, which allows web browsing, email, fax, and navigation service. Then, based on 3G, high speed packet access (HSPA) and evolved high speed packet access (HSPA+) were launched, which were also known as 3G+ and can support up to 5.76 Mbps data rate [6]. In late 2000s, long term evolution (LTE) and mobile worldwide interoperability for microwave access (WiMAX) were generally viewed as the standards of the forth generation (4G) wireless

communication. The indication of 4G was to implement reliable communication solutions based on internet protocol (IP), which means some emerging applications, such as multimedia messaging service (MMS), digital video broadcasting (DVB), high definition (HD) TV that rely on higher data rate and lower latency can be fulfilled [7–9].

2.1.2 The Motivation of 5G

In the era of 4G, the number of connected device and mobile data are explosively increased, and new emerging applications are unquestionable becoming popular recently. It is reported that the video traffic has occupied more than half of the global data traffic since 2012. The average mobile user is expected to download about 1 terabyte of data annually by 2020 [10]. Besides, the academia and industry are both working towards exploring other applications in the domain of augmented reality (AR), internet of things (IoT), internet of vehicles (IoV), machine to machine (M2M) communication, device to device (D2D) communication, e-healthcare. In order to satisfy the requirement of these enormous applications, the present 4G LTE network is no longer a sufficient choice. For example, the standard 4G LTE network can theoretically has 150 Mbps data rate in the downlink, which can only be able to support full HD video streaming. However, most of current video streaming are normally 4K. Besides, in M2M and IoT networks, thousands of devices are required to be connected, while 4G is only designed to support maximally 600 connected devices in single cell [8]. Also, the IoV network is extremely sensitive to round trip latency, which is about 1 ms, while 4G can only have 15 ms latency. Hence, it is time to shift to the future fifth generation (5G) wireless communication and develop various techniques to satisfy the exponential rise of the emerging applications.

2.1.3 Use Cases for 5G

In 5G, it is expected to realize a great deal of applications, such as connected vehicles, remote controlled robotics, mobile computing, AR, VR, etc. According to the definition of international mobile telecommunications 2020 (IMT2020), 5G has three categories of user cases, which are shown in Fig. 2.1.

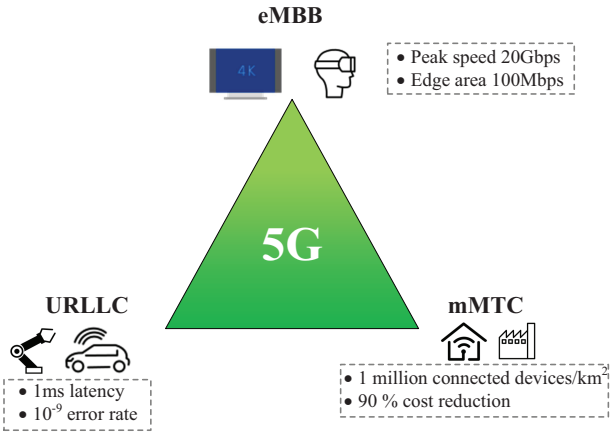


FIGURE 2.1: Use cases of 5G

- **Enhanced mobile broadband (eMBB)**: this use case is considered to adapt to different scenarios, such as wide-area coverage, hotspots, etc. For the wide-area coverage, the network is designed to provide higher data rate when the user is in high mobility, while the goal of the seamless coverage is to guarantee that users can be and kept connected, no matter where they move to. For the hotspots, when the user density is high, the network needs to provide high traffic capacity [11].
- **Ultra reliable and low latency communication (URLLC)**: The target of URLLC is to meet the extremely strict requirement for network reliability, latency in some scenarios, such as smart grids, autonomous system, remote control, tactile internet [12], etc.
- **Massive machine type communications (mMTC)**: the so-called mMTC is to offer connections to a massive of devices that are not sensitive to the latency and data volume. To realize this, the low power wide area networks (LPWAN) is triggered in the machine-to-machine (M2M) domain. Also, other potential technologies, like the enhanced machine type communication (eMTC) and the narrowband internet-of-things (NB-IoT) are developed to extend battery life and reduce the cost of devices in the network [13].

2.2 UAV-assisted Wireless Communication

In order to meet the QoS requirement of 5G, UAV-assisted wireless communication is becoming a hot topic. In this section, we illustrate the technique of UAV-assisted

wireless communication.

2.2.1 Research Background of UAV

Historically, UAVs are normal known as drones that are remotely controlled by pilots or embedded computer programs. They are widely applied in the domain of military for remote surveillance, attack, sensing, etc, which are originally designed to reduce the pilot losses. Due to the constant reduction in the cost of UAVs and the mature technology in UAVs' manufacturing, UAVs are becoming vital in enormous commercial applications, such as traffic control, search, rescue, delivery, agriculture, telecommunication, photography, etc. For standardizing the operation of launching UAVs, the U.S. Federal Aviation Administration (FAA) released the operational rules for controlling small unmanned aircraft systems (UASs) that are within 25 kg in 2016. One year after, in order to further explore the usage of UAVs, including beyond-visual-line-of-sight (BVLoS) flights, night-time operations, flight above people, FAA also launched a national program named "Drone Integration Pilot Program". These programs and guidance pave the way for the growth of UAV industry, whose scale enormously increased in the next decade.

Generally, according to the usage and applications of UAVs, there are massive types of UAVs. The commercial market prefer to categorize UAVs into two types, such as fix-wing UAVs, and rotary-wing UAVs, according to the wing configuration. Specifically, fixed-wing UAVs have advantages in terms of flying speed, payload, which are suitable for long-distance flying task in general, but they may rely on a launcher for landing or taking off, and can not hover in the sky. Compared to fixed-wing UAVs, rotary-wing UAVs are more flexible, and can land or take off vertically, and even keep static at a certain location. The details about the classifications of the types of UAVs can be found in [14].

2.2.2 Categories of UAV-assisted Wireless Communications

In order to achieve the metrics in terms of latency, throughput, coverage in 5G, UAVs have been a vital component in wireless communications. If the configuration of UAVs is properly designed, they can provide much reliable solutions in most practical scenarios

with relatively tolerant cost. According to the description in [15], the application of UAVs in wireless communications can be normally categorized into 4 types:

1. **Direct link**: In this setting, the direct LoS communication links between UAVs and ground UEs can be easily established over the industrial scientific medical (ISM) 2.4 GHz band. The ground UEs can be a remote controller, ground station and so on. However, the communication range will be significantly influenced by the complicated environment. For example, when UAVs are deployed in urban areas. High buildings, trees will block the communication signals, which will have negative influences on latency and throughput. Besides, this setting also relies on gateway for accessing the Internet, which will definitely cause longer latency. Also, when the interference is severe, the QoS and secure are not guaranteed, which means this simple setting is not a good choice for large-scale deployment of UAVs.
2. **Satellite communications**: With the help of global coverage provided by satellites, the UAVs can establish the link between any ground gateways through the relay of satellites. This setting is particularly useful for the UAVs that are deployed in ocean, desert, or other areas without terrestrial coverage. Furthermore, satellites can also provide navigation and location service for UAVs. However, there are still challenges for satellite-assisted UAV communications. First of all, the propagation loss will be extreme severe as the distance between satellites and UAVs are quite long. Secondly, as UAVs are normally sensitive to their physical size, weight, and power, they can not carry equipment for satellite communication. Besides, operating satellites is extreme expensive, which limits its usage in commercial market.
3. **Ad Hoc network**: Mobile ad hoc network (MANET) is a type of self-organizing network that does not rely on terrestrial infrastructure. It can dynamically adjust with the environment and offer peer-to-peer communications with different mobile devices, such as laptops, smartphones. As mobile devices in MANET can change their locations randomly, the communication links will also change frequently. Additionally, in order to establish the link between two far devices, MANET enables multi-hop relaying among other devices, which significantly reduce the energy consumption and end-to-end delay. As the extension of MANET, vehicular ad hoc network (VANET) and flying ad hoc network (FANET) are designed to integrate UAVs for serving ground vehicles that are in high mobility. However, this setting is only suitable for small-scale FANET network.

4. **Cellular network**: Have discussed the above settings, it is clear to see that they cannot support massive UAV communications with sufficient performance or cost efficiency. The integration of UAVs in cellular network is envisioned as a good choice. With the aid of ubiquitous coverage of cellular network and mature wireless communication techniques, the metrics of applying UAVs can be easily met. For example, the UAVs can be deployed as relaying node in cellular network, for achieving nearly 100 % coverage and support 1 ms latency.

2.2.3 Use Cases of UAVs in Wireless Communications

According to the components that UAVs play in wireless communications, there are three typical use cases, which are described as follows:

1. **UAV-aided ubiquitous coverage**: In this use case, UAVs can be viewed as flying or aerial BSs to provide seamless coverage within certain areas. Compared to traditional terrestrial BSs, the UAVs can flexibly adjust the altitude and location according to communication conditions. Also, for temporary events, natural disasters, UAVs are particularly useful to enhance the system performance and recover the communication functionality.
2. **UAV-aided relaying**: In this use case, UAVs can be employed as relaying nodes to extend the coverage of BSs and strengthen the connectivity for UEs that are far from BSs.
3. **UAV-aided information dissemination and data collection**: In IoT or wireless sensor network (WSN), UAVs can be employed as aerial access points (APs) to collect data from ground nodes.

2.2.4 Advantages and Challenges of UAV-assisted Wireless Communications

In this subsection, we summary the advantages of UAV-assisted wireless communications as follows:

1. **High altitude**: Compared to traditional BSs with fixed altitude, UAVs can serve UEs with much higher altitude. Precisely, for urban micro deployment, the height of BSs is normally about 10 m, for urban macro deployment, the height is around

25 m [16], while for UAV-aided cellular network, UAVs are allowed to fly up tp 122 m. Additionally, in terrestrial networks, traditional BSs only support 2-D coverage while UAVs can offer 3-D coverage.

2. **High LoS probability**: In terrestrial wireless networks, ground UEs often suffer from severe path loss because of the influence of shadowing and fading. While in UAV-assisted wireless communications, the air-ground and LoS channel can be established with a much higher profitability, which offers much reliable links. Moreover, as the LoS links are less sensitive to the change in time and frequency domain, the techniques of communication schedule and resource allocation can be sufficiently realized, comparing to the traditional wireless networks.
3. **Flexible 3-D mobility**: Compared to the fixed location wireless networks, UAVs can move with high speed in 3-D space. They can flexibility fly to certain locations to adjust the dynamic communication conditions and realize different QoS requirements.

Although UAVs play a significant role in wireless communication, they still have to face massive research challenges. In what follows, we summarize some of challenges:

1. **UAV placement**: In order to achieve the optimal communication performance, the location and altitude of UAVs should be appropriate designed, which is extremely flexible, fast and short-term. In addition, when the communication environment is complicated, the obstacle avoidance is also needed to be considered.
2. **Channel modeling**: Practically, the channel characteristics of UAV-assisted wireless communication is more sophisticated than the terrestrial wireless communication in various aspects. In general, there are three types of communication links named UAV-BS link, UAV-UAV link, and UAV-UE link. The UAV-UAV link can be simply characterized by free-space path-loss channel [17, 18]. While the other two links are more complicated and their characteristics may vary with different communication environments. Some of channel models, such altitude-dependent channel parameters [19], elevation angle-dependent channel parameters [20], depression angle-depdendent excess path loss model [21], elevation angle-dependent probabilistic LoS model [22], 3GPP GBS-UAV channel model, are summarized in [15].
3. **Path planning (Trajectory control)**: Different from the traditional terrestrial network, the performance of deploying UAVs in wireless communication network is strictly limited by massive constraints in the domain of UAV trajectory or path planning. We list some of constraints as follows:

- Minimize/maximum altitude
- Initial/final location
- Maximum/minimum speed
- Maximum/minimum acceleration
- Obstacle avoidance
- Collision avoidance

The above constraints make the optimization of UAV trajectory difficult in general. On one hand, the optimization problem is normally nonconvex with respect to the trajectory variables. On the other hand, the optimization problem also involves infinite variables related to continuous time. There are several directions for tackling the optimization of UAV trajectory. First of all, for the initial/final path planning, the optimization problem can be transferred to the classic traveling salesman problem (TSP) or pickup-and-deliver problem (PDP). Secondly, in order to make the optimization problem more tractable, the UAV trajectory can be discretized and be solved with finite variables. Thirdly, the most difficult method for optimizing the UAV trajectory is to tackle it directly, which is nonconvex and involves continuous variables. Most approaches are based on block coordinates descent (BCD) or successive convex approximation (SCA) techniques and they are normally sub-optimal and need iterations.

4. **Interference management**: As the LoS communication link incurs in UAV-assisted wireless communication system with high probability, the interference is an indispensable issue. To this end, some novel techniques for interference management should be proposed.
5. **SWAP limitations**: In traditional terrestrial wireless communication systems, ground BSs or UEs normally equip with a stable power supply from grid network or rechargeable battery. However, UAVs deployed in wireless communication systems have stringent constraints in size, weight, and power (SWAP), which limit the UAVs' endurance and communication performance. On one hand, UAVs can only equip with compact/tiny communication hardware for catering the limitation in size and weight, which have to sacrifice some performance gain in terms of latency, throughput. On the other hand, considering the communication related energy consumption, UAVs need to consume more energy for moving and hovering, which is more important and critical. To this end, the energy-efficient design of UAVs should be carefully

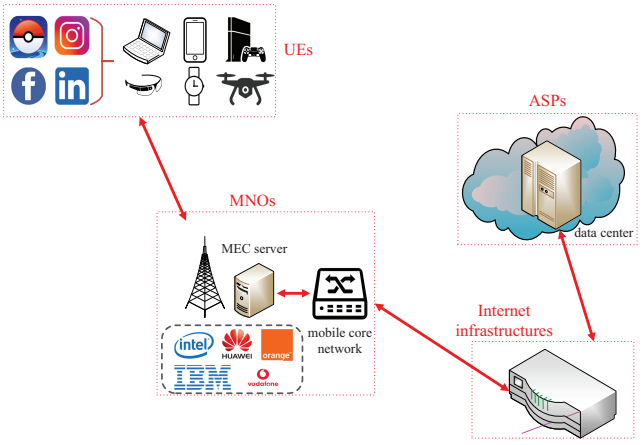


FIGURE 2.2: Structure of MEC

considered.

6. **Security issue**: In order to serve more and more devices in UAV-assisted wireless communication systems, security issue is another research challenge. To avoid the network jamming or attacks from eavesdroppers, there is a need to develop more advanced communication techniques in protocol and physical layer.

2.3 MEC

In this section, some fundamental elements of MEC are discussed.

2.3.1 The Motivation of MEC

Nowadays, more and more emerging mobile applications become popular recently and most of them relay on extensive data and services provided by cloud computing servers from remote data centers, which poses a huge challenge in network load. It is expected that the demands of bandwidth will constantly double each year [23], and the novel augmented reality based mobile applications will lead to higher requirements for bandwidth and latency. In order to guarantee adequate QoS, current solution is to apply mobile cloud computing (MCC) technique, which deploys storage, computation at the centralized cloud. However, MCC is no longer a good choice in terms of latency, security, coverage, and data rate transmission. Thus, thanks to the advanced 5G, MEC is developed to address the challenges that MCC cannot tackle. The concept of MEC was

first presented by european telecommunications standards institute (ETSI) and industry specification group (ISG), which is further defined recently as 'Multi-Access Edge Computing'. According to the definition of ETSI, MEC is one of the emerging technology in 5G networks, which enables IT service environment and cloud computing equipments at the edge of the mobile network. Through radio access network (RAN), the direct connection between UEs and the nearest MEC server can be established. Fig. 2.2 illustrates the architecture of MEC. Specifically, the MEC system consists of four components: 1) UEs. 2) network infrastructures owned by mobile network operators (MNOs), such as base stations, access points, core network. 3) Internet infrastructures like routers. 4) application infrastructures provided by application service providers (ASPs), such as MEC server, data center, content delivery network (CDN).

2.3.2 Advantages of MEC

Compared to the centralized MCC technique, the distributed MEC is well-known in terms of low latency, proximity, high bandwidth. There characteristics can bring massive profits for both consumers, MNOs, and ASPs:

1. For consumers, they can experience mobile applications (AR, VR, face recognition, etc) that rely on low latency, high through, high computational resource through offloading their tasks to MEC server. Furthermore, considering the limited battery of mobile devices, the energy efficiency can be further improved, which also improves the QoS.
2. For ASPs, they can obtain profits via building a MEC-based infrastructure-as-a-service (IaaS) platform at the edge of network. Besides, due to the characteristics of MEC, more mobile applications can be developed.
3. For MNOs, as the RAN access is allowed for third party companies, the deployment of applications and service is more complex, which brings income for MNOs according to the used storage, computation resource, bandwidth. Additionally, applying MEC server at the edge of network will significantly reduce the network load as most of service requests can be solved by MEC server instead of sending them to the data center through the core network.

2.3.3 Applications of MEC

In this subsection, we introduce some recognized applications of MEC.

1. **Augmented Reality (AR)**: AR-based applications gain their popularity recently. The idea of AR is to apply sound and visual contents for combining real and virtual environment. However, AR-based applications normally require high computational resource and low latency, in order to provide adequate QoS.
2. **Healthcare**: Recently, some human-interaction and wearable devices, such as smartphone, Google glass, smart watch are widely applied in healthcare. As these devices are equipped with gyroscopes, accelerometers, these devices can help to prevent fall incidents when patients have strokes. For example, recent U-fall that employ smartphone and MEC technology is proposed. With the combination of fall detection algorithm, smart sensors on smartphones, and MEC server, U-fall is capable of providing real-time, accurate fall detection, which is more reliable and dependable.
3. **Connected Vehicles**: Recently, as vehicles are capable of connecting with other vehicles on the road via an Internet access. However, this V2V communication relies on MEC technology. When traffic jam or car accident happens, the vehicle can communication with other appropriate vehicles on the road and inform them risks.
4. **Video Analytics**: Surveillance cameras are widely deployed in secure system. However, as the number of surveillance cameras is increasing, the data volume for storing is also increasing. The traditional centralized client-server architecture is not capable of managing millions of cameras, which inevitably stress the network. To this end, MEC can assist surveillance cameras with a distributed manner. For example, with the help of MEC, surveillance cameras can monitor traffic congestion and work in face recognition applications, which can help to reduce the crime.

2.3.4 Research Challenges in MEC

Having discussed the above applications of MEC, we introduce some of research challenges in MEC.

1. **Computation Offloading**: Generally, in MEC system, most mobile applications are sensitive to resource and power. Basically, there are two categories of offloading strategies, which are partial offloading, and binary offloading. For partial offloading,

the task is divided into two parts, one can be offloaded to the MEC server and another part can be executed locally. For binary offloading, the task can be executed locally or offloaded to MEC server. Considering the requirement of QoS of UEs, it is crucial to optimization the offloading strategy. On one hand, the optimization of offloading should be real-time, especially in complicated scenarios. On the other hand, considering the constraints in computation and storage resource of MEC server, the process of optimizing offloading strategy is extremely difficult.

2. **Storage and Computation Resource Allocation**: Although the storage and computation resource that MEC server can provide is much more sufficient than UEs, the resource of MEC is still insufficient in some cases, especially when large number of UEs need to be served. Thus, the resource allocation of MEC should be optimized for guaranteeing adequate QoS of UEs.
3. **Security and Privacy Issues**: In the MEC ecosystem, the security issue mainly includes network security, core network security, MEC server security, virtualization security, and end devices security. For instance, when deploying MEC server at the edge of network, the network management policy will be more complicated and the MEC server may be exposed by various DoS attacks and viruses, which can damage the MEC server can lead to network traffic. Therefore, the appropriate security mechanisms for identification, authentication and data encryption should be implemented.

   While deploying MEC server that is close proximity to the UEs, the privacy information, such as credit card, emails, password, location may become easy to be obtained by attackers. To this end, some privacy-preserving aggregation schemes, such as homomorphic encryption can be applied [24]. Furthermore, the location information is another privacy issue. Current mobile applications obtain the location information or provide navigation service through the global positioning system (GPS). In order to solve this issue, several security mechanisms can be applied to confuse the attacker, such as the mobishare system [25].

2.4 UAV-enabled MEC

In recent years, researchers are expecting to integrate UAV into MEC systems [26], due to the characteristics of UAV and MEC. In UAV-enabled MEC, the UAV can be viewed

as a UE that has computational intensive tasks, a flying MEC server that has efficient computation resource, or a flying relaying node to assist UEs to offload their tasks. Compared to the traditional terrestrial MEC system, UAV-enabled MEC can be flexibly deployed and has promising advantages in different scenarios, such as wild, forest, desert, or even crowded areas, where terrestrial MEC system can not be reliably deployed. Some leading companies, like Google, Huawei, Nokia, Amazon have established their UAV-enabled MEC systems, and more and more applications that rely on it have been developed.

2.4.1 Applications of UAV-enabled MEC

In this subsection, some overview applications of UAV-enabled MEC are discussed. According to the description in [27], there are mainly four applications.

1. **Hotspots**: In terrestrial MEC system, massive UEs will temporarily gather within a crowded area and they constantly generate computation intensive tasks. For example, in the city center, there can be more than one million people simultaneously using their phones for taking different applications in rush hour. Traditional MEC system may not be reliable because of the extremely high traffic volume and tremendous computation tasks. In this case, UAV-enabled MEC can be temporarily exploited as a powerful assistance to improve the QoS of UEs and reduce the outage probability.
2. **Without terrestrial MEC system**: Although terrestrial MEC system is ubiquitous within most of cellular networks, which cover most of areas, there are still many areas, such as forests, deserts, wilderness, that are out of the coverage of cellular networks. The government and organization need to constantly monitor the environment and take necessary measures to protect it. However, it is impossible to establish the terrestrial MEC system, which is affordable, unreliable and inconvenient. In this case, deploying UAV-enabled MEC is a good alternative.
3. **Battle field**: UAV-enabled MEC could be a much sufficient and reliable system. For example, in battle fields, there are many computational intensive tasks generated by special missions like missile navigation. Establishing the terrestrial MEC system is not reliable which can be easily destroyed and sensed by the enemy. On the contrary, UAV-enabled MEC is more flexible and cost-efficient.

4. **Unforeseen disasters and resuce**: When the unforeseen disasters happen, the terrestrial MEC system would be severely destroyed. To this end, UAV-enabled MEC could be a vital role that completes rescue and reconstruction tasks.

2.4.2 Architectures of UAV-enabled MEC

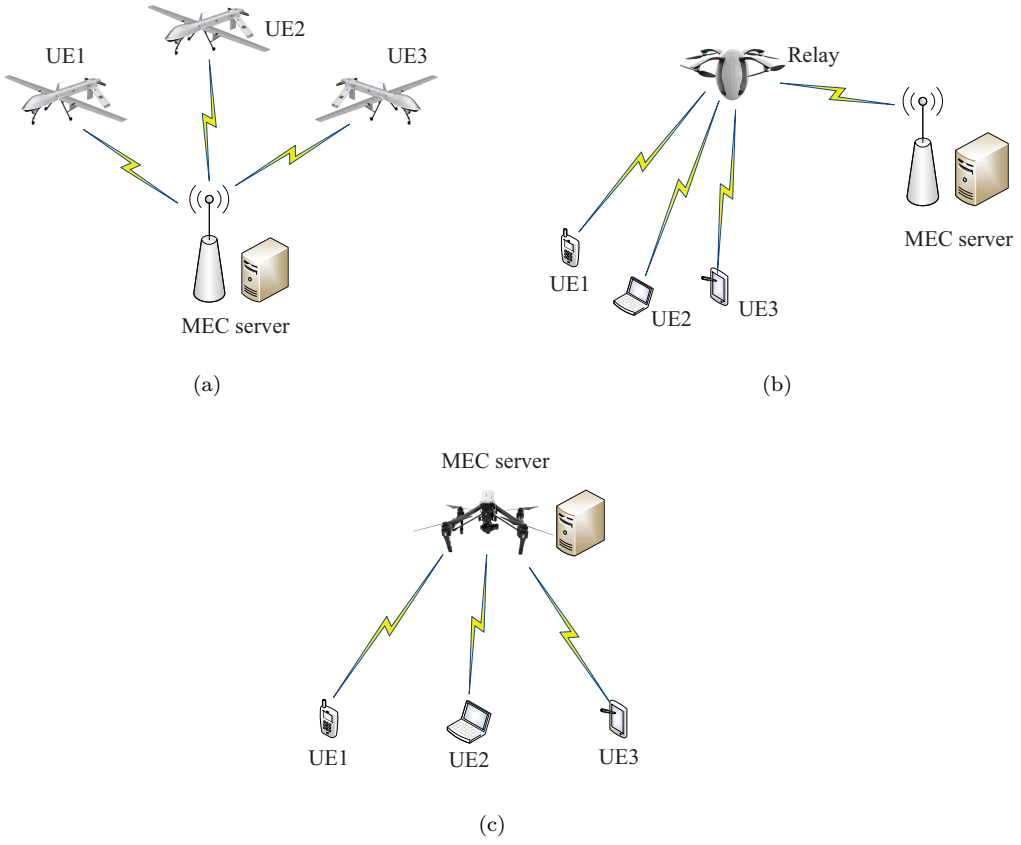


FIGURE 2.3: Three architectures of UAV-enabled MEC.

According to the usage of UAVs applied, there are normally three architectures in UAV-enabled MEC. The first architecture is shown in Fig. 2.3(a), in which the UAV is viewed as a flying UE that needs to offload its computational intensive task to the ground MEC server, due to the finite battery capacity. Then, the second architecture is shown in Fig. 2.3(b), where the UAV is not equipped with MEC server but it works as relaying node to assist ground UEs to offload their tasks to the nearest MEC server. Finally, as shown in Fig. 2.3(c), the UAV can be viewed as a flying MEC server, which provides computational resource and enables task offloading from ground UEs.

Overall, the above three architectures can be flexibly applied in different scenarios. For instance, the first architecture is suitable for the scenario that the UAV has limited computation resource and battery capacity. The second architecture is appropriate for the scenario that ground UEs is far away from MEC server and the UAV can assist them to offload their tasks through relaying. The third architecture can be applied in the scenario where the UAV is equipped with considerable battery and computation resource. In this case, the ground UEs can benefit from offloading in unforeseen disasters and hotspots.

2.5 Machine Learning

In this section, we start to introduce the overview of machine learning. Precisely, we first present the basics of machine learning. Then, we introduce the concept of artificial neural network. Finally, we give the introduction of reinforcement learning.

2.5.1 Basics of Machine Learning

The concept of machine learning is firstly described in [28], and its idea is to automatically improve computer algorithms through experience. Basically, through building a model that is based on training data, the machine learning algorithm can make predictions or decisions without explicit programs. In recent years, in order to intelligently analysis the growing volumes of data generated in the domain of healthcare, financial services, government, technology, marketing, etc, various successful machine learning based algorithms are ubiquitous. There are three categories of machine learning approaches, which are:

1. **Supervised Learning**: In the framework of supervised learning, a training data set is needed, in which each training sample consists of a pair of input and output objects. Note that the input object is also named as 'label'. The basic idea of supervised learning is to learn a function that can approximately map the input and output objects in the training data set by learning and analyzing the training samples and predicting the output objects. Thus, when the supervised learning model is trained and learned enough, it can generate the optimal output objects for unseen examples.

2. **Unsupervised Learning**: The concept of unsupervised learning is to learn the patterns and discipline from an unlabeled data set. Precisely, the training data set in the framework of unsupervised learning is not tagged or labeled, which means the training sample only has input object. The unsupervised learning model is trained and learned to find the structure or distribution in the dataset instead of finding the optimal output objects.
3. **Reinforcement Learning**: Reinforcement learning has been widely used in a variety of control problems like gaming, robotics and navigation while considering the dynamics of environment. Essentially, it is different from supervised learning that learns from a training set of labeled training examples. Also, reinforcement learning is different from unsupervised learning that aims at finding structure hidden in the unlabeled training data set. The idea of reinforcement learning is to learn how to map states and actions in order to maximize the accumulated reward. Theoretically, in the framework of reinforcement learning, there is an agent deployed to interact with the environment (emulator). It aims at choosing the optimal *actions* that can maximize the accumulated *rewards* by given a series of *states*. At time step t, the accumulated reward can be expressed as $R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$, where γ is the discount factor, $r_{t'}$ is the reward function at time step t'. Also, the optimal action-value function, which is also known as the Bellman equation is described as

$$Q^*(s,a) = \mathbb{E}_{s'}\left[r + \gamma \max_{a'} Q^*(s',a')|_{s,a}\right], \tag{2.1}$$

where s', a' denote the sequence of states and actions. Lots of reinforcement learning algorithms estimate the action-value function iteratively by using the above equation.

2.5.2 DQN

In a standard reinforcement learning, an agent is assumed to interact with the environment and select the optimal actions that can maximize the accumulated reward. In [29], a Deep Q Network (DQN) structure developed by Google Deepmind, integrates the deep neural networks with traditional reinforcement learning. The DQN is used to estimate

the well-known Q-value defined as

$$Q(s(t), c(t)) = \mathbb{E}[Z(t)|s(t), c(t)], \tag{2.2}$$

where $s(t)$ and $c(t)$ denote the state and action respectively, $\mathbb{E}[\cdot]$ denotes the expectation, whereas $Z(t) = \sum_{t'=t}^{T} \gamma z(t')$ is a reward and $\gamma \in [0, 1]$ is the discount factor and $z(t')$ is a reward function in the t'-th time step (or time slot). As the objective is to maximize the reward, a widely used policy is $\pi(s(t)|\phi^Q) = \text{argmax}_{c(t)} Q(s(t), c(t)|\phi^Q)$, where ϕ^Q is the parameter of the deep neural network. Then, the DQN can be trained by minimizing the loss function [29]. Also, since the deep networks are known to be unstable and very difficult to converge, two effective approaches, i.e., target network and experience replay, have been introduced in [29]. The target network has the same structure as the original DQN but the parameters are updated more slowly. The experience replay stores the state transition samples which can help the DQN converge. However, the DQN was originally designed to solve the problem with discrete variables. Although we can adapt the DQN to continuous problems by discretizing the action space, it may unfortunately result in a huge searching space and therefore intractable to deal with.

2.5.3 DDPG

To deal with the problem with continuous variables, e.g., the trajectory control of UAV, one may apply the actor-critic approach, which was developed in [30]. DeepMind has proposed a deep deterministic policy gradient (DDPG) approach [31] by integrating the actor-critic approach into DRL. DDPG includes two DQNs, one of the DQNs, named actor network with function $\pi(s(t)|\phi^\pi)$ is applied to generate action $c(t)$ for a given state $s(t)$. The other DQN named critic network with function $Q(s(t), c(t)|\phi^Q)$, is used to generate the Q-value, which evaluates the action produced by the actor network. In order to improve the learning stability, two adjacent target networks corresponding to the actor and critic networks, $\pi'(\cdot)$, $Q'(\cdot)$ with respective parameters $\phi^{\pi'}$, $\phi^{Q'}$, are also applied.

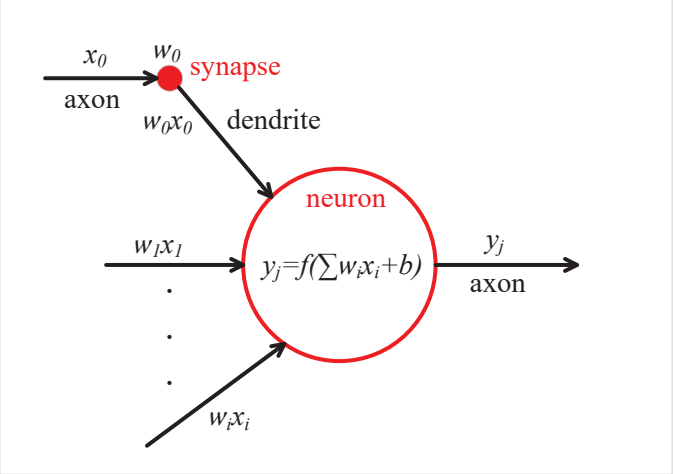


FIGURE 2.4: The structure of a neuron ($x_i, w_i, f(\cdot), b$ represent the activations, weights, nonlinear function, bias separately).

Then, the critic network can be updated with the loss function, $L(\phi^Q)$, as

$$L(\phi^Q) = \frac{1}{K}\sum_{k=1}^{K}\delta_k^2, \tag{2.3}$$

where in each time step, the mini-batch randomly samples K constituting experiences from experience replay buffer, and δ_k is temporal difference (TD)-error [32] which is given by

$$\delta_k = z(k) + \gamma Q'(s(k+1), \pi'(s(k+1)|\phi^{\pi'})|\phi^{Q'}) - Q(s(k), \pi(s(k)|\phi^{\pi})|\phi^{Q}). \tag{2.4}$$

On the other hand, the actor network can be updated by applying the policy gradient, which is described as [31].

$$\begin{aligned}\nabla_{\phi^\pi} J \approx \frac{1}{K}\sum_{k=1}^{K}\nabla_c Q(s,c|\phi^Q)|_{s=s(k),c=\pi(s(k)|\phi^\pi)} = \\ \frac{1}{K}\sum_{k=1}^{K}\left[\nabla_c Q(s,c|\phi^Q)|_{s=s(k),c=\pi(s(k))}\cdot\nabla_{\phi^\pi}\pi(s|\phi^\pi)|_{s=s(k)}\right].\end{aligned} \tag{2.5}$$

2.5.4 Artificial Neural Network

As the subfield of machine learning, artificial neural network (ANN) was a hot topic in last few decades and firstly called by John McCarthy in the 1950s. Its' basic idea is to implement intelligent machines that have the ability of learning like humans in various domains. Since the most powerful 'machine' for learning and solving problems is brain, current scientists are working towards exploring the details of our brains and some popular brain-inspired programs and algorithms are naturally developed to emulate how our brains work in some aspects. Specifically, in the human brain, the basic computational element is the neuron. There are about 68 billions in the brain, each of which is connected with massive elements named dendrites and each element has a mechanism named axon. For better expression, we shown the structure of the neuron in Fig. 2.4. First fo all, the neuron receives the signals through dendrites, then, it performs the computation on the received signals and generates a signal on the axon. Note that the mentioned input and output signals are referred as activations and the axon of a neuron is connected to the dendrites of other neurons. We name the connections between axon and dendrite as a synapse and it is estimated that there are 10^{14} to 10^{15} sysnapses in human brain. Thus, for the certain neuron, its mathematical expression is given as follows:

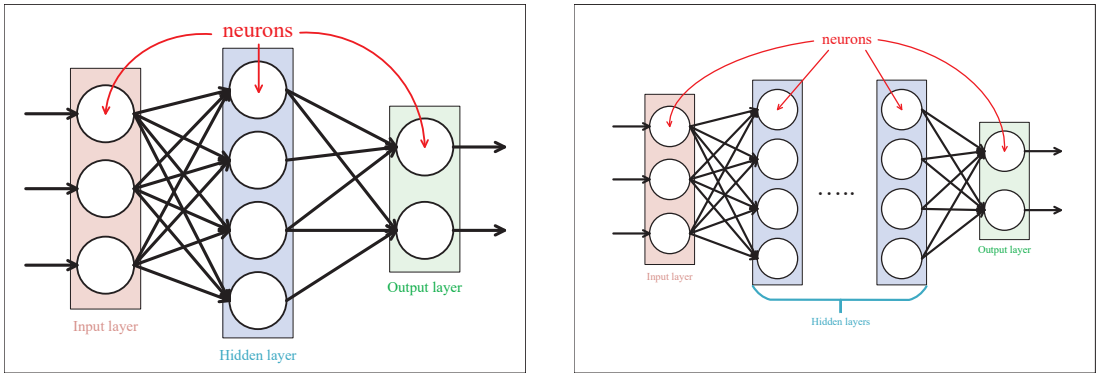
$$y_j = f\bigg(\sum_i \omega_i x_i + b\bigg), \tag{2.6}$$

where x_i denotes the activations, ω_i is the weights, $f(\cdot)$ means the nonlinear function, and b is the bias. It is worth noting that the $f(\cdot)$ normally refers some common functions, such as Sigmoid$(\cdot)$, Tanh$(\cdot)$, ReLu$(\cdot)$. Thus, for the certain input, the different weights will lead to different responses and the learning process of ANN is to adjust the weights while its structure dose not change. This characteristics of ANN makes a remarkable motivation for a machine learning-based algorithm.

Essentially, a standard Fully-connected ANN has the following components:

1. **Input layer**: It is a network layer with a number of neurons for representing the input signal.
2. **Output layer**: It is a network layer with a number of neurons for representing the output signal.

3. **Hidden layer**: It consists of a number of neurons that are used to represent the structure of brain.



(a) (b)

FIGURE 2.5: Structure of shallow neural network (a), DNN (b). [1]

Additionally, according to the number of hidden layers deployed in ANN, there are two categories of ANN, named shallow ANN and deep neural network (DNN) [1] and their main difference can be found in Fig. 2.5. More precisely, in shallow neural network, there is only one hidden layer used, while DNN has more than 1 hidden layers.

In order to learn the information from the given input data, the weights of ANN is needed to be updated, which we denote as training process. For simplicity, we take the image classification problem in supervised learning as an example. Essentially, the main objective of training the ANN for image classification is to minimize the errors between the predicted output signals and the desired labeled signals, which can be mathematically expressed as

$$\mathbb{E}(\mathbf{W},\mathbf{b}) = 0.5 \cdot \sum_{\mathcal{S}} \left(||\mathbf{y}(\mathbf{W},\mathbf{b},\mathbf{x}) - \mathbf{y}_D||_2 \right), \tag{2.7}$$

where $\mathcal{S}$ denotes the training dataset, $\mathbf{W}$ is the set of weights, $\mathbf{b}$ means the set of bias, $\mathbf{x}$ denotes the set of input signals or vector and $\mathbf{Y}_D$ is the desired labeled signals. In order to minimize $\mathbb{E}(\mathbf{W},\mathbf{b})$, the weights of each neuron in ANN are updated via the commonly used gradient descent algorithm. For example, for a certain neuron j, the error between the labeled signal $\mathbf{y}_{D,j}$ and the output signal $\mathbf{y}_j(\mathbf{w}_j, \mathbf{b}_j, \mathbf{x}_j)$ is $E_j(\mathbf{w}_j, \mathbf{b}_j) = 0.5 \cdot \left(||\mathbf{y}_j(\mathbf{w}_j, \mathbf{b}_j, \mathbf{x}_j) - \mathbf{y}_{D,j}||_2 \right)$. Then each element $w_{i,k}$ in $\mathbf{w}_j$ will be updated with the following equation

$$w_{j,k,z+1} = w_{j,k,z} - \gamma \frac{\partial E_j(\mathbf{w}_j, \mathbf{b}_j)}{\partial w_{j,k,z}}, \tag{2.8}$$

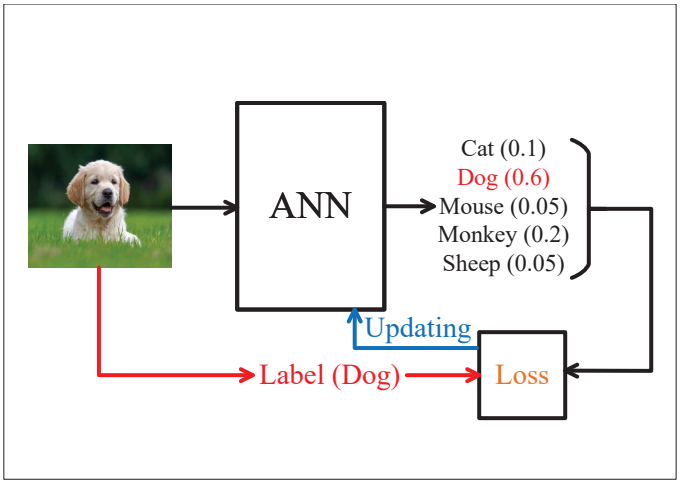


FIGURE 2.6: The overall process of solving image classification problem via using ANN.

and the corresponding bias will also be updated by

$$b_{j,k,z+1} = b_{j,k,z} - \gamma \frac{\partial E_j(\mathbf{w}_j, \mathbf{b}_j)}{\partial b_{j,k,z}}, \tag{2.9}$$

where z denotes the z-th updating iteration, γ is the learning rate.

We further demonstrate the training process of image classification problem in Fig. 2.6. Specifically, we first give the sampled image from the training dataset as the input vector. Then, we give the output signals as the predicted vector, which represents the corresponding probabilities of possible classes. Note that the class with the highest probability indicates the most likely class in the image. The target of training the ANN is to adjust the weights that can successfully determine the correct classes from training dataset.

Chapter 3

Related Work

There are many related works that study UAV, MEC and DRL separately, but only a very few consider them holistically. For UAV aided wireless communications, several scenarios have been studied, such as in areas of relay transmissions [33], cellular system [34], data collection [35], wireless power transfer [36], caching networks [37], and D2D communication [38]. In [39], the authors presented an approach to optimize the altitude of UAV to guarantee the maximum radio coverage on the ground. In [40], the authors presented a fly-hover-and-communicate protocol in a UAV-enabled multiuser communication system. They partitioned the ground terminals into disjoint clusters and deployed the UAV as a flying base station. Then, by jointly optimizing the UAV altitude and antenna beamwidth, they optimized the throughput in UAV-enabled downlink multicasting, downlink broadcasting, and uplink multiple access models. In [41], to maximize the minimum average throughput of covered users in OFDMA system, the authors proposed an efficient iterative algorithm based on block coordinate descent and convex optimization techniques to optimize the UAV trajectory and resource allocation. Furthermore, UAV trajectory optimization research were also investigated. For instance in [18], Zeng *et al.* proposed an efficient design by optimizing UAV's flight radius and speed for the sake of maximizing the energy efficiency of UAV communication. In order to maximize the minimum throughput of all mobile terminals in cellular networks, Lyu *et al.* [42] developed a new hybrid network architecture by deploying UAV as an aerial mobile base station. Different from [18, 39–41] with the single UAV system, a multi-UAV enabled wireless communication system was considered to serve a group of

users in [43]. Also, in [44], resource allocation between communication and computation has been investigated in multi-UAV systems. In [45], Mozaffari *et al.* investigated the application of UAVs in Internet of Things (IoT) network, and they optimized the mobility of UAVs, the device-UAV association and uplink power control, for minimizing the overall transmit power of ground IoT devices.

In addition, some recent literature made efforts to mobile edge computing (MEC), which is considered to be a promising technology for bringing computing resource to the edge of wireless networks [46], where UEs can benefit from offloading their tasks to MEC servers. In [47], partial computation offloading was studied. The computation tasks can be divided into two parts, where one part is executed locally and the other part is offloaded to MEC servers. In [48], binary computation offloading was studied, where the computation tasks can either be executed locally or offloaded to MEC servers.

By taking advantage of the mobility of UAVs, UAV-enabled MEC has been studied in [49, 50]. In [49], authors proposed a heterogeneous MEC (H-MEC) architecture that consists of fixed ground stations and UAVs. In [50], the authors studied UAV-enabled MEC, where wireless power transfer technology is applied to power Internet of things devices and collect data from them. In [51], Zhou *et al.* investigated an UAV-enabled MEC wireless-powered system, and they tackled the computation maximization problem through optimizing UAV's speed, partial and binary computation offloading modes. In [52], Asheralieva *et al.* studied network operation problem in UAV-enabled MEC network, and they developed a framework based on hierarchical game-theoretic and reinforcement learning. In [53], Zhang *et al.* established a communication and computation optimization model in an MEC-enabled UAV network, where the successful transmission probability was derived through using stochastic geometry.

For most of the above works, optimization theory are mainly applied in order to obtain the optimal and / or suboptimal solutions, e.g., trajectory design and resource allocation. However, solving such optimization problems normally requires plenty of computational resources and take much time. To address this problem, DRL has been applied and attracted much attention recently. In [29], the authors proposed a RL framework that uses DQN as the function approximator. In addition, two important ingredients experience replay and target network are used for improving the convergence performance. In [54], the authors pointed out that the classical DQN algorithm may suffer from substantial

overestimations in some scenarios, and proposed a double Q-learning algorithm. In order to solve control problems with continuous state and action space, Lillicrap *at al.* [31] proposed a policy gradient based algorithm. For the purpose of obtaining faster learning and state-of-art performance, in [55], the authors proposed a more robust and scalable approach named prioritized experience replay. Although DRL has achieved remarkable successes in game-playing scenarios, it is still an open research area in UAV-enabled MEC.

Chapter 4

Q-Learning based User Association and Resource Allocation in Multi-UAV enabled MEC

4.1 Introduction

Nowadays, user equipments (UEs) such as smart phones, tablets, wearable devices and other Internet of smart things are becoming increasingly popular and bringing huge convenience to our daily life. Moreover, many emerging mobile applications (e.g., augmented reality, smart navigation and interactive service) are receiving more and more attention but most of those applications are resource intensive, which makes the UEs very difficult to execute them, due to limited battery and computation resource (e.g. CPU, storage or memory) in UEs.

Fortunately, mobile edge computing (MEC) has recently been proposed as a means to enable UEs with intensive computational tasks to offload them to the edge cloud, which can not only prolong the battery life of UEs, but also increase UEs' computational capacity. Offloading decision making and resource allocation have been studied in [56, 57], while MEC with Cloud Radio Access Network (C-RAN) has been investigated in [58–60]. The above works either consider there is only one MEC (e.g., [56, 61]), or consider the

MECs have fixed location (e.g., [58, 62]), which may not be practical in some scenarios. For instance, the single MEC is normally resource-limited and may not be able to meet the requirement of all the UEs at the same time. Also, MEC with fixed location lacks flexibility and may not be suitable to the cases where the number and the requirement of UEs keep changing.

Unmanned aerial vehicle (UAV), due to the features of low cost, high flexibility and easy to deployment, has recently attracted much attention in wireless communication, e.g., serving as base station [63] or mobile relays [64]. UAV enabled MEC (e.g., [65]) has been proposed by integrating MEC server to UAVs (i.e., UAVE), to provide computing resource to ground UEs. Compared with the traditional fixed location MEC, UAVE is of particular interest to the scenario such as 1) temporary events (i.e., in case of a large number of people gathering in the ground celebrating a big event or watching football match); 2) emergency situations (i.e., in case of earthquake and the infrastructure may be destroyed or temporary unavailable) or other on-demand services. However, the operation of UAVE faces many challenges, two of which are how to achieve 1) the association between multiple UEs and UAVs and 2) the resource allocation from the UAVs to the UEs, while meeting the quality of service (QoS) and minimizing the whole energy consumption for all the UEs.

To address these challenges, we formulate above problem into a mixed integer non-linear programming (MINLP), which is very difficult to be addressed in general, especially in the large-scale scenario (e.g., when there is a large number of UEs in the ground waiting to be served). We then propose a Reinforcement Learning-based user Association and resource Allocation (RLAA) algorithm to deal with this problem efficiently and effectively. Numerical results show that the proposed RLAA can achieve the optimal performance compared to the exhaustive search in small scale, and have considerable performance gain over other typical algorithms in large-scale scenarios.

The rest of this chapter is organized as follows. We show the system model and the optimization problem in Section 4.2. Then, our proposed RLAA algorithm is introduced in Section 4.3. The simulation result is given in Section 4.4, followed by the summary remarks in Section 4.5.

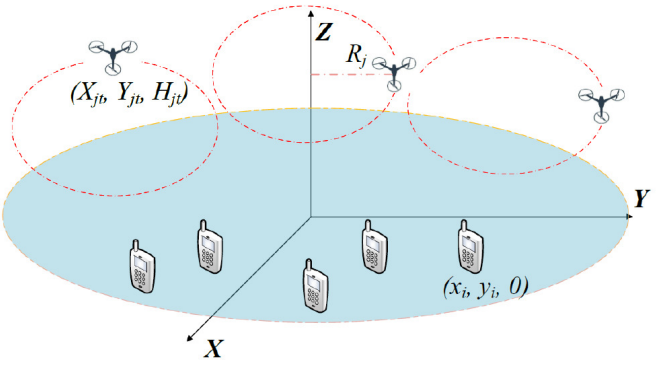


FIGURE 4.1: A Multi-UAV enabled MEC system

4.2 System Model

As shown in Fig. 4.1, we consider there are $i \in \mathcal{N} = \{1, 2, ..., N\}$ UEs, each of which has a computation-intensive task to be executed. Also, we consider there are $j \in \mathcal{M} = \{1, 2, ..., M\}$ UAVs deployed as the MEC platform, flying in a circle with radius R_j. Define a new vector $j \in \mathcal{M}' = \{0, \mathcal{M}\}$ to denote the possible place where the tasks from ground UEs can be executed at, in which $j = 0$ denotes that UE conducts task itself without offloading. Similar to [65], we assume that the j-th UAV's flight period can be discretized into $t \in \mathcal{T}_j = \{1, 2, ..., T_j\}$ time slots. Define a new vector $t \in \mathcal{T}_j' = \{0, \mathcal{T}_j\}$ to denote the possible time slots when the tasks from ground UEs can be executed at, in which $t = 0$ denotes that UE conducts task itself. Also we assume that the UAV's location change within each time slot can be ignored, compared to the distances from the UAV to all UEs. Denote the coordinate of the j-th UAV at t-th time slot as $[X_{jt}, Y_{jt}, H_{jt}]$ and the coordinate of the i-th UE as $[x_i, y_i, 0]$.

Similar to [58], assume i-th UE has a computational intensive task I_i to be executed as

$$I_i = (D_i, F_i), \ \forall i \in \mathcal{N} \tag{4.1}$$

where F_i denotes the total number of CPU cycles required to complete this task and D_i denotes the amount of data needed to be transmitted to UAV if deciding to offload, in which D_i and F_i can be obtained by using the approaches provided in [66]. Assume that each UE can decide either to execute the task locally or choose to offload to one of the UAVs in one time slot and also assume that the task can be completed in this time slot. Similar to [56], we do not consider the time for returning the results back to UE

from UAV. Thus, one can have

$$C1 : a_{ijt} = \{0,1\}, \forall i \in \mathcal{N}, j \in \mathcal{M}', t \in \mathcal{T}_j' \tag{4.2}$$

where $a_{ijt} = 1$, $j \neq 0$, $t \neq 0$ denotes that the i-th UE choose the j-th UAV in the t-th time slot to offload, while $a_{ijt} = 1$, $j = 0$, $t = 0$ denotes that i-th UE execute the task itself and otherwise, $a_{ijt} = 0$. Note that $t = 0$, if and only if $j = 0$.

Also, assume that the j-th ($j \in \mathcal{M}$) UAV can serve more than one UE in each time slot and this task has to be completed either via offloading or local execution. Therefore, one can have

$$C2 : \sum_{j=0}^{M} \sum_{t=0}^{T_j} a_{ijt} = 1, \ \forall i \in \mathcal{N} \tag{4.3}$$

4.2.1 Task Offloading

In offloading scenario, we assume the horizontal distance between i-th UE and the j-th UAV in t-th time slot as

$$R_{ijt} = \sqrt{(X_{jt} - x_i)^2 + (Y_{jt} - y_i)^2} \tag{4.4}$$

Then, the offloading data rate can be given by

$$r_{ijt} = B \log_2(1 + \frac{\alpha P_i^{Tr}}{H_{jt}^2 + R_{ijt}^2}), \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}_j \tag{4.5}$$

where B is denoted as the channel bandwidth, P_i^{Tr} as the transmission power of the i-th UE, $\alpha = \frac{g_0 G_0}{\sigma^2}$, $G_0 \approx 2.2846$, g_0 as the channel power gain at the reference distance 1 m and σ^2 as the noise power [40].

Also, one can see that the time to offload the data from i-th UE to the j-th UAV in t-th time interval can be given as

$$T_{ijt}^{Tr} = \frac{D_i}{r_{ijt}}, \ \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}_j \tag{4.6}$$

Also, the time to execute the task can be expressed as

$$T_{ijt}^{C} = \frac{F_i}{f_{ijt}}, \ \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}_j \tag{4.7}$$

where f_{ijt} is the computation resource that the jth UAV could provide to the i-th UE. Then, we can have the total time consumption as

$$T_{ijt} = T_{ijt}^{Tr} + T_{ijt}^{C},\ \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}_j \tag{4.8}$$

Moreover, the total energy consumption of the i-th UE to the j-th UAV in t-th time slot can be given as

$$E_{ijt}^{Tr} = P_i^{Tr} T_{ijt}^{Tr},\ \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}_j \tag{4.9}$$

Similar to [56], we assume each UAV in every time slot can only accept limited amount of offloaded task. Then, one has

$$C3 : \sum_{i=1}^{N} a_{ijt} \leq K,\ \forall j \in \mathcal{M}, t \in \mathcal{T}_j \tag{4.10}$$

where K is the maximal number of UEs that each UAV can accept in each time slot.

4.2.2 Local Execution

If the UE decides to execute the task locally, the power consumption for the i-th UE can be given by $k_i(f_{ijt})^{v_i}$, where $j = 0$, $t = 0$, and $k_i \geq 0$ is the effective switched capacitance and v_i can be normally to 3 [56]. Then, the local execution time can be given by $T_{ijt}^{C} = \frac{F_i}{f_{ijt}}$, $(j = 0, t = 0)$ and then, the total energy consumption can be given as $k_i(f_{ijt})^{v_i} T_{ijt}^{C}$.

4.2.3 Problem Formulation

Then, one can have the energy consumption of each UE as

$$[l]E_{ijt} = \begin{cases} k_i(f_{ijt})^{v_i} T_{ijt}^{C}, & \text{if } j = 0,\ t = 0 \\ P_i^{Tr} T_{ijt}^{Tr}, & \text{if } j \in \mathcal{M}, t \in \mathcal{T}_j \end{cases} \tag{4.11}$$

Also, the total time spent to complete each task can be expressed as

$$T_{ijt} = \begin{cases} T_{ijt}^{C}, & \text{if } j = 0,\ t = 0 \\ T_{ijt}^{Tr} + T_{ijt}^{C}, & \text{if } j \in \mathcal{M}, t \in \mathcal{T}_j \end{cases} \tag{4.12}$$

One can assume that the maximal computation resource which the j-th UAV can provide is as f_j^{max}. Then, one can have

$$C4: \sum_{i=1}^{N} a_{ijt} f_{ijt} \leq f_j^{max}, \ \forall j \in \mathcal{M}, t \in \mathcal{T}_j \tag{4.13}$$

Also, as the task normally has to be completed in certain amount of the time and thus without loss of generality, we assume the task must be completed in time T^{max}. In our chapter, assume all the transmitting and computing process for each task must be completed within one time interval T^{max}. For the complicated cases, if the task can not be executed within a single time interval, it can be automatically transformed to the next interval. Then, we have

$$C5: \sum_{j=0}^{M} a_{ijt} T_{ijt} \leq T^{max}, \ \forall i \in \mathcal{N}, t \in \mathcal{T}_j' \tag{4.14}$$

Denote $\boldsymbol{A} = \{a_{ijt}, \forall i \in \mathcal{N}, j \in \mathcal{M}', t \in \mathcal{T}_j'\}$, $\boldsymbol{F} = \{f_{ijt}, \forall i \in \mathcal{N}, j \in \mathcal{M}', t \in \mathcal{T}_j'\}$. Then, one can have

$$\mathcal{P}: \min_{\boldsymbol{A}, \boldsymbol{F}} \sum_{i=1}^{N} \sum_{j=0}^{M} \sum_{t=1}^{T_j} a_{ijt} E_{ijt} \tag{4.15a}$$

$$\text{subject to} \tag{4.15b}$$

$$C1: a_{ijt} \in \{0, 1\}, \ \forall i \in \mathcal{N}, j \in \mathcal{M}', t \in \mathcal{T}_j' \tag{4.15c}$$

$$C2: \sum_{j=0}^{M} \sum_{t=0}^{T_j} a_{ijt} = 1, \forall i \in \mathcal{N} \tag{4.15d}$$

$$C3: \sum_{i=1}^{N} a_{ijt} \leq K, \ \forall j \in \mathcal{M}, t \in \mathcal{T}_j \tag{4.15e}$$

$$C4: \sum_{i=1}^{N} a_{ij} f_{ijt} \leq f_j^{max}, \ \forall j \in \mathcal{M}, t \in \mathcal{T}_j \tag{4.15f}$$

$$C5: \sum_{j=0}^{M} a_{ijt} T_{ijt} \leq T^{max}, \forall i \in \mathcal{N}, t \in \mathcal{T}_j' \tag{4.15g}$$

Note that the above problem $\mathcal{P}$ is a MINLP problem, which is difficult to be solved optimally in general. Some existing algorithms like exhaustive search or branch and

bound algorithm may solve this problem, but with prohibitive complexity. Therefore, in this chapter, we aim to obtain an efficient solution to solve this problem. To this end, we propose the RLAA algorithm to deal with $\mathcal{P}$ effectively and efficiently.

4.3 Proposed Algorithm

In this section, we show our proposed RLAA algorithm. First, we introduce three important elements in RLAA (i.e., actions, states, and reward functions).

- **Actions:** At each episode *eps*, each UE takes an action. If the UE decides to offload the task to the j-th UAV in t-th time interval, the action is denoted as ρ_{jt}, $\forall j \in \mathcal{M}, t \in \mathcal{T}_j$. If UE decides to execute the task locally, the action is as ρ_{00}. Then, one can define the collection of actions as follows:

$$\mathcal{C} = \{\rho_{00}, \rho_{11}, ..., \rho_{1T_1}, ..., \rho_{M1}, ..., \rho_{MT_M}\}. \tag{4.16}$$

  For above offloading action ρ_{jt}, $\forall j \in \mathcal{M}, t \in \mathcal{T}_j$, the minimal computation resources of the i-th UE can be given by

$$f_{ijt}^{min} = \frac{F_i}{T^{max} - \frac{D_i}{r_{ijt}}}, \ \forall j \in \mathcal{M}, t \in \mathcal{T}_j \tag{4.17}$$

  For local execution action ρ_{00}, the minimal computation resources of the i-th UE is given as

$$f_{ijt}^{min} = \frac{F_i}{T^{max}}, \ j = 0, t = 0 \tag{4.18}$$

  Note that not all actions can guarantee that the task can be completed within one time interval, as the available computation resources may be less than the minimal computation resources (i.e., f_{ijt}^{min} in (4.17) and (4.18)). Similarly, the communication resource can also not be guaranteed (i.e., $\mathcal{C}3$ in (15e)). Therefore we may remove some actions in $\mathcal{C}$, resulting in the collection of feasible actions for the i-th UE as $\mathcal{C}_i$.
- **States:** Then, we define the states as follows:

$$\mathbf{s} = \{\omega_1, \ldots, \omega_i, \ldots, \omega_N\} \tag{4.19}$$

where ω_i represents the decision of the i-th UE. Specifically, if the i-th UE ($i \in \mathcal{N}$) offloads the task to the j-th UAV in t-th time interval, we assign action ρ_{jt} $\forall j \in \mathcal{M}, t \in \mathcal{T}_j$ to state ω_i. It is worth mentioning that if the i-th UE decides to execute the task locally, we assign action ρ_{00} to state ω_i.

- **Reward Functions:** We define the reward function as

$$Z(\mathbf{s}, \rho_{jt}) = \frac{1}{E_{ijt}} \tag{4.20}$$

The above proposed reward function can keep reducing the energy consumption of each UE and may finally achieve the minimization of the energy consumption of all UEs.

Then, we present RLAA in Algorithm 1. In the beginning, states $\mathbf{s}$ is initialized. The Q-table is also initialized, which is used to record every state and action (i.e., line 1 in Algorithm 1). At each episode, we obtain the collection of the actions $\mathcal{C}_i$ for the i-th UE. Then, according to the ϵ-greedy policy [29], the i-th UE either chooses a random action with probability ϵ or follows the greedy policy with probability $1-\epsilon$, which is expresses as

$$\rho_{jt} = \begin{cases} \rho, & \text{if rand}(0,1) < \epsilon \\ \underset{\rho_{jt} \in \mathcal{C}_i}{\text{argmax}} Q(\mathbf{s}, \rho_{jt}), & \text{otherwise} \end{cases} \tag{4.21}$$

where ρ is an action randomly selected from $\mathcal{C}_i$, rand(0,1) denotes a random number uniformly distributed over the interval [0,1] (i.e., line 4 - line 8 in Algorithm 1).

Then, the resource allocation is conducted for the i-th UE (i.e., line 9 in Algorithm 1). If the i-th UE offload the task to the j-th UAV in t-th time slot, the minimal computation resource f_{ijt}^{min} in (4.17) is allocated. If the i-th UE execute task locally, the minimal computation resource f_{ijt}^{min} in (4.18) is allocated. Based on the proposed reward function in (4.20), the i-th UE can then obtain a reward (i.e., line 10 in Algorithm 1).

Next, we update the Q-table (line 11), where the updating rule of Q-table is given as

$$Q(\mathbf{s}, c) \leftarrow Q(\mathbf{s}, c) + \beta\{Z(\mathbf{s}, c) + \gamma \underset{c \in \mathcal{C}_i}{\text{max}} Q(\mathbf{s}', c) - Q(\mathbf{s}, c)\}, \tag{4.22}$$

where γ is the reward decay over the interval [0,1], β is the learning rate over the interval [0,1], and $\mathbf{s}'$ is the next state. Also, states $\mathbf{s}$ is updated based on action ρ_{jt}. Specifically,

we assign action ρ_{jt}, $\forall j \in \mathcal{M}', t \in \mathcal{T}_j'$ to state ω_i.

The above process will be repeated until the maximum episode (eps^{max}) is reached. Finally, each UE selects an action according to Q-table (line 16). Specifically, for the i-th UE, the action in C_i corresponding to the largest value of Q-table is selected.

Algorithm 1 Our proposed RLAA

1: Initialize $\mathbf{s}$ and Q-table;
2: **while** $eps \leq eps^{max}$
3: **for** $i = 1{:}N$
4: **if** rand(0,1)$< \epsilon$
5: Select an action ρ_{jt} randomly from $\mathcal{C}_i$ for the i-th UE;
6: **else**
7: Select an action $\rho_{jt} = \underset{\rho_{jt} \in \mathcal{C}_i}{\text{argmax}} Q(\mathbf{s}, \rho_{jt})$ for the i-th UE;
8: **end if**
9: Allocate computation resource f_{ijt}^{min} from (4.17) and (4.18) for the i-th UE;
10: Obtain a reward $Z(\mathbf{s}, \rho_{jt})$ according to (4.20);
11: Update Q-table according to (4.22);
12: Update $\mathbf{s}$;
13: **end for**
14: $eps \leftarrow eps + 1$;
15: **end while**
16: Select an action for each UE.

4.4 Simulation Results

In this section, the simulation is conducted with Python 3.6, where the parameters of the tests are shown in Table. 4.1, in which the channel bandwidth is set to $B = 1$ MHz, the noise variance is set to $\sigma^2 = -90$ dbm/Hz, the channel power gain at the reference distance 1 m is set to $g_0 = 1.42 \times 10^{-4}$ [40], the transmission power P_i^{Tr} is set to 1 W, the time interval T^{max} is set to 1 s, the k_i is set to 10^{-27} for all the UEs. Also, we assume each UAV can support $K = 150$ UEs in one time slot. All UEs are assumed to be randomly distributed in a rectangle area of coordinates $[-2000, 2000] \times [-1000, 1000]$ m. We randomly select the data size D_i of each task from the interval of $[100, 1000]$ KB and select F_i from the interval of $[10^8, 10^9]$ cycles.

In order to evaluate the performance of our proposed RLAA, the following four algorithms are used as comparison algorithms.

TABLE 4.1: Simulation Parameters

Parameters	**Settings**
Radius R_j for all UAVs	800 m
Flying height H_{jt} for all UAVs	350 m
Bandwidth B	1 MHz
Transmission power P_i^{Tr}	1 W
Noise variance σ^2	-90 dbm/Hz
G_0	2.2846
Channel power gain g_0	1.42×10^{-4}
Data Size D_i	$[100, 1000]$ KB
Execution task F_i	$[10^8, 10^9]$ cycles
Time duration T^{max}	1 s
Location of UEs	$[-2000, 2000] \times [-1000, 1000]$ m
f_j^{max}	150 GHz
ϵ-greedy policy probability	0.9
Reward decay γ	0.9
Learning rate β	0.2
k_i for all UEs	10^{-27}
v_i for all UEs	3
eps^{max}	10000
T_j for all UAVs	12

- **Exhaustive search (ES):** We examine all the possibilities, with the objective of minimizing the overall energy consumption for all the UEs.
- **Local execution (LE):** We assume all tasks are executed locally and there are no offloading.
- **Random offloading (RO):** Each UE randomly selects the UAV and the time slot to offload its task.
- **Greedy offloading (GO):** Each UE selects the nearest UAV to offload its task. If the UAV is overloaded (i.e., $C3$ is violated), then selects the second nearest UAV to offload and so on.

Firstly, we compare the performance of RLAA with its four compared algorithms on a set of small scale instances (i.e., the number of UEs ranges from 3 to 7). We assume that there are two UAVs flying in circles with the same radius and the center coordinates of two UAVs are set to $[1200, 1200, 350]$ and $[-1200, -1200, 350]$, respectively. From Fig. 4.2, one can see that RLAA has the same performance as ES, both of which can achieve the minimal enery consumption. Also, one can see that GO achieves better performance than RO, whereas LE achieve the worst performance for all examined values. This is because that our proposed RLAA can choose most energy-efficient action for all

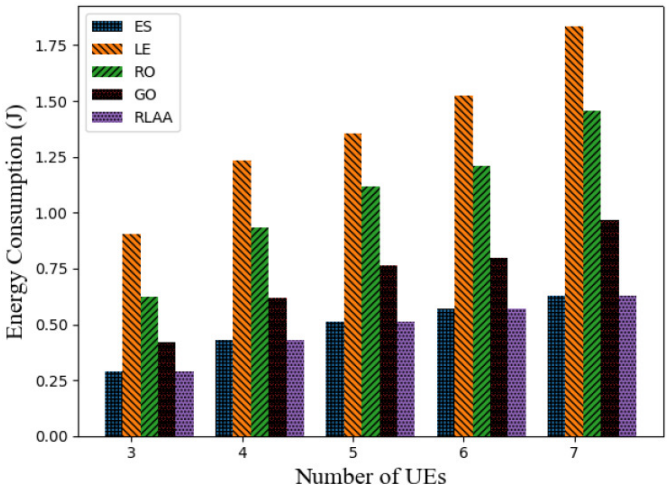


FIGURE 4.2: The overall energy consumption of ES, LE, RO, GO and RLAA versus the number of UEs.

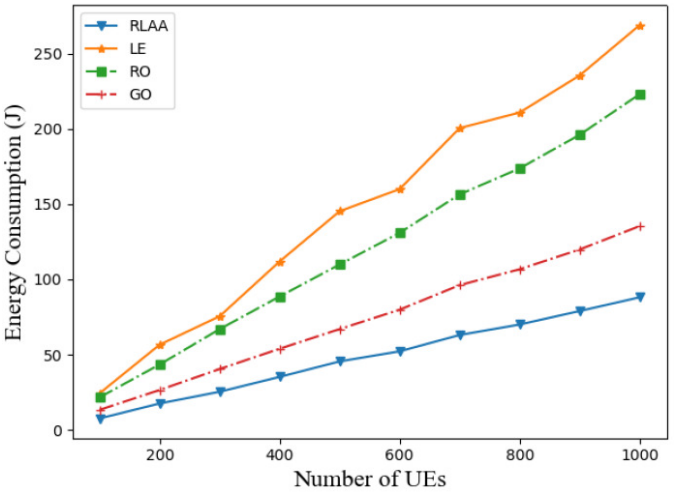


FIGURE 4.3: The overall energy consumption of RLAA, LE, RO and GO versus the number of UEs with 3 UAVs.

the UEs according to computation and communication requirement, while others either make UE to execute all the task locally (i.e., LE), or randomly offload the tasks (i.e., RO), or just find the nearest UAV (i.e., GO), resulting in worse performance.

Next, we compare the performance of RLAA with LE, RO and GO on a set of large scale instances, where the number of UEs is increased to 100∼1000. The number of the UAVs is set to 3, where the center coordinates are $[1200, 1200, 350]$, $[-1200, -1200, 350]$ and $[-1200, 1200, 350]$, respectively. Note that we do not examine ES here, due to its prohibitive complexity. From Fig. 4.3, one can see that our proposed RLAA still performs best, followed by GO, RO and LE, as expected.

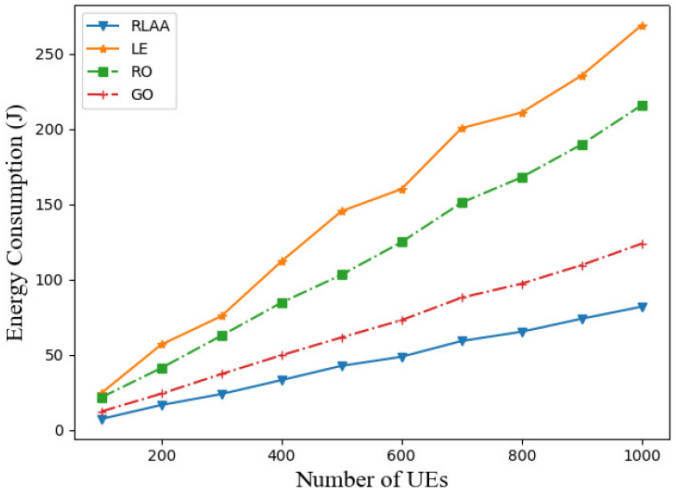


FIGURE 4.4: The overall energy consumption of RLAA, LE, RO and GO versus the number of UEs with 5 UAVs.

In Fig. 4.4, we further increase the number of UAVs to 5, where the center coordinates are set to as $[1200, 1200, 350]$, $[-1200, -1200, 350]$, $[-1200, 1200, 350]$, $[1200, -1200, 350]$ and $[0, 0, 350]$, respectively. One sees that our proposed RLAA still outperforms other compared algorithms, with significant amount of energy being saved for all the UEs.

4.5 Summary

In this chapter, we studied a multi-UAV enabled MEC system, in which the UAVs are assumed to fly in circles over the ground UEs to provide the computation services. The proposed problem is formulated as a MINLP, which is hard to deal with in general. We propose a RLAA algorithm to address it effectively. Simulation results show that RLAA can achieve the same performance as the exhaustive search in small scale cases, whereas in large case scenario, RLAA still have considerable performance gain over other traditional approaches.

Chapter 5

DRL-based Continuous Trajectory Control for Dynamic UAV-enabled MEC

5.1 Introduction

With the popularity of computationally-intensive tasks, e.g., smart navigation and augmented reality, people are expecting to enjoy more convenient life than ever before. However, current smart devices and user equipments (UEs), due to small size and limited resource, e.g., computation and battery, may not be able to provide satisfactory Quality of Service (QoS) and Quality of Experience (QoE) in executing those highly demanding tasks.

Mobile edge computing (MEC) has been proposed by moving the computation resource to the network edge and it has been proved to greatly enhance UE's ability in executing computation-hungry tasks [67]. Recently, flying mobile edge computing (F-MEC) has been proposed, which goes one step further by considering that the computing resource can be carried by unmanned aerial vehicles (UAVs) [68]. F-MEC inherits the merits of UAV and it is expected to provide more flexible, easier and faster computing service than traditional fixed-location MEC infrastructures. However, the F-MEC also brings several challenges: 1) how to minimize the long-term energy consumption of all UEs by choosing proper user association (i.e., whether UE should offload the tasks and if so, which UAV

to offload to, in the case of multiple flying UAVs); 2) how much computations the UAV should allocate to each offloaded UE by considering the limited amount of on-board resource; 3) how to control each UAV's trajectory in real time (namely, flying direction and distance), especially considering the dynamic environment (i.e., the UAV may serve UEs from different taking off points). Traditional approaches like exhaustive search are hardly to tackle the above problems due to the fact that the decision variable space of F-MEC, e.g., deciding the optimal trajectory and resource allocation, is continuous instead of discrete. In [56], the authors propose a quantized dynamic programming algorithm to address the resource allocation problem of MEC. However, the complexity of this approach is very high as the flying choice of UAV is nearly infinite (as continues variables). Moreover, the authors in [41] discretize the UAV trajectory into a sequence of UAV locations and make their proposed problem tractable. Similarly, in [69], the authors assume that the UAV's trajectory can be approximated by using the discrete variables and then they deal with it by using the traditional convex optimization approaches. However, the above treatment may decrease the control accuracy of the UAV and also is not flexible. Furthermore, the above contributions only considered a single UAV case. In practice, one UAV may not have enough resource to serve all the users. If the served area is very large, more than one UAV are normally needed, which will undoubtedly increase the decision space and make it very difficult for the traditional convex optimization-based approaches to obtain the optimal control strategies of each UAV. In [70], Liu *et al.* propose a deep reinforcement learning based DRL-EC^3 algorithm, which can control the trajectory of multiple UAVs but did not consider the user association and resource allocation.

Inspired by the challenges mentioned above, in this chapter, we first propose a Convex optimizAtion based Trajectory control algorithm (CAT) to minimize the energy consumption of all the UEs, by jointly optimizing user association, resource allocation and UAV trajectory. Specifically, by applying block coordinate descent (BCD) method, CAT is divided into two parts, i.e., subproblems for deciding UAV trajectories and for deciding user association and resource allocation. In each iteration, we solve each part separately while keep the other part fixed, until the convergence is achieved.

Next, we propose a deep Reinforcement leArning based Trajectory control algorithm (RAT) to facilitate the real-time decision making. In RAT, two deep Q networks

(DQNs), i.e., actor and critic networks are applied, where the actor network is responsible for deciding the direction and flying distance of the UAV, while the critic network is in charge of evaluating the actions generated by the actor network. Then, we propose a low-complexity matching algorithm to decide the user association and resource allocation with the UAVs. We choose the overall energy consumption of all the UEs as a reward of the RAT. In addition, we deploy a mini-batch to collect samples from the experience replay buffer by using a Prioritized Experience Replay (PER) scheme.

Different from traditional optimization based algorithms which normally need iterations and are susceptible to initial points, the proposed RAT can be adapted to any taking off points of the UAVs and can obtain the solutions very rapidly once the training process has been completed. In other words, if the taking off points of UAV are input to the RAT, the trajectories of UAVs will be determined by the proposed RAT with only some simple algebraic calculations instead of solving the original optimization problem through traditional high-complexity optimization algorithms. This attributes to the fact that during the training stages, excessive randomly taking off points of UAV are generated and used to train the networks until they are converged. Also, with the help of prioritized experience reply (PER), the convergence speed will be increased significantly. RAT can be applied to the practical scenarios where the UAVs needs to act and fly swiftly such as the battlefields. By inputting the current coordinates as the taking off points to the networks, the trajectories of the UAVs will be immediately obtained and then all the UAVs can take off and fly according to the obtained trajectories. Also, the resource allocation and user association are determined by the proposed low-complexity matching algorithm. This is particularly useful to some emergence scenarios (e.g., battlefields, earthquake, large fires), as fast decision making is crucial in these areas.

In the simulation, we can see that the proposed RAT can achieve the similar performance as the convex-based solution CAT. They both have considerable performance gain over other traditional algorithms. In addition, we can see that during the learning procedure, the proposed RAT is less sensitive to the hyperparameters, i.e., the size of mini-batch and the experience replay buffer, when comparing to tradtional reinforcement learning where PER is not applied.

The remainder of this chapter is organized as follows. Section 5.2 describes the system model. Section 5.3 introduces the proposed CAT algorithm, whereas Section 5.4 gives

the proposed RAT algorithm including the preliminaries of DRL. Section 5.5 extends the application of proposed RAT algorithm to 3-D scenario. The simulation results are reported in Section 5.6. Finally, summary is given in Section 5.7.

5.2 System Model

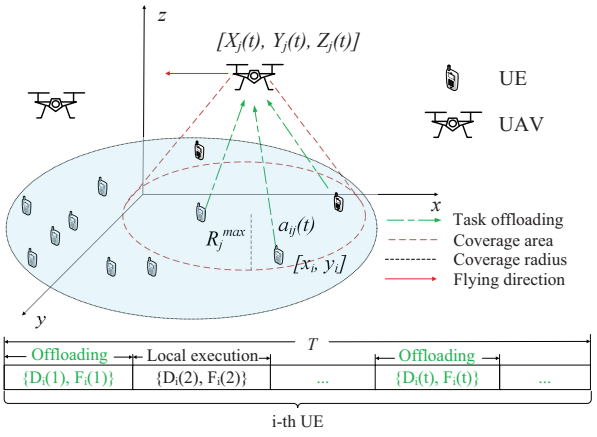


FIGURE 5.1: Multi-UAV enabled F-MEC architecture.

As shown in Fig. 5.1, we consider a scenario that there are N UEs with the set denoted as $\mathcal{N} = \{1, 2, ..., N\}$ and M UAVs with the set denoted as $\mathcal{M} = \{1, 2, ..., M\}$, which form an F-MEC platform. To make it clear, the main notations used in this chapter are listed in Table. 5.1.

We assume that the i-th UE generates one task $I_i(t)$ in the t-th time slot, which has to be executed within a maximal time duration $T^{\max}$, due to the QoS requirement. In this chapter, we assume the entire process lasts for T time slots. Thus, T tasks will be generated for each UE and we have $t \in \mathcal{T} = \{1, 2, ..., T\}$ and

$$I_i(t) = \{D_i(t),\ F_i(t)\},\ \forall i \in \mathcal{N}, t \in \mathcal{T}, \tag{5.1}$$

where $D_i(t)$ denotes the size of data required to be transmitted to a UAV if the UE chooses to offload the task, and $F_i(t)$ denotes the total number of CPU cycles needed to execute this task. Assume that each UE can choose either to offload the task to one of

TABLE 5.1: Main Notations.

Notation	Definition
$i, N, \mathcal{N}$	index, number, set of of UEs.
$j, M, \mathcal{M}$	index,number,set of UAVs.
$t, T, \mathcal{T}$	index, number, set of time slots.
$I_i(t), D_i(t), F_i(t)$	i-th UEs' task in t-th time slot.
$a_{ij}(t)$	user association between i-th UE and j-th UAV.
$R_j^{\max}$	maximal horizontal coverage radius of j-th UAV.
$\theta_j^h(t), \theta_j^v(t), d_j(t)$	flying action of j-th UAV.
$d^{\max}, v_j(t)$	maximal distance, velocity of j-th UAV.
$[X_j(t), Y_j(t), Z_j(t)]$	coordinate of j-th UAV.
$X^{\max}, Y^{\max}$	side length of rectangle-shaped area.
$T^{\max}$	maximal time duration.
$V^{\max}, f^{\max}$	maximal number of tasks, computation resource.
$[x_i, y_i]$	coordinate of i-th UE.
$R_{ij}(t)$	horizontal distance between UE and UAV.
B, P^{Tr}	channel bandwidth, transmitting power.
g_0, σ^2	channel power gain, noise power.
$T_{ij}^{\mathrm{O}}(t), T_{ij}^{\mathrm{Tr}}(t), T_{ij}^{\mathrm{C}}(t)$	time for task completion, offloading, executing.
$E_{ij}^{\mathrm{Tr}}(t), E_{ij}^{\mathrm{L}}(t)$	energy for offloading, local execution.
$\boldsymbol{U}, \boldsymbol{G}$	set of UAV trajectory, UAV coordinates.
$\boldsymbol{A}, \boldsymbol{F}$	set of user association, resource allocation.
$s(t), a(t), z(t)$	state, action and reward.
$\pi(\cdot), Q(\cdot), L(\cdot)$	policy function, Q function, loss function.
K, X	size of mini-batch, experience replay buffer.
ϕ, δ, J	network parameter, TD-error, policy gradient.
$Z^{\min}, Z^{\max}$	minimal, maximal altitude value.
$d_{ij}(t)$	distance between the j-th UAV and i-th UE.

the UAVs or execute the task locally. Then one can have

$$a_{ij}(t) = \{0, 1\}, \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.2}$$

where $a_{ij}(t) = 1$, $j \neq 0$ implies that the i-th UE decides to offload the task to the j-th UAV in the t-th time slot, while $a_{ij}(t) = 1$, $j = 0$ means that the i-th UE executes the task itself in the t-th time slot, and otherwise, $a_{ij}(t) = 0$. Define a new set $j \in \mathcal{M}' = \{0, 1, 2, ..., M\}$ to represent the possible place where the tasks from UEs can be executed, where $j = 0$ indicates that UE conducts its own task locally without offloading.

In addition, we assume that each UE can only be served by at most one UAV or itself, and each task only has one place to execute. Then, it follows

$$\sum_{j=0}^{M} a_{ij}(t) = 1, \forall i \in \mathcal{N}, t \in \mathcal{T}. \tag{5.3}$$

Additionally, in this chapter, the OFDM is applied, which means that each UAV can only accept $V^{\max}$ tasks in each time slot, due to the number of limited sub-carriers. Thus, one has

$$\sum_{i=1}^{N} a_{ij}(t) \leq V^{\max}, \forall j \in \mathcal{M}, t \in \mathcal{T}. \tag{5.4}$$

5.2.1 UAV Movement

Assume that the j-th UAV flies at the altitude and it has a maximal horizontal coverage, which depends on the azimuth angle of antennas and the flying altitude [40]. Also, assume that in the t-th time slot, the j-th UAV can fly with a horizontal direction as

$$0 \leq \theta_j^h(t) \leq 2\pi, \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.5}$$

and distance as

$$0 \leq d_j(t) \leq d^{\max}, \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.6}$$

where $d^{\max}$ is the maximal flying distance that the UAV can move in each time slot, due to the limited power budget. In our chapter, we describe the UAV's movement based on the Cartesian Coordinate system. Thus, we denote the coordinate of the j-th UAV in the t-th time slot as $[X_j(t), Y_j(t), Z_j]$, where $X_j(t) = X_j(0) + \sum_{l=1}^{t} d_j(l)\cos\big(\theta_j^h(l)\big)$, $Y_j(t) = Y_j(0) + \sum_{l=1}^{t} d_j(l)\sin\big(\theta_j^h(l)\big)$ and $[X_j(0), Y_j(0), Z_j]$ is the initial coordinate of the j-th UAV.

Additionally, each UAV can only move within a rectangle-shaped area, whose side length is denoted as $X^{\max}$, and $Y^{\max}$. Then, it has

$$0 \leq X_j(t) \leq X^{\max}, \ \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.7}$$

and

$$0 \leq Y_j(t) \leq Y^{\max},\ \forall j \in \mathcal{M}, t \in \mathcal{T}. \tag{5.8}$$

We denote that the j-th UAV can move with a constant velocity $v_j(t)$, which varies with the flying distance $d_j(t)$ in each time slot. Thus, it has

$$v_j(t) = \frac{d_j(t)}{T^{\max}},\ \forall j \in \mathcal{M}, t \in \mathcal{T}. \tag{5.9}$$

In this chapter, we ignore the communication related energy, including communication circuitry and signal processing.

5.2.2 Task Execution

If the i-th UE decides to offload the task to the j-th UAV in the t-th time slot, then the horizontal distance $R_{ij}(t)$ can be written as

$$R_{ij}(t) = \sqrt{(X_j(t) - x_i)^2 + (Y_j(t) - y_i)^2}, \tag{5.10}$$

where $[x_i, y_i]$ is the coordinate of the i-th UE. Additionally, we assume that each UAV has a maximal azimuth angle $\theta^{\max}$ [1]. Thus, in each time slot, the maximal horizontal coverage of the j-th UAV $R^{\max}$ can be obtained as follows

$$R^{\max} = Z_j \tan(\theta^{\max}). \tag{5.11}$$

Thus, it has

$$a_{ij}(t) R_{ij}(t) \leq R^{\max},\ \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}. \tag{5.12}$$

In this chapter, the free space channel model is applied. Thus, the uplink data rate is given by

$$r_{ij}(t) = B \log_2 \left(1 + \frac{\alpha P^{\mathrm{Tr}}}{Z_j^2 + R_{ij}^2(t)}\right),\ \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.13}$$

where B is the bandwidth for each communication channel; P^{Tr} is the transmitting power of the i-th UE; $\alpha = \frac{g_0 G_0}{\sigma^2}$ with $G_0 \approx 2.2846$ [44]; g_0 is the channel power gain at the reference distance 1 m and σ^2 is the noise power.

[1] We define the azimuth angle with respect to a 3-D reference axis, such as x axis, y axis, z axis.

If the i-th UE decides to offload its task to the j-th UAV in the t-th time slot, the total task completion time is given by

$$T_{ij}^{\mathrm{O}}(t) = T_{ij}^{\mathrm{Tr}}(t) + T_{ij}^{\mathrm{C}}(t),\ \forall t \in \mathcal{T}, \tag{5.14}$$

where $T_{ij}^{\mathrm{Tr}}(t)$ is the time to offload the data from the i-th UE to the j-th UAV in the t-th time slot, given by

$$T_{ij}^{\mathrm{Tr}}(t) = \frac{D_i(t)}{r_{ij}(t)},\ \forall t \in \mathcal{T}, \tag{5.15}$$

and $T_{ij}^{\mathrm{C}}(t)$ is the time required to execute the task at the UAV as

$$T_{ij}^{\mathrm{C}}(t) = \frac{F_i(t)}{f_{ij}^{\mathrm{C}}(t)},\ \forall t \in \mathcal{T}, \tag{5.16}$$

where $f_{ij}^{\mathrm{C}}(t)$ is the computation resource that the j-th UAV can provide to the i-th UE in the t-th time slot.

Note that the time needed for returning the results back to UE from UAV is ignored, similar to [62]. The overall energy consumption of the i-th UE to the j-th UAV in the t-th time slot is given by

$$E_{ij}^{\mathrm{Tr}}(t) = P^{\mathrm{Tr}} T_{ij}^{\mathrm{Tr}}(t),\ \forall t \in \mathcal{T}. \tag{5.17}$$

If the UE decides to execute the task locally, the power consumption can be evaluated as $k_i(f_{ij}^{\mathrm{L}}(t))^{v_i}$ [71], where $k_i \geq 0$ is the effective switched capacitance, v_i is typically set to 3, and $f_{ij}^{\mathrm{L}}(t)$ is the computation resource that the i-th UE applies to execute the task. The overall time for local execution can be given by

$$T_{ij}^{\mathrm{L}}(t) = \frac{F_i(t)}{f_{ij}^{\mathrm{L}}(t)}. \tag{5.18}$$

Thus, the total energy consumption for local execution is

$$E_{ij}^{\mathrm{L}}(t) = k_i(f_{ij}^{\mathrm{L}}(t))^{v_i} T_{ij}^{\mathrm{L}}(t),\ t \in \mathcal{T}. \tag{5.19}$$

To sum up, the overall energy consumption for task execution $E_{ij}(t)$ is given by

$$[l]E_{ij}(t) = \begin{cases} E_{ij}^{\mathrm{L}}(t), & \text{local execution}, \\ E_{ij}^{\mathrm{Tr}}(t), & \text{offloading}, \end{cases} \tag{5.20}$$

and the time to complete the task $T_{ij}(t)$ is expressed as

$$[l]T_{ij}(t) = \begin{cases} T_{ij}^{\mathrm{L}}(t), & \text{local execution,} \\ T_{ij}^{\mathrm{O}}(t), & \text{offloading.} \end{cases} \tag{5.21}$$

Without loss of generality, we assume that each task has to be completed within maximal time duration $T^{\max}$, which is consistent with the maximal flying time in each time slot as

$$T_{ij}(t) \leq T^{\max}, \ \forall i \in \mathcal{N}, j \in \mathcal{M}', t \in \mathcal{T}. \tag{5.22}$$

In each time slot, since the computation resource that each UAV can provide is limited, we have

$$\sum_{i=1}^{N} a_{ij}(t) f_{ij}^{\mathrm{C}}(t) \leq f^{\max}, \ \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.23}$$

where $f^{\max}$ is the maximal computation resource that the j-th UAV can provide in each time slot. Next, we show our proposed problem formulation.

5.2.3 Problem Formulation

Denote $\boldsymbol{U} = \{\theta_j^h(t), d_j(t), \forall j \in \mathcal{M}, t \in \mathcal{T}\}$, $\boldsymbol{A} = \{a_{ij}(t), \forall i \in \mathcal{N}, j \in \mathcal{M}', t \in \mathcal{T}\}$, $\boldsymbol{F} = \{f_{ij}(t), \forall i \in \mathcal{N}, j \in \mathcal{M}', t \in \mathcal{T}\}$. Then, the energy minimization for all UEs is formulated as

$$\mathcal{P}1: \min_{\boldsymbol{U},\boldsymbol{A},\boldsymbol{F}} \sum_{i=1}^{N}\sum_{j=0}^{M}\sum_{t=1}^{T} a_{ij}(t)E_{ij}(t) \tag{5.24a}$$

subject to:

$$a_{ij}(t) = \{0,1\}, \forall i \in \mathcal{N}, j \in \mathcal{M}', t \in \mathcal{T}, \tag{5.24b}$$

$$\sum_{j=0}^{M} a_{ij}(t) = 1, \forall i \in \mathcal{N}, t \in \mathcal{T}, \tag{5.24c}$$

$$\sum_{i=1}^{N} a_{ij}(t) \leq V^{\max}, \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.24d}$$

$$0 \leq \theta_j^h(t) \leq 2\pi, \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.24e}$$

$$0 \leq d_j(t) \leq d^{\max}, \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.24f}$$

$$0 \leq X_j(t) \leq X^{\max}, \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.24g}$$

$$0 \leq Y_j(t) \leq Y^{\max}, \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.24h}$$

$$a_{ij}(t)R_{ij}(t) \leq R^{\max}, \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.24i}$$

$$T_{ij}(t) \leq T^{\max}, \ \forall i \in \mathcal{N}, j \in \mathcal{M}', t \in \mathcal{T}, \tag{5.24j}$$

$$\sum_{i=1}^{N} a_{ij}(t) f_{ij}^{\mathrm{C}}(t) \leq f^{\max}, \ \forall j \in \mathcal{M}, t \in \mathcal{T}. \tag{5.24k}$$

One can see that the above problem $\mathcal{P}1$ is a mixed integer nonlinear programming (MINLP), as it includes both integer variable, $\boldsymbol{A}$ and continuous variables, $\boldsymbol{F}$ and $\boldsymbol{U}$, which is very difficult to solve in general. We first propose a convex optimization based algorithm CAT to address it iteratively. Then, we propose a Deep Reinforcement Learning (DRL) based RAT to facilitate fast decision-making, which can be applied in dynamic environment. Note that in practice, if the i-th UE does not generate the tasks in the t-th time slot and then the corresponding $D_i(t)$ and $F_i(t)$ can be set to zero.

5.3 Proposed CAT Algorithm

In this section, a convex optimization based CAT is proposed to solve the above problem $\mathcal{P}1$. We first define a set of new variables to denote the trajectories of UAVs as $\boldsymbol{G} = \{G_j(t), \forall j \in \mathcal{M}, t \in \mathcal{T}\}$, where the coordinate is $G_j(t) = [X_j(t), Y_j(t)]$, $X_j(t) = X_j(0) + \sum_{l=1}^{t} d_j(l)\cos\big(\theta_j^h(l)\big)$ and $Y_j(t) = Y_j(0) + \sum_{l=1}^{t} d_j(l)\sin\big(\theta_j^h(l)\big)$. Thus, the optimization problem $\mathcal{P}1$ can be reformulated as

$$\mathcal{P}2: \min_{\boldsymbol{G},\boldsymbol{A},\boldsymbol{F}} \sum_{i=1}^{N}\sum_{j=0}^{M}\sum_{t=1}^{T} a_{ij}(t)E_{ij}(t) \tag{5.25a}$$

subject to: (5.24b), (5.24c), (5.24d), (5.24g), (5.24h), (5.24j), (5.24k),

$$a_{ij}(t)||G_j(t) - q_i||^2 \leq (R^{\max})^2, \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.25b}$$

$$||G_j(t+1) - G_j(t)||^2 \leq (d^{\max})^2, \forall t \in \{0, 1, ..., T-1\}, \tag{5.25c}$$

where $q_i = [x_i, y_i]$. In order to solve $\mathcal{P}2$, we divide it into two subproblems and apply the block coordinate descent (BCD) method to address it. To this end, we first optimize the user association $\boldsymbol{A}$ and resource allocation $\boldsymbol{F}$ given the UAV trajectory $\boldsymbol{G}$. Then, we optimize the UAV trajectory $\boldsymbol{G}$ given the user association $\boldsymbol{A}$ and resource allocation $\boldsymbol{F}$. We solve the two optimization problems iteratively, until the convergence is achieved.

5.3.1 User Association and Resource Allocation

Given the UAV trajectory $\boldsymbol{G}$, the subproblem to decide user association $\boldsymbol{A}$ and resource allocation $\boldsymbol{F}$ can be formulated as

$$\min_{\boldsymbol{A},\boldsymbol{F}} \sum_{i=1}^{N}\sum_{j=0}^{M}\sum_{t=1}^{T} a_{ij}(t)E_{ij}(t) \tag{5.26a}$$

subject to: (5.24b), (5.24c), (5.24d), (5.24j), (5.24k), (5.25b).

One can see that (5.24j) can be written as

$$f_{ij}^{\mathrm{C}}(t) \geq \frac{F_i(t)}{T^{\max} - \frac{D_i(t)}{r_{ij}(t)}}, \ \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.27}$$

if the i-th UE chooses to offload the task, and

$$f_{ij}^{\mathrm{L}}(t) \geq \frac{F_i(t)}{T^{\max}}, \ j = 0, \forall t \in \mathcal{T}, \tag{5.28}$$

if the i-th UE decides to execute the task locally. It is readily to see that equality holds for both (5.27) and (5.28).

Then, (5.26) can be re-written as

$$\min_{\boldsymbol{A},\boldsymbol{F}} \sum_{i=1}^{N}\sum_{j=0}^{M}\sum_{t=1}^{T}\left(a_{ij}(t)E_{ij}^{\mathrm{Tr}}(t)+(1-a_{ij}(t))E_{ij}^{\mathrm{L}}(t)\right) \tag{5.29a}$$

subject to: (5.24b), (5.24c), (5.24d), (5.25b),

$$f_{ij}^{\mathrm{L}}(t)=\frac{F_i(t)}{T^{\max}},\ j=0,\forall t\in\mathcal{T}, \tag{5.29b}$$

$$\sum_{i=1}^{N} a_{ij}(t)\frac{F_i(t)}{T^{\max}-\frac{D_i(t)}{r_{ij}(t)}}\le f^{\max},\ \forall j\in\mathcal{M},t\in\mathcal{T}. \tag{5.29c}$$

It is ready to find (5.29) is similar to a Multiple-Choice Multi-Dimensional 0-1 Knapsack Problem (MMKP), which is difficult to solve in general. Fortunately, it may be addressed by applying Branch and Bound method via a standard Python package PULP [72].

5.3.2 UAV Trajectory Optimization

Given the user association and resource allocation from (5.29) and removing the constant, $\mathcal{P}2$ can be simplified as

$$\min_{\boldsymbol{G}} \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{t=1}^{T} a_{ij}(t)\frac{P^{\mathrm{Tr}}D_i(t)}{B\log_2(1+\frac{\alpha P^{\mathrm{Tr}}}{Z_j^2+||G_j(t)-q_i||^2})} \tag{5.30a}$$

subject to: (5.24g), (5.24h), (5.25b), (5.25c),

$$\frac{D_i(t)}{B\log_2(1+\frac{\alpha P^{\mathrm{Tr}}}{Z_j^2+||G_j(t)-q_i||^2})}+\frac{F_i(t)}{f_{ij}^{\mathrm{C}}(t)}\le T^{\max},\forall i\in\mathcal{N},j\in\mathcal{M},t\in\mathcal{T}. \tag{5.30b}$$

It is easy to see that the above optimization problem is non-convex with respect to $G_j(t)$. Next, we introduce a set $\boldsymbol{\eta}=\{\eta_{ij}(t),\forall i\in\mathcal{N},j\in\mathcal{M},t\in\mathcal{T}\}$, where $\eta_{ij}(t)=a_{ij}(t)\frac{P^{\mathrm{Tr}}D_i(t)}{B\log_2(1+\frac{\alpha P^{\mathrm{Tr}}}{Z_j^2+||G_j(t)-q_i||^2})}$, then, problem (5.30) can be transformed into

$$\min_{\boldsymbol{G},\boldsymbol{\eta}} \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{t=1}^{T}\eta_{ij}(t) \tag{5.31a}$$

subject to: (5.24g), (5.24h), (5.25b), (5.25c),

$$B\log_2\left(1+\frac{\alpha P^{\mathrm{Tr}}}{Z_j^2+||G_j(t)-q_i||^2}\right)\ge\frac{a_{ij}(t)P^{\mathrm{Tr}}D_i(t)}{\eta_{ij}(t)},\forall i\in\mathcal{N},j\in\mathcal{M},t\in\mathcal{T}, \tag{5.31b}$$

$$B\log_2\left(1+\frac{\alpha P^{\mathrm{Tr}}}{Z_j^2+||G_j(t)-q_i||^2}\right)\ge\frac{D_i(t)}{T^{\max}-\frac{F_i(t)}{f_{ij}^{\mathrm{C}}(t)}},\forall i\in\mathcal{N},j\in\mathcal{M},t\in\mathcal{T}. \tag{5.31c}$$

One observes that (5.31b) and (5.31c) are convex with respect to $||G_j(t) - q_i||$, respectively. Thus, (5.31b) and (5.31c) are non-convex constraints. Then, similar to [41, 69], we apply the successive convex approximation (SCA) to solve this problem. Specifically, for any given local point $G_j^r(t)$ in $\boldsymbol{G}^r = \{G_j^r(t), \forall j \in \mathcal{M}, t \in \mathcal{T}\}$, one can have the following inequality as

$$\begin{aligned} w_{ij}(t) &= B\log_2\big(1 + \frac{\alpha P^{\mathrm{Tr}}}{Z_j^2 + ||G_j(t) - q_i||^2}\big) \\ &\geq K_{ij}^r(t)(||G_j(t) - q_i||^2 - ||G_j^r(t) - g_i||^2) + B_{ij}^r(t) \\ &\triangleq w_{ij}^{lb,r}(t), \end{aligned} \tag{5.32}$$

where $K_{ij}^r(t) = -\frac{B\alpha P^{\mathrm{Tr}}\log_2(e)}{(Z_j^2+||G_j^r(t)-q_i||^2)(Z_j^2+||G_j^r(t)-q_i||^2+\alpha P^{\mathrm{Tr}})}$, and

$B_{ij}^r(t) = B\log_2\big(1 + \frac{\alpha P^{\mathrm{Tr}}}{Z_j^2+||G_j^r(t)-q_i||^2}\big)$.

Then, problem (5.31) can be written as

$$\min_{\boldsymbol{G},\boldsymbol{\eta}} \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{t=1}^{T}\eta_{ij}(t) \tag{5.33a}$$

subject to: (5.24g), (5.24h), (5.25b), (5.25c),

$$w_{ij}^{lb,r}(t) \geq \frac{a_{ij}(t)P^{\mathrm{Tr}}D_i(t)}{\eta_{ij}(t)}, \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.33b}$$

$$w_{ij}^{lb,r}(t) \geq \frac{D_i(t)}{T^{\max} - \frac{F_i(t)}{f_{ij}^{\mathrm{C}}(t)}}, \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}. \tag{5.33c}$$

The above problem is a convex quadratically constrained quadratic program (QCQP) and it can be solved by a standard Python package CVXPY [73].

5.3.3 Overall Algorithm Design

In this section, a convex optimization-based CAT is proposed to solve Problem $\mathcal{P}2$, where we optimize user association and resource allocation subproblem iteratively with the UAV trajectory subproblem until the convergence is achieved. We describe the pseudo code of proposed CAT in Algorithm 2.

Discussions: Algorithm 1 needs to run once the initial taking off locations of the UAVs change. However, the complexity of Algorithm 1 is high as the solutions are iteratively

Algorithm 2 CAT Algorithm

1: Set $r = 0$, and initialize $\boldsymbol{G}^r$;
2: **repeat**
3: Solve Problem (5.29) by Branch and Bound method for given $\boldsymbol{G}^r$, and denote the optimal solution as $\boldsymbol{A}^{r+1}$ and $\boldsymbol{F}^{r+1}$;
4: Solve Problem (5.33) for given $\boldsymbol{A}^{r+1}$ and $\boldsymbol{F}^{r+1}$, and denote the solution as $\boldsymbol{G}^{r+1}$;
5: $r = r + 1$;
6: **until** the convergence is achieved.

obtained and each subproblem involves a huge number of optimization variables especially when the total number of time slots is high. Precisely, as shown in Algorithm 2, assume that the overall iteration number is K^r. In each iteration, Problem (5.29) has $N(M+1)T$ variables, and it can be solved by Branch and Bound method, in which the Simplex technique for solving linear programs is used. Thus, the computational complexity is $\mathcal{O}\big(2^{N(M+1)T}\big)$ in the worst case. Furthermore, according to the analysis in [41, 74], in Problem (5.33), $\boldsymbol{G}$ has $2MT$ variables, $\boldsymbol{\eta}$ has NMT variables. Hence, the total number of variables is $(N+2)MT$. As a result, the number of iterations required is $\mathcal{O}\big(\sqrt{(N+2)MT}\log_2(\frac{1}{\epsilon_1})\big)$, where ϵ_1 is the accuracy of SCA for solving Problem (5.33). Similarly, the overall number of constraints in Problem (5.33) is $MT(3N+2)+T$. Then, the computational complexity is $\mathcal{O}\Bigg(\big((N+2)MT\big)^2\sqrt{(N+2)MT}\log_2(\frac{1}{\epsilon_1})\big(MT(3N+2)+T\big)\Bigg)$, which is equivalent to $\mathcal{O}\big(3(NMT)^{3.5}\log_2(\frac{1}{\epsilon_1})\big)$. Overall, the total complexity of CAT algorithm is $\mathcal{O}\big(K^r\big(2^{N(M+1)T}+3(NMT)^{3.5}\log_2(\frac{1}{\epsilon_1})\big)\big)$. Hence, Algorithm 1 is not suitable for some emergence scenarios (e.g., battlefields, earthquake, large fires), where fast decision making is highly demanded. This motivates the algorithm developed based on DRL in the following section.

5.4 Proposed RAT Algorithm

To facilitate the fast decision making, the DRL-based RAT algorithm is proposed in this section.

5.4.1 The RAT Algorithm

In this section, we introduce the DRL based RAT algorithm, which includes deep neural networks (i.e., actor and critic networks) and the matching algorithms. In order to apply

the DRL, we first define the state, action and reward as follows:

1) **State** $s(t)$: $s(t) = \{[X_j(t), Y_j(t), Z_j],\ \forall j \in \mathcal{M}\}$, $s(t)$ is the set of the coordinates of all UAVs.
2) **Action** $c(t)$: $c(t)$ is the set of the actions of all UAVs, including the horizontal direction $\theta_j^h(t)$ and distance $d_j(t)$. Then, the action set can be defined as $c(t) = \{[\theta_j^h(t), d_j(t)],\ \forall j \in \mathcal{M}\}$.
3) **Reward** $z(t)$: $z(t)$ is defined as the minus of the overall energy consumption of all the UEs in each time slot as

$$z(t) = -\sum_{i=1}^{N}\sum_{j=0}^{M} a_{ij}(t)E_{ij}(t) - p, \tag{5.34}$$

where p is the penalty if any of UAV flies out of the target area, which means (5.24g) or (5.24h) is not satisfied.

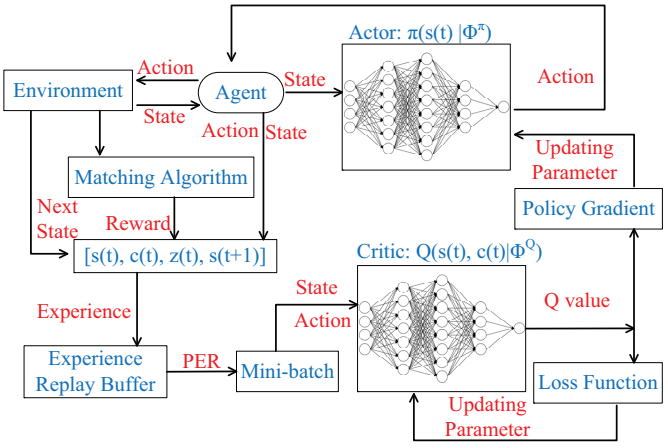


FIGURE 5.2: The structure of RAT algorithm.

The algorithm framework used in this chapter is depicted in Fig. 5.2, where an agent, which could be deployed in the control center of the base station, is assumed to interact with the environment. An actor network $\pi(s(t)|\phi^\pi)$ is applied to generate the action, which includes the flying direction and distance for each UAV. The critic network $Q(s(t), c(t)|\phi^Q)$ is used to obtain the Q-value of the action (i.e., to evaluate the action generated by actor network). In each time slot, the agent sends the action generated by actor network to each UAV. Then, each UE tries to associate with one UAV in its coverage, i.e., (5.12) by using a matching algorithm in Algorithm 4. More specifically, each UE tries to connect the UAV which can save more offloading energy. If the

minimum offloading energy is larger than the energy of local execution, the UE will decide to conduct the task locally. Note that RAT has the same optimization strategy for resource allocation as CAT.

Also, each UAV selects the UEs based on the following criteria: 1) UE should be within its coverage area; 2) UE could save more energy, i.e., the more of $E_{ij}^{\mathrm{L}}(t) - E_{ij}^{\mathrm{C}}(t)$ will be given higher priority in offloading to this UAV. We will introduce the details of the proposed matching algorithm in Algorithm 4. After the matching algorithm, the reward in (5.34) can be obtained.

We assume that there is an experience replay buffer for the agent to store the experience $[s(t), c(t), z(t), s(t+1)]$. Once the experience replay buffer is full, the learning procedure starts. A mini-batch with K experiences can be obtained from the experience replay buffer to train the networks.

In the classical DRL algorithms, such as Q-learning [75], SARSA [76] and DDPG [31], the mini-batch uniformly samples experiences from the experience replay buffer. However, since TD-error in (2.4) is used to update the Q-value network, experience with high TD-error often indicates the successful attempts. Therefore, a better way to select the experience is to assign different weights to samples. Schaul *et al.* [55] developed a prioritized experience replay scheme, in which the absolute TD-error $|\delta_k|$ is used to evaluate the probability of the sampled k-th experience from the mini-batch. Then, the probability of sampling the k-th experience can be given by

$$P(k) = \frac{p_k^{\beta}}{\sum_{m \in K} p_m^{\beta}}, \tag{5.35}$$

where $p_k = |\delta_k| + \epsilon$, $\epsilon = 0.001$ is a positive constant to avoid the edge-case of transitions not being revisited if $|\delta_k|$ is 0, $\beta = 0.6$ is denoted as a factor to determine the prioritization [55].

However, frequently sampling experiences with high $|\delta_k|$ can cause divergence and oscillation. To tackle this issue, the importance-sampling weight [77] is introduced to represent the importance of sampled experience, which can be given by

$$\omega_k = \frac{1}{(X \cdot P(k))^{\mu}}, \tag{5.36}$$

where X is size of experience replay buffer, μ is given as 0.4 [55]. Thus, the loss function $L(\phi^Q)$ in (2.3) is updated as

$$L(\phi^Q) = \frac{1}{K}\sum_{k=1}^{K} \omega_k \delta_k^2, \tag{5.37}$$

which is used in our proposed RAT to train the networks. Next, we describe the pseudo code of the overall RAT framework in Algorithm 3.

Algorithm 3 RAT Algorithm

1: Initialize actor network $\pi(s(t)|\phi^\pi)$ with parameters ϕ^π and critic network $Q(s(t), s(t)|\phi^Q)$ with parameters ϕ^Q;
2: Initialize target networks $Q'(\cdot)$ with parameters $\phi^{Q'} = \phi^Q$ and $\pi'(\cdot)$ with parameters $\phi^{\pi'} = \phi^\pi$;
3: Initialize experience replay buffer $\mathcal{X}$;
4: **for** epoch =1,..., $k^{\max}$ **do**
5: Initialize $s(t)$;
6: **for** time slot t =1,..., T **do**
7: $\pi(s(t)|\phi^\pi) + \rho N'$ where N' is the random noise and ρ decays with t;
8: **for** UAV j=1,..., M **do**
9: Execute $c(t)$;
10: Obtain $s(t+1)$;
11: **end for**
12: Obtain the user association with UAVs using matching algorithm proposed in Algorithm 4;
13: Obtain the reward $z(t)$ from (5.34);
14: Store experience $[s(t), c(t), z(t), s(t+1)]$ into the replay buffer;
15: **if** the replay buffer is full **then**
16: **for** k = 1,..., K **do**
17: Sample k-th experience with probability $P(k)$ from (5.35);
18: Calculate $|\delta_k|$ and ω_k from (2.4) and (5.36) respectively;
19: **end for**
20: Update parameters of the critic network ϕ^Q by minimizing its loss function according to (5.37);
21: Update parameters of the actor network ϕ^π by using policy gradient approach according to (2.5);
22: Update two target networks with the updating rate τ:
23: **end if**
24: **end for**
25: **end for**

We first initialize the actor, critic, two target networks, and experience replay buffer in Line 1 - 3. In the beginning of each epoch, all UAVs start to serve UEs from different taking off points. Note that for better exploration, we add a random noise N' to the action, where N' follows a normal distribution with 0 mean and variance 1, ρ is set to 2 and decays with a rate of 0.9995 in each time step. From Line 8-11, each UAV flies according to the generated action $c(t)$ and enters the next state $s(t+1)$. Then,

we obtain the user association by using Algorithm 4. Next, the reward $z(t)$ is obtained according to (5.34) (i.e., Line 13). The experience is also stored in the replay buffer. When the buffer is full, the mini-batch samples K experiences by applying the prioritized experience replay (i.e., Line 16-19). Then, we update the actor and critic networks by using loss function in (5.37) and policy gradient in (2.5) respectively. Finally, we update the target networks by using the following equations as (i.e., Line 22)

$$\phi^{Q'} \leftarrow \tau\phi^{Q} + (1-\tau)\phi^{Q'}, \tag{5.38}$$

and

$$\phi^{\pi'} \leftarrow \tau\phi^{\pi} + (1-\tau)\phi^{\pi'}, \tag{5.39}$$

where τ is the updating rate.

Algorithm 4 Matching Algorithm

1: Initialize $\boldsymbol{A}$ and $\boldsymbol{F}_j$, $\forall j \in \mathcal{M}$, $\forall i \in \mathcal{N}$;
2: **for** UAV $j = 1,..., M$ **do**
3: **for** UE $i = 1,..., N$ **do**
4: **if** (5.12) is met **then**
5: Calculate $E_{ij}^{\mathrm{L}}(t)$, $E_{ij}^{\mathrm{Tr}}(t)$ and $f_{ij}^{\mathrm{C}}(t)$;
6: **if** $E_{ij}^{\mathrm{L}}(t) > E_{ij}^{\mathrm{Tr}}(t)$ **then**
7: Store i into $\boldsymbol{E}_j$;
8: **end if**
9: **end if**
10: **end for**
11: Sort the element in $\boldsymbol{E}_j$ in descending order with respect to $E_{ij}^{\mathrm{L}}(t) - E_{ij}^{\mathrm{Tr}}(t)$;
12: **end for**
13: **repeat**
14: **for** UAV $j = 1,..., M$ **do**
15: $i = GetTopItem(\boldsymbol{E}_j)$;
16: **if** (5.4), (5.23) are met **then**
17: **if** $E_{ij}^{\mathrm{Tr}}(t) < E_{i\boldsymbol{A}(i)}^{\mathrm{Tr}}(t)$ or $\boldsymbol{A}(i) = 0$ **then**
18: $\boldsymbol{A}(i) = j$;
19: **end if**
20: $RemoveTopItem(\boldsymbol{E}_j)$;
21: **end if**
22: **end for**
23: **until** Each UE in $\boldsymbol{E}_j$ is checked.
24: Return $\boldsymbol{A}$

Next, we introduce the low-complexity matching algorithm which can decide the user association and resource allocation given UAVs' trajectories, as shown in Algorithm 4. First, we denote $\boldsymbol{A}$ with size N to record the user association between UEs and UAVs.

If $\boldsymbol{A}(i) = j$, the i-th UE matches with the j-th UAV, while if $\boldsymbol{A}(i) = 0$, the i-th UE is not matched yet and has to execute its task locally. In addition, we denote a preference list $\boldsymbol{E}_j$ for the j-th UAV to record UEs that can benefit from offloading. Then, from Line 2 to 10, we generate the preference list $\boldsymbol{E}_j$ for the j-th UAV. Precisely, if constraint (5.12) is met, we obtain $E_{ij}^{\mathrm{L}}(t)$, $E_{ij}^{\mathrm{Tr}}(t)$, and $f_{ij}^{\mathrm{C}}(t)$ according to (5.19), (5.17), and (5.27), respectively. UEs that benefit from offloading will be stored in $\boldsymbol{E}_j$. Since UAVs need to save as much energy of UEs as possible, we sort the preference list $\boldsymbol{E}_j$ with descending order with respect to $E_{ij}^{\mathrm{L}}(t) - E_{ij}^{\mathrm{Tr}}(t)$, as shown in Line 11. The UE that can save more energy via offloading will be matched with a higher priority. Next, from Line 13 to 23, we conduct the matching process. Each UAV keeps selecting UEs according to its preference list, and constantly checking the constraints (5.4) and (5.23) based on $\boldsymbol{A}$. In the meantime, the selected UE will determine whether to match with the UAV or not. Precisely, from Line 17 to 19, if the selected UE is not matched before, or matching with the j-th UAV could save more energy than previous match, the corresponding $\boldsymbol{A}(i)$ will be updated. We do this process until all the UEs in each preference list are checked. Then, the final user association can be obtained from $\boldsymbol{A}$.

According to [31], our RAT algorithm is an offline learning and off-policy DRL-based algorithm as the experience replay mechanism is applied, and the mini-batch will sample several uncorrelated experiences for training networks in each time step. Additionally, the training procedure can be deployed in a simulator, and the RAT model can be easily deployed in reality when the convergence is achieved, which will inevitably reduce the payoff of implementation. Furthermore, once the whole networks are converged, the solutions can be generated very fast with only some simple algebraic calculations instead of solving the original MINLP. This is due to the fact that during the training stages, random taking off points of all the UAVs are generated and the networks are trained to converge.

Discussions: after adequate training process, the RAT model, including the networks is saved for testing. In each time slot, the action of all UAVs is generated together by actor network. In our chapter, as the fully-connected hidden layers are applied, the computational complexity for generating action of UAVs is $\mathcal{O}\big(\sum_{l=1}^{L} n_l \cdot n_{l-1}\big)$, where L is the number of network layers, n_l is the number of neurons in the l-th layer. Then, the computational complexity of matching algorithm is $\mathcal{O}(NM)$. The overall complexity of RAT algorithm in testing process is $\mathcal{O}\big((\sum_{l=1}^{L} n_l \cdot n_{l-1} + NM)T\big)$.

5.5 Extension to 3-D Channel Model

In this section, in order to consider the more practical environment and the impacts of blockage and shadowing, we extend the previous free-space to 3-D channel model proposed in [39]. In each time slot, we assume the UAV can fly with a vertical direction $\theta_j^v(t) \in [0, \pi]$, a horizontal direction $\theta_j^h(t) \in [0, 2\pi]$, and a flying distance $d_j(t) \in [0, d^{\max}]$. We define the coordinate of the j-th UAV in the t-th time slot as $[X_j(t), Y_j(t), Z_j(t)]$, where

$$X_j(t) = X_j(0) + \sum_{l=1}^{t} d_j(l)\sin\big(\theta_j^v(l)\big)\cos\big(\theta_j^h(l)\big), \tag{5.40}$$

and

$$Y_j(t) = X_j(0) + \sum_{l=1}^{t} d_j(l)\sin\big(\theta_j^v(l)\big)\sin\big(\theta_j^h(l)\big), \tag{5.41}$$

and

$$Z_j(t) = Z_j(0) + \sum_{l=1}^{t} \cos\big(\theta_j^v(l)\big), \tag{5.42}$$

in which, $[X_j(0), Y_j(0), Z_j(0)]$ is the initial coordinate of the UAV. For collision avoidance, we consider

$$Z^{\min} \leq Z_j(t) \leq Z^{\max}, \forall t \in \mathcal{T}, \tag{5.43}$$

where $Z^{\min}$ and $Z^{\max}$ are the minimal and maximal flying altitude of the UAV.

Thus, the distance between the j-th UAV and the i-th UE in t-th time slot is given by

$$d_{ij}(t) = \sqrt{\big(X_j(t) - x_i\big)^2 + \big(Y_j(t) - x_i\big)^2 + Z_j^2(t)}, \ \forall j \in \mathcal{M}, i \in \mathcal{N}, t \in \mathcal{T}. \tag{5.44}$$

The coverage radius of the j-th UAV in the t-th time slot can be given by

$$R_j^{\max}(t) = Z_j(t)\tan(\theta^{\max}). \tag{5.45}$$

The mean path loss between the j-th UAV and the i-th UE in the t-th time slot can be expressed as [39]

$$L_{ij}(t) = \frac{\eta_{\text{LoS}} - \eta_{\text{NLoS}}}{1 + a\exp(-b(\theta_{ij}(t) - a))} + 20\log_{10}\big(d_{ij}(t)\big) + 20\log_{10}\big(\frac{4\pi f_c}{c}\big) + \eta_{\text{NLoS}}, \tag{5.46}$$

where η_{LoS}, η_{NLoS} are the path loss of achieving LoS and NLoS links, a and b are constant values that can be obtained in [39], $\theta_{ij}(t) = \arctan\left(\frac{Z_j(t)}{R_{ij}(t)}\right)$ is the elevation angle between the UAV and the UE, f_c is the carrier frequency, and c is the light speed. Then, we can show the data rate as follows:

$$r_{ij}(t) = B\log_2\left(1 + \frac{P^{\text{Tr}}}{\sigma^2}10^{-\frac{L_{ij}(t)}{10}}\right). \tag{5.47}$$

Additionally, we consider to maximize the energy efficiency of UAVs and motivated by [78], we show the power consumed by the j-th UAV in the t-th time slot as follows

$$\begin{aligned} P_j(t) =& P_o\left(1 + 3(\frac{v_j(t)}{U_b})^2\right) + P_s\left(\sqrt{1 + \frac{1}{4}(\frac{v_j(t)}{V_h})^4} - \frac{1}{2}(\frac{v_j(t)}{V_h})^2\right)^{\frac{1}{2}} \\ &+ \frac{\pi}{2}d_0\rho_a r_s R_r^2 v_j(t)^3 + wgv_j(t)\cos\left(\theta_j^v(t)\right), \end{aligned} \tag{5.48}$$

where P_o and P_s are fixed constants that can be obtained in [79], U_b is the tip speed of the rotor blade, V_h denotes the mean rotor induced velocity when hovering, d_0 is the drag ratio of main body, ρ_a is the air density, r_s is the rotor solidity, R_r means the rotor radius, w is the weight of UAV, and g is the gravity acceleration.

Thus, the remaining energy of the j-th UAV in the t-th time slot is defined as

$$e_j(t) = e^{\max} - \sum_{l=1}^{t} P_j(l)T^{\max}, \tag{5.49}$$

where $e^{\max}$ is the maximal energy of each UAV.

Thus, the optimization problem can be written as follows:

$$\mathcal{P}1: \min_{\boldsymbol{U},\boldsymbol{A},\boldsymbol{F}} \sum_{t=1}^{T}\left(\sum_{j=0}^{M}\sum_{i=1}^{N} a_{ij}(t)E_{ij}(t) + k_z\sum_{j=1}^{M} P_j(t)T^{\max}\right) \tag{5.50a}$$

subject to: (5.24b), (5.24c), (5.24d), (5.24e), (5.24f),

(5.24g), (5.24h), (5.24j), (5.24k),

$$0 \le \theta_j^v(t) \le \pi,\ \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.50b}$$

$$Z^{\min} \le Z_j(t) \le Z^{\max},\ \forall j \in \mathcal{M}, t \in \mathcal{T}, \tag{5.50c}$$

$$a_{ij}(t)R_{ij}(t) \le R_j^{\max}(t),\ \forall i \in \mathcal{N}, j \in \mathcal{M}, t \in \mathcal{T}. \tag{5.50d}$$

where $\boldsymbol{U} = \{\theta_j^v(t), \theta_j^h(t), d_j(t),\ \forall j \in \mathcal{M}, t \in \mathcal{T}\}$, k_z is the weight factor.

To solve the above problem, we define the state and action as follows:

1) **State** $s(t)$: $s(t) = \{[X_j(t), Y_j(t), Z_j(t), e_j(t)],\ \forall j \in \mathcal{M}\}$.
2) **Action** $c(t)$: the action set can be defined as $c(t) = \{[\theta_j^v(t), \theta_j^h(t), d_j(t)],\ \forall j \in \mathcal{M}\}$.
3) **Reward** $z(t)$: we define the reward as follows

$$z(t) = -\sum_{j=0}^{M}\sum_{i=1}^{N} a_{ij}(t)E_{ij}(t) - k_z\sum_{j=1}^{M} P_i(t)T^{\max} - p, \tag{5.51}$$

where p is the penalty if any of UAV flies out of the target area, i.e., if (5.24g), (5.24h) or (5.50c) is not satisfied.

Thus, having defined the state, action and reward, the above problem can be solved by the proposed RAT algorithm as introduced before.

5.6 Simulation Results

In this section, both convex optimization-based CAT and DRL-based RAT are evaluated with simulations implemented on Intel i5-3450t, NVIDIA GTX 1050Ti, Python 3.6, PULP 1.6.10, CVXPY 1.1.7, and Tensorflow 1.15.0. We deploy three fully-connected hidden layers with 1024, 800 and 600 neurons in both actor and critic networks in RAT. The actor network is trained by applying RMSPropOptimizer with the learning rate 0.001, whereas the critic network is trained by using AdamOptimzer with the learning rate 0.001. In the simulation, we assume there are 60 time slots in each training epoch. There are 100 UEs randomly distributed in a rectangle-shaped area with the side length of $X^{\max} = 400$ m and $Y^{\max} = 400$ m. Additionally, there are 2 UAVs deployed to serve UEs within the target area. Note that for RAT, each UAV has 20 different taking off points during the training procedure. Besides, in each time slot, UE generates a task with communication requirement $D_i(t) \in [10, 50]$ KB and computation requirement $F_i(t) \in [2 \times 10^9, 2 \times 10^{10}]$ cycles. Other parameters are summarized in Table 5.2. We assume in each time slot, UAVs will send a signal to activate the corresponding UEs, which will either offload the task or execute locally, within the delay requirement.

TABLE 5.2: Simulation Parameters

Parameters	Settings	Parameters	Settings
T	60	N	100
M	2	$V^{\max}$	30
$d^{\max}$	30 m	$T^{\max}$	1 s
$X^{\max}$	400 m	$Y^{\max}$	400 m
$\theta^{\max}$	$\frac{\pi}{4}$	$Z_j(0)$	75 m
v_i	3	g_0	1.42×10^{-4}
P^{Tr}	0.1 W	B	10 MHz
σ^2	-90 dbm	$e^{\max}$	10^6 J
k_i	10^{-28}	$f^{\max}$	100 GHz
γ	0.999	p	100
$k^{\max}$	3000	ρ	2
w	2 kg	g	10 m/s^2
τ	0.001	$Z^{\min}$	50 m
$Z^{\max}$	120 m	η_{LoS}	1.6
η_{NLoS}	23	a	12.08
b	0.11	f_c	2.5 GHz
c	3×10^8 m/s	k_z	0.0025
P_o	79.86	U_b	120 m/s
P_s	88.63	V_h	4.03
d_0	0.6	ρ_a	1.25 kg/m^3
r_s	0.05	R_r	0.4 m

In order to evaluate the performance of the proposed CAT and RAT, we present the following three algorithms for comparison purpose.

- **Local Execution (LE):** All tasks are executed locally without offloading.
- **Random moving (RM):** In this setting, each UAV randomly selects the horizontal direction and flying distance to take.
- **Cluster moving (CM):** We group all the UEs into 10 clusters and each UAV flies in the trajectory connecting all the cluster center one by one. Note that it takes $\frac{T}{10}$ time slots for each UAV to move from one cluster center to another one.
- **Deep Deterministic Policy Gradient (DDPG) [31]:** We set the parameter of DDPG the same as actor and critic networks of RAT, but do not apply the prioritized experience replay. In other words, DDPG uniformly samples the experiences from the experience replay buffer in the training procedure.

Note that both RM, CM, DDPG apply the matching algorithm proposed in Algorithm 4 to decide the user association and resource allocation.

5.6.1 Convergence Evaluation of CAT and RAT

In this subsection, we show the convergence of proposed CAT and RAT. In Fig. 5.3, we depict the convergence performance of CAT with three different pairs of initial trajectories. Specifically, we group all UEs into one cluster and the UAVs fly in a circle around the cluster center with radius 80 m, 100 m, and 120 m respectively. We denote these three pairs of UAV trajectories as the initial trajectories. As shown in Fig. 5.3, we can conclude that for any initial trajectory, the overall energy consumption of UEs achieved by CAT always decreases and finally remains stable after several iteration times. However, one can also observe that the convergent solution achieved by CAT will be influenced by the initial trajectory.

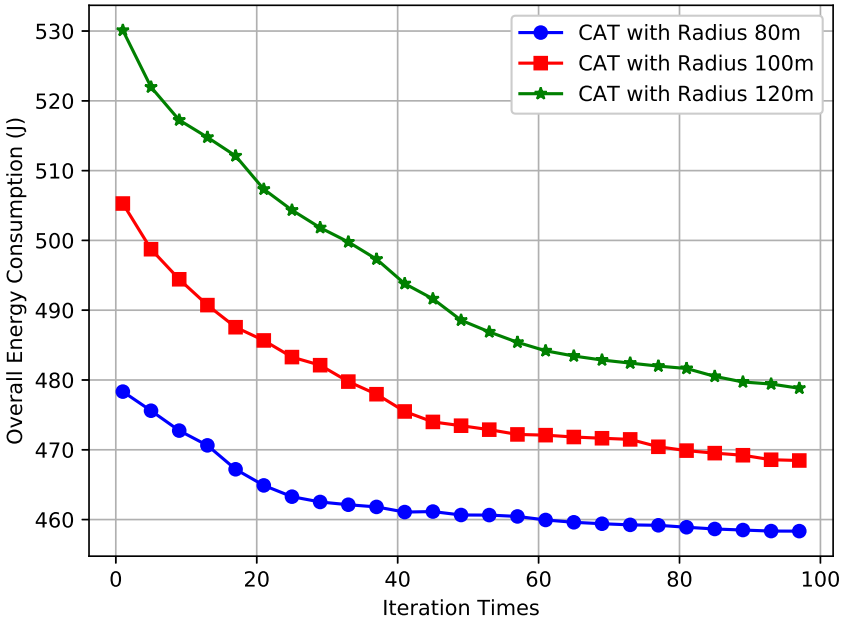
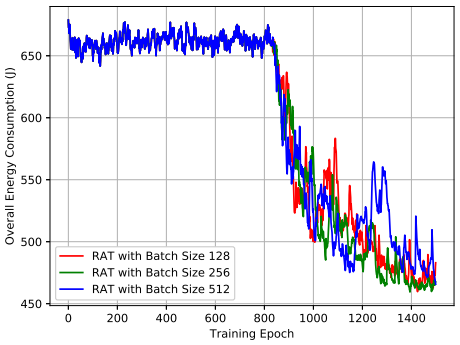


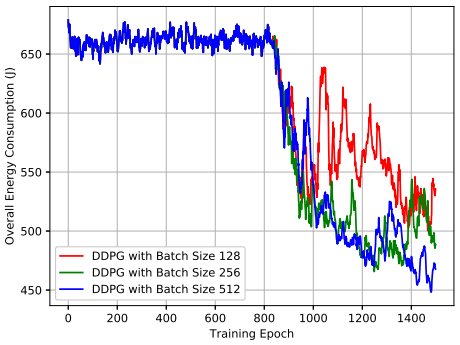
FIGURE 5.3: The convergence performance of proposed CAT.

It is worth noting from Fig. 5.3, 5.4, and 5.5 that when the training cure or iteration curve keeps within particular range, we can conclude that the convergence is obtained. Additionally, as the agent in DRL framework keeps trial and error during the training process, that is to say the training curve will be unavoidable fluctuating.

Then, we show the convergence performance of RAT in training process. From Fig. 5.4 to Fig. 5.5, we compare the influence of hyperparameters to both DDPG and RAT. Prioritized experience replay is applied in RAT. Both RAT and DDPG start the learning

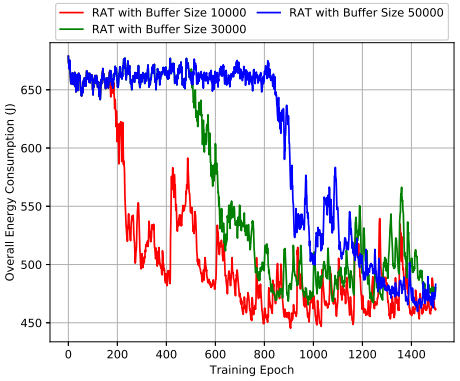


(a) The overall energy consumption of RAT with different batch size.

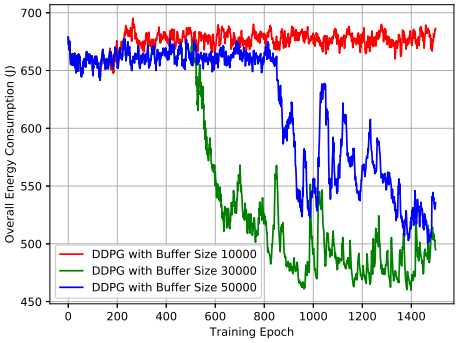


(b) The overall energy consumption of DDPG with different batch size.

FIGURE 5.4: The convergence performance of RAT and DDPG with different size of mini-batch.



(a) The overall energy consumption of RAT with different buffer size.



(b) The overall energy consumption of DDPG with different buffer size.

FIGURE 5.5: The convergence performance of RAT and DDPG with different experience replay buffer.

procedure once the experience replay buffer is full. In Fig. 5.4, we depict the overall energy consumption of RAT and DDPG for different size of mini-batches, where the size of experience replay buffer is 50000. To be more specific, from Fig. 5.4(a), we can see that RAT has the similar convergence performance for different size of mini-batches and it becomes more stable during the learning procedure. In Fig. 5.4(b), when the batch size is 128, DDPG has an obvious fluctuation during the learning procedure. When the batch size is 256, the convergence performance of DDPG becomes worse after the 1400-th epoch. While DDPG can only have a promising convergence performance when the batch size is 512. Overall, from Fig. 5.4, it is clear to see that the RAT is less sensitive to the change of mini-batch than DDPG.

In Fig. 5.5, we depict the overall energy consumption of RAT and DDPG for different

sizes of experience replay buffer, where the size of mini-batch is set as 128. From Fig. 5.5(a) and 5.5(b), when the buffer size is 10000, the proposed RAT finally remains stable between 450 J and 500 J, although it has an obvious fluctuation during the learning process. The DDPG has no convergence tendency during the entire learning procedure. When the buffer size is 50000, DDPG becomes worse after 1000-th epoch, and finally reaches 550 J. Overall, we can observe that DDPG can only have a promising performance when the buffer size is 30000, while RAT can always converge and remain stable during the learning procedure, no matter which the buffer size is. Thus, we can conclude that RAT is less sensitive to the size of experience replay buffer than DDPG.

5.6.2 Trajectory Evaluation of CAT and RAT

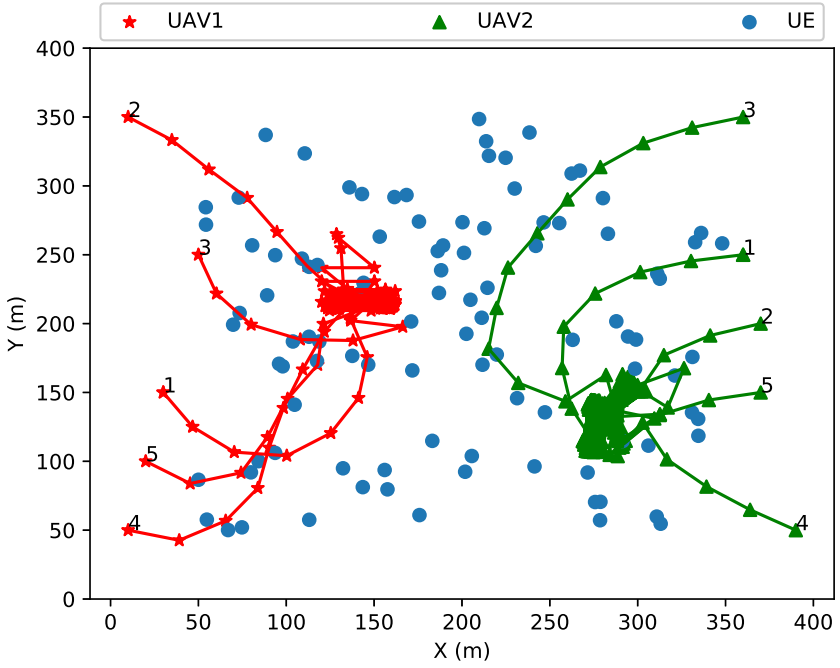


FIGURE 5.6: Multi-UAV enabled F-MEC controlled by RAT.

In Fig. 5.6 and Fig. 5.7, we show the trajectories obtained by RAT and CAT, respectively. Note that during the training procedure, the UAVs controlled by RAT always starts to serve UEs from 20 different taking off points. Additionally, for fairness, the UAVs controlled by CAT have the same taking off points as RAT. For the initial trajectories, we group all the UEs into 6 clusters and each UAV flies in the trajectory connecting all cluster centers one by one. Note that the iteration number of CAT is 10.

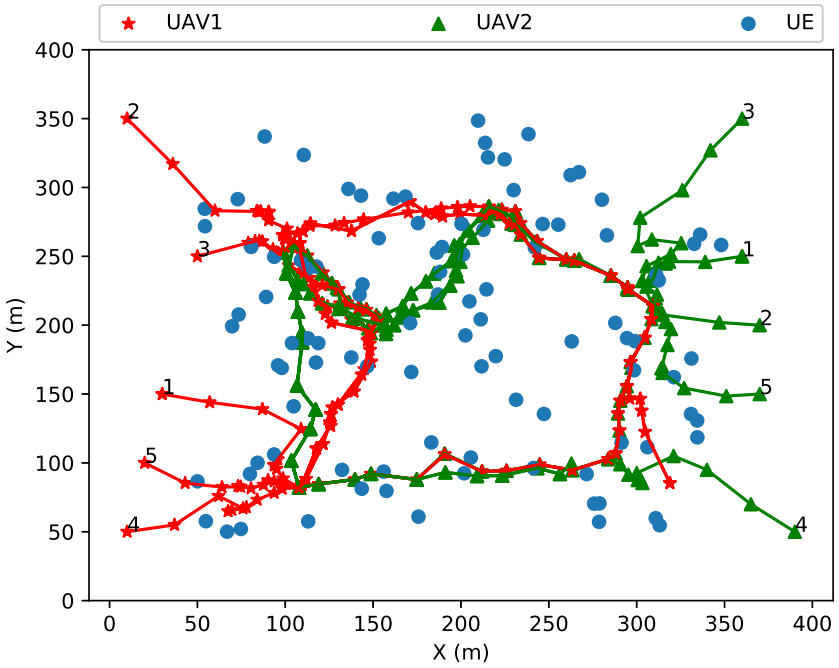


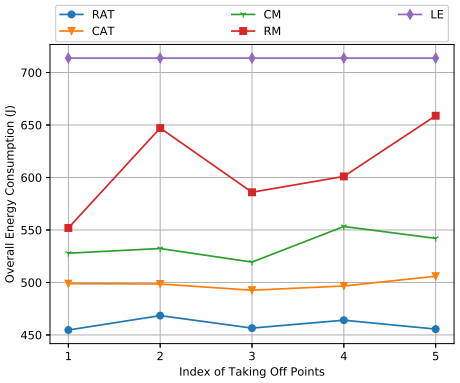
FIGURE 5.7: Multi-UAV enabled F-MEC controlled by CAT.

As shown in Fig. 5.6, we randomly select 5 pairs of taking off points for comparison. One can observe that no matter which the taking off points of the UAVs are, the proposed RAT can guide the UAVs to their certain areas and move around to serve different UEs. This is due to the fact that we train the RAT to converge during the training stage by randomly generating several taking off points of the UAVs. Then, during the testing stage, RAT can intermediately output the best solutions once taking off points are given.

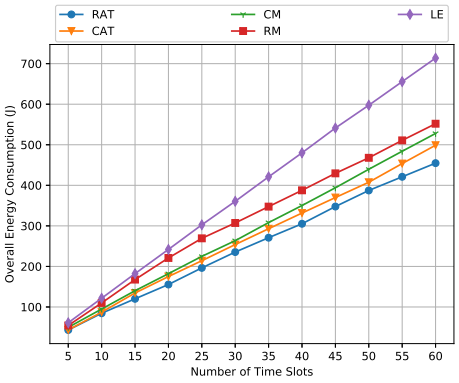
In Fig. 5.7, one can also see that the trajectories obtained by CAT are similar with the initial trajectories. This may indicate that CAT may fall into the local optimum, whereas the proposed RAT has the global search ability due to the exploration feature of DRL.

5.6.3 Energy Consumption Evaluation of CAT and RAT

In Fig. 5.8, we compare the performance of RAT, CAT, CM, RM and LE in terms of energy consumption of UEs. As shown in Fig. 5.8(a), we depict the overall energy consumption of UEs achieved by RAT, CAT, CM, RM, and LE with different taking off points. It is obvious to see that LE has the worst performance. This is because



(a) The overall energy consumption of RAT, CAT, RM, CM, LE with different taking off points.



(b) The overall energy consumption of RAT, CAT, RM, CM, LE in different number of time slots.

FIGURE 5.8: The performance comparison of RAT, CAT, RM, CM, and LE.

all UEs execute their tasks locally without offloading, which will inevitably consume more energy. RM outperforms LE but it fluctuates with the index of taking off points. CM has better performance than RM, which always remains between 520 J and 550 J. CAT outperforms LE, RM, and CM, which remains about 500 J. Additionally, one can observe that RAT achieves the best performance, as expected.

Furthermore, we depict the overall energy consumption of UEs achieved by RAT, CAT, RM, CM, and LE in different number of time slots in Fig. 5.8(b), with the index of taking off points setting as 1. It is readily to see that both the energy consumption of RAT, CAT, RM, CM, and LE increase as the number of time slots increases. LE performs the worst, which consumes above 700 J eventually. Additionally, we can observe that RAT outperforms other algorithms. Moreover, CAT still has considerable performance, which is only slightly worse than RAT.

TABLE 5.3: Executed Time of CAT and RAT

Index	**CAT** (s)	**RAT**	
		Training (s)	**Testing** (s)
1	1405.23	10534.88	1.23
2	1491.74		1.22
3	1460.46		1.20
4	1445.11		1.21
5	1402.48		1.21

In Table 5.3, we show the time consumed by CAT and RAT for each pair of taking off points in Fig. 5.8. Note that RAT is trained for 3000 epochs, while the iteration number

of CAT is 10. One can see that for all the taking off points, the proposed CAT takes over 1400 seconds to find solutions, while RAT only takes 1.2 seconds in average, although it takes longer time in training process. This is because once the RAT are trained properly, it only needs a few number of algebra calculations to obtain the solution.

Additionally, in Fig. 5.9, we analyse the overall energy consumption of RAT, CAT, RM, CM and LE when we have different number of UAVs. Note that for fairness, the UAVs controlled by RAT, CAT, RM, CM have the same taking off points. Specifically, in Fig. 5.9, one observes that the energy consumption of UEs achieved by RAT, CAT, RM, and CM decrease with the increasing number of UAVs. This is because deploying more UAVs provides higher computational capacity. Therefore, more UEs will benefit from offloading, which will decrease their overall energy consumption. Besides, we observe that for all the cases, RAT can achieve the best performance, whereas CAT performs slightly worse than RAT. Also, CM, LM and RM have worse performance than CAT, as expected.

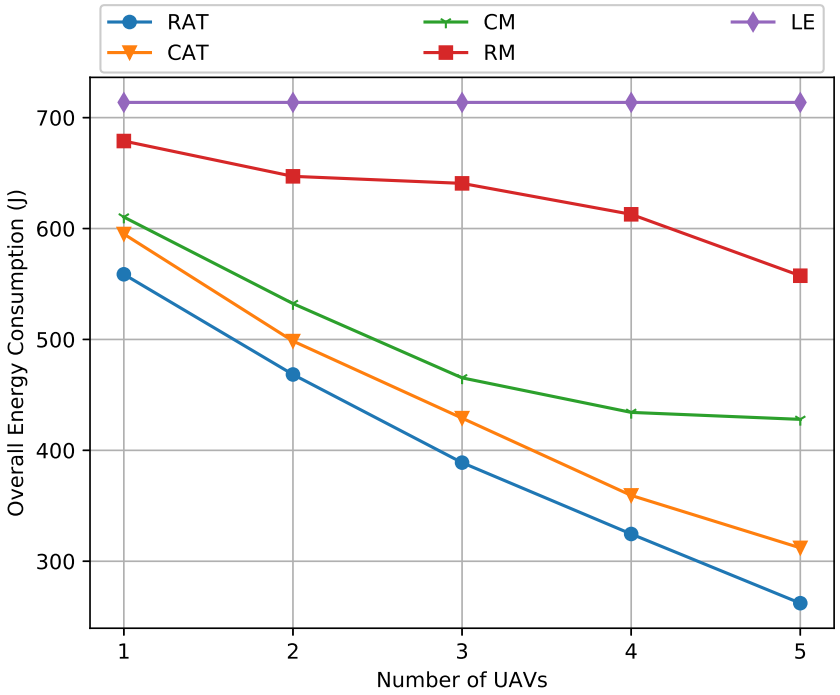


FIGURE 5.9: The overall energy consumption of RAT, CAT, RM, CM, LE with different number of UAVs.

5.6.4 Extension to 3-D channel model

In this subsection, we analyse the performance of proposed RAT in 3-D channel model. We set the number of time slots T as 50, the channel bandwidth as 20 MHz, $D_i(t) \in [5, 10]$ KB, $F_i(t) \in [7.5 \times 10^8, 2 \times 10^9]$ cycles, the size of mini-batch is 512, and the size of experience replay buffer is 100000. In each training epoch, each UAV starts to serve UEs with the altitude of $Z_j(0) = 50$ m. Firstly, we depict the overall energy consumption achieved by the proposed RAT algorithm during the training procedure in Fig. 5.10. One can see that the overall energy consumption of UEs remains between 600 J and 700 J in the beginning. When the learning process starts, the curve decreases and eventually remains slightly above 350 J.

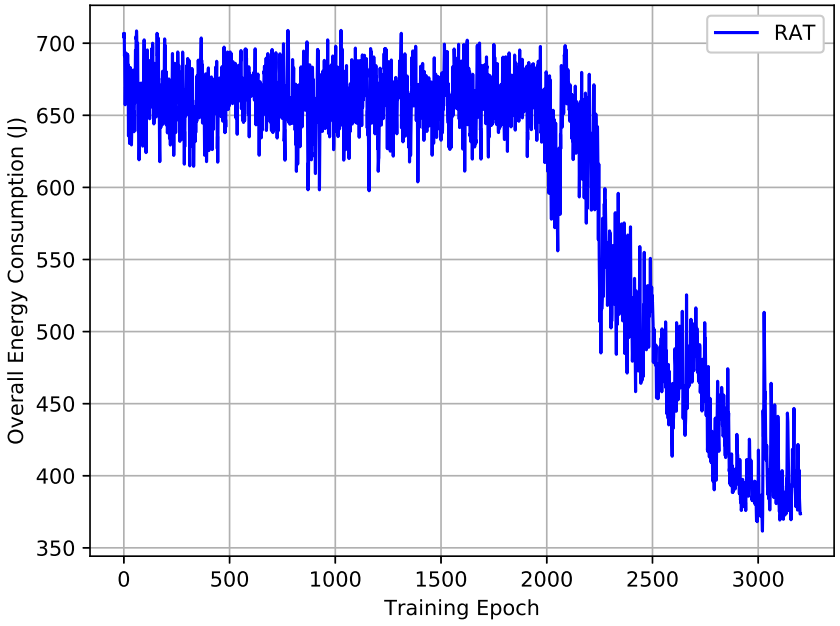


FIGURE 5.10: The convergence performance of proposed RAT in 3-D UAV trajectory and 3-D channel model scenario.

Then, we depict the UAV trajectories obtained by RAT during testing phase in Fig. 5.11. Note that blue dots represent UEs, red stars represent the trajectories of UAV1 and green triangles represent the trajectories of UAV2. As shown in Fig. 5.11, one can see that the UAVs always move from their taking off points to the certain areas, and move around to serve different UEs with the most sufficient distance. In addition, one can observe that each UAV will increase its altitude at the beginning. This is because higher altitude

may increase the coverage radius of the UAV, thereby serving more UEs, although it also decreases the data rate of the offloading process.

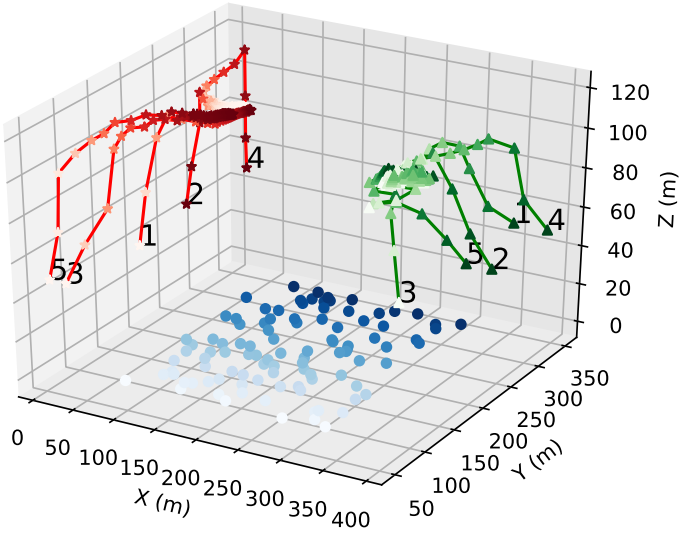
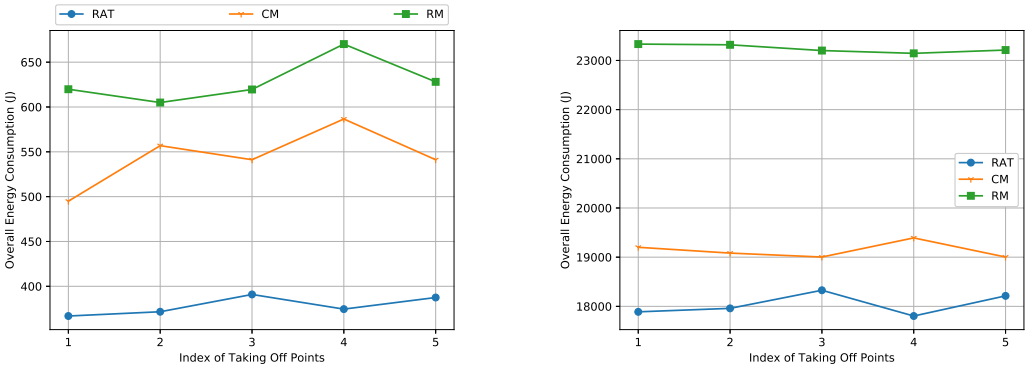


FIGURE 5.11: 3-D trajectories obtained by RAT in 3-D scenario (blue dots for UEs, red stars for UAV1 and green triangles for UAV2).

Furthermore, we analyse the overall energy consumption of UEs and UAVs achieved by RAT, CM, and RM in different scenarios in Fig. 5.12, where the UAVs controlled by CM first climb from the minimal altitude $Z^{\min}$ to the maximal altitude $Z^{\max}$ in the first 10 time slots, and after that fly horizontally. Also, the RM randomly selects the available flying action for each UAV, including the horizontal flying direction, the vertical flying direction, and the flying distance. More precisely, in Fig. 5.12(a), one can observe that our proposed RAT consistently outperforms CM and RM, whereas CM performs worse than RAT but better than RM, as expected.

Finally, we show the overall energy consumption of UAVs achieved by RAT, CM and RM in Fig. 5.12(b). One observes that our proposed RAT has the best performance, whereas CM has the worse performance than RAT, but better than RM.



(a) The overall energy consumption of UEs achieved by RAT, CM, and RM with different taking off points.

(b) The overall energy consumption of UAVs achieved by RAT, CM, and RM with different taking off points.

FIGURE 5.12: The performance comparison of RAT, CM, and RM.

5.7 Summary

In this chapter, we have considered the flying mobile edge computing architecture, by taking advantage of the UAVs to serve as the moving platform. We aim to minimize the energy consumption of all the UEs by optimizing the UAVs' trajectories, user associations and resource allocation. To tackle the multi-UAVs' trajectories problem, a convex optimization-based CAT has been first proposed. Then, in order to conduct fast decision, a DRL-based RAT including a matching algorithm has also been proposed. Simulation results show that CAT and RAT have considerable performance.

Chapter 6

Multi-Agent DRL-based Trajectory Planning for Cooperative UAV-enabled MEC

6.1 Introduction

As a benefit of their compelling features, unmanned aerial vehicles (UAVs) are expected to play a vital role in wireless communication systems. To elaborate a little further, UAVs are capable of providing wireless connectivity even without network infrastructure, or complement the conventional base stations (BSs), whose coverage may suffer from severe blockage due to tall buildings or by the damage caused by natural disasters [26]. In order to support reliable communication links, UAVs can promptly adjust their locations according to the dynamic communication environment. Furthermore, since UAVs can be deployed freely and flexibly in three-dimensional (3D) space, direct line-of-sight (LoS) communication with ground-UEs can be readily established, which can potentially boost the throughput in practical scenarios [41]. As a benefit of the above appealing features, in [80] and [81], both fixed-wing UAVs and rotary-wing UAVs were considered as the relaying nodes, for providing seamless connectivity. In [82], Wang *et al.* investigated a fixed-wing UAV-to-UAV communication system, and they proposed a path planning algorithm for minimizing the latency of information transmission, under the constraints of accelerations, location uncertainties and throughput. In [83], Cui *et*

al. studied the problem of maximizing the average data rate among UEs in mobile-UAV-enabled networks both in orthogonal multiple access (OMA) and non-orthogonal multiple access (NOMA) modes. Furthermore, in agricultural applications, as well as in weather monitoring and wildfire management, UAV can be utilized as a mobile data collector [42]. As a future development, in [36], the authors deployed the UAV as the mobile energy transmitter (ET) in a wireless power transfer (WPT) system.

In order to fully exploit the potential of UAVs in wireless communication systems, it is important to investigate their path planning, hovering altitude and trajectory control [84–86]. In [86], Wang *et al.* creatively proposed a joint UAV altitude and power allocation optimization method, which beneficially alleviated the inter-cell interference of each UAV network. In [22], Al-Hourani *et al.* optimized the latitude of UAVs in order to provide the maximum radio coverage area on the ground. In [87], both static and mobile UAVs were considered in device-to-device (D2D) networks. Additionally, the UAV's altitude was optimized for maximizing the system's sum-rate and coverage probability. To tackle the throughput maximization problem of UAV-aided mobile relaying systems, Zeng *et al.* [88] proposed an iterative algorithm to optimize the UAV's trajectory and power allocation. In the content of multi-UAV enabled multiuser systems, Wu *et al.* [43] maximized the minimum throughput over all ground users by jointly optimizing the user scheduling, power control and UAV trajectories. In order to meet the different quality-of-service (QoS) requirement of users, Alzenad *et al.* [89] investigated coverage-placement problem of UAV-BSs and proposed an optimal placement algorithm for maximizing the number of users supported.

In recent years, mobile edge computing (MEC) has been shown to dramatically improve the user experience [67, 90]. By providing both computing and storage hardware at the network edges, namely at the BSs or access points (APs), the resource-limited UEs have the option of offloading their computation-intensive and latency-critical applications to the MEC servers [91]. Due to the mobility of UAVs, recent years have seen research progress on the integration of UAVs with MEC [92, 93]. In [94], Motlagh *et al.*, were amongst the first who proposed UAV-enabled MEC, in which UEs can significantly reduce the energy consumption via offloading. In order to minimize the overall energy dissipation of UEs while meeting their QoS requirement, Jeong *et al.* [95] proposed an efficient successive convex approximation-based algorithm for jointly optimizing the bit

allocation and UAV's trajectory. Considering a multi-UAV system, Hua *et al.* [96] investigated the multi-UAV scenario, and they optimized the UAVs' trajectories, transmit power and user scheduling.

Given the recent advances in machine learning [97], the combination of deep neural networks (DNNs) [98] and reinforcement learning (RL) [99], i.e., deep reinforcement learning (DRL) has become a hot research topic. In DRL, an agent is assumed to interact with the environment for learning the optimal policy with the aid of exploration. Compared to traditional RL, DRL facilitates more accurate convergence and approximation by exploiting the power of DNNs for estimating the associated functions in RL [100]. The great potential of DRL in solving complex control problems has also been demonstrated in [29, 31, 32, 101, 102]. In [29], Mnih *et al.* introduced the deep Q network (DQN) philosophy, which ignited the field of DRL. For instance, Wang *et al.*[101] systematically investigated the problem of distributed Q-learning aided heterogeneous network association in the content of energy-efficient Internet of things (IoT). In order to improve the training procedure, DQN relies on a pair of techniques namely, experience replay and target networks. For the sake of tackling the typical over-estimation problem of RL, a double DQN (D-DQN) was proposed by Van Hasselt *et al.* [32]. However, DQN may suffer from the curse of high-dimensional action spaces and cannot be readily applied to continuous domains. Thus, motivated by this, Lillicrap *et al.* [31] proposed a deep deterministic policy gradient (DDPG) technique based on the so-called actor-critic architecture, which can be readily applied for a range of challenging problems. A comprehensive survey of multi-agent RL, have also been provided by Bu *et al* [102].

Against the above background, we conceive a cooperative UAV-enabled MEC framework, where each UAV is controlled by a dedicated agent. We aim for jointly maximizing the geographical fairness[1] among the UEs covered, the fairness of UE-load of each UAV[2], while minimizing the overall energy consumption of UEs by optimizing each UAV's trajectory and offloading decisions. This is a complex problem which includes both integer and continuous variables. Hence it is challenging to address it by traditional algorithms, such as convex optimization and dynamic programming. Therefore, we conceive a multi-agent deep reinforcement learning based solution, with the help of the popular Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [103] for solving

[1]The geographical fairness reflects the QoS level of UEs served by UAVs from the initial time slot to the current time slot.

[2]The UE-load of UAV is defined in (6.18).

it. Given the UAVs' trajectories, a low-complexity approach is introduced for optimizing the offloading decisions of UEs. Our simulation results will show that the proposed DRL based algorithm outperforms the benchmark algorithms. We summarize the difference between our work and the existing literature in Table 6.1.

TABLE 6.1: Comparison between our work and the existing literature.

Reference	Single UAV	Multi UAV	Mobile edge computing (incl.)	Path planning	Offloading decision	Reinforcement learning (e.g., Q-learning)	Multi agent learning	DNN
[41]	✓			✓				
[81]	✓			✓				
[82]		✓		✓				
[83]	✓			✓				
[42]	✓			✓				
[36]	✓			✓				
[84]		✓		✓				✓
[85]		✓		✓		✓	✓	
[86]		✓		✓				
[87]	✓			✓				
[88]		✓		✓				
[43]		✓		✓				
[89]	✓			✓				
[92]		✓	✓	✓	✓			
[93]	✓			✓	✓			
[95]	✓		✓	✓				
[96]		✓		✓				
[101]						✓	✓	
[103]						✓	✓	✓
Our work		✓	✓	✓	✓	✓	✓	✓

The rest of the chapter is organized as follows. In Section 6.2, we introduce the system model and the optimization problem. In Section 6.3, our multi-agent based DRL algorithm is proposed. Our experimental results are shown in Section 6.4. Finally, our summary is drawn in Section 6.5.

The main notations used in this chapter are summarized in Table 6.2.

TABLE 6.2: List of main notations

Notation	**Description**
n, N, $\mathcal{N}$	The index, number and the set of UEs
m, M, $\mathcal{M}$	The index, number and the set of UAVs
t, T, $\mathcal{T}$	The index, number and the set of TSs
$z_{n,m,t}$	Offloading decision of UE n
$S_{n,t}$	Computation task of UE n in TS t
$D_{n,t}$	Data volume of task $S_{n,t}$
$F_{n,t}$	Overall CPU cycles required for task $S_{n,t}$
$f_{n,m,t}$	Computation capacity of UAV m allocated to UE n
$T^{C}_{n,m,t}$	Execution time of UAV m to UE n in TS t
$T^{Tr}_{n,m,t}$	Transmission time of UE n to UAV m in TS t
T^{max}	Maximal time duration of each TS
$\alpha_{m,t}, d_{m,t}$	Flying angle and distance of UAV m in TS t
d^{max}	Maximal flying distance of UAV in each TS
$[X_{m,t}, Y_{m,t}, H]$	Coordinates of UAV m in TS t
$[x_n, y_n]$	Coordinates of UE n
$R_{n,m,t}$	Horizontal distance between UAV m and UE n
$R_{m,m',t}$	Horizontal distance between UAV m and UAV m'
R^{max}	Maximal horizontal coverage radius of UAV
$r_{n,m,t}$	Transmitting data rate of UE n to UAV m
$E^{C}_{n,m,t}$	Energy consumption for task execution
$E^{Tr}_{n,m,t}$	Energy consumption for offloading
$c_{m,t}$	Relative UE-load of UAV m in TS t
f^{u}_{t}	Fairness index of UE-load of each UAV in TS t
f^{e}_{t}	Fairness index of UEs in TS t

6.2 System Model

In this section, we describe the system model. As shown in Fig. 6.1, we assume that there are N UEs randomly distributed in a square-shaped area with side length l^{max}, and the set of UEs is denoted as $\mathcal{N} \triangleq \{n = 1, 2, ...N\}$. There are M UAVs flying at a fixed altitude H over the target area to serve the ground UEs, and the set of UAVs is denoted as $\mathcal{M} \triangleq \{m = 1, 2, ..., M\}$. We also assume that UAVs can be deployed

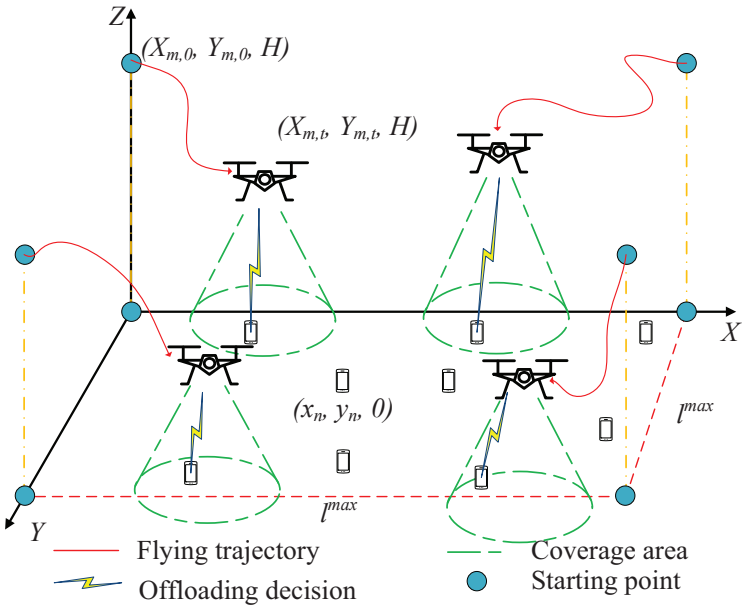


FIGURE 6.1: Overall System Architecture

and easily charged on the building roof when UAVs run out of their energy. Assume that each UE has a computational task to be executed at each time slot (TS) over T consecutive TSs, $\mathcal{T} \triangleq \{t = 1, 2, ..., T\}$. Each of the tasks can be executed either by the UE or offloaded to one of the UAVs. We define a new set $m \in \mathcal{M}' \triangleq \{0, 1, ..., M\}$ to denote the possible places where the tasks can be executed, with $m = 0$ representing local execution. Then, we define the offloading decision variable $z_{n,m,t}$ as

$$z_{n,m,t} = \{0, 1\}, \forall n \in \mathcal{N}, m \in \mathcal{M}', t \in \mathcal{T}, \tag{6.1}$$

where $z_{n,m,t} = 1, m \neq 0$ means that UE n decides to offload the task to UAV m in TS t, while $z_{n,m,t} = 1, m = 0$ represents that UE n carries out the task itself in TS t, and otherwise $z_{n,m,t} = 0$. Furthermore, we assume that each task can only be executed at a single place. Thus, we have

$$\sum_{m=0}^{M} z_{n,m,t} = 1, \forall n \in \mathcal{N}, t \in \mathcal{T}. \tag{6.2}$$

Similarly to [104], in the TS t, we assume that UE n has a computationally intensive task $S_{n,t}$ to be executed, which is defined as

$$S_{n,t} = \{D_{n,t}, F_{n,t}\}, \forall n \in \mathcal{N}, t \in \mathcal{T}, \tag{6.3}$$

where $D_{n,t}$ denotes the data volume to be processed, while $F_{n,t}$ describes the total number of the CPU cycles required for executing this task. Both $D_{n,t}$ and $F_{n,t}$ can be characterized as in [105].

Furthermore, in TS t, each of UAV flies in a direction determined by the angle of $\alpha_{m,t} \in [0, 2\pi)$, distance of $d_{m,t} \in [0, d^{max}]$, and cannot go beyond the border of the target area. We assume that the initial coordinates of UAV m are set as $[X_{m,0}, Y_{m,0}, H]$. Then, the coordinates of UAV m in TS t can be calculated as $[X_{m,t}, Y_{m,t}, H]$, where $X_{m,t} = X_{m,0} + \sum_{t'=1}^{t} d_{m,t'}\cos(\alpha_{m,t'})$ and $Y_{m,t} = Y_{m,0} + \sum_{t'=1}^{t} d_{m,t'}\sin(\alpha_{m,t'})$. Thus, we have

$$0 \leq X_{m,t} \leq l^{max}, \ \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{6.4}$$

and

$$0 \leq Y_{m,t} \leq l^{max}, \ \forall m \in \mathcal{M}, t \in \mathcal{T}. \tag{6.5}$$

Additionally, we denote the distance between UAV m and UAV m' in TS t as $R_{m,m',t}$, which can be expressed as

$$R_{m,m',t} = \sqrt{(X_{m,t} - X_{m',t})^2 + (Y_{m,t} - Y_{m',t})^2}. \tag{6.6}$$

We assume that the UAVs should keep a minimal distance of R^u for avoiding their collision in each TS. Then, we have

$$R_{m,m',t} \geq R^u, \forall m, m' \in \mathcal{M}, m \neq m'. \tag{6.7}$$

The horizontal distance between UE n and UAV m in TS t is calculated as

$$R_{n,m,t} = \sqrt{(X_{m,t} - x_n)^2 + (Y_{m,t} - y_n)^2}, \ \forall n \in N, m \in \mathcal{M}, t \in \mathcal{T}, \tag{6.8}$$

where $[x_n, y_n]$ is assumed to be the coordinate of UE n. Note that if UE n decides to offload a task to UAV m in TS t, it must be in the coverage of UAV m. Then, we have

$$z_{n,m,t} R_{n,m,t} \leq R^{max}, \ \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, \tag{6.9}$$

where R^{max} is the maximal horizontal coverage radius of the UAVs.

Then, the offloading data rate can be expressed by

$$r_{n,m,t} = B\log_2\left(1 + \frac{\rho P_n}{H^2 + R_{n,m,t}^2}\right), \ \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, \tag{6.10}$$

where B is the channel's bandwidth, P_n is the transmission power of UE n, $\rho = g_0 G_0/\sigma^2$, $G_0 \approx 2.2846$, g_0 is the channel's power gain at the reference distance of 1 m and σ^2 is the noise power [106]. Here we do not consider any particular modulation and coding scheme.

Thus, if UE n decides for offloading its task to UAV m in TS t, the time required for offloading the data is given by

$$T_{n,m,t}^{Tr} = \frac{D_{n,t}}{r_{n,m,t}}, \ \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, \tag{6.11}$$

and the execution time of the task can be expressed as

$$T_{n,m,t}^{C} = \frac{F_{n,t}}{f_{n,m,t}}, \forall n \in \mathcal{N}, m \in \mathcal{M}', t \in \mathcal{T}, \tag{6.12}$$

where $f_{n,m,t}$ represents the computational capability of UAV m that can be allocated to UE n, and $m = 0$ indicates local execution. Thus, the overall time required for executing the task can be described as

$$[l]T_{n,m,t} = \begin{cases} T_{n,m,t}^{C}, & \text{if local execution,} \\ T_{n,m,t}^{Tr} + T_{n,m,t}^{C}, & \text{if offloading.} \end{cases} \tag{6.13}$$

We also assume that all tasks should be executed within the maximal time duration T^{max} of TS. Then, we have

$$z_{n,m,t} T_{n,m,t} \leq T^{max}, \ \forall n \in \mathcal{N}, m \in \mathcal{M}', t \in \mathcal{T}. \tag{6.14}$$

According to [56], if the UE n decides to execute a task locally, the energy consumption is given by

$$E^{C}_{n,m,t} = k_n (f_{n,m,t})^{v_n} T^{C}_{n,m,t}, \ \forall n \in \mathcal{N}, t \in \mathcal{T}, \tag{6.15}$$

where $k_n \geq 0$, $v_n \geq 1$ are positive coefficients.

If UE n decides to offload a task, the energy consumption of offloading is

$$E^{Tr}_{n,m,t} = P_n T^{Tr}_{n,m,t}, \ \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}. \tag{6.16}$$

Thus, the energy consumption at UE n can be expressed as

$$[l]E_{n,m,t} = \begin{cases} E^{C}_{n,m,t}, & \text{if local execution,} \\ E^{Tr}_{n,m,t}, & \text{if offloading.} \end{cases} \tag{6.17}$$

Then, we define $c_{m,t} \in [0, 1]$ as the relative UE-load of UAV m in TS t, as:

$$c_{m,t} = \frac{\sum_{n=1}^{N} z_{n,m,t}}{N}, \ \forall m \in \mathcal{M}, t \in \mathcal{T}. \tag{6.18}$$

In this chapter, our first objective is to minimize the total energy consumption of UEs via optimizing both the offloading decisions and the UAVs' trajectories. However, this may lead to an unfair process since some UAVs may serve more UEs than others. To address this issue, we define a fairness index f^{u}_{t} as

$$f^{u}_{t} = \frac{\left(\sum_{m=1}^{M} \sum_{t'=1}^{t} c_{m,t'}\right)^2}{M \sum_{m=1}^{M} \left(\sum_{t'=1}^{t} c_{m,t'}\right)^2}, \tag{6.19}$$

where f^{u}_{t} reflects the level of fairness among the UAVs physically, if all the UAVs have a similar UE-load commencing from the initial TS up to TS t, the value of f^{u}_{t} is closer to 1.

Then, to avoid the situation that some UEs are served during many TSs, while others are never served at all, we define another geographical fairness f^{e}_{t} as follows

$$f^{e}_{t} = \frac{\left(\sum_{n=1}^{N} \sum_{t'=1}^{t} z_{n,m,t'}\right)^2}{N \sum_{n=1}^{N} \left(\sum_{t'=1}^{t} z_{n,m,t'}\right)^2}, \tag{6.20}$$

where f_t^e reflects the level of fairness among the UEs, explicitly, if all UEs are served for a similar number of TSs commencing from the initial TS to the TS t, the value of f_t^e is closer to 1.

Then, we formulate our optimization problem as follows

$$\mathcal{P}1: \max_{\boldsymbol{P},\boldsymbol{Z}} \sum_{t=1}^{T} \frac{f_t^u \cdot f_t^e}{\sum_{n=1}^{N}\sum_{m=0}^{M} z_{n,m,t} E_{n,m,t}} \tag{6.21a}$$

subject to:

$$z_{n,m,t} = \{0,1\}, \forall n \in \mathcal{N}, m \in \mathcal{M}', t \in \mathcal{T}, \tag{6.21b}$$

$$\sum_{m=0}^{M} z_{n,m,t} = 1, \forall n \in \mathcal{N}, t \in \mathcal{T}, \tag{6.21c}$$

$$0 \leq X_{m,t} \leq l^{max}, \ \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{6.21d}$$

$$0 \leq Y_{m,t} \leq l^{max}, \ \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{6.21e}$$

$$0 \leq \alpha_{m,t} < 2\pi, \ \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{6.21f}$$

$$0 \leq d_{m,t} \leq d^{max}, \ \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{6.21g}$$

$$R_{m,m',t} \geq R^u, \forall m, m' \in \mathcal{M}, m \neq m', \tag{6.21h}$$

$$z_{n,m,t} R_{n,m,t} \leq R^{max}, \ \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, \tag{6.21i}$$

$$z_{n,m,t} T_{n,m,t} \leq T^{max}, \ \forall n \in \mathcal{N}, m \in \mathcal{M}', t \in \mathcal{T}. \tag{6.21j}$$

where $\boldsymbol{P} = \{\alpha_{m,t}, d_{m,t}, \forall m \in \mathcal{M}, t \in \mathcal{T}\}$ and $\boldsymbol{Z} = \{z_{n,m,t}, \forall n \in \mathcal{N}, m \in \mathcal{M}', t \in \mathcal{T}\}$. Our objectives are to maximize the fairness of UE-load of each UAV and the fairness of the number of times that each UE is served by UAVs over all the TSs, while minimizing the overall energy consumption of UEs. It is readily observed that the optimization problem cannot be solved by traditional approaches, since it involves both the continuous variables $\boldsymbol{P}$ and the discrete variables $\boldsymbol{Z}$. Thus, in this chapter, a Multi-Agent deep reinforcement learning based Trajectory control algorithm (MAT) is proposed.

6.3 The Proposed Algorithm

In this section, we present our proposed algorithm.

6.3.1 MAT

In this section, by applying the popular MADDPG [103], we conceive a multi-agent MDP, namely an observable Markov game [107]. It is assumed that there are M agents interacting with the environment characterized by a set of states $\mathcal{S} \triangleq \{s_t, t \in \mathcal{T}\}$ and a set of actions $\mathcal{A} \triangleq \{a_t, t \in \mathcal{T}\}$. The state s_t consists of the private observation $o_{m,t}$ and some other extra information known by each agent. Additionally, each UAV is controlled by its dedicated agent. In each TS, each agent obtains its private observation $o_{m,t}$ and takes its own action $a_{m,t}$ as well as receives a reward $r_{m,t}$. Then, the environment updates the state and traverses to a new state. Note that each agent is equipped with an actor network $a_{m,t} = \pi^m(o_{m,t})$, a critic network $Q^m(s_t, a_t)$, their target networks $a_{m,t+1} = \pi^{m'}(o_{m,t+1})$ and $Q^{m'}(s_{t+1}, a_{t+1})$, as well as an experience replay buffer B_m.

The proposed algorithm is based on the framework of centralized training combined with decentralized execution. During the training process, each agent sends its own private observation $o_{m,t}$ and action $a_{m,t}$ to the environment, and then the states s_t which consist of the observations of all the agents and actions are sent back to each agent. Here, all the agents can exchange their private information simultaneously with each other, including coordinates. Furthermore, the critic network of each agent is trained with the states and actions that includes all the agents' observations and actions. Then, during the testing process, each agent can execute its action by only receiving its own private observations $o_{m,t}$, which can potentially maximize the accumulated rewards.

Thus, we define the observation, action and reward function for each agent in TS t as follows:

1. Observation $o_{m,t}$: we first add the coordinates $[X_{m,t}, Y_{m,t}]$ of UAV m in TS t into the observation of agent m. For avoiding collisions between each pair of UAVs, we define the set of relative UAV distances $\{R_{m,m',t},\ m' \in \mathcal{M}, m' \neq m\}$ as part of the observation. Additionally, for better exploration, we also add the set of accumulated times of UEs served by UAVs and UE-load of UAVs commencing from the initial TS up to TS t, i.e., $\{\sum_{t'=1}^{t} z_{n,m,t'},\ \forall n \in \mathcal{N}\}, \{\sum_{t'=1}^{t} c_{m,t'},\ \forall m \in \mathcal{M}\}$, respectively into the observation set.
2. Action $a_{m,t}$: we define the UAV's flying direction and distance as the action $a_{m,t} = \{\alpha_{m,t}, d_{m,t}\}$ of the m-th UAV in the t-th TS.

3. Reward Function $r_{m,t}$: we define the reward function as:

$$r_{m,t} = \frac{f_t^u \cdot f_t^e}{\frac{1}{N}\sum_{n=1}^{N}\sum_{m=0}^{M} z_{n,m,t} \cdot E_{n,m,t}} - p_m, \tag{6.22}$$

where p_m is the penalty incurred if UAV m flies out of the target area or UAV m is collided with another UAV (i.e., the relative distance is under the defined limit).

Then, we define the entire state s_t, and action a_t as follows

1. State s_t: the state consists of the observations of all the agents, which is expressed as $s_t = \{o_{m,t},\ \forall m \in \mathcal{M}\}$.
2. Action a_t: the action consists of the actions of all the agents, which is $a_t = \{a_{m,t},\ \forall m \in \mathcal{M}\}$.

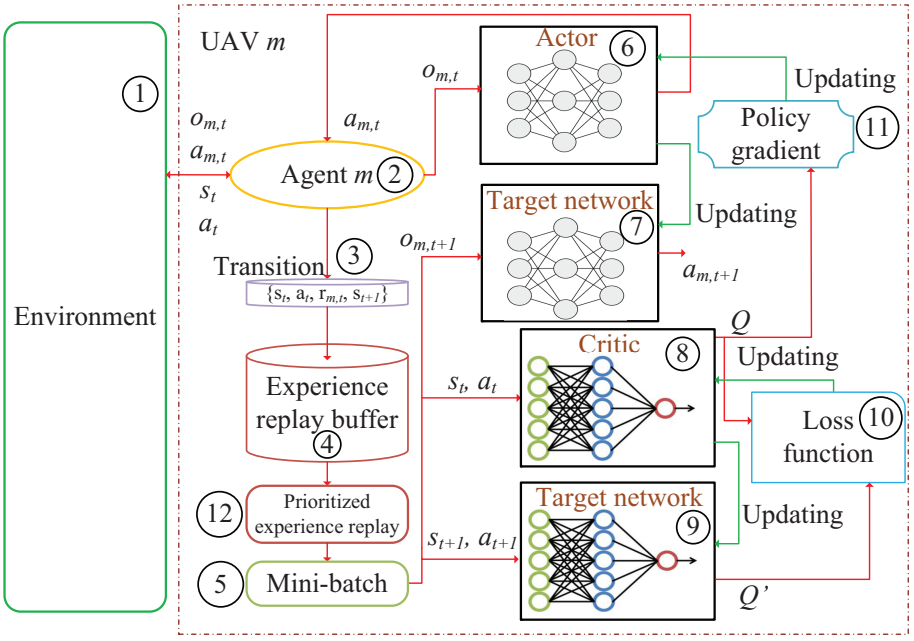


FIGURE 6.2: Structure of UAV m (i.e., controlled by Agent m)

We show the structure of agent m in Fig. 6.2. During its interaction with the environment ①, each UAV (controlled by agent) ② selects the optimal action associated with its actor network $\pi^m(\cdot)$ ⑥, and then obtains the Q value from the critic network $Q^m(\cdot)$ ⑧ as well as its target action and target Q value from $\pi^{m'}$ ⑦ and $Q^{m'}(\cdot)$ ⑨ respectively. The profile of observation, action and reward, which determine the transition are defined as $e_{m,t} \triangleq \{s_t, a_t, r_{m,t}, s_{t+1}\}$ that are stored in the experience replay buffer ④. However, during the training procedure, randomly sampling the mini-batch ⑤ may have

unpredictable effects, since some transitions associated with poor attempts may lead to the termination of the training procedure or may not converge. As a result, Schaul *et al.* [55] pointed out that transitions having high Temporal Difference (TD)-error often indicate successful attempts. The TD-error δ_m of agent m can be defined as follows

$$\delta_m = r_{m,t} + \gamma Q^{m'}(s_{t+1}, a_{t+1}|\theta^{Q^{m'}}) - Q^m(s_t, a_t|\theta^{Q^m}), \ \forall m \in \mathcal{M}, t \in \mathcal{T}. \tag{6.23}$$

Additionally, motivated by [55], we utilize a prioritized experience replay scheme, in which the absolute TD-error $|\delta_{m,k}|$ was used for evaluating the probability of the k-th sampled transition in the mini-batch. Then, the probability of sampling the k-th transition is expressed as

$$P_{m,k} = \frac{(|\delta_{m,k}| + \varepsilon)^\beta}{\sum_{k'=1}^{K}(|\delta_{m,k}| + \varepsilon)^\beta}, \ \forall m \in \mathcal{M}, \tag{6.24}$$

where K is the size of mini-batch, ε is a positive constant value, and β is 0.6. Thus, the loss function ⑩ of the agent m is defined as

$$L(\theta^{Q^m}) = \mathbb{E}\big[\frac{1}{(K \cdot P_{m,k})^\mu}(\delta_m)^2\big], \tag{6.25}$$

where μ is given as 0.4.

Then, the critic network ⑧ of agent m can be updated by the loss function ⑩ provided in (6.25). Furthermore, the actor network ⑥ of agent m can be trained by the policy gradient ⑪ defined as

$$\nabla_{\theta^{\pi^m}} J = \mathbb{E}\big[\nabla_{\theta^{\pi^m}} \pi^m(o_{m,t}|\theta^{\pi^m}) \nabla_{a_{m,t}} Q^m(s_t, a_t)|\theta^{Q^m}\big], \ \forall m \in \mathcal{M}, t \in \mathcal{T}. \tag{6.26}$$

Given the UAVs' trajectories, we introduce a low-complexity approach for optimizing the offloading decisions of UEs. Here, we do not consider the constraint of the maximal available computing resource in each UAV. This can be readily extended to more practical scenarios, where each UAV can only have a certain amount of the computing resources, with the introduction of the matching algorithm. We will leave this idea for our future work. For each UE in TS t, we select the offloading decision based on the

following expression

$$[l]z_{n,m,t} = \begin{cases} 1, & m = \underset{m' \in \mathcal{M}'}{\operatorname{argmin}}\{E_{n,m',t}\}, \\ 0, & \text{otherwise.} \end{cases} \tag{6.27}$$

Specifically, after the movement of UAVs, each UE can select the most suitable UAV for offloading, which consumes the least energy. Otherwise, the UE may execute the task itself. If UE n decides to offload a task to UAV m, the computational capacity allocated to UE from the UAV is expressed as

$$f_{n,m,t} = \frac{F_{n,t}}{T^{max} - T^{Tr}_{n,m,t}}. \tag{6.28}$$

Algorithm 5 MAT

for UAV m in $\mathcal{M}$ **do**
  Initialize actor network $\pi^m(\cdot)$, critic network $Q^m(\cdot)$ with parameters θ^{π^m} and θ^{Q^m};
  Initialize target networks $\pi^{m'}(\cdot)$ and $Q^{m'}(\cdot)$ with parameters $\theta^{\pi^{m'}} = \theta^{\pi^m}$ and $\theta^{Q^{m'}} = \theta^{Q^m}$;
  Initialize experience replay buffer B_m;
end for
for Episode = 1,2,...,e^{max} **do**
  for UAV m in $\mathcal{M}$ **do**
    Initialize observation $o_{m,t}$;
  end for
  for TS t in $\mathcal{T}$ **do**
    Obtain s_t;
    for UAV m in $\mathcal{M}$ **do**
      Obtain action $a_{m,t} = \pi^m(o_{m,t}|\theta^{\pi^m}) + \epsilon$;
      Execute $a_{m,t}$. Note that the UAV will stay at the current location if it flies out of the target area or it is collided with another UAV;
    end for
    Obtain a_t;
    for UE n in $\mathcal{N}$ **do**
      Obtain the available offloading decision $z_{n,m,t}$ that consumes the least energy according to (6.27);
      Calculate $E_{n,m,t}$;
    end for
    for UAV m in $\mathcal{M}$ **do**
      Obtain $r_{m,t}$ according to (6.22) ;
      Obtain $o_{m,t+1}$;
    end for
    Obtain s_{t+1};
    for UAV m in $\mathcal{M}$ **do**
      Store transition $\{s_t, a_t, r_{m,t}, s_{t+1}\}$ into experience replay buffer B_m with priority $|\delta_m| + \varepsilon$;
      if learning process starts **then**
        Sample a mini-batch of K transitions from B_m with probability $P_{m,k}$;
        Update critic network according to (6.25);
        Update actor network according to (6.26);
        Update target networks with updating rate τ:
        $\theta^{\pi^{m'}} \leftarrow \tau\theta^{\pi^m} + (1-\tau)\theta^{\pi^{m'}}$;
        $\theta^{Q^{m'}} \leftarrow \tau\theta^{Q^m} + (1-\tau)\theta^{Q^{m'}}$;
        Update priorities of K transitions;
      end if
    end for
  end for
end for

We provide the pseudo code of proposed procedure in Algorithm 6. Specifically, we carry out the initialization between Line 1 and 5 at the beginning, where each UAV initializes its actor, critic and two target networks. Then, the training procedure starts from Line 6, where each UAV first obtains its observation from the environment ①. Note that each UAV is controlled by its dedicated agent ②. Then, based on the achieved observation, each UAV selects the action $a_{m,t}$, which is generated by its actor network ⑥. In order to achieve a better exploration, we add a noise parameter ϵ, which follows a normal distribution with zero mean and a variance of 1. The exploration noise decays with the rate of 0.9995. Then, the UAV executes the action. Note that the UAV will stay at the current location and obtains a penalty p_m, if the next location is obtained outside the target area or the UAV is collided with other UAVs. Then, UE selects the UAV which consumes the least energy according to (6.27). Next, we obtain the reward $r_{m,t}$ and the next observation $o_{m,t+1}$. Then, each UAV stores the transition ③ into its experience replay buffer B_m ④. From Line 28 to 34, when the learning procedure starts, the mini-batch ⑤ with prioritized experience replay ⑫ scheme samples K transitions from B_m. Furthermore, the critic network ⑧ is updated by the loss function ⑩ provided in (6.25), and the actor network ⑥ is also updated by the policy gradient ⑪ provided in (6.26). After that, the pair of target networks are updated at a rate of τ. Finally, we update the priorities of the K sampled transitions.

6.4 Simulation Results

In this section, we rely on our simulations for evaluating the performance of the proposed MAT algorithm. The simulations are conducted by using Python 3.7 and Tensorflow 1.15.0. We employ four fully-connected hidden layers having $[400, 300, 200, 200]$ neurons in both the actor and critic networks. The actor network is trained at the learning rate of 3×10^{-5}, while the critic network is trained at the learning rate of 10^{-4}. The AdamOptimizer [108] is used for updating the actor and critic networks. We set the target region to be a square-shaped area with side length of $l^{max} = 100$ m, where 50 UEs are randomly and uniformly distributed. We set the initial coordinates of UAVs to $[10, 10]$, $[90, 90]$, $[10, 90]$ and $[90, 10]$ m. Additionally, each UE generates a single task in each TS. The rest of the parameters can be found in Table 6.3.

TABLE 6.3: Simulation parameters

Notation	Description
N	50
T	20
l^{max}	100 m
$D_{n,t}$	$[10, 14]$ Kb
$F_{n,t}$	$[1800, 2000]$ cycles/bit
T^{max}	1 s
d^{max}	20 m
R^{max}	20 m
R^{u}	1 m
H	50 m
B	10 MHz
P_n	0.1 Watt
σ^2	-90 dBm
k_n	10^{-28}
v_n	3
g_0	1.42×10^{-4}
γ	0.95
K	256
τ	0.01
ε	0.001
B_m	10^5
e^{max}	3000
p_m	10

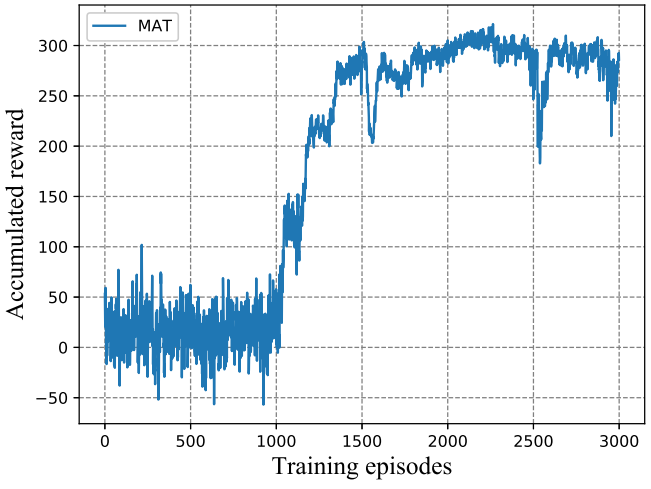


FIGURE 6.3: Accumulated reward versus training episodes (with 3 UAVs).

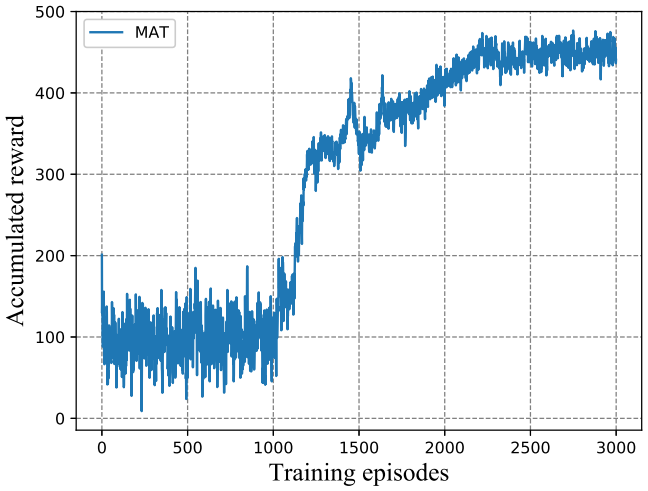


FIGURE 6.4: Accumulated reward versus training episodes (with 4 UAVs).

Firstly, we depict the training curve of MAT in Fig. 6.3, where 3 UAVs are deployed. Observe from Fig. 6.3 that the accumulated reward achieved by MAT remains under 50 at the beginning and starts increasing from the 1000-th episode. After about 2000 training episodes, the curve reaches about 300 and then convergence is achieved.

Then, we increase the number of UAV to 4 and in Fig. 6.4, we depict the accumulated reward achieved by MAT during the training process. Similarly, the curve remains below 200 at the beginning and then increases after the 1000-th episode. It finally saturates around 450. Observe that the accumulated reward seen in Fig. 6.4 is higher than that in Fig. 6.3. This is because deploying more UAVs can serve more UEs at the same time, hence resulting in increased accumulated rewards.

After the training stage, both the model and the network parameters are saved for testing. Next, we compare our algorithm in the cases of 3 and 4 UAVs to the following benchmark solutions:

- RANDOM: In this setup, each UAV randomly selects a flying direction within $\alpha_{m,t} \in [0, 2\pi)$, and a flying distance $d_{m,t} \in [0, d^{max}]$. Note that the UAVs are restricted to the target area.
- CIRCLE: We group all the UEs into a single cluster according to the UEs' coordinates and then all the UAVs fly in a circle twice around the center of the cluster having a radius of R^{max}.

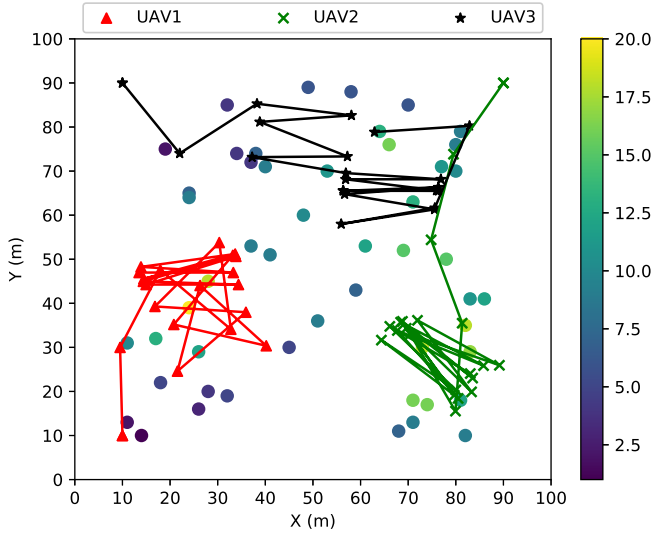


FIGURE 6.5: UAVs' trajectories (with 3 UAVs and the locations of UEs are represented by dots.)

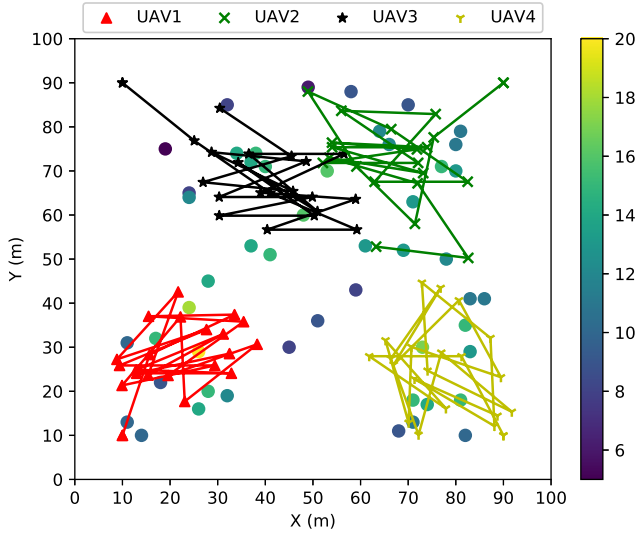


FIGURE 6.6: UAVs' trajectories (with 4 UAVs and the locations of UEs are represented by dots.)

Note that the MAT, RANDOM, and CIRCLE benchmarks have the same starting points for the UAVs and their offloading decisions are described in Eq. (6.27).

We first depict the UAV trajectories in Fig. 6.5, where 3 UAVs are deployed. In this figure, dots represent the location of UEs. We apply a heat map to show the number of times that each UE is served by the UAV commencing from the initial TS to the final TS. The darker the dots, the less amount of time that the UE is spent by the UAV serving. Observe from this figure that all the UAVs move around certain areas,

since their coverage range is limited and they have to move for the sake of serving more UEs to increase the fairness index. Additionally, we can see that each UAV covers the particular area in a cooperative manner, so as to maximize the reward defined. For instance, 'UAV2' moves to the lower right corner from its initial location for serving more UEs, while 'UAV3' moves to the upper right corner to help users in this region.

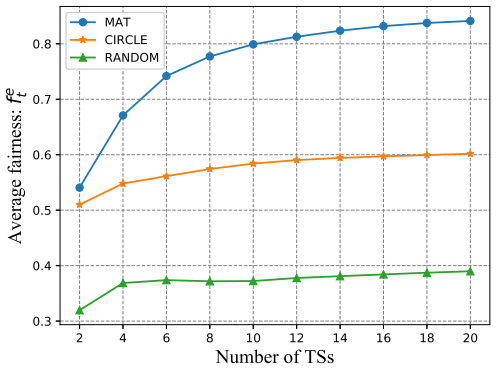
Then, we increase the number of UAVs to 4 and depict the trajectories in Fig. 6.6. Observe that more UAVs result in better coverage. Again, the UAVs cooperate for serving more UEs within the required number of TSs. Furthermore, compared to the heat map shown in Fig. 6.5, 4 UAVs can serve each UE more times than 3. More specially, 4 UAVs can increase the minimum number of serving occurrences from about 2.5 TSs in Fig. 6.5 to about 6 TSs in Fig. 6.6.

In Fig. 6.7, we show the fairness attained by 3 UAVs while serving all UEs, the fairness of each UAV's UE-load and the overall energy consumption of all the UEs. Observe from Fig. 6.7(a) that the average fairness f_t^e among all the served UEs achieved by the MAT, CIRCLE and RANDOM regimes increases with the increase of the number of TSs, as expected. Specifically, MAT increases from 0.53 to 0.85, while CIRCLE increases from about 0.5 to 0.6. Finally, RANDOM remains under 0.4.

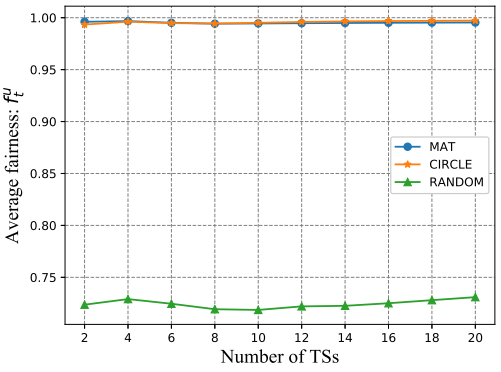
Then, we show the fairness f_t^u of each UAV's UE-load achieved by the MAT, CIRCLE and RANDOM regimes in Fig. 6.7(b). Observe that both MAT and CIRCLE approach the fairness of 1, because both solutions can control the UAVs to serve a similar number of UEs. However, RANDOM can only achieve a fairness of 0.75.

Next, in Fig. 6.7(c), we analyse the energy consumed by UEs. We can see that our proposed MAT achieves the best performance, followed by CIRCLE and RANDOM. This is because after training, MAT assists the UAVs in a cooperative way serving the UEs. Hence, more UEs can offload their tasks to UAVs, which results in reduced energy consumption for all the UEs.

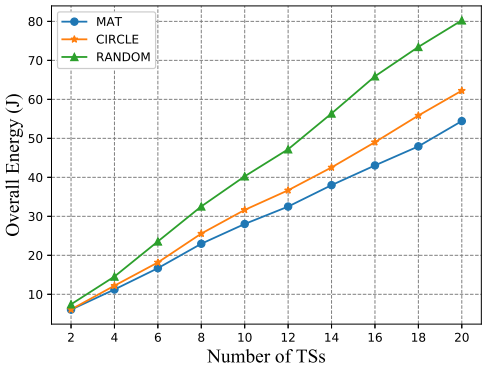
Next, in Fig. 6.8, we increase the number of UAVs to 4 and evaluate the performance of three compared solutions. One can see from Fig. 6.8(a) that the average fairness f_t^e increases with the increase of TSs, as expected. Our proposed MAT can achieve the best performance, reaching at 0.9, whereas the RANDOM performs the worst, which can only achieve about 0.5.



(a) The performance of MAT, CIRCLE and RANDOM versus different number of TSs, in terms of fairness index f_t^e (with 3 UAVs).

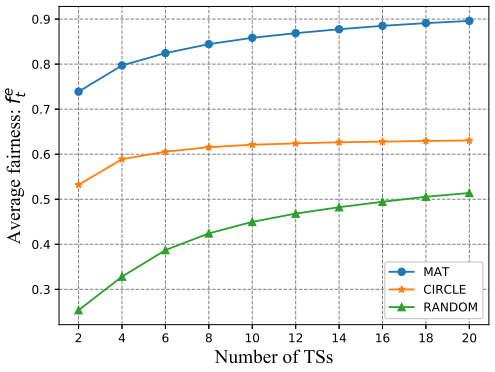


(b) The performance of MAT, CIRCLE and RANDOM versus different number of TSs, in terms of fairness index f_t^u (with 3 UAVs).

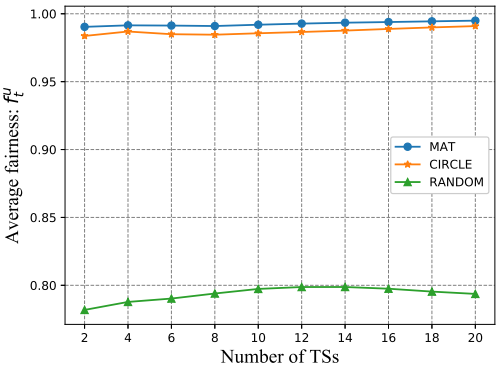


(c) The performance of MAT, CIRCLE and RANDOM versus different number of TSs, in terms of overall energy consumption of all the UEs (with 3 UAVs).

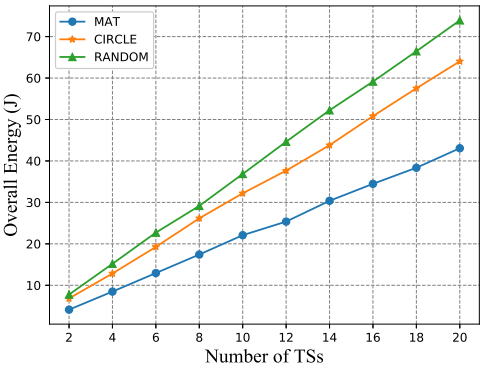
FIGURE 6.7: The performance of MAT, CIRCLE and RANDOM versus different number of TSs, in terms of (a) fairness index f_t^e, (b) fairness index f_t^u and (c) overall energy consumption of all the UEs (with 3 UAVs).



(a) The performance of MAT, CIRCLE and RANDOM versus different number of TSs, in terms of fairness index f_t^e (with 4 UAVs).



(b) The performance of MAT, CIRCLE and RANDOM versus different number of TSs, in terms of fairness index f_t^u (with 4 UAVs).



(c) The performance of MAT, CIRCLE and RANDOM versus different number of TSs, in terms of overall energy consumption of all the UEs (with 4 UAVs).

FIGURE 6.8: The performance of MAT, CIRCLE and RANDOM versus different number of TSs, in terms of (a) fairness index f_t^e, (b) fairness index f_t^u and (c) overall energy consumption of all the UEs (with 4 UAVs).

Then, in Fig. 6.8(b), we draw the fairness of each UAV's UE-load f_t^u achieved by MAT, CIRCLE and RANDOM. One sees that MAT outperforms CIRCLE and RANDOM, as expected. CIRCLE performs worse than MAT but has much better performance than RANDOM.

Additionally, we show the performance of energy consumed by UEs in Fig. 6.8(c). Similar with before, one can observe that MAT can always achieve the best performance and help UEs to save the energy consumption, while CIRCLE performs the second, followed by RANDOM. This further proves that with proper training, MAT can control the UAVs to provide better service to UEs.

6.5 Summary

In this chapter, we have proposed a multi-agent deep reinforcement learning based trajectory control algorithm for jointly maximizing the fairness among all the UEs and the fairness of UE-load of each UAV, as well as minimizing the energy consumption of all the UEs by optimizing each UAV' trajectory and offloading decision from all the UEs. Simulation results show that the proposed MAT has the considerable performance gain over the compared benchmark algorithms.

Chapter 7

DQN based Discrete Trajectory Design for UAV-Aided Emergency Communications

7.1 Introduction

Unmanned aerial vehicles (UAVs), also known as drones, have been playing an increasingly important role in emergency situations such as earthquake and large fires, where UAVs could be deployed to provide emergency communications for user equipments (UEs) and support life saving activities. It also has the potential to provide other wireless communication related services, such as ubiquitous coverage, relaying, information dissemination, mobile edge computing (MEC) and data collection [109–111]. Considering their low cost, high mobility, fast deployment and the direct Line-of-Sight (LoS) connectivity, UAV-enabled wireless communications are expected to achieve higher throughput compared to traditional terrestrial wireless communications.

In order to fully exploit the potential of UAVs, much research has been conducted in the trajectory design of UAV-enabled communications [50, 65, 112]. In [88], Zeng *et al.* maximized the throughput of UAV-enabled mobile relaying system, whereas in [113], the authors maximized the energy efficiency in a point-to-point UAV-ground communication system. In [39], the authors optimized the altitude of UAV to maximize the radio coverage on the ground. In [65], the UAV was utilized as a mobile base station to

serve the ground UEs, and the authors proposed a successive convex approximation (SCA) based algorithm to maximize the minimum average throughput of UEs. In [114], Lyu *et al.* proposed a new cyclical multiple access scheme, where UAV flies cyclically to serve the ground users. In [37], an UAV-enabled secure transmission scheme was proposed in hyper dense networks. For UAV-enabled wireless power transfer networks, Xu *et al.* optimized the trajectory of UAV for the purpose of maximizing the sum of energy received by users. For multi UAV-enabled multiuser system, Yang *et al.* [92] minimized the sum power of user equipment via jointly optimizing the user association, power control, computation capacity allocation, and location planning in a mobile edge computing (MEC) network.

Recently, UAV has been playing an increasingly important role in emergency communications. For instance, during the earthquake, if the local ground station is destroyed, UAV could be deployed to serve as the flying base station to serve the users. They can dynamically move towards the UEs that are out of the communication range, and transmit/receive the data to/from them. In [38], Mozaffari *et al.* addressed some key challenges of deploying UAVs to serve the ground users, such as the optimal deployment and energy efficiency of UAVs. In [115], multiple UAVs were deployed to receive the information from ground UEs, and in order to achieve the reliable uplink communications, the authors proposed to optimize the UAV trajectory and the transmit power of UEs. In [116], Huang *et al.* proposed a differential evolution algorithm to minimize the energy consumption via optimizing the UAV's deployment, such as the number and location of stop points.

Among the recent development in the field of artificial intelligence (AI) and machine learning (ML), reinforcement learning (RL) [117] has become a hot topic both in academia and industry. In [75], Watkins *et al.* introduced a model-free reinforcement learning: Q-learning, which can be viewed as a method of asynchronous dynamic programming (DP). Also, some fundamental elements like agent, state, action, penalty, reward and Q-value were discussed. However, Q-learning is not practical for complicated applications since the number of states and actions will increase exponentially. Thus, combining deep neural networks (DNNs) with RL creates a feasible approach, which could provide more accurate convergence and approximation. In [29], Mnih *et al.* developed a novel solution, i.e., a deep Q-network (DQN), which has achieved an outstanding performance in the challenging domain of Atari 2600 games.

Against the above background, in this chapter, we propose a joint UAV trajectory and power control optimization problem to maximize the number of served UEs, the fairness and the overall uplink data rate of UEs in the emergency communication scenario. To this end, we address the UAV trajectory problem by applying DQN framework. Then, based on the given UAV trajectory, we solve the power control problem via using the convex optimization based algorithm.

The rest of this chapter is organized as follows. Section 7.2 introduces the system model. In Section 7.3, we introduce the proposed algorithm. In Section 7.4, numerical results are presented to verify the proposed algorithm. Finally, we summary the chapter in Section 7.5.

The main notations used in this chapter are summarized in Table 7.1.

TABLE 7.1: Main Notations.

Notation	**Definition**
$n, N, \mathcal{N}$	the index, the number, and the set of UEs,
$t, T, \mathcal{T}$	the index, the number, and the set of TSs
l^{max}	the side length of the square area
Z^{min}, Z^{max}	minimal and maximal height of the UAV
e^{max}	the maximum energy level of UAV
e_t	the remaining energy level of UAV in TS t
$\alpha_t, \beta_t, \omega_t$	the flying action of UAV in TS t
X_t, Y_t, Z_t	the coordinate of UAV in TS t
x_n, y_n	the coordinate of UE n
$d_{n,t}$	distance between UE n with UAV in TS t
$c_{n,t}$	coverage status of UE n in TS t
$L(\theta_{n,t}, d_{n,t})$	path loss between UE n and UAV is TS t
$\gamma_{n,t}$	SINR at UAV from UE n in TS t
$r_{n,t}$	uplink data rate from UE n to UAV in TS t

7.2 System Model

As shown in Fig. 7.1, we consider the emergence situation, where the ground base station is destroyed and the UAV is deployed to provide communication to all the UEs. Assume the UAV flies over a square area with the side length l^{max}. We assume there are N UEs randomly distributed in the target area, and the set of UEs is denoted as $\mathcal{N} \triangleq \{n = 1, 2, ..., N\}$. Also assume the uplink data transmission lasts for T time slots

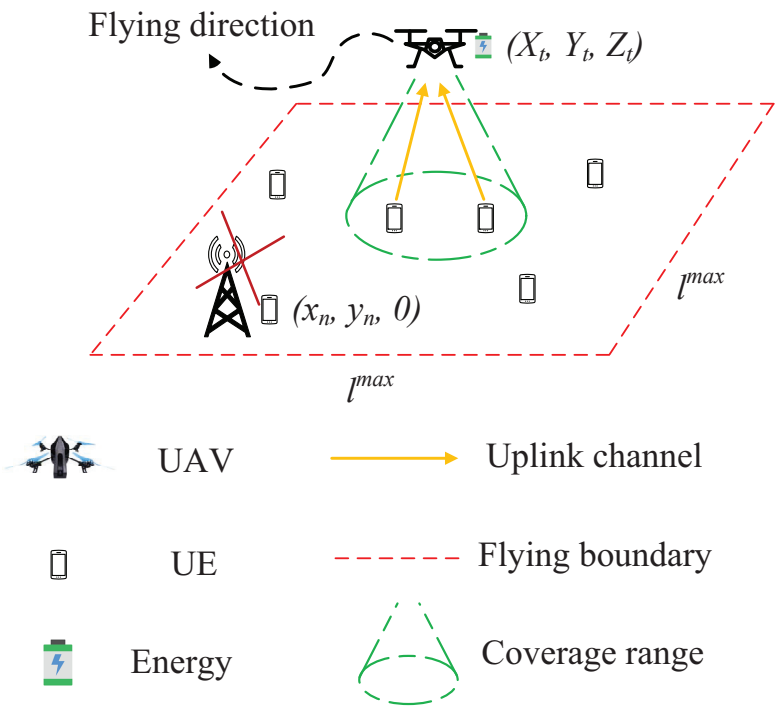


FIGURE 7.1: UAV-Aided Emergency Communication System

(TSs), and the set of TSs is denoted as $\mathcal{T} \triangleq \{t = 1, 2, ...T\}$. In each TS, the UAV has a flying action $[\alpha_t, \beta_t, \omega_t]$ to conduct, where α_t is the horizontal angle of the flying direction, β_t is the vertical angle of the flying direction, and ω_t is the flying distance. For simplicity, in this chapter, we assume that the possible action A_t is chosen from the following set:

$$A_t = \{[\alpha_t, \beta_t, \omega_t] = [\frac{2\pi}{N_\alpha} i, \frac{\pi}{N_\beta} j, \frac{\omega^{max}}{N_\omega} k], \ \forall i \in 0, ...N_\alpha, \ j \in 0, ...N_\beta, \ k \in 0, ...N_\omega\}, t \in \mathcal{T}, \tag{7.1}$$

where N_α, N_β, and N_ω are the numbers of flying angles and distance that UAV can move in each TS. This means that the UAV can only fly with some specific angles and distance values. ω^{max} is the maximal flying distance in each TS. Note that if the UAV stays at the current location, the action $[\alpha_t, \beta_t, \omega_t] = [0, 0, 0]$, where one can see $i = 0$, $j = 0$, $k = 0$. Otherwise, it moves with the corresponding angles $\frac{2\pi}{N_\alpha} i$, $\frac{\pi}{N_\beta} j$ and the distance $\frac{\omega^{max}}{N_\omega} k$. Hence, the coordinate of UAV in TS t can be denoted as $[X_t, Y_t, Z_t]$, where $X_t = X_0 + \sum_{t'=1}^{t} \omega_{t'} \sin(\beta_{t'}) \cos(\alpha_{t'})$, $Y_t = Y_0 + \sum_{t'=1}^{t} \omega_{t'} \sin(\beta_{t'}) \sin(\alpha_{t'})$, and $Z_t = Z_0 + \sum_{t'=1}^{t} \omega_{t'} \cos(\beta_{t'})$, with $[X_0, Y_0, Z_0]$ being the initial coordinate of the UAV.

Since the UAV can not fly out of the target area, we have

$$0 \leq X_t \leq l^{max}, \ \forall t \in \mathcal{T}, \tag{7.2}$$

and

$$0 \leq Y_t \leq l^{max}, \ \forall t \in \mathcal{T}. \tag{7.3}$$

Additionally, in this chapter, we set

$$Z^{min} \leq Z_t \leq Z^{max}, \ \forall t \in \mathcal{T}, \tag{7.4}$$

where Z^{min}, Z^{max}, are the minimal and maximal flying height of the UAV, for collision avoidance.

Thus, the distance between the UAV and UE n in TS t can be given by

$$d_{n,t} = \sqrt{(X_t - x_n)^2 + (Y_t - y_n)^2 + Z_t^2}, \ \forall n \in \mathcal{N}, t \in \mathcal{T}, \tag{7.5}$$

where $[x_n, y_n]$ is the coordinate of UE n.

Furthermore, in this chapter, the UAV has a azimuth angle value of antenna θ', which is based on 3-D Cartesian coordinate, such as x axis, y axis, z axis. Hence, in TS t, the UAV has a maximal horizontal coverage circle with the radius of $R_t^{max} = Z_t \tan(\theta')$ [92] and it varies with the height of the UAV. We also assume that the UAV has the energy constraint e^{max}. We define the remaining energy level e_t of the UAV in TS t as:

$$e_t = e^{max} - \sum_{t'=0}^{t} \nabla e_{t'}. \ \forall t \in \mathcal{T}, \tag{7.6}$$

where $\nabla e_{t'}$ is the energy consumed by UAV in TS t', which is defined as

$$\nabla e_{t'} = \left(P_0 \left(1 + 3\frac{v_t^2}{V_r^2} \right) + P_1 \left(\sqrt{1 + \frac{v_t^4}{4V_0^4}} - \frac{v_t^2}{2V_0^2} \right)^{\frac{1}{2}} + \frac{1}{2} d_0 \rho s \pi R_b^2 v_t^3 \right) T^{max}, \tag{7.7}$$

where v_t is the flying velocity of UAV in TS t, T^{max} is the maximal time duration of each TS, V_r is the tip speed of the rotor blade, V_0 is the mean rotor velocity when

hovering, d_0 is the drag ratio, ρ means the air density, s denotes the rotor solidity, R_b is the radius value of rotor disc. And P_0, P_1 are constant values that can be found in [79]. For simplicity, in this chapter, we set $v_t = \frac{\omega_t}{T^{max}}$. Note that we do not consider the energy consumption of data receiving/transmission since it is negligible compared with the moving and hovering energy consumption. Also, to simply the model, we adopt the simplified energy consumption model above, which could be readily extended to the more general model considering different types of UAVs. In practice, we also assume there is some preserved battery for UAV flying back to the ground, which is ignored here to make the model compact.

In this chapter, the 3-D channel model proposed in [39] is adopted. Thus, the mean path loss between the UAV and the UE n in TS t is given by

$$L(\theta_{n,t}, d_{n,t}) = \frac{\eta_{\text{LoS}} - \eta_{\text{NLoS}}}{1 + a\text{exp}\big(-b(\theta_{n,t} - a)\big)} + 20\log_{10}(d_{n,t}) + 20\log_{10}\big(\frac{4\pi f_c}{c}\big) + \eta_{\text{NLoS}}, \quad (7.8)$$

where η_{LoS} and η_{NLoS} (in dB) are the path loss corresponding to the LoS and non-LoS links respectively. a and b are positive constants which can be obtained in [39]. f_c is the carrier frequency (Hz), c is the light speed (m/s), and $\theta_{n,t} = \arctan\big(\frac{Z_t}{\sqrt{(X_t - x_n)^2 + (Y_t - y_n)^2}}\big)$.

We denote $c_{n,t}$ as the coverage status of UE n in TS t, and it can be defined as

$$[l] c_{n,t} = \begin{cases} 1, & \text{if } \sqrt{(X_t - x_n)^2 + (Y_t - y_n)^2} \leq R_t^{max}, \\ 0, & \text{Otherwise.} \end{cases} \quad (7.9)$$

Additionally, we assume that if the UE n is under the coverage of UAV in TS t, i.e., $c_{n,t} = 1$, the UE n is served by UAV and the data collection from UE n to UAV is started. Thus, the corresponding signal-to-interference-plus-noise ratio (SINR) at the UAV can be expressed as

$$\gamma_{n,t} = \frac{c_{n,t} P_{n,t} 10^{-\frac{L(\theta_{n,t}, d_{n,t})}{10}}}{\sum_{n'=1, n' \neq n}^{N} c_{n',t} P_{n',t} 10^{-\frac{L(\theta_{n',t}, d_{n',t})}{10}} + \sigma^2}, \quad (7.10)$$

where $P_{n,t}$ means the transmit power of UE n in TS t; σ^2 is the additive white Gaussian noise (AWGN) at the receiver. Therefore, the uplink data rate from UE n to the UAV

in TS t is expressed as

$$r_{n,t} = \log_2\big(1 + \gamma_{n,t}\big),\ \forall n \in \mathcal{N}, t \in \mathcal{T}. \tag{7.11}$$

One can also apply the power constraint as follows, then we have

$$0 \leq P_{n,t} \leq P^{max},\ \forall n \in \mathcal{N}, t \in \mathcal{T}, \tag{7.12}$$

where P^{max} is the maximum transmit power of UEs.

In this chapter, we also aim to maximize the number of UEs served by UAV via optimizing the UAV trajectory. Then we define C_t as follows

$$C_t = \frac{1}{N}\sum_{n=1}^{N} c_{n,t}, \forall t \in \mathcal{T}, \tag{7.13}$$

which can represent the proportion of the number of UEs served by UAV in TS t. However, this may lead to unfair serving process since some UEs are covered for many TSs and the rest UEs may be never covered at all. Therefore, similar to [118, 119], we apply the fairness index among all UEs, which is defined as

$$f_t = \frac{\big(\sum_{n=1}^{N}\sum_{t'=1}^{t} c_{n,t'}\big)^2}{N\sum_{n=1}^{N}\big(\sum_{t'=1}^{t} c_{n,t'}\big)^2}, \tag{7.14}$$

where f_t reflects the quality of service (QoS) level that the UEs served by UAV from the initial TS to the TS t. More precisely, if all the UEs are served for the similar number of TSs, the fairness value f_t is closer to 1.

Additionally, we define the overall data rate of UEs served by UAV in TS t as

$$R_t = \sum_{n=1}^{N} c_{n,t} r_{n,t}, \forall t \in \mathcal{T}. \tag{7.15}$$

Thus, we formulate the optimization problem as follows

$$\mathcal{P}1: \max_{\boldsymbol{U},\boldsymbol{P}} \sum_{t=1}^{T} \big(f_t \cdot C_t \cdot R_t\big), \tag{7.16a}$$

subject to:

$$A_t = \{[\alpha_t, \beta_t, \omega_t] = \big[\frac{2\pi}{N_\alpha} i, \frac{\pi}{N_\beta} j, \frac{\omega^{max}}{N_\omega} k\big], \\ \forall i \in 0, ...N_\alpha,\ j \in 0, ...N_\beta,\ k \in 0, ...N_\omega\}, t \in \mathcal{T}, \tag{7.16b}$$

$$0 \leq X_t \leq l^{max},\ \forall t \in \mathcal{T}, \tag{7.16c}$$

$$0 \leq Y_t \leq l^{max},\ \forall t \in \mathcal{T}, \tag{7.16d}$$

$$Z^{min} \leq Z_t \leq Z^{max},\ \forall t \in \mathcal{T}, \tag{7.16e}$$

$$0 \leq P_{n,t} \leq P^{max},\ \forall n \in \mathcal{N}, t \in \mathcal{T}, \tag{7.16f}$$

where $\boldsymbol{U} = \{X_t, Y_t, Z_t, \forall t \in \mathcal{T}\}$ and $\boldsymbol{P} = \{P_{n,t}, \forall n \in \mathcal{N}, t \in \mathcal{T}\}$. It is readily to see that the above problem cannot be solved by traditional optimization approach as it involves discrete variables $\boldsymbol{U}$ and continuous variables $\boldsymbol{P}$. Additionally, all three factors cannot be achieved optimally at the same time since each factor will have a negative effect on others. Thus, we aim to achieve the optimal balance between them. Then, in this chapter, we first propose a DQN-based algorithm to solve the UAV trajectory problem. Next, based on the optimized UAV trajectory, we further propose a successive convex approximation (SCA) based algorithm to solve the power control problem.

7.3 Proposed Algorithm

Before presenting the proposed algorithm, we first introduce some important knowledge of deep reinforcement learning.

7.3.1 Background Knowledge

In the traditional reinforcement learning structure, there is an agent interacting with the environment through a series of states, actions and rewards. In each time step, the agent selects the policy that maps the state and action with the aim of maximizing the accumulated reward. Specifically, the process of interacting with the environment can

be expressed with an action-value function named Q-function, which is defined as

$$Q(s,a) = \max_{\pi} \mathbb{E}\big[Z|s_t = s, a_t = a\big], \tag{7.17}$$

where Q is known as Q-value, π denotes the policy by taking the action a at the state s and Z is the reward.

Although DRL combines DNN with Q-learning, it may still have instability or divergence. Since DNN may be seen as the non-linear function approximator, small updates to Q-value may significantly vary the policy, or even change the data distribution as well as the correlations between action-value and target value. Therefore, to address this issue, in [29], Mnih *et al.* introduced the DQN framework, which contains a pair of mechanisms: Firstly, they applied the experience replay, where the mini-batch randomly samples several transitions $\{s_t, a_t, z_t, s_{t+1}\}$ to train the DQN. This mechanism removes the correlation of state sequences and smooths over changes in the data distribution. Secondly, an iterative updating mechanism was deployed. Specifically, there is a target network periodically updating for the purpose of adjusting the action-value towards the target value.

7.3.2 The Proposed DQN based UAV trajectory design Algorithm

In this section, the proposed DQN algorithm is presented, where we assume there is an agent interacting with the environment. The agent controls the UAV and aims to select the optimal policy that can maximize the accumulated reward $Z_t = \sum_{t'=t}^{T} \gamma^{t'-t} z_{t'}$ by giving a set of states $\mathcal{S} \triangleq \{s_t = s_1, s_2, ...s_T\}$ and actions $\mathcal{A} \triangleq \{a_t = a_1, a_2, ..., a_T\}$, where $\gamma \in [0, 1]$ is the discount factor. More specifically, we describe the state, action and reward in TS t as follows:

1. State s_t: the state of agent in TS t has two components.
   (a) UAV 's current coordinate: $\{X_t, Y_t, Z_t\}$.
   (b) UAV's current energy level: $\{e_t\}$.
2. Action a_t: we define action $a_t = \{\alpha_t, \beta_t, \omega_t\}$ as the UAV's horizontal angle α_t, vertical angle β_t and distance ω_t in TS t, where $a_t \in A_t$.
3. Reward Function z_t: we define the reward function as:

$$z_t = f_t \cdot C_t \cdot R_t - p, \tag{7.18}$$

where p is the penalty if UAV flies out of the target area and R_t can be obtained by the proposed convex optimization based solutions in Algorithm 7.

In the proposed DRL shown in Fig. 7.2, there are two DQN networks, namely evaluation and target networks, respectively [29]. Note that the evaluation and target networks have the same structure but the latter updates periodically. The agent selects the action according to the evaluation network and the agent follows an ϵ-greedy policy.

According to the state s_t and action a_t, the agent obtains the reward r_t and then the environment transfers to the next state s_{t+1}. The transition $\{s_t, a_t, z_t, s_{t+1}\}$ can be stored in the experience replay memory with size M^{max}. Once the learning process starts, the mini-batch randomly samples K transitions from the memory. The evaluation network is trained by the sequence of the loss function, which can be expressed as

$$L_i(\theta_i) = \mathbb{E}_{s,a}\big[\big(y_i - Q(s,a|\theta_i)\big)^2\big], \tag{7.19}$$

where i is the index of iteration, $y_i = \mathbb{E}\big[z + \gamma \max_{a'} Q(s', a'|\theta_{i-1})\big]$ and it can be obtained by the target network.

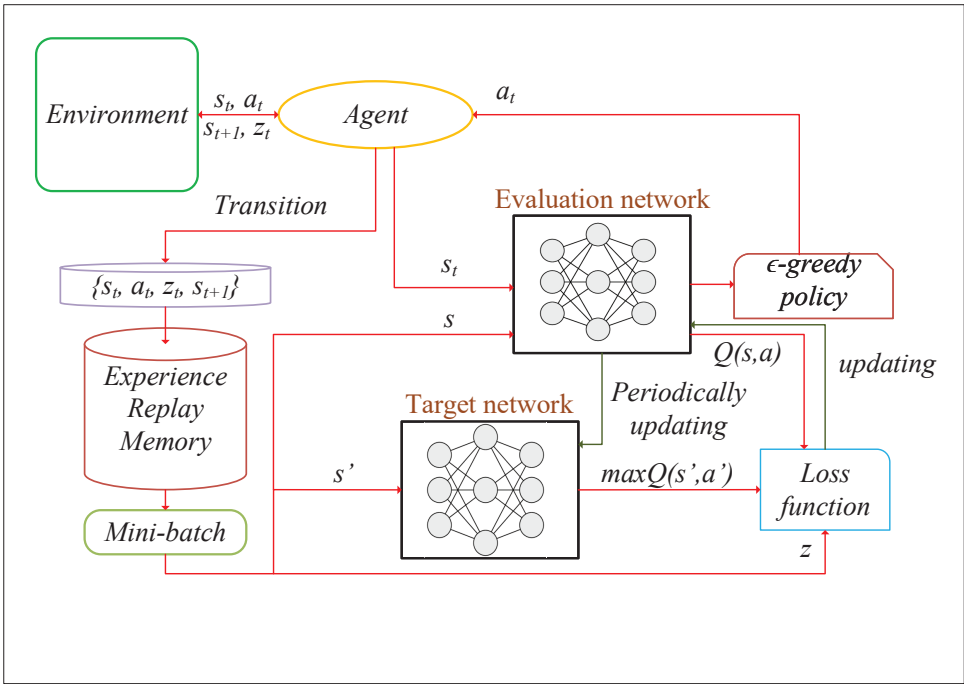


FIGURE 7.2: Structure of proposed DQN

Algorithm 6 The proposed DQN algorithm

1: Initialize evaluation network, target network with parameters θ;
2: Initialize experience replay memory with size M^{max};
3: **for** Episode = 1,2,...,E^{max} **do**
4: Initialize state $s_t = [X_0, Y_0, Z_0, e^{max}]$;
5: **for** TS = 1,2,...T **do**
6: Obtain s_t;
7: ϵ_t = rand(0,1);
8: **if** $\epsilon_t \leq \epsilon$ **then**
9: $a_t = \text{argmax}Q(s_t, a_t)$;
10: **else**
11: Select a random action a_t from A_t;
12: **end if**
13: Execute a_t;
14: Obtain z_t according to Algorithm.7;
15: Obtain s_{t+1};
16: Store transition $\{s_t, a_t, z_t, s_{t+1}\}$ into experience replay memory;
17: **if** the learning process starts **then**
18: Randomly sample K transitions from memory;
19: Obtain loss value according to (7.19);
20: Perform a gradient descent step on loss value with respect to the network parameters θ;
21: Update evaluation network;
22: Update target network periodically;
23: **end if**
24: **end for**
25: **end for**

During the interaction with the environment, the agent selects the optimized action of UAV associated with the evaluation network, which follows a ϵ-greedy policy. Specifically, the agent can select the action that has the largest Q-value with probability ϵ, or randomly select the action from the action set A_t with probability $1-\epsilon$. Also, the agent obtains state s_t, next state s_{t+1} and reward z_t from the environment. Note that the data rate R_t of reward z_t is calculated by the proposed convex optimization based algorithm provided in Algorithm 7. Then, the transition, which consists of $\{s_t, a_t, z_t, s_{t+1}\}$, is stored in the experience replay memory. Once the learning procedure starts, the mini-batch randomly samples K transitions from the experience replay memory. Given the Q-value $Q(s, a)$ and the target value y_i obtained by the evaluation and target network, the loss function provided by (7.19) is used to update the evaluation network and the target network is updated periodically.

Furthermore, we provide the pseudo code of proposed DQN algorithm in Algorithm 6. Specifically, from Line 1 to 2, we initialize the evaluation network, target network and

experience replay memory. Then, in each training episode, we initialize the state s_t, and the evaluation network generates the action by the given state s_t. Note that a ϵ-greedy policy is employed to select the optimized action. Specifically, a variable $\epsilon_t \in [0, 1]$ is generated. If $\epsilon_t \leq \epsilon$, we select the action a_t that has the largest Q-value. Otherwise, we select a random action a_t. Next, the agent executes the action a_t, obtains the reward z_t provided by (7.18) and the environment transfers to the next state s_{t+1}. Note that the UAV stays at the current location and the agent receives a penalty if the UAV flies out of the target area. The transition is stored in the experience replay memory. From line 17, once the learning process starts, the learning procedure starts. The mini-batch randomly samples K transitions from the memory for calculating the loss value. Then, we perform a gradient descent step on loss value calculated by loss function with respect to the network parameters θ. Finally, we update evaluation network and target network periodically.

7.3.3 Power Control Algorithm

In order to maximize the reward z_t with the given trajectory, we further propose a convex optimization-based algorithm for handling the power control of all UEs. Then, in TS t, given the UAV trajectory, the maximization problem of reward function (7.18) is transformed into the following problem:

$$\max_{\boldsymbol{P}} \ f_t \cdot C_t \cdot R_t - p, \tag{7.20a}$$

subject to:

$$0 \leq P_{n,t} \leq P^{max}, \ \forall n \in \mathcal{N}, \tag{7.20b}$$

from which, both f_t, C_t and p are fixed. Motivated by [43], via introducing the auxiliary variable η, the problem is transformed into

$$\max_{\eta, \boldsymbol{P}} \ \eta, \tag{7.21a}$$

subject to:

$$f_t \cdot C_t \cdot \sum_{n=1}^{N} c_{n,t} r_{n,t} - p \geq \eta, \ \forall n \in \mathcal{N}, \tag{7.21b}$$

$$0 \leq P_{n,t} \leq P^{max}, \ \forall n \in \mathcal{N}. \tag{7.21c}$$

Problem (7.21) is a non-convex optimization since (7.21b) is a non-convex constraint. It is noted that $r_{n,t}$ can be expressed as

$$\begin{aligned} r_{n,t} &= \log_2\left(1+\frac{c_{n,t}P_{n,t}10^{-\frac{L(\theta_{n,t},d_{n,t})}{10}}}{\sum_{n'=1,n'\neq n}^{N} c_{n',t}P_{n',t}10^{-\frac{L(\theta_{n',t},d_{n',t})}{10}}+\sigma^2}\right) \\ &= \log_2\left(\sum_{n=1}^{N} c_{n,t}P_{n,t}10^{-\frac{L(\theta_{n,t},d_{n,t})}{10}}+\sigma^2\right)-\widetilde{r}_{n,t},\ \forall n\in\mathcal{N}, \end{aligned} \tag{7.22}$$

where

$$\widetilde{r}_{n,t} = \log_2\Big(\sum_{n'=1,n'\neq n}^{N} c_{n',t}P_{n',t}10^{-\frac{L(\theta_{n',t},d_{n',t})}{10}}+\sigma^2\Big),\ \forall n\in\mathcal{N}. \tag{7.23}$$

In order to solve the above non-convex constraint of (7.21b), we apply the successive convex approximation (SCA) to calculate the value of $\widetilde{r}_{n,t}$. Specifically, we define $\boldsymbol{P}^k = \{P_{n,t}^k, \forall n\in\mathcal{N}\}$ as the given transmission power of UEs in TS t in the k-th iteration. Inspired by [120], any concave function can be globally upper-bounded by its first-order Taylor expansion at any point. Hence, by given $\boldsymbol{P}^k$, one has

$$\begin{aligned} \widetilde{r}_{n,t} &= \log_2\left(\sum_{n'=1,n'\neq n}^{N} c_{n',t}P_{n',t}10^{-\frac{L(\theta_{n',t},d_{n',t})}{10}}+\sigma^2\right) \\ &\leq \sum_{n'=1,n'\neq n}^{N} \frac{c_{n',t}10^{-\frac{L(\theta_{n',t},d_{n',t})}{10}}\log_2(e)}{\sum_{l=1,l\neq n}^{N} c_{l,t}P_{n',t}^k 10^{-\frac{L(\theta_{l,t},d_{l,t})}{10}}+\sigma^2}(P_{n',t}-P_{n',t}^k) \\ &\quad+\log_2\left(\sum_{n'=1,n'\neq n}^{N} c_{n',t}P_{n',t}^k 10^{-\frac{L(\theta_{n',t},d_{n',t})}{10}}+\sigma^2\right)\triangleq\widetilde{r}_{n,t}^{up}. \end{aligned} \tag{7.24}$$

With any given local point $\boldsymbol{P}^k$ and the upper bound $\widetilde{r}_{n,t}^{up}$, Problem (7.21) can be transformed into

$$\max_{\boldsymbol{\eta}^k, \boldsymbol{P}} \eta^k, \tag{7.25a}$$

subject to:

$$f_t \cdot C_t \cdot \sum_{n=1}^{N} c_{n,t} \left(\log_2 \left(\sum_{n=1}^{N} c_{n,t} P_{n,t} 10^{-\frac{L(\theta_{n,t}, d_{n,t})}{10}} + \sigma^2 \right) - \widetilde{r}_{n,t}^{up} \right)$$

$$\geq \eta^k, \ \forall n \in \mathcal{N}, \tag{7.25b}$$

$$0 \leq P_{n,t} \leq P^{max}, \ \forall n \in \mathcal{N}. \tag{7.25c}$$

One can see that the above problem is now been converted to the convex optimization, which can be solved efficiently by the standard convex optimization solver, e.g., CVX [120]. Then, we provide the pseudo code in Algorithm 7.

Algorithm 7 The proposed convex optimization based power control algorithm

1: Obtain a_t according to the DQN network;
2: Execute a_t;
3: Obtain f_t, C_t according to Eq. (7.14) and Eq. (7.9);
4: Initialize $\boldsymbol{P}^0$;
5: $k = 0$;
6: **repeat**
7: Solve Problem (7.25) for given $\boldsymbol{P}^k$;
8: Denote the optimal solution as $\boldsymbol{P}^{k+1}$;
9: $k = k + 1$;
10: **until** The convergence is achieved

As shown in Algorithm 7, we first obtain the state of UAV s_t, execute a_t and obtain f_t and C_t. Then, we initialize $\boldsymbol{P}^0$, and solve Problem (7.25) for given $\boldsymbol{P}^k$. Next, we repeat the process until the convergence is achieved.

7.4 Simulation Result

In this section, we evaluate the performance of proposed DQN and convex optimization based algorithm. The simulation is executed by using Python 3.7, Tensorflow 1.15 [121]. CVXPY 1.0.24 [73] is used in the convex optimization based algorithm. We deploy two fully-connected hidden layer with $[400 \times 300]$ neurons in DQN networks. The learning rate is 0.001 and RMSOptimizer is used to update DQN networks. We set the target

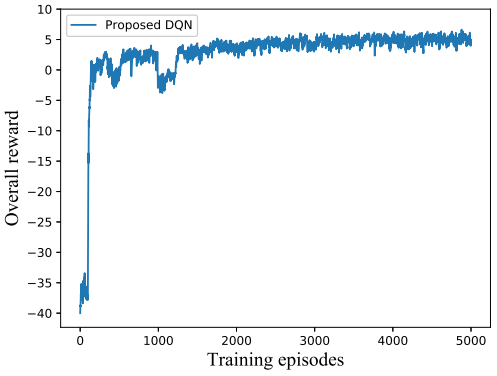


FIGURE 7.3: Overall reward versus training episodes.

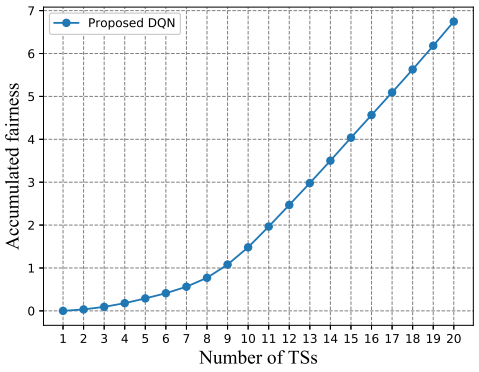
area to be a square with side length $l^{max} = 400$ m and 30 UEs are randomly distributed in the target area. In each training episode, the UAV always starts from the same initial point, i.e., $[X_0, Y_0, Z_0] = [5, 5, 70]$. In each TS, once the UE is covered by UAV, UAV starts data collection from the UE. More parameters can be found in Table. 7.2.

TABLE 7.2: Parameter Setting.

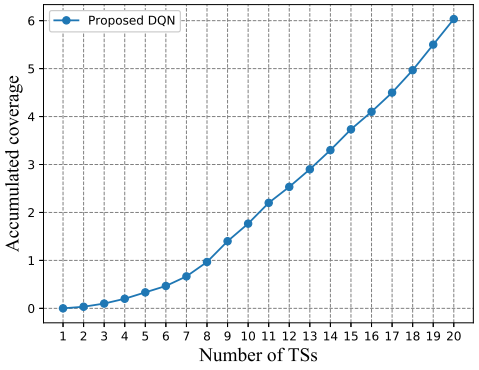
Parameter	**Description**	**Parameter**	**Description**
N	30	ω^{max}	30 m
l^{max}	400 m	N_α	6
N_β	5	N_ω	4
Z_{min}	50 m	Z^{max}	100 m
P^{max}	0.1 W	ϵ	0.9
e^{max}	200 KJ	T^{max}	1 s
θ'	$\frac{\pi}{4}$	P_0	79.85
P_1	88.63	V_r	120
V_0	4.03	d_0	0.6
ρ	1.225	s	0.05
R_b	0.4 m	η_{LoS}	1.6 dB
η_{NLoS}	23 dB	f_c	2.5 GHz
c	3×10^8 m/s	a	12.08
b	0.11	σ^2	-100 dBm
γ	0.99	K	256
M^{max}	10^5	p	2

We first analyze the overall reward achieved by DQN algorithm in each training episode (i.e., 20 TSs) in Fig. 7.3, from which, we observe that the overall reward remains negative at the beginning. This is because the UAV always flies out of the target area, which means the penalty is always incurred. When the learning process starts, the agent learns to optimize the UAV trajectory from the exploration process and the DQNs start converging, which increases the overall reward. Once the convergence is achieved, the

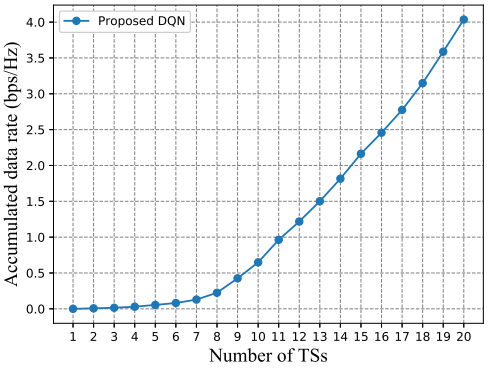
overall reward remains about 5, which shows the best UAV trajectory and transmission power of each UE are obtained.



(a) The accumulated fairness over one episode during testing.



(b) The accumulated coverage over one episode during testing.



(c) The accumulated data rate over one episode during testing.

FIGURE 7.4: The accumulated (a) fairness, (b) coverage and (c) data rate over one episode during testing.

After adequate training, the model and their parameters are saved for testing. We

analyse the performance of the proposed DQN algorithm during the testing procedure in Fig. 7.4. Specifically, we first evaluate the accumulated fairness in different numbers of TSs in Fig. 7.4(a), from which we can observe that the accumulated fairness keeps rising from 0 and stabilizes at 7. In Fig. 7.4(b), one can see that the accumulated coverage increases from 0 to 6 eventually. Then, we evaluate the accumulated data rate (bps/Hz) of UEs served by UAV in Fig. 7.4(c), from which we observe that the data rate keeps rising with the increase of the number of TSs. It reaches about 4 bps/Hz finally. Overall, one can see from Fig. 7.4 that our proposed DQN algorithm can learn from experience and reach the considerable performance.

Then, for comparison, we present the following baseline algorithms:

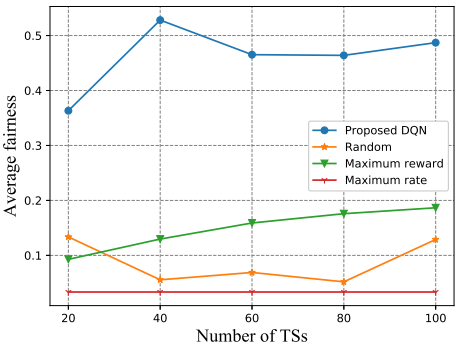
- Random: In each TS, UAV randomly selects a horizontal angle value α_t, a vertical angle value β_t and distance value ω_t from the action set A_t. Additionally, it randomly selects the power control $P_{n,t}$ for each UE. It is worth mentioning that the UAV is restricted to the target area.
- Maximum rate: In each TS, the UAV always selects the action a_t from A_t that can maximize the instantaneous data rate, which is defined as

$$a_t = \max_{a_t} R_t|_{a_t \in A_t}. \tag{7.26}$$

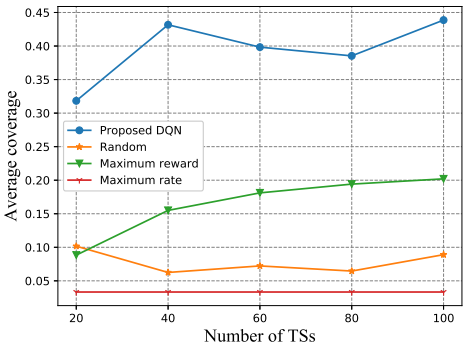
  Note that in this solution, the UEs served by UAV always transmit their data with maximal power consumption as P^{max}.
- Maximum reward: In each TS, the UAV selects the action of UAV a_t that can maximize the reward.

  Similar as before, the maximum transmission power P^{max} is applied for each UE.

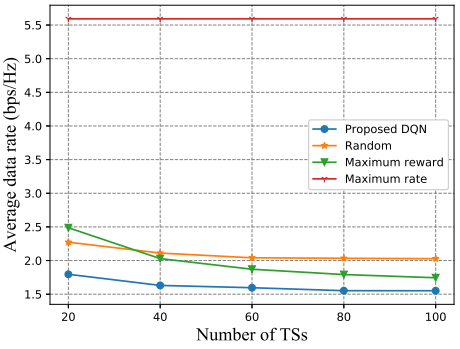
Then, we evaluate the performance of the proposed DQN algorithm and the above baseline solutions in different number of TSs in Fig. 7.5. It is worth mentioning that it is quite challenging to achieve the best solution in all three factors, i.e., fairness, coverage and data rate at the same time. On one hand, the UAV will keep flying for serving different UEs for maximizing the fairness, which will inevitably reduce the data rate and consume more energy of UAV. On the other hand, the UAV will tend to stay at the location that can maximize the data rate, which will have a negative effect on fairness and coverage. Besides, maximizing the number of UEs served by the UAV will lead to severe interference between UEs, which will also reduce the overall data rate.



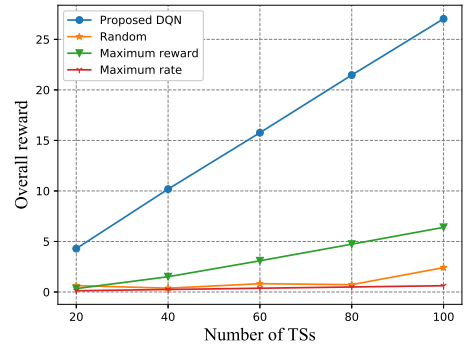
(a) The average fairness overall reward versus different number of TSs which UAV possesses.



(b) The average coverage versus different number of TSs which UAV possesses.



(c) The average data rate overall reward versus different number of TSs which UAV possesses.



(d) The average overall reward versus different number of TSs which UAV possesses.

FIGURE 7.5: The average (a) fairness, (b) coverage, (c) data rate, and (d) overall reward versus different number of TSs which UAV possesses.

However, as our objective is to maximize the overall reward consisting of all the three factors. Our proposed solution can achieve the best performance in this regard and will be shown below.

First, in Fig. 7.5(a), we analyse the impact of the number of TSs on fairness. One can observe that the proposed DQN algorithm outperforms other baselines in all the examined cases. It can always achieve the fairness above 0.35, where as the other three algorithms can only achieve fairness below 0.2.

Then, in Fig. 7.5(b), one can see that in terms of coverage, our proposed DQN-based solution performs the best, which can reach close to 0.45, However, other benchmark algorithms can reach at most around 0.2.

Furthermore, we evaluate the performance in terms of average data rate of UEs served by UAV in Fig. 7.5(c). One observes that the "maximum rate" solution has the best

performance, as it aims at maximizing the data rate of the users, while "random" and "maximum reward" perform slightly better than our proposed DQN. The explanation is that the UAV controlled by "maximum rate" only serves a few UEs, which will lead to lower interference between UEs, however, it cannot guarantee the coverage and fairness, as shown before. Our proposed DQN-based solution, as it will also consider the coverage, and it may serve several UEs at the same time, resulting in lower data rate due to interference among different UEs.

Then, as shown in Fig. 7.5(d), we depict the overall reward achieved by the DQN-based solution and other baselines in a single episode with respect to different number of TSs. One can observe that with the increase of the number of TSs, the overall reward of all algorithms increase. The proposed DQN has the best performance, as expected. Other benchmark algorithms have lower performance, as they only focus on one factor, such as data rate.

7.5 Summary

In this chapter, we have considered the UAV-aided emergency communications, where the UAV is deployed in the case that the ground base station is destroyed. We propose a DRL based DQN algorithm to optimize the UAV trajectory. Additionally, we present a convex optimization based algorithm to optimize the power transmission of UEs served by UAV. Simulation results show that the proposed algorithm can achieve the considerable performance gain over the existing algorithms in terms of fairness, the total number of UEs served and the overall data rate of UEs.

Chapter 8

Conclusions and Future Work

8.1 Summary of Conclusions

The forthcoming 5G and beyond network has to support massive UEs and enable various kinds of emerging applications in terms of connectivity, data rate, latency, etc. In order to meet the technical requirements of 5G, the combination of UAV and MEC has shown huge potential in current wireless communication systems. However, when facing the research challenges in UAV-enabled MEC, traditional approaches are not sufficient enough and they are normally sub-optimal and require plenty of computational resource. In this regard, the main purpose of this thesis is to develop novel AI-based algorithms for solving challenges in UAV-enabled MEC. The main conclusions of this thesis is given as follows:

- First, we have studied a multi-UAV enabled MEC system, in which the UAVs are assumed to fly in circles over the ground UEs to provide the computation services. The proposed problem is formulated as a MINLP, which is hard to deal with in general. We propose a RLAA algorithm to address it effectively. Simulation results show that RLAA can achieve the same performance as the exhaustive search in small scale cases, whereas in large case scenario, RLAA still have considerate performance gain over other traditional approaches.
- Second, we have considered the flying mobile edge computing architecture, by taking advantage of the UAVs to serve as the moving platform. We aim to minimize the

energy consumption of all the UEs by optimizing the UAVs' trajectories, user associations and resource allocation. To tackle the multi-UAVs' trajectories problem, a convex optimization-based CAT has been first proposed. Then, in order to conduct fast decision, a DRL-based RAT including a matching algorithm has also been proposed. Simulation results show that CAT and RAT have considerable performance.

- Third, we have proposed a multi-agent deep reinforcement learning based trajectory control algorithm for jointly maximizing the fairness among all the UEs and the fairness of UE-load of each UAV, as well as minimizing the energy consumption of all the UEs by optimizing each UAV' trajectory and offloading decision from all the UEs. Simulation results show that the proposed MAT has the considerable performance gain over the compared benchmark algorithms.
- Forth, we have considered the UAV-aided emergency communications, where the UAV is deployed in the case that the ground base station is destroyed. We propose a DRL based DQN algorithm to optimize the UAV trajectory. Additionally, we present a convex optimization based algorithm to optimize the power transmission of UEs served by UAV. Simulation results show that the proposed algorithm can achieve the considerable performance gain over the existing algorithms in terms of fairness, the total number of UEs served and the overall data rate of UEs.

8.2 Future Work

The algorithms and approaches proposed in this thesis have addressed some of existing issues in UAV-enabled MEC system. In order to sufficiently broaden the usage of UAV-enabled MEC in the future wireless network, in this section, we overview some of future research challenges as follows:

- **UAV trajectory control for multi objectives**: Current DRL-based solutions for optimizing the UAV trajectory is based on the single objective, which is not sufficient enough for practical scenarios. For some complicated tasks or objectives that have spare rewards, such as UAV can fly back to the initial area, or land at certain area after serving process, the DRL-based solutions can not handle this. In this end, it is essential to develop novel hierarchical reinforcement learning (HRL)-based algorithms [122].

- **Combination of UAV and autonomous vehicles (AVs) in MEC**: With the recent development in AV systems, various kinds of mobile MEC systems are appearing. The MEC server can be deployed on the UAV or other AVs, which means the mobile MEC system is hybrid. Thus, in order to fulfill the potential of MEC system and guarantee the QoS of UEs, the hybrid MEC system should be carefully designed. However, as the mobile MEC servers are deployed with different usages and playing with different roles, it is quite challenging to control them simultaneously. In this end, one future direction is to develop general multi-agent reinforcement learning (MARL) algorithms that can cooperate a group of UAVs, AVs or other autonomous components in mobile MEC systems.
- **Training Data Acquisition**: As discussed above, the AI-based algorithms need to take advantages of training data that is obtained from the real-world. However, the acquisition of real training data is normally expensive and time consuming. In addition, most of data is related to the privacy of consumers, which is well protected by the commercial companies. In this case, how to generate relevant and diverse data that can be used in the training process is a huge challenge. In this end, the generative adversarial network (GAN) [123] is becoming a suitable alternative.

- **Online Training**: In order to make the training performance more efficient and practical, it is necessary to consider about online training method, especially considering the research challenge on how to guarantee the well-trained DRL model or DNN can work still well in real-world. In this case, in the future, the combination of online training method and DRL will be further studied.

Bibliography

[1] M. Bianchini and F. Scarselli, "On the complexity of neural network classifiers: A comparison between shallow and deep architectures," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 8, pp. 1553–1565, 2014.

[2] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5g wireless networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 1617–1655, 2016.

[3] A. Hanscom and M. Bedford, "Unmanned aircraft system (uas) service demand 2015-2035," *literature review & projections of future usage, Res. Innov. Technol. Admin., US Dept. Transp., Washington, DC, USA, Tech. Rep. DOT-VNTSC-DoD-13-01*, 2013.

[4] A. Merwaday and I. Guvenc, "UAV assisted heterogeneous networks for public safety communications," in *2015 IEEE wireless communications and networking conference workshops (WCNCW)*. IEEE, 2015, pp. 329–334.

[5] K. Santhi, V. Srivastava, G. SenthilKumaran, and A. Butare, "Goals of true broad band's wireless next wave (4g-5g)," in *2003 IEEE 58th Vehicular Technology Conference. VTC 2003-Fall (IEEE Cat. No.03CH37484)*, vol. 4, 2003, pp. 2317–2321 Vol.4.

[6] T. Halonen, J. Romero, and J. Melero, *GSM, GPRS and EDGE performance: evolution towards 3G/UMTS*. John Wiley & Sons, 2004.

[7] T. S. Rappaport *et al.*, *Wireless communications: principles and practice*. prentice hall PTR New Jersey, 1996, vol. 2.

[8] B. Furht and S. A. Ahson, *Long Term Evolution: 3GPP LTE radio and cellular technology*. Crc Press, 2016.

[9] S. Sesia, I. Toufik, and M. Baker, *LTE-the UMTS long term evolution: from theory to practice*. John Wiley & Sons, 2011.

[10] T. S. Rappaport, W. Roh, and K. Cheun, "Wireless engineers long considered high frequencies worthless for cellular systems. they couldn't be more wrong," *IEEE SPECTRUM*, vol. 51, no. 9, pp. 34–+, 2014.

[11] M. Shafi, A. F. Molisch, P. J. Smith, T. Haustein, P. Zhu, P. De Silva, F. Tufvesson, A. Benjebbour, and G. Wunder, "5g: A tutorial overview of standards, trials, challenges, deployment, and practice," *IEEE journal on selected areas in communications*, vol. 35, no. 6, pp. 1201–1221, 2017.

[12] M. Simsek, A. Aijaz, M. Dohler, J. Sachs, and G. Fettweis, "5g-enabled tactile internet," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 460–473, 2016.

[13] W. Ejaz, A. Anpalagan, M. A. Imran, M. Jo, M. Naeem, S. B. Qaisar, and W. Wang, "Internet of things (iot) in 5g wireless communications," *IEEE Access*, vol. 4, pp. 10 310–10 314, 2016.

[14] K. P. Valavanis and G. J. Vachtsevanos, *Handbook of unmanned aerial vehicles*. Springer, 2015, vol. 2077.

[15] Y. Zeng, Q. Wu, and R. Zhang, "Accessing from the sky: A tutorial on uav communications for 5g and beyond," *Proceedings of the IEEE*, vol. 107, no. 12, pp. 2327–2375, 2019.

[16] G. T. 38.913, "Technical specification group radio access network; study on scenarios and requirements for next generation access technologies;(release 14)," *Tech. Rep.*, 2016.

[17] N. Goddemeier, K. Daniel, and C. Wietfeld, "Role-based connectivity management with realistic air-to-ground channels for cooperative uavs," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 5, pp. 951–963, 2012.

[18] Y. Zeng and R. Zhang, "Energy-efficient UAV communication with trajectory optimization," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 3747–3760, June 2017.

[19] R. Amorim, H. Nguyen, P. Mogensen, I. Z. Kovács, J. Wigard, and T. B. Sørensen, "Radio channel modeling for UAV communication over cellular networks," *IEEE Wireless Communications Letters*, vol. 6, no. 4, pp. 514–517, Aug 2017.

[20] M. M. Azari, F. Rosas, K.-C. Chen, and S. Pollin, "Ultra reliable uav communication using altitude and cooperation diversity," *IEEE Transactions on Communications*, vol. 66, no. 1, pp. 330–344, 2017.

[21] A. Al-Hourani and K. Gomez, "Modeling cellular-to-uav path-loss for suburban environments," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 82–85, 2017.

[22] A. Al-Hourani, S. Kandeepan, and S. Lardner, "Optimal LAP altitude for maximum coverage," *IEEE Wireless Communications Letters*, vol. 3, no. 6, pp. 569–572, Dec 2014.

[23] I. Update, "Ericsson mobility report," *Ericsson: Stockholm, Sweden*, 2018.

[24] R. Lu, X. Liang, X. Li, X. Lin, and X. Shen, "Eppa: An efficient and privacy-preserving aggregation scheme for secure smart grid communications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1621–1631, 2012.

[25] W. Wei, F. Xu, and Q. Li, "Mobishare: Flexible privacy-preserving location sharing in mobile online social networks," in *2012 Proceedings IEEE INFOCOM*. IEEE, 2012, pp. 2616–2620.

[26] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: Opportunities and challenges," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 36–42, 2016.

[27] F. Zhou, R. Q. Hu, Z. Li, and Y. Wang, "Mobile edge computing in unmanned aerial vehicle networks," *IEEE Wireless Communications*, vol. 27, no. 1, pp. 140–146, 2020.

[28] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[29] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level

control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[30] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Advances in neural information processing systems*, 2000, pp. 1008–1014.

[31] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[32] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Thirtieth AAAI Conference on Artificial Intelligence*, Mar. 2016.

[33] L. Kong, L. Ye, F. Wu, M. Tao, G. Chen, and A. V. Vasilakos, "Autonomous relay for millimeter-wave wireless communications," *IEEE J. Select. Areas Commun.*, vol. 35, no. 9, pp. 2127–2136, 2017.

[34] U. Challita, A. Ferdowsi, M. Chen, and W. Saad, "Machine learning for wireless connectivity and security of cellular-connected UAVs," *IEEE Wireless Communications*, vol. 26, no. 1, pp. 28–35, 2019.

[35] C. Zhan, Y. Zeng, and R. Zhang, "Energy-efficient data collection in UAV enabled wireless sensor network," *IEEE Wireless Communications Letters*, vol. 7, no. 3, pp. 328–331, June 2018.

[36] J. Xu, Y. Zeng, and R. Zhang, "UAV-enabled wireless power transfer: Trajectory design and energy optimization," *IEEE Transactions on Wireless Communications*, vol. 17, no. 8, pp. 5092–5106, Aug 2018.

[37] N. Zhao, F. Cheng, F. R. Yu, J. Tang, Y. Chen, G. Gui, and H. Sari, "Caching UAV assisted secure transmission in hyper-dense networks based on interference alignment," *IEEE Transactions on Communications*, vol. 66, no. 5, pp. 2281–2294, 2018.

[38] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Transactions on Wireless Communications*, vol. 15, no. 6, pp. 3949–3963, 2016.

[39] A. Al-Hourani, S. Kandeepan, and S. Lardner, "Optimal LAP altitude for maximum coverage," *IEEE Wireless Communications Letters*, vol. 3, no. 6, pp. 569–572, 2014.

[40] H. He, S. Zhang, Y. Zeng, and R. Zhang, "Joint Altitude and Beamwidth Optimization for UAV-Enabled Multiuser Communications," *IEEE Communications Letters*, vol. 22, no. 2, pp. 344–347, Feb 2018.

[41] Q. Wu and R. Zhang, "Common throughput maximization in UAV-enabled OFDMA systems with delay consideration," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6614–6627, Dec. 2018.

[42] J. Lyu, Y. Zeng, and R. Zhang, "UAV-aided offloading for cellular hotspot," *IEEE Transactions on Wireless Communications*, vol. 17, no. 6, pp. 3988–4001, 2018.

[43] Q. Wu, Y. Zeng, and R. ZHANG, "Joint trajectory and communication design for multi-UAV enabled wireless networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 2109–2121, 2018.

[44] Z. Yang, C. Pan, K. Wang, and M. Shikh-Bahaei, "Energy efficient resource allocation in UAV-enabled mobile edge computing networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 9, p. 4576–4589, Sep 2019. [Online]. Available: http://dx.doi.org/10.1109/twc.2019.2927313

[45] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Mobile unmanned aerial vehicles (UAVs) for energy-efficient internet of things communications," *IEEE Transactions on Wireless Communications*, vol. 16, no. 11, pp. 7574–7589, 2017.

[46] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[47] 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," *IEEE Transactions on Wireless Communications*, vol. 16, no. 8, pp. 4924–4938, Aug 2017.

[48] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, pp. 4569–4581, Sep. 2013.

[49] F. Jiang, K. Wang, L. Dong, C. Pan, W. Xu, and K. Yang, "AI driven heterogeneous MEC system with UAV assistance for dynamic environment: Challenges and solutions," *IEEE Network*, pp. 1–9, 2020.

[50] Y. Du, K. Yang, K. Wang, G. Zhang, Y. Zhao, and D. Chen, "Joint resources and workflow scheduling in UAV-enabled wirelessly-powered MEC for IoT systems," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 10, pp. 10 187–10 200, Oct 2019.

[51] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 9, pp. 1927–1941, 2018.

[52] A. Asheralieva and D. Niyato, "Hierarchical game-theoretic and reinforcement learning framework for computational offloading in UAV-enabled mobile edge computing networks with multiple service providers," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8753–8769, 2019.

[53] Q. Zhang, J. Chen, L. Ji, Z. Feng, Z. Han, and Z. Chen, "Response delay optimization in mobile edge computing enabled UAV swarm," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3280–3295, 2020.

[54] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," 2015.

[55] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, Nov. 2015.

[56] X. Lyu, H. Tian, W. Ni, Y. Zhang, P. Zhang, and R. P. Liu, "Energy-Efficient Admission of Delay-Sensitive Tasks for Mobile Edge Computing," *IEEE Transactions on Communications*, vol. 66, no. 6, pp. 2603–2616, June 2018.

[57] K. Yang, S. Ou, and H. Chen, "On effective offloading services for resource-constrained mobile devices running heavier mobile internet applications," *IEEE Communications Magazine*, vol. 46, no. 1, pp. 56–63, January 2008.

[58] L. Zhang, K. Wang, D. Xuan, and K. Yang, "Optimal Task Allocation in Near-Far Computing Enhanced C-RAN for Wireless Big Data Processing," *IEEE Wireless Communications*, vol. 25, no. 1, pp. 50–55, February 2018.

[59] H. Mei, K. Wang, and K. Yang, "Multi-layer cloud-ran with cooperative resource allocations for low-latency computing and communication services," *IEEE Access*, vol. 5, pp. 19 023–19 032, 2017.

[60] X. Wang, K. Wang, S. Wu, S. Di, K. Yang, and H. Jin, "Dynamic resource scheduling in cloud radio access network with mobile cloud computing," in *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*, June 2016, pp. 1–6.

[61] X. Chen, "Decentralized Computation Offloading Game for Mobile Cloud Computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974–983, April 2015.

[62] X. Wang, K. Wang, S. Wu, S. Di, H. Jin, K. Yang, and S. Ou, "Dynamic Resource Scheduling in Mobile Edge Cloud with Cloud Radio Access Network," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 11, pp. 2429–2445, Nov 2018.

[63] J. Lyu, Y. Zeng, R. Zhang, and T. J. Lim, "Placement Optimization of UAV-Mounted Mobile Base Stations," *IEEE Communications Letters*, vol. 21, no. 3, pp. 604–607, March 2017.

[64] Y. Chen, N. Zhao, Z. Ding, and M. Alouini, "Multiple UAVs as Relays: Multi-Hop Single Link Versus Multiple Dual-Hop Links," *IEEE Transactions on Wireless Communications*, vol. 17, no. 9, pp. 6348–6359, Sept 2018.

[65] Q. Wu and R. Zhang, "Common throughput maximization in UAV-enabled OFDMA systems with delay consideration," *IEEE Transactions on Communications*, vol. 66, no. 12, pp. 6614–6627, 2018.

[66] L. Yang, J. Cao, S. Tang, T. Li, and A. T. S. Chan, "A Framework for Partitioning and Execution of Data Stream Applications in Mobile Cloud Computing," in *2012 IEEE Fifth International Conference on Cloud Computing*, June 2012, pp. 794–802.

[67] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—a key technology towards 5G," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[68] Y. Du, K. Wang, K. Yang, and G. Zhang, "Energy-efficient resource allocation in UAV based MEC system for IoT devices," in *IEEE Global Communications Conference*, 2018, pp. 1–6.

[69] Z. Li, M. Chen, C. Pan, N. Huang, Z. Yang, and A. Nallanathan, "Joint trajectory and communication design for secure UAV networks," *IEEE Commun. Lett.*, pp. 1–4, Feb. 2019.

[70] C. H. Liu, Z. Chen, J. Tang, J. Xu, and C. Piao, "Energy-efficient UAV control for effective and fair communication coverage: A deep reinforcement learning approach," *IEEE J. Select. Areas Commun.*, vol. 36, no. 9, pp. 2059–2070, Sep. 2018.

[71] F. Jiang, K. Wang, L. Dong, C. Pan, W. Xu, and K. Yang, "Deep learning based joint resource scheduling algorithms for hybrid MEC networks," *IEEE Internet of Things Journal*, 2019.

[72] S. Mitchell, M. G. O. Sullivan, and I. Dunning, "Pulp : A linear programming toolkit for python," in *Python*, 2011.

[73] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[74] C. Pan, H. Zhu, N. J. Gomes, and J. Wang, "Joint precoding and RRH selection for user-centric green MIMO C-RAN," *IEEE Transactions on wireless Communications*, vol. 16, no. 5, pp. 2891–2906, 2017.

[75] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[76] J. Hamari, J. Koivisto, H. Sarsa *et al.*, "Does gamification work?-a literature review of empirical studies on gamification." in *HICSS*, vol. 14, no. 2014, 2014, pp. 3025–3034.

[77] A. R. Mahmood, H. P. Van Hasselt, and R. S. Sutton, "Weighted importance sampling for off-policy learning with linear function approximation," in *Advances in Neural Information Processing Systems*, 2014, pp. 3014–3022.

[78] R. Ding, F. Gao, and X. S. Shen, "3D UAV trajectory design and frequency band allocation for energy-efficient and fair communication: A deep reinforcement learning approach," *IEEE Transactions on Wireless Communications*, vol. 19, no. 12, pp. 7796–7809, 2020.

[79] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE Transactions on Wireless Communications*, vol. 18, no. 4, pp. 2329–2345, 2019.

[80] L. Kong, L. Ye, F. Wu, M. Tao, G. Chen, and A. V. Vasilakos, "Autonomous relay for millimeter-wave wireless communications," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 9, pp. 2127–2136, Sep. 2017.

[81] R. Fan, J. Cui, S. Jin, K. Yang, and J. An, "Optimal node placement and resource allocation for UAV relaying network," *IEEE Communications Letters*, vol. 22, no. 4, pp. 808–811, 2018.

[82] H. Wang, J. Wang, G. Ding, J. Chen, F. Gao, and Z. Han, "Completion time minimization with path planning for fixed-wing UAV communications," *IEEE Transactions on Wireless Communications*, vol. 18, no. 7, pp. 3485–3499, 2019.

[83] F. Cui, Y. Cai, Z. Qin, M. Zhao, and G. Y. Li, "Multiple access for mobile-UAV enabled networks: Joint trajectory design and resource allocation," *IEEE Transactions on Communications*, vol. 67, no. 7, pp. 4980–4994, 2019.

[84] T. Q. Duong, L. D. Nguyen, H. D. Tuan, and L. Hanzo, "Learning-aided realtime performance optimisation of cognitive UAV-assisted disaster communication," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[85] X. Liu, Y. Liu, Y. Chen, and L. Hanzo, "Trajectory design and power control for multi-UAV assisted wireless networks: A machine learning approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7957–7969, 2019.

[86] J. Wang, C. Jiang, Z. Wei, C. Pan, H. Zhang, and Y. Ren, "Joint UAV hovering altitude and power control for space-air-ground IoT networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1741–1753, April 2019.

[87] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Transactions on Wireless Communications*, vol. 15, no. 6, pp. 3949–3963, June 2016.

[88] Y. Zeng, R. Zhang, and T. Lim, "Throughput maximization for UAV-enabled mobile relaying systems," *IEEE Transactions on Communications*, vol. 64, no. 12, pp. 4983–4996, 2016.

[89] M. Alzenad, A. El-Keyi, F. Lagum, and H. Yanikomeroglu, "3-D placement of an unmanned aerial vehicle base station (UAV-BS) for energy-efficient maximal coverage," *IEEE Wireless Communications Letters*, vol. 6, no. 4, pp. 434–437, Aug 2017.

[90] K. Wang, P. Huang, K. Yang, C. Pan, and J. Wang, "Unified offloading decision making and resource allocation in ME-RAN," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 8159–8172, Aug 2019.

[91] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2322–2358, Fourthquarter 2017.

[92] Z. Yang, C. Pan, K. Wang, and M. Shikh-Bahaei, "Energy efficient resource allocation in UAV-enabled mobile edge computing networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 9, pp. 4576–4589, 2019.

[93] Y. Zhou, C. Pan, P. L. Yeoh, K. Wang, M. Elkashlan, B. Vucetic, and Y. Li, "Secure communications for UAV-enabled mobile edge computing systems," *IEEE Transactions on Communications*, vol. 68, no. 1, pp. 376–388, 2020.

[94] N. H. Motlagh, M. Bagaa, and T. Taleb, "UAV-based iot platform: A crowd surveillancce use case," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 128–134, 2017.

[95] S. Jeong, O. Simeone, and J. Kang, "Mobile Edge Computing via a UAV-Mounted Cloudlet: Optimization of Bit Allocation and Path Planning," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2049–2063, March 2018.

[96] M. Hua, Y. Wang, Q. Wu, H. Dai, Y. Huang, and L. Yang, "Energy-efficient cooperative secure transmission in multi-UAV-enabled wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7761–7775, 2019.

[97] J. Wang, C. Jiang, H. Zhang, Y. Ren, K. Chen, and L. Hanzo, "Thirty years of machine learning: The road to Pareto-optimal wireless networks," *IEEE Communications Surveys Tutorials*, pp. 1–46, 2020.

[98] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[99] R. S. Sutton, A. G. Barto *et al.*, *Introduction to reinforcement learning.* MIT press Cambridge, 1998, vol. 2, no. 4.

[100] Y. Li, "Deep reinforcement learning: An overview," *arXiv preprint arXiv:1701.07274*, 2017.

[101] J. Wang, C. Jiang, K. Zhang, X. Hou, Y. Ren, and Y. Qian, "Distributed Q-learning aided heterogeneous network association for energy-efficient IIoT," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2756–2764, April 2020.

[102] L. Bu, R. Babu, B. De Schutter *et al.*, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 2, pp. 156–172, 2008.

[103] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6382–6393.

[104] K. Wang, K. Yang, and C. S. Magurawalage, "Joint Energy Minimization and Resource Allocation in C-RAN with Mobile Cloud," *IEEE Transactions on Cloud Computing*, vol. 6, no. 3, pp. 760–770, July 2018.

[105] L. Yang, J. Cao, Y. Yuan, T. Li, A. Han, and A. Chan, "A framework for partitioning and execution of data stream applications in mobile cloud computing,"

ACM SIGMETRICS Performance Evaluation Review, vol. 40, no. 4, pp. 23–32, 2013.

[106] H. He, S. Zhang, Y. Zeng, and R. Zhang, "Joint altitude and beamwidth optimization for UAV-enabled multiuser communications," *IEEE Communications Letters*, vol. 22, no. 2, pp. 344–347, 2017.

[107] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine learning proceedings 1994*. Elsevier, 1994, pp. 157–163.

[108] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[109] Y. Zeng, Q. Wu, and R. Zhang, "Accessing from the sky: A tutorial on UAV communications for 5G and beyond," *Proceedings of the IEEE*, vol. 107, no. 12, pp. 2327–2375, Dec 2019.

[110] L. Wang, K. Wang, C. Pan, W. Xu, N. Aslam, and L. Hanzo, "Multi-agent deep reinforcement learning-based trajectory planning for multi-uav assisted mobile edge computing," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 1, pp. 73–84, 2021.

[111] L. Wang, P. Huang, K. Wang, G. Zhang, L. Zhang, N. Aslam, and K. Yang, "Rl-based user association and resource allocation for multi-uav enabled mec," in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2019, pp. 741–746.

[112] Z. Yang, C. Pan, M. Shikh-Bahaei, W. Xu, M. Chen, M. Elkashlan, and A. Nallanathan, "Joint altitude, beamwidth, location, and bandwidth optimization for UAV-enabled communications," *IEEE Communications Letters*, vol. 22, no. 8, pp. 1716–1719, Aug 2018.

[113] Y. Zeng and R. Zhang, "Energy-efficient UAV communication with trajectory optimization," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 3747–3760, 2017.

[114] J. Lyu, Y. Zeng, and R. Zhang, "Cyclical multiple access in UAV-aided communications: A throughput-delay tradeoff," *IEEE Wireless Communications Letters*, vol. 5, no. 6, pp. 600–603, 2016.

[115] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Mobile internet of things: Can UAVs provide an energy-efficient mobile architecture?" in *2016 IEEE Global Communications Conference (GLOBECOM)*. Piscataway: IEEE Press, Dec 2016, pp. 1–6.

[116] P. Huang, Y. Wang, K. Wang, and K. Yang, "Differential evolution with a variable population size for deployment optimization in a UAV-assisted IoT data collection system," *IEEE Transactions on Emerging Topics in Computational Intelligence*, pp. 1–12, 2019.

[117] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[118] R. Jain, A. Durresi, and G. Babic, "Throughput fairness index: An explanation," in *ATM Forum contribution*, vol. 99, no. 45, 1999.

[119] C. H. Liu, Z. Chen, and Y. Zhan, "Energy-efficient distributed mobile crowd sensing: A deep learning approach," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1262–1276, 2019.

[120] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[121] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," 2016.

[122] A. G. Barto and S. Mahadevan, "Recent advances in hierarchical reinforcement learning," *Discrete event dynamic systems*, vol. 13, no. 1, pp. 41–77, 2003.

[123] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[124] T. Q. Dinh, Q. D. La, T. Q. S. Quek, and H. Shin, "Distributed Learning for Computation Offloading in Mobile Edge Computing," *IEEE Transactions on Communications*, pp. 1–1, 2018.

[125] T. Li, C. S. Magurawalage, K. Wang, K. Xu, K. Yang, and H. Wang, "On Efficient Offloading Control in Cloud Radio Access Network with Mobile Edge Computing," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, June 2017, pp. 2258–2263.

[126] T. Soyata, R. Muraleedharan, C. Funai, M. Kwon, and W. Heinzelman, "Cloud-Vision: Real-time face recognition using a mobile-cloudlet-cloud acceleration architecture," in *2012 IEEE Symposium on Computers and Communications (ISCC)*, July 2012, pp. 000 059–000 066.

[127] L. Wang, K. Wang, C. Pan, W. Xu, N. Aslam, and A. Nallanathan, "Deep reinforcement learning based dynamic trajectory control for uav-assisted mobile edge computing," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2021.

[128] L. Wang, K. Wang, C. Pan, X. Chen, and N. Aslam, "Deep q-network based dynamic trajectory design for uav-aided emergency communications," *Journal of Communications and Information Networks*, vol. 5, no. 4, pp. 393–402, 2020.

[129] J. Xu, Y. Zeng, and R. Zhang, "UAV-enabled wireless power transfer: Trajectory design and energy optimization," *IEEE Transactions on Wireless Communications*, vol. 17, no. 8, pp. 5092–5106, 2018.

[130] A. Osseiran, O. Elloumi, J. Song, and J. Montserrat, "Internet of things," *IEEE Communications Standards Magazine*, vol. 1, pp. 84–84, 01 2017.

[131] Z. Dawy, W. Saad, A. Ghosh, J. G. Andrews, and E. Yaacoub, "Toward massive machine type cellular communications," *IEEE Wireless Communications*, vol. 24, no. 1, pp. 120–128, 2016.

[132] L. Wei, R. Hu, Y. Qian, and G. Wu, "Enable device-to-device communications underlaying cellular networks: challenges and research aspects," *IEEE Communications Magazine*, vol. 52, no. 6, pp. 90–96, 2014.

[133] F. Zhou, Y. Wu, H. Sun, and Z. Chu, "UAV-enabled mobile edge computing: Offloading optimization and trajectory design," *arXiv preprint arXiv:1802.03906*, 2018.

[134] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," 2011.

[135] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, "Performance optimization in mobile-edge computing via deep reinforcement learning," *arXiv preprint arXiv:1804.00514*, 2018.

[136] L. Huang, S. Bi, and Y.-J. A. Zhang, "Deep reinforcement learning for online offloading in wireless powered mobile-edge computing networks," 2018.

[137] X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui, "Joint computation and communication cooperation for mobile edge computing," in *2018 16th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*. IEEE, 2018, pp. 1–6.

[138] M.-A. Messous, H. Sedjelmaci, N. Houari, and S.-M. Senouci, "Computation offloading game for an UAV network in mobile edge computing," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–6.

[139] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy, "Deep exploration via bootstrapped dqn," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 4026–4034. [Online]. Available: http://papers.nips.cc/paper/6501-deep-exploration-via-bootstrapped-dqn.pdf

[140] K. Wang and K. Yang, "Resource allocation between service computing and communication computing for mobile operator," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–6.

[141] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[142] L.-J. Lin, "Reinforcement learning for robots using neural networks," CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE, Tech. Rep., 1993.

[143] J. L. McClelland, B. L. McNaughton, and R. C. O'reilly, "Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory." *Psychological review*, vol. 102, no. 3, p. 419, 1995.

[144] J. O'Neill, B. Pleydell-Bouverie, D. Dupret, and J. Csicsvari, "Play it again: reactivation of waking experience and memory," *Trends in neurosciences*, vol. 33, no. 5, pp. 220–229, 2010.

[145] S. Adam, L. Busoniu, and R. Babuska, "Experience replay for real-time reinforcement learning control," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 2, pp. 201–212, 2012.

[146] J. Peters and S. Schaal, "Natural actor-critic," *Neurocomputing*, vol. 71, no. 7-9, pp. 1180–1190, 2008.

[147] P. Zhan, K. Yu, and A. L. Swindlehurst, "Wireless relay communications with unmanned aerial vehicles: Performance and optimization," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 3, pp. 2068–2085, July 2011.

[148] J. Gong, T.-H. Chang, C. Shen, and X. Chen, "Aviation time minimization of UAV for data collection from energy constrained sensor networks," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2018, pp. 1–6.

[149] J. Gu, T. Su, Q. Wang, X. Du, and M. Guizani, "Multiple moving targets surveillance based on a cooperative network for multi-UAV," *IEEE Communications Magazine*, vol. 56, no. 4, pp. 82–89, 2018.

[150] H. Q. Le, H. Al-Shatri, and A. Klein, "Efficient resource allocation in mobile-edge computation offloading: Completion time minimization," in *2017 IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 2513–2517.

[151] S. Garg, A. Singh, S. Batra, N. Kumar, and L. T. Yang, "UAV-empowered edge computing environment for cyber-threat detection in smart vehicles," *IEEE Network*, vol. 32, no. 3, pp. 42–51, May 2018.

[152] R. K. Jain, D.-M. W. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination," *Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA*, 1984.

[153] D. P. Bertsekas, *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 1995, vol. 1, no. 2.

[154] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ser. ICML'14, 2014.

[155] P. Zhan, K. Yu, and A. L. Swindlehurst, "Wireless relay communications with unmanned aerial vehicles: Performance and optimization," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 3, pp. 2068–2085, 2011.

[156] J. Lyu, Y. Zeng, and R. Zhang, "UAV-aided offloading for cellular hotspot," *IEEE Transactions on Wireless Communications*, vol. 17, no. 6, pp. 3988–4001, June 2018.

[157] J. Gu, T. Su, Q. Wang, X. Du, and M. Guizani, "Multiple moving targets surveillance based on a cooperative network for multi-UAV," *IEEE Communications Magazine*, vol. 56, no. 4, pp. 82–89, April 2018.

[158] A. E. A. A. Abdulla, Z. M. Fadlullah, H. Nishiyama, N. Kato, F. Ono, and R. Miura, "An optimal data collection technique for improved utility in UAS-aided networks," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, April 2014, pp. 736–744.

[159] Z. Han, A. L. Swindlehurst, and K. J. R. Liu, "Optimization of MANET connectivity via smart deployment/movement of unmanned air vehicles," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 7, pp. 3533–3546, Sep. 2009.

[160] T. Schouwenaars, B. De Moor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," in *2001 European Control Conference (ECC)*, Sep. 2001, pp. 2603–2608.

[161] S. Garg, A. Singh, S. Batra, N. Kumar, and L. T. Yang, "UAV-empowered edge computing environment for cyber-threat detection in smart vehicles," *IEEE Network*, vol. 32, no. 3, pp. 42–51, 2018.

[162] T. Hoßfeld, L. Skorin-Kapov, P. E. Heegaard, and M. Varela, "Definition of QoE fairness in shared systems," *IEEE Communications Letters*, vol. 21, no. 1, pp. 184–187, Jan 2017.

[163] T. Bai, J. Wang, Y. Ren, and L. Hanzo, "Energy-efficient computation offloading for secure UAV-edge-computing systems," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 6, pp. 6074–6087, 2019.