Applications of Model-Free Learning in Wireless Networks

by

Huijin Cao

A Thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements of the degree of

DOCTOR OF PHILOSOPHY

Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg

Supervisor: **Prof. Jun Cai**

Abstract

When wireless decision-making entities have complete and global information, the network control problems are frequently addressed in the model-based paradigm. However, due to the practical limitation of information incompleteness/locality, directly applying the model-based solutions will face difficulties since a model of the network dynamics may even not be available in advance, or in most cases its details may be inaccurate or not instantaneously known. Thus, wireless decision-making entities may face a black-box network control problem and the model-based network management mechanisms will be no longer applicable. As a result, the method of controlling-by-learning without the need for the *a priori* network model, namely, the model-free learning, has been considered as one promising implementation approach to wireless networks. In this thesis, the applications of the model-free learning in three networks with different characteristics and objectives are investigated, they are an opportunistic spectrum access (OSA) network with multiple heterogeneous secondary users (SUs) and primary channels, a cloudlet-based mobile cloud computing (MCC) network with multi-mobile-device computation offloading in a multi-channel wireless contention environment, and an energy harvesting based network comprising a single macro-cell base station (MBS) with grid power supply and multiple small-cells with renewable energy supply. Effective and efficient model-free learning mechanisms in these networks are aimed to be designed. In particular, i) a new Stochastic Learning Automata (SLA) based algorithm, called N-SLA, is designed for the OSA network to

adapt each SU's spectrum access strategy in an unknown and dynamic environment so as to converge towards a Nash Equilibrium (NE) in a fully distributed way, ii) a fully distributed computation offloading (FDCO) algorithm based on machine learning technology is proposed for the cloudlet-based MCC network so that each mobile user can make the offloading decisions independently and adaptively, and a pure-strategy NE can be finally achieved, and iii) a Post-Decision State based Approximate Reinforcement Learning (PDS-ARL) algorithm is proposed for the energy harvesting based network, which learns on-the-fly the optimal context-aware proactive caching policy with a high learning efficiency, to fully reap the benefits of energy harvesting. Both analytical and numerical results validate the efficacy of the proposed algorithms.

Keywords: Cognitive radio networks, opportunistic spectrum access, Nash Equilibrium, ordinal potential games, stochastic learning automata, cloudlet-based mobile cloud computing, computation offloading, exact potential game, machine learning, energy harvesting, proactive caching, post-decision state, approximate reinforcement learning, Markov decision process.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my advisor Prof. Jun Cai for his continuous support during my Ph.D. study. Prof. Cai taught me how to do decent research, how to improve the logical thinking ability, and how to write an attractive paper. He always inspired me not to give up during my tough days, and encouraged me to move forward when I made achievements. Without his patience, motivation, and immense knowledge, I would not have completed my Ph.D. study.

Besides, I would like to thank my examining committee members: Prof. Attahiru S. Alfa, Prof. Qingjin Peng, and Prof. Vincent Wong for their valuable feedback and constructive suggestions. In addition, I would also like to express my sincere thanks to my previous supervisor Prof. Yanhui Lu for her moral encouragement and invaluable support. All of these are very helpful in finishing my research work and the writing of this thesis.

I thank all my colleagues in the Network Intelligence and Innovation (NI2) Lab for their insightful comments and suggestions during our discussions. My sincere thanks go also to the teaching, administrative and technical staff of the Department of Electrical and Computer Engineering, the University of Manitoba, which provides a peaceful and productive working environment.

Last but not least, I must express my profound gratitude to my dear husband Ran Zi for his continuous support, constant encouragement and wholehearted dedications, and to my parents and sister who are always my strong backing. I also appreciate the continued encouragement that my best friends Lele Li, Sui Chan, and Huixin Zhang gave me, I am truly lucky to have them in my life.

This thesis is dedicated to my family.

Contents

List of Figures

List of Tables

List of Abbreviations

CR	Cognitive Radio
PHY	Physical Layer
MAC	Medium Access Control
OSA	Opportunistic Spectrum Access
RL	Reinforcement Learning
PU	Primary User
BR	Best Response
NE	Nash Equilibrium
FP	Fictitious Play
GP	Gradient Play
LA	Learning Automata
DP	Dynamic Programming
CE	Correlated Equilibrium
SG	Stochastic Games
RF	Radio Frequency
SU	Secondary User
SLA	Stochastic Learning Automata
N-SLA	New-Stochastic Learning Automata
MCC	Mobile Cloud Computing
MDP	Markov Decision Process

MBS	Macro-Cell Base Station
SBS	Small-Cell Base Station
FDCO	Fully Distributed Computation Offloading
PDS	Post-Decision State
ARL	Approximate Reinforcement Learning
eICIC	enhanced Inter-Cell Interference Coordination (eICIC)
CPP	Content Preference Profile
PDS	Post-Decision State
ARL	Approximate Reinforcement Learning
UCB1	Upper-Confidence-Bound 1
ODE	Ordinary Differential Equation
CSMA	Carrier Sense Multiple Access
WAN	Wide Area Network
QoE	Quality of Experience
TL	Transfer Learning

Chapter 1

Introduction

Wireless networks operate within a dynamic environment, which entails many time-varying characteristics such as variable link qualities, dynamic spectrum usage, and changing traffic patterns [1]. The changing nature of these characteristics will significantly affect network performance, and hence has a profound impact on the efficient network control. However, in many practical scenarios, the complexity of network dynamics makes it difficult to determine the network evolution model in advance. As a result, without knowing the accurate network model in advance, the wireless decision-making entities may face a black-box network control problem and the traditional model-based network control mechanisms will be no longer applicable. Thus, the methods of model-free learning without the need for the *a priori* network model are considered promising and more appropriate [2–4].

Many existing studies on model-free learning in wireless networks focused on radio environment awareness in order to enhance spectrum efficiency. This leads to the concept of Cognitive Radio (CR) networks [5], which are featured by a novel physical

layer and medium access control (PHY-MAC) architecture, namely, spectrum sharing between the licensed/primary networks and the unlicensed/secondary networks, for opportunistic spectrum access based on the detection of spectrum holes [6]. Specifically, in CR networks, a secondary network relies on spectrum cognition modules to make proper decisions for seamless spectrum access without interfering the primary transmissions. For this category of works in the literature, "learning" is a set of techniques for feature classification of primary signal identification [7]. For an overview of the relevant techniques, the readers may refer to recent survey works in [8–10].

In addition to being aware of the environment, learning is also necessary for the efficient adaptation of the wireless networks to their environment with unknown and time-varying characteristics. Specifically, due to the incomplete information and the complex interactions among the transmit parameters and policies (e.g., transmit power, coding scheme, modulation scheme, communication protocol, etc.) and their impacts on the environment, the precise effects of the network inputs, i.e., the adopted transmission strategies, on the network outputs, i.e., the achieved various network performance such as data rates and delay, is unavailable. In this case, the decision-making entities not only need to dynamically characterize their network situations, but also have to accordingly infer the proper transmission strategies for better adaptation to the environment. Since learning enables the decision-making entities to adapt their behaviors based on the reinforcement from their interaction with the environment and build their understanding of the environment from scratch through trial-and-error, it has been considered as one key implementation approach to adaptive, self-organized network control in wireless networks [11, 12]. In the context of

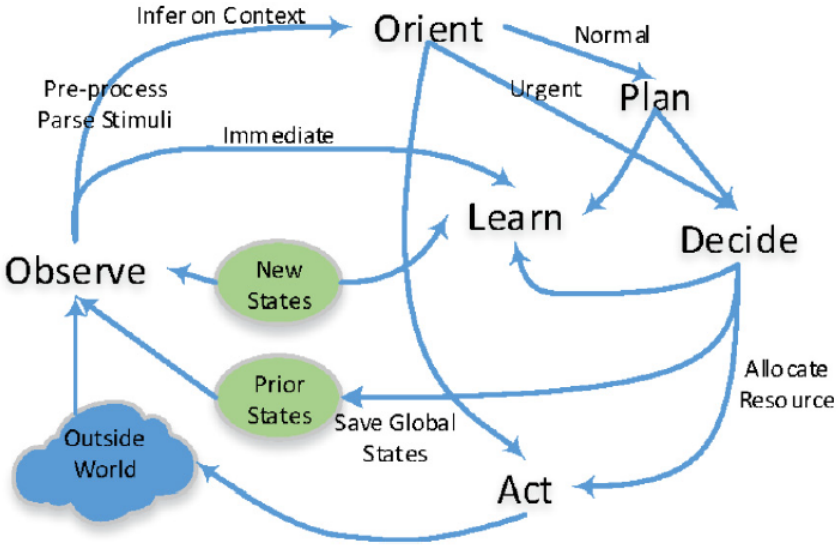

Figure 1.1: Cognition cycle of a single decision-making entity [13]

adaptive control, controlling-by-learning in wireless networks is usually described by the cognition-decision paradigm as shown in Fig. 1.1 [13]. This paradigm describes the learning-based strategy-taking process of a single decision-making entity from a high-level perspective and interprets it as a cognition cycle to present the information flow from environment cognition to the final network control decision. Specifically, the single decision-making entity observes its environment by parsing incoming information streams to infer the communication context. Then, the entity orients itself. Normally, the incoming network message would be dealt with by generating plans. The "Decide" phase selects among the candidate plans. "Acting" initiates the selected plan. "Learning" is a function of observations and decisions, which directs the entity's behaviour.

1.1 Overview of Model-Free Learning Techniques on Network Adaptive Control

In the literature, the existing learning algorithms on network adaptive control are generally classified as either centralized or decentralized. In a centralized scenario, there exists a single decision-making entity, also called as a single agent, which takes charge of the whole network control but with limited ability of network modeling or environment observation. The objective is to design an efficient centralized learning algorithm for the single entity to adapt to the unknown and time-varying environment by properly and adaptively configuring the transmission parameters. In contrast, the problem becomes more complicated in a decentralized scenario, where multiple distributed decision-making entities/agents exist. In this scenario, each agent not only has to adapt to the environment, but also has to coordinate its strategy with others in the network, based on only a limited amount of information exchange among them. Furthermore, how to guarantee the network convergence under the condition of interest conflicts among these agents creates even more challenges.

1.1.1 Centralized Learning Techniques

In a centralized scenario, if the agent-environment interaction forms the Markov Decision Process (MDP) [14], Reinforcement Learning (RL) algorithm [15–17], which was proposed by Watkins in 1989 [18] to solve the MDP problem without knowledge of the transition probabilities, can be utilized to obtain the optimal solution. RL is a technique that permits an agent to modify its behavior by interacting with its

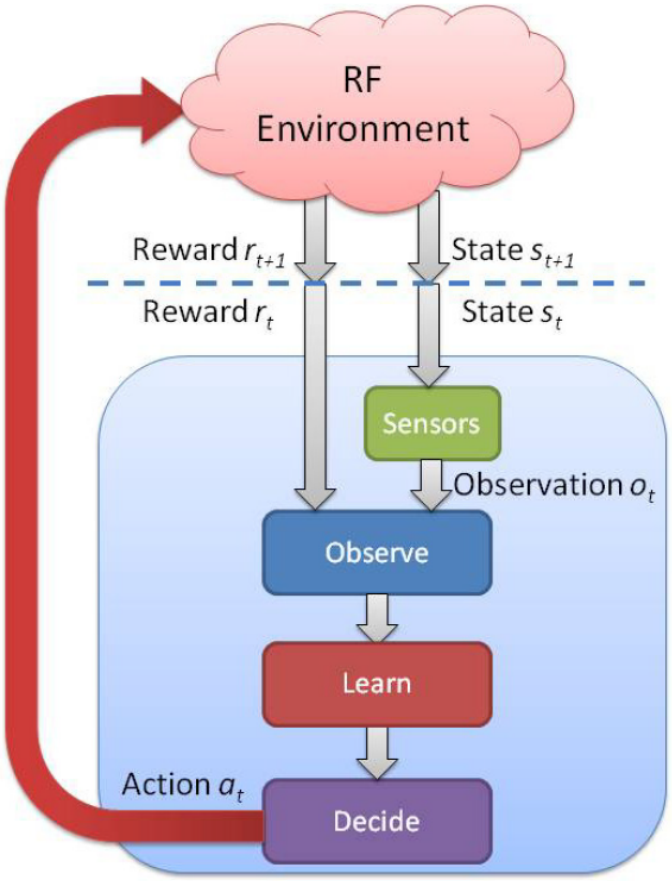

Figure 1.2: The reinforcement learning cycle [7]: At the beginning, the agent receives an observation of the current state and the accrued reward, based on which, the agent updates its policy, and then selects a certain action according to the updated policy.

environment, and the only source of knowledge is the feedback the agent receives from its environment after executing an action. The feedback can be a positive (reward) or negative (cost) quantity, telling how good or bad an action is. The objective of the agent is to find the proper stationary policy that probabilistically maps the environment state to action so that the accumulated long-term utility of the agent is optimized. In Fig. 1.2 [7, 19], an RL-based cognition cycle for an agent is illustrated.

In time epoch t, the learning agent receives an observation o_t of the state s_t. The observation is accompanied by a delayed reward $r_t(s_{t-1}, a_{t-1})$ representing the reward received at time t resulting from taking action a_{t-1} in state s_{t-1} at time $t-1$. The learning agent uses the current observation o_t and the past experience such as the delayed reward $r_t(s_{t-1}, a_{t-1})$ to compute the action a_t that should be taken at time t. The action a_t results in a state transition from s_t to s_{t+1} and a delayed reward $r_{t+1}(s_t, a_t)$. Thus, the learning agent is not passive and does not only observe the outcomes from the environment, but also affects the environment via its actions such that it might be able to drive the environment to a desired state that brings the highest reward to it. It has been demonstrated that the RL algorithm can converge to the optimal policy when applied to single-agent MDP models [15, 18].

In the literature, RL algorithms have been mostly applied into CR networks for determining the spectrum sensing and access policies. For example, in [20], by leveraging the model-free RL algorithm, a novel spectrum sensing strategy for a CR network with a single SU and multiple licensed channels was proposed, aiming to determine the best channel to sense so as to reduce the sensing time overhead and further improve the SU's achievable throughput. An RL-based scheme, which enabled effective OSA to improve the efficiency of spectrum utilization with no prior knowledge of the environment's characteristics and dynamics, was proposed in [21]. The dynamic load-balancing spectrum decision for a CR network that dynamically distributed packets from SUs to different available primary channels was studied in [22], where an RL algorithm was applied to find the minimum delay policy when the traffic and channel characteristics were unknown. In [23], a two-stage RL approach was proposed to not

only select a channel for data transmission, but also to predict how long the channel would remain unoccupied so that the time spent on channel sensing could be minimized. Macaluso *et al.* in [24] further explored the question of when RL algorithm improved the performance of opportunistic channel selection by characterizing the primary user (PU) activity using the concept of Lempel-Ziv complexity, and showed that RL was beneficial only for some levels of PU activity and complexity.

While RL algorithms can lead to optimal policies for MDP problems, their performance in non-Markovian environments remains questionable [25, 26]. Hence, the authors in [25–27] proposed the policy-search approach as an alternative method for non-Markovian learning tasks. Policy-search algorithms directly look for optimal policies in the policy space itself, without having to estimate the actual states of the systems [25, 26]. In particular, by adopting policy-gradient algorithms, the policy vector can be updated to reach an optimal solution (or a local optimum) in non-Markovian environments. Policy-gradient algorithms have shown promising results in robotics applications [28, 29] .

1.1.2 Decentralized Learning Techniques

In decentralized scenarios, according to the degree of coupling among the distributed agents, learning techniques are built upon either loosely coupled multi-agent systems or game based multi-agent systems.

When considering the learning mechanism in a multi-agent system, it is natural to simply adopt the standard single-agent learning algorithms by assuming that each agent is an independent learner with a local utility function. In doing so, the activities

of the other agents are treated as part of a stationary environment and the learning agents update their policies without considering their interactions with other agents. Since multi-agent learning based on single-agent learning requires the joint learning process to be decomposed into local ones, individual agent behaviors are relatively disjoint, and the agents are able to ignore the information raised by the interactions with each other. This is the reason for us to call it as a "loosely coupled multi-agent system". This approach enjoys popularity especially within the studies in the cooperative decision-making domain [30, 31]. Its typical applications can be found in modeling the hunter-prey systems [32] and team coordination [33].

However, it is worth pointing out that for most multi-agent systems, convergence of single-agent learning based algorithms is not guaranteed. Furthermore, even when convergence can be reached, it usually takes a significant amount of time for merely determining switching between one pair of actions [4]. As a result, most of the practical single-agent learning based mechanisms are limited in the special scenarios such as the fully-cooperative multi-agent systems or two-agent multi-agent systems [4].

To better address the problems raised by agent interactions in multi-agent systems, the decentralized network control is always modelled as games [34]. One important reason for adopting game theoretic models in multi-agent systems lies in the requirements that decisions are to be made in a distributed manner with the limited ability of both information acquisition and action coordination. This may be either due to the overwhelming dimension of the state-action space as the number of agents grows, or due to the overhead for information exchange among agents. In the game-based

decision-making model, the individual rationality property of the agents leads to the concept of the best response. The best response of an agent is defined as the local strategy under which the local payoff is not worse than that under any other strategies, given the joint adversary strategies. In the context of games, the goal of learning now becomes finding the strategy-updating rules for converging to a specific equilibrium (such as NE).

When the network evolution is not subject to a stochastic environment, most of the network control problems that require considering the interactions among distributed agents can be formulated as repeated games [35]. In contrast to the MDP-based learning mechanisms that heavily depend on value iteration, policy-search now plays an important role in deriving the learning rules for repeated games. In the literature, there are four prototypical learning schemes that repeated games are based on, including (i) fictitious play (FP), (ii) gradient play (GP), (iii) learning automata (LA), and (iv) no-regret learning. The basic prerequisite of FP and GP is that the agents are willing to reveal their action information to the others after each round of play, so that each agent can track the frequency of action selection by the other agents [36]. In this sense, FP and GP are frequently considered as model-dependent learning mechanisms since they try to build the model of the opponents' joint policies from accumulated experience. However, compared with other model-based, non-learning mechanisms such as dynamic programming (DP) for MDPs [14], FP and GP do not need any *a priori* knowledge of the network or other agents. Therefore, as long as the learning agents are able to observe the actions of the rival agents or afford the overhead for action information exchange, FP and GP can be employed as the basic

solutions for many resource management games in wireless networks [36–40]. In contrast, LA and no-regret learning are featured by the process of action selection using only local information, without action information exchange among agents. Most of the LA based and no-regret learning based schemes were used to address the problem of channel contention in CR networks, which can be found in [41–46], where LA were employed to obtain Nash Equilibrium (NE) policies [41–43], while no-regret learning were widely applied for learning the Correlated Equilibrium (CE) [44–46].

When the network evolution is subject to a stochastic environment, repeated games are then extended into stochastic games (SG) [47], in which the payoff of each agent at each round of the game is also dependent on the state variable, whose evolution is influenced by the joint actions of all agents. Compared with repeated games, SG is considered as a more practical tool for modeling the agent interaction in a stochastic wireless environment, where the elements of the wireless environment (e.g., channel states, buffer states and collision states) evolve stochastically and are influenced by the transmission strategies of each agent. In the context of SGs, learning schemes are mainly referred to the value iteration based and the conjecture based algorithms. Value iteration based learning algorithms can be considered as a combination of a matrix game solver and a value iteration based state value learner, aiming to find a specific equilibrium (NE or CE) of the SG [48–51]. However, for both the NE based and the CE based value iteration learning algorithms, since it is necessary for the SG to be of complete information in order to immediately obtain the NE/CE of the matrix game, it is required that the learning agents should keep track of the entire Q-table from all the other agents at each state. By contrast, in the condition

without knowing what the strategies of the other agents are, or what their payoff functions are, the concept of "conjecture" [52] about one agent's opponent policies is introduced in the conjecture based learning algorithms [53–55]. Compared with the standard value iteration based learning mechanism, the Q-table in conjecture based learning algorithms is constructed with only local states and actions. Hence, to address the unaffordable transmission overhead resulting from information exchanging, the policy conjecture can be implemented to approximately learn the matrix game equilibrium strategy and the Q-table of the SG.

1.2 Motivation and Objectives

In the literature, most of the learning applications in wireless networks are limited to CR networks. The main reason is that CR networks have been widely recognized as intelligent wireless communication networks. In the framework of CR networks, the abilities of being aware of the ongoing Radio Frequency (RF) activities in surrounding environment and learning from the sensed observation to adapt to the unknown and time-varying environment are typically emphasized. However, there still exist some open issues that are yet to be addressed in the area of learning for distributed spectrum access in CR networks. For example, the existing works mainly focused on situations with homogeneous SUs and uniform MAC protocols, which makes it possible to achieve convergence to NEs autonomously by using the standard SLA [56] algorithm. Thus, learning algorithm design for more general situations with heterogeneous SUs needs to be developed. In addition, with the development of advanced machine learning techniques and the increase of demand on adaptive and self-organized

network control in wireless networks, learning techniques are expected to be applied to more and more wireless networks. Actually, learning becomes necessary in scenarios where i) the network evolution mathematical model can not be built accurately due to the unknown or complicated environment dynamics, or ii) each user has to optimize its own behaviour without the accurate behaviour model of other users due to the unaffordable overhead of information exchange. Since different networks have different characteristics, the existing learning algorithms for CR networks may not be applicable to other different networks. Thus, specialized learning algorithm designs for other different wireless networks need to be developed.

Motivated by all above, in this thesis, learning applications are investigated in three different networks: an OSA network with multiple heterogeneous SUs and primary channels, a cloudlet-based MCC network with multi-mobile-device computation offloading in a multi-channel wireless contention environment, and an energy harvesting based network comprising a single MBS with grid power supply and multiple small-cells with renewable energy supply. The main objective of this thesis is to design effective and efficient learning algorithms for the three wireless networks. These learning designs are based on the distinct characteristics of the corresponding networks.

1.2.1 An OSA network

In CR networks, autonomous learning algorithms are desired in order to enable all CR nodes to learn on their own the unknown RF environment and to optimize their behaviours such as spectrum access independently and adaptively. To achieve

this, SLA approach has been widely applied. For example, in [57], Zandi *et al.* proposed a distributed adaptive learning and access policy for SUs, where the channel selection and access could effectively adapt to a wide range of traffic load patterns in the primary network, and could converge towards a pure strategy NE without any information exchange. In [58], Xu *et al.* studied the problem of multi-user sequential channel sensing and access in dynamic cognitive radio networks, and proposed a distributed stochastic learning algorithm to cope with the uncertain, dynamic, and incomplete information constraints. Both [57] and [58] focused on the scenario where the number of primary channels was no less than the number of SUs. Such scenario leaded to a deterministic transmission, i.e., the transmission probability for each SU was one. In [59–62], the scenario where the number of SUs was larger than the number of primary channels was considered. Wu *et al.* in [59] studied the problem of distributed channel selection for interference mitigation in a time-varying radio environment without information exchange. Zheng *et al.* in [60] investigated a more general and practical system model where the users' activities were dynamically variable, and designed a low-complexity fully distributed no-regret learning algorithm for channel adaptation. Nevertheless, these studies focused on minimizing the experienced interference in the physical layer only, and ignored the consideration of multiple access control mechanisms at higher layers. In [61, 62], authors employed the uniform MAC protocol to coordinate transmissions among SUs that tried to access the same idle channel.

Unlike all these existing works that mainly focused on homogeneous SUs and uniform MAC protocols, a more general scenario with heterogeneous SUs is considered

here. Specifically, in the OSA network, distributed channel selection is considered by jointly considering the following aspects: (1) the availability of spectrum holes are time-varying and its statistics are unknown to SUs; (2) there is no central controller and no information exchange among SUs; (3) each SU is allowed to opportunistically occupy the idle channel with a probability, and in general, such probabilities for different SUs are different, depending on their individual service requirements; and (4) the interests of SUs are conflicting since each SU tries to selfishly maximize its own expected throughput. I am trying to answer the fundamental questions that (1) without channel statistics, how could a SU choose a channel with high availability to improve its chance for access? (2) without a central controller and any information exchange, how could a SU effectively resolve collisions and achieve self-coordination for channel access?

1.2.2 A Cloudlet-based MCC network

Mobile devices, such as smartphones, tablets, and notebooks, are increasingly penetrating our daily lives as convenient tools for communication, entertainment, business, social networking, etc. With such enormous popularity, users expect to run desktop-level applications, such as interactive gaming, virtual reality, and natural language processing, on mobile devices [63, 64]. However, these emerging mobile applications typically require intensive computation and high energy consumption [65, 66], which does not match the limited computation capabilities and battery power on mobile devices. To address these challenges, a new architecture, known as MCC [67], was emerged as a promising approach to broaden the capability of mobile devices by mi-

grating part or all of computational tasks from mobile devices to infrastructure-based cloud servers, which ordinarily have powerful computing capability, high computation power, and huge storage.

However, since the infrastructure-based cloud servers are located centrally in the core network and ordinarily far away from the mobile users/devices, some major obstacles exist that limit an effective deployment of MCC strategies: i) the extra transmission energy cost at mobile devices, and ii) the long latency experienced through a wide area network (WAN). In fact, in macro-cellular networks, the transmit power necessary for cell edge users to access a remote cloud might null all potential benefits coming from offloading [68]. Moreover, for latency-sensitive mobile applications, such as augmented reality, online games, and speech recognition, the user Quality of Experience (QoE) may be greatly degraded due to high latency introduced by WAN. To tackle these challenges, the cloudlet-based MCC network was proposed recently to better support latency-sensitive and resource-intensive mobile applications [69, 70]. Specifically, in the cloudlet-based MCC framework, the mobile devices can speed up their mobile application executions by offloading their computation tasks to the nearby resource-rich computing servers/clusters via one-hop wireless access, rather than relying on a remote cloud [71].

The cloudlet-based MCC networks have attracted significant attention in recent years. Many existing works (e.g. [72–74]) optimized offloading strategies in a single-mobile-device scenario, while only a few works (e.g. [68, 75, 76]) addressed the multi-mobile-device computation offloading problem. Moreover, under the setting of multiple mobile devices, most existing works adopted centralized optimization for managing

both radio and computing resource allocations. However, since the complete information is required for centralized optimization, all devices need to report their local information to the central controller, which results in very high network overhead. In addition, the centralized optimization problem is commonly complex, and leads to some kind of heuristic solutions. What's more, since the mobile devices are owned by different individuals who may act according to their own interests, they may not have the incentive to follow the centralized optimal solution. All of these make the design of an efficient distributed multi-mobile-device computation offloading mechanism imperative.

Motivated by the aforementioned discussions, the design of an efficient distributed multi-mobile-device computation offloading algorithm for a cloudlet-based MCC network is studied. In such a network, each mobile device can make its decision independently, based on the execution cost in terms of both energy consumption and time cost. However, under multi-channel setting, it is critical to achieve efficient wireless access coordination among multiple mobile devices. Otherwise, if too many mobile devices choose the same wireless channel for offloading simultaneously, severe collisions may happen and the achievable average transmission rate for each mobile device can be significantly reduced. This results in very low energy efficiency and very long transmission time. Therefore, to achieve an efficient computation offloading mechanism in a distributed cloudlet-based MCC network, two key challenges need to be carefully tackled: 1) How does each mobile device make a decision independently between the local computing and the cloud computing? 2) How does a mobile device choose a proper channel in order to achieve a high transmission rate without

information exchange with other devices?

1.2.3 An Energy Harvesting based network

Resulted from rapid growth of wireless multimedia traffic, enormous energy consumption of wireless communication networks has become one of the major concerns for future 5G networks. At the same time, providing stable grid power supply may be very costly or even infeasible in some remote or hazardous locations [77], and grid-tied servers may violate environmental quality regulations in rural areas that are ecologically sensitive [78, 79]. In view of the soaring electricity prices as well as the significant carbon footprint of grid power, off-grid renewable energy, harvested from ambient vibrations, heat, wind, or solar radiation, has been embraced as a major or even the sole power supply for some small networks, thanks to the recent advance in energy harvesting techniques [80, 81].

Here, an energy harvesting based network consisting of a single MBS with grid power supply and multiple small-cells with renewable energy supply is considered. When an user request is denied by its SBS, it needs to be served by the MBS instead, which typically consumes more energy and also increases CO2 emissions. Thus, for both energy conservation and environmental protection, it is necessary to maximize the service ratio at the SBSs. However, due to the randomness of the renewable energy generation and the limitation on the battery capacity, energy shortage or waste will occur at the SBSs when the traffic patten mismatches the energy harvesting process, which makes it challenging to fully reap the benefits of energy harvesting.

In the literature, there have been extensive studies on this mismatching issue

[82–84]. However, in most of these works, the traffic pattern was adjusted through postponing the early demands, which may degrade their delay performance. To avoid such performance degradation, proactive caching was introduced in energy harvesting based networks [85]. Specifically, when the harvested energy is sufficient, contents can be proactively pushed to users before their actual demands, so that the energy waste due to the battery overflow can be avoided. Later, the content requests can be satisfied by users' local storage. Obviously, the benefits of proactive caching are i) transferring the harvested energy into the future and ii) enhancing user experience through achieving zero delay.

In past years, proactive caching has received an increasing amount of attention [86–91]. However, most of these existing works considered stable grid power supply and focused on designing low-rate transmission strategy for power saving, which can not be directly applied to energy harvesting based networks. To apply proactive caching in energy harvesting based networks, the following issues have to be addressed: i) both the random energy generation and the limited battery capacity have to be considered, ii) proactive caching decisions should jointly consider the content popularity distribution and the dynamic battery status, and iii) the objective is to match random energy arrivals with random user requests over time.

Motivated by all above, the proactive caching design in the energy harvesting based network is investigated. However, designing an effective proactive caching policy is very challenging due to the following reasons:

- In order to match random energy arrivals with random user requests for fully reaping the benefits of energy harvesting, the stochastic models for both random

processes should be known *a priori*. However, due to the high unpredictability of renewable energy, it is difficult, if not impossible, to obtain the stochastic model of energy harvesting process.

- To make proactive caching decisions, the battery states under which the proactive caching action should be taken need to be determined, and which contents should be proactively cached at which users also has to be determined. Unfortunately, without the stochastic model of energy harvesting process, it is difficult to make efficient proactive caching decision based on the instantaneous battery status only. Besides, since contents may only be interesting to users for a finite period of time, and the content popularity distribution may change over time, it is also difficult to make efficient proactive caching decisions on which contents should be proactively cached at which users according to the instantaneous content popularity distribution only. Moreover, different users may have different content preferences based on their own contexts, e.g., age, gender, and personality, which further complicates proactive caching by increasing both the state and action dimensionalities significantly.

- Since complete stochastic information about network dynamics is not available, offline optimization algorithms (such as DP [14]) become infeasible, and online approaches are preferred. However, due to the "curse of dimensionality" (the joint system state including battery status, content life span, content popularity distribution at each user, and the content lists cached at each user), the standard tabular based online learning algorithms (such as R-learning [92, 93]) become inapplicable.

Therefore, designing an online learning algorithm which learns the optimal context-aware proactive caching policy on-the-fly with a high learning efficiency is important.

1.3 Summary of Contributions

The main contributions are summarized as follows.

1) Distributed opportunistic spectrum access in an unknown and dynamic environment: A stochastic learning approach (this work has been published in the IEEE Transactions on Vehicular Technology).

- The distributed throughput maximization problem in the OSA network is first formulated as a noncooperative game. Then, the game is proved to be an ordinal potential game [94] by carefully constructing an ordinal potential function. According to the property of the ordinal potential game, the formulated game has at least one pure-strategy NE.

- To derive the NEs, a Best Response (BR) [95–97] based algorithm is first proposed, which can guarantee the convergence towards NEs within finite time, based on the assumptions that there exists a coordinator for SUs to work in a round-robin fashion and a common control channel for SUs to exchange their information.

- SLA is then introduced into the formulated game to adapt decision-making in an unknown and dynamic environment. To investigate the convergence property of the SLA in ordinal potential games, which has not been addressed in literature, a weighted potential game is defined, which is proved to have the same NE set with

the formulated ordinal potential game. After that, a new SLA based algorithm is proposed, called N-SLA, to achieve the NEs of the weighted potential game, or equivalently the NEs of the formulated ordinal potential game.

- Simulation results are provided to demonstrate the convergence of both the BR based and the SLA based algorithms, and illustrate the efficiency of them in terms of sum log expected throughput.

2) Distributed multi-mobile-device computation offloading for cloudlet-based mobile cloud computing: A game-theoretic machine learning approach (this work has been published in the IEEE Transactions on Vehicular Technology).

- The multi-mobile-device computation offloading decision making problem is formulated as a noncooperative game, which takes into account both communication and computation costs of computation offloading in the cloudlet-based MCC network. By carefully constructing an exact potential function, the formulated game is then proved to be an exact potential game [98]. According to the property of the exact potential game, the formulated game has at least one pure-strategy NE.

- To achieve the NEs in a fully distributed way, the machine learning technology is introduced to adapt each mobile device's decision-making in an unknown and dynamic environment. An FDCO algorithm is proposed, which can converge towards a NE without any information exchange among mobile devices.

- The NE solutions achieved by the proposed FDCO algorithm are further quantified in terms of the number of beneficial cloudlet computing mobile devices

and the network-wide execution cost. The proposed FDCO algorithm is analytically shown to maximize the number of beneficial cloudlet computing mobile devices without incurring higher network-wide execution cost than computing locally by all mobile devices.

3) Learning based online context-aware proactive caching for an energy harvesting based network (this work has been submitted to the IEEE Transactions on Wireless Communications).

- A context-aware proactive caching problem in an energy harvesting based network is presented. By taking into account various unique aspects of the considered network, the original problem is formulated as an MDP framework.

- To conquer both incomplete stochastic information and "curse of dimensionality" in the formulated MDP, an PDS-ARL algorithm is proposed, which can learn the optimal context-ware proactive caching policy on-the-fly and achieve a remarkable improvement on the learning rate by reducing both the state and action dimensionalities significantly.

- Extensive simulations are carried out to validate the effectiveness of the proposed algorithm. By comparing with the standard R-learning algorithm and the non-proactive strategy, the proposed algorithm shows significant performance improvement in terms of the learning rate and the service ratio at SBSs.

1.4 Organization of the Thesis

The rest of the thesis is organized as follows. In Chapter 2, the problem of distributed spectrum access in an OSA network is considered and an SLA based approach is investigated. In Chapter 3, the problem of distributed multi-mobile-device computation offloading for cloudlet-based MCC is considered and a game-theoretic machine learning approach is investigated. In Chapter 4, the problem of online context-aware proactive caching for an energy harvesting based network is considered and an RL based algorithm is investigated, followed by conclusions and future work in Chapter 5.

Chapter 2

Distributed Opportunistic Spectrum Access in an Unknown and Dynamic Environment: A Stochastic Learning Approach

In this chapter, the problem of distributed throughput maximization in an OSA network with multiple heterogeneous SUs and primary channels is investigated. This problem is first formulated as a noncooperative game, which is further proved to be an ordinal potential game. Then, a BR based algorithm is proposed to achieve the NEs of the formulated game, given that there exists a coordinator for SUs to work in a round-robin fashion and a common control channel for SUs to exchange their information. To further relieve the network overhead due to information exchange among SUs, a N-SLA algorithm is designed, which can converge to a NE of the formulated ordinal

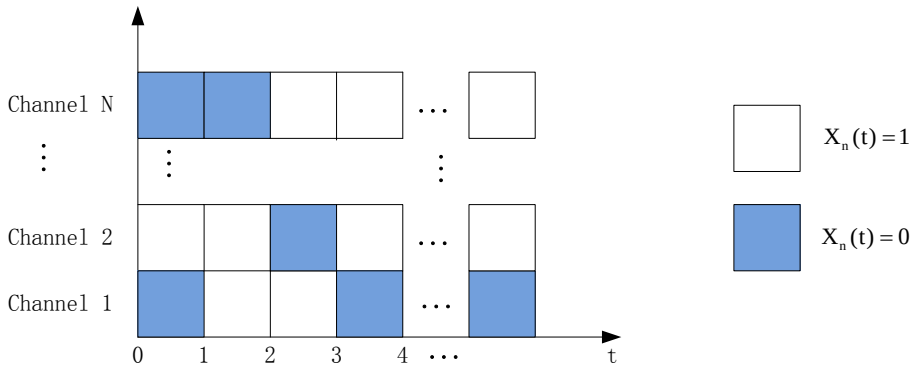

Figure 2.1: System Architecture.

potential game in a fully distributed way. To my best knowledge, I'm the first to address the convergence issue of the SLA based algorithms for ordinal potential games. Simulation results validate the effectiveness of the proposed algorithms.

2.1 System Model and Game Formulation

2.1.1 System Model

Consider a distributed OSA network consisting of M SUs (or equivalently, M pairs of secondary transceivers) and N independent primary channels, owned by N PUs. Time is divided into slots with an equal length. Let $X_n(t)$ denote the availability status of channel n at time slot t. $X_n(t) = 1$ means channel n is available and $X_n(t) = 0$ otherwise, as shown in Fig. 2.1. Assume that $X_n(t)$, $n = 1, \cdots, N$, follows a stationary random process over t, with the mean $\theta_n = \mathbb{E}[X_n(t)] \in [0, 1]$ [57, 58].

For explanation purpose, a case where all SUs locate in a small-scale mutually interfering area [57, 58] is considered. At the beginning of time slot t, each SU selects

one channel for sensing. Based on the sensing outcome [1], each SU makes a decision on access. If the selected channel n ($n = 1, \cdots, N$) is sensed to be idle at SU m ($m = 1, \cdots, M$), it will access this channel with a transmission probability P_m ($0 < P_m \leq 1$). Otherwise, it will keep silent in the current slot. A transmission is successful only if there is a single SU in transmission in the given channel. Otherwise, a collision occurs. The collision-free achievable rate of SU m ($m = 1, \cdots, M$) on each channel can be calculated as

$$r_m^f(t) = \log_2(1 + \frac{p_m d_m^{-\alpha} |\xi_m(t)|^2}{\sigma^2}) \tag{2.1}$$

(2.1) is obtained based on Shannon theory, where p_m is the transmission power of SU m, σ^2 is the noise power, d_m is the distance between the transmitter and the receiver of SU m, α is the path loss factor, and $|\xi_m(t)|^2$ is an exponentially distributed random fading coefficient with unit mean. Note that the channel bandwidth of each primary channel has been normalized to be 1 for simplicity.

2.1.2 Game Formulation

Define a_m as the channel selection action of SU m, $\boldsymbol{a}_{-m}$ as the set of channel selection actions of all SUs except SU m, i.e., $\boldsymbol{a}_{-m} = \{a_1, \cdots, a_{m-1}, a_{m+1}, \cdots, a_M\}$, and $\mathcal{U}_n$ as the set of SUs who select the channel n ($n = 1 \cdots, N$), i.e., $\mathcal{U}_n = \{m \in \{1, \cdots, M\} : a_m = n\}$. Then, the achievable throughput of SU m in time slot t can be written as:

$$r_m(a_m, \boldsymbol{a}_{-m}, t) = X_{a_m}(t) I_m(\mathcal{U}_{a_m}, t) r_m^f(t) \tag{2.2}$$

[1] For simplicity, it is assumed that the channel sensing is perfect. However, the analysis in this chapter can easily be extended to the scenario with imperfect channel sensing by introducing the detection probability P_d and the false alarm probability P_f.

where $I_m(\mathcal{U}_{a_m}, t) = 1$ if only SU m from $\mathcal{U}_{a_m}$ transmits over the channel a_m in time slot t, and 0 otherwise.

Hence, the expected throughput of SU m is given by

$$\begin{aligned} R_m(a_m, \boldsymbol{a}_{-m}) &= \mathbb{E}[r_m(a_m, \boldsymbol{a}_{-m}, t)] \\ &= \mathbb{E}[X_{a_m}(t)]\mathbb{E}[I_m(\mathcal{U}_{a_m}, t)]\mathbb{E}[r_m^f(t)] \\ &= \theta_{a_m} P_m \prod_{l \in \mathcal{U}_{a_m}, l \neq m} (1 - P_l)\bar{r}_m^f \end{aligned} \tag{2.3}$$

Here, a distributed throughput maximization problem is considered, where each SU tries to maximize its own expected normalized rate without a central controller:

$$\max_{a_m = 1, \cdots, N} \hat{R}_m(a_m, \boldsymbol{a}_{-m}), \qquad \forall m = 1, \cdots, M \tag{2.4}$$

where

$$\begin{aligned} \hat{R}_m(a_m, \boldsymbol{a}_{-m}) &= \frac{R_m(a_m, \boldsymbol{a}_{-m})}{\bar{r}_m^f} \\ &= \theta_{a_m} P_m \prod_{l \in \mathcal{U}_{a_m}, l \neq m} (1 - P_l) \end{aligned} \tag{2.5}$$

According to problem (2.4), each SU m tries to maximize its own utility function $\hat{R}_m$, which complies with the property of noncooperative games. Thus, to solve this distributed throughput maximization problem, a noncooperative game denoted by $\mathcal{G}_{OSA} = \{\mathcal{M}, \mathcal{N}, \{\hat{R}_m(a_m, \boldsymbol{a}_{-m})\}_{m \in \mathcal{M}}\}$ is formulated, where $\mathcal{M} = \{1, \cdots, M\}$ is the set of players (or SUs), $\mathcal{N} = \{1, \cdots, N\}$ is the set of actions (or primary channels) that each player can take, and $\hat{R}_m(a_m, \boldsymbol{a}_{-m})$ is the utility of the player m ($m = \{1, \cdots, M\}$) upon taking action $a_m \in \mathcal{N}$ while other players taking $\boldsymbol{a}_{-m}$. Each player independently and selfishly adjusts its strategy to maximize its individual utility $\hat{R}_m(a_m, \boldsymbol{a}_{-m})$.

Definition 2.1. A channel selection profile $\boldsymbol{a}^* = (a_1^*, \cdots, a_M^*)$ is a pure strategy NE of $\mathcal{G}_{OSA}$ if and only if no SU can improve its utility function by deviating unilaterally, i.e.,

$$\hat{R}_m(a_m^*, \boldsymbol{a}_{-m}^*) \geq \hat{R}_m(a_m, \boldsymbol{a}_{-m}^*), \quad \forall m \in \mathcal{M}, \quad \forall a_m \in \mathcal{N} \tag{2.6}$$

2.2 Properties of NEs in $\mathcal{G}_{OSA}$

In this section, the properties of the NEs in $\mathcal{G}_{OSA}$ are investigated. The formulated game $\mathcal{G}_{OSA}$ will be proved to be an ordinal potential game [94].

Definition 2.2. A game is called an ordinal potential game if the incentives of all players of the game for changing their actions can be reflected by a function $\Phi : \boldsymbol{a} = (a_1, \cdots, a_M) \to \mathbb{R}$, called an ordinal potential function, i.e.,

$$\begin{aligned} &\hat{R}_m(a_m, \boldsymbol{a}_{-m}) - \hat{R}_m(a'_m, \boldsymbol{a}_{-m}) > 0 \Longleftrightarrow \Phi(a_m, \boldsymbol{a}_{-m}) - \Phi(a'_m, \boldsymbol{a}_{-m}) > 0, \\ &\forall m \in \mathcal{M}, \quad \forall a_m, a'_m \in \mathcal{N}, a_m \neq a'_m \end{aligned} \tag{2.7}$$

where " $\Longleftrightarrow$" means "equivalent to".

In general, the proof of the existence of an ordinal potential function in a game is sufficient to prove the game being an ordinal potential game.

Theorem 2.1. *The formulated noncooperative game $\mathcal{G}_{OSA}$ is an ordinal potential game.*

Proof. To prove the theorem, two scenarios are considered separately: (I) the number of channels N is no less than the number of SUs M, i.e., $N \geq M$, and (II) $N < M$.

In scenario (I), since the spectrum resource is redundant, each SU can transmit with probability 1. Thus, effective orthogonalization mechanism to avoid collisions among SUs is crucial. In scenario (II), each SU m $(\forall m \in \mathcal{M})$ transmits with a probability P_m, where $0 < P_m < 1$, to share the limited spectrum with other SUs.

Scenario (I): With $P_m = 1$, $\hat{R}_m(a_m, \boldsymbol{a}_{-m})$ $(\forall m \in \mathcal{M})$ can be rewritten as

$$\hat{R}_m(a_m, \boldsymbol{a}_{-m}) = \upsilon_{a_m}(|\mathcal{U}_{a_m}|) = \begin{cases} \theta_{a_m}, & |\mathcal{U}_{a_m}| = 1 \\ 0, & \text{otherwise} \end{cases} \tag{2.8}$$

where $|\mathcal{U}_{a_m}|$ denotes the number of SUs taking the same action a_m.

Then, a bounded function $\Phi_1 : \boldsymbol{a} = (a_1, \cdots, a_M) \rightarrow \mathbb{R}$ can be defined as follows,

$$\Phi_1(a_m, \boldsymbol{a}_{-m}) = \sum_{n=1}^{N} \sum_{k=1}^{|\mathcal{U}_n|} \upsilon_n(k) \tag{2.9}$$

Note that equation (2.9) follows the same form of the Rosenthal's potential function [99]. Following similar proof given in [57],

$$\begin{aligned} &\Phi_1(a'_m, \boldsymbol{a}_{-m}) - \Phi_1(a_m, \boldsymbol{a}_{-m}) = \hat{R}_m(a'_m, \boldsymbol{a}_{-m}) - \hat{R}_m(a_m, \boldsymbol{a}_{-m}), \\ &\forall m \in \mathcal{M}, \forall a_m, a'_m \in \mathcal{N}, a_m \neq a'_m \end{aligned} \tag{2.10}$$

From (2.10), it is obvious that the property in (2.7) holds. Therefore, the function $\Phi_1 : \boldsymbol{a} = (a_1, \cdots, a_M) \rightarrow \mathbb{R}$ is an ordinal potential function so that the game $\mathcal{G}_{OSA}$ with $P_m = 1$ $(\forall m \in \mathcal{M})$ is an ordinal potential game. Moreover, since the deviation in the utility of an arbitrary player m is exactly reflected by the deviation in the ordinal potential function, the game $\mathcal{G}_{OSA}$ with $P_m = 1$ $(\forall m \in \mathcal{M})$ is also an exact potential game [98].

Scenario (II): With $0 < P_m < 1$ $(\forall m \in \mathcal{M})$, a bounded function $\Phi_2 : \boldsymbol{a} =$

$(a_1, \cdots, a_M) \to \mathbb{R}$ is defined as:

$$\Phi_2(a_m, \boldsymbol{a}_{-m}) = -\sum_{m\in\mathcal{M}} \beta_m\Big(\sum_{l:a_l=a_m} \beta_l + \beta_m + 2y_m^{a_m}\Big) \tag{2.11}$$

where

$$\beta_m = \log_2(1 - P_m), \tag{2.12}$$

$$y_m^{a_m} = \log_2\Big(\frac{\theta_{a_m} P_m}{1 - P_m}\Big) \tag{2.13}$$

Suppose that an arbitrary SU m ($m \in \mathcal{M}$) unilaterally changes its action from a_m to a'_m ($a_m, a'_m \in \mathcal{N}, a_m \neq a'_m$). According to (2.11),

$$\begin{aligned}\Phi_2(a_m, \boldsymbol{a}_{-m}) = -[&\beta_m\Big(\sum_{l\in\mathcal{U}_{a_m}|(a_m,\boldsymbol{a}_{-m})} \beta_l + \beta_m + 2y_m^{a_m}\Big) + \sum_{p\in\mathcal{U}_{a_m}|(a_m,\boldsymbol{a}_{-m}),p\neq m} \beta_p(\beta_m + \\ &\sum_{l\in\mathcal{U}_{a_p}|(a_m,\boldsymbol{a}_{-m}),l\neq m} \beta_l + \beta_p + 2y_p^{a_p}) + \sum_{q\in\mathcal{U}_{a'_m}|(a_m,\boldsymbol{a}_{-m})} \beta_q\Big(\sum_{l\in\mathcal{U}_{a_q}|(a_m,\boldsymbol{a}_{-m})} \beta_l + \beta_q + 2y_q^{a_q}\Big) + \\ &\sum_{e:a_e\neq a_m,a'_m} \beta_e\Big(\sum_{l\in\mathcal{U}_{a_e}|(a_m,\boldsymbol{a}_{-m})} \beta_l + \beta_e + 2y_e^{a_e}\Big)]\end{aligned} \tag{2.14}$$

and

$$\begin{aligned}\Phi_2(a'_m, \boldsymbol{a}_{-m}) = -[&\beta_m\Big(\sum_{l\in\mathcal{U}_{a'_m}|(a'_m,\boldsymbol{a}_{-m})} \beta_l + \beta_m + 2y_m^{a'_m}\Big) + \sum_{q\in\mathcal{U}_{a'_m}|(a'_m,\boldsymbol{a}_{-m}),q\neq m} \beta_q(\beta_m + \\ &\sum_{l\in\mathcal{U}_{a_q}|(a'_m,\boldsymbol{a}_{-m}),l\neq m} \beta_l + \beta_q + 2y_q^{a_q}) + \sum_{p\in\mathcal{U}_{a_m}|(a'_m,\boldsymbol{a}_{-m})} \beta_p\Big(\sum_{l\in\mathcal{U}_{a_p}|(a'_m,\boldsymbol{a}_{-m})} \beta_l + \beta_p + 2y_p^{a_p}\Big) \\ &+ \sum_{e:a_e\neq a_m,a'_m} \beta_e\Big(\sum_{l\in\mathcal{U}_{a_e}|(a'_m,\boldsymbol{a}_{-m})} \beta_l + \beta_e + 2y_e^{a_e}\Big)]\end{aligned} \tag{2.15}$$

Then,

$$\begin{aligned}&\Phi_2(a'_m, \boldsymbol{a}_{-m}) - \Phi_2(a_m, \boldsymbol{a}_{-m}) \\ &= \beta_m\Big(\sum_{l\in\mathcal{U}_{a_m}|(a_m,\boldsymbol{a}_{-m})} \beta_l + \beta_m + 2y_m^{a_m}\Big) - \beta_m\Big(\sum_{l\in\mathcal{U}_{a'_m}|(a'_m,\boldsymbol{a}_{-m})} \beta_l + \beta_m + 2y_m^{a'_m}\Big) + \beta_m\end{aligned}$$

$$\sum_{p\in\mathcal{U}_{a_m}|(a_m,\boldsymbol{a}_{-m}),p\neq m}\beta_p-\beta_m\sum_{q\in\mathcal{U}_{a'_m}|(a'_m,\boldsymbol{a}_{-m}),q\neq m}\beta_q$$

$$=2\beta_m(\sum_{l\in\mathcal{U}_{a_m}|(a_m,\boldsymbol{a}_{-m})}\beta_l+y_m^{a_m}-\sum_{l\in\mathcal{U}_{a'_m}|(a'_m,\boldsymbol{a}_{-m})}\beta_l-y_m^{a'_m}) \tag{2.16}$$

Define

$$\tilde{R}_m(a_m,\boldsymbol{a}_{-m})=\log_2(\hat{R}_m(a_m,\boldsymbol{a}_{-m})),\quad \forall m\in\mathcal{M},a_m\in\mathcal{N} \tag{2.17}$$

Substituting (2.5) into (2.17),

$$\begin{aligned}&\tilde{R}_m(a_m,\boldsymbol{a}_{-m})\\&=\log_2(\theta_{a_m}P_m\prod_{l\in\mathcal{U}_{a_m}|(a_m,\boldsymbol{a}_{-m}),l\neq m}(1-P_l))\\&=\log_2(\frac{\theta_{a_m}P_m}{1-P_m})+\sum_{l\in\mathcal{U}_{a_m}|(a_m,\boldsymbol{a}_{-m})}\log_2(1-P_l)\end{aligned} \tag{2.18}$$

According to (2.12) and (2.13),

$$\tilde{R}_m(a_m,\boldsymbol{a}_{-m})=y_m^{a_m}+\sum_{l\in\mathcal{U}_{a_m}|(a_m,\boldsymbol{a}_{-m})}\beta_l \tag{2.19}$$

Substituting (2.19) into (2.16),

$$\begin{aligned}&\Phi_2(a'_m,\boldsymbol{a}_{-m})-\Phi_2(a_m,\boldsymbol{a}_{-m})\\&=2\beta_m(\tilde{R}_m(a_m,\boldsymbol{a}_{-m})-\tilde{R}_m(a'_m,\boldsymbol{a}_{-m}))\\&=2\beta_m(\log_2(\hat{R}_m(a_m,\boldsymbol{a}_{-m}))-\log_2(\hat{R}_m(a'_m,\boldsymbol{a}_{-m})))\end{aligned} \tag{2.20}$$

Since $\beta_m=\log_2(1-P_m)<0$,

$$\hat{R}_m(a_m,\boldsymbol{a}_{-m})-\hat{R}_m(a'_m,\boldsymbol{a}_{-m})>0\Longleftrightarrow\Phi_2(a_m,\boldsymbol{a}_{-m})-\Phi_2(a'_m,\boldsymbol{a}_{-m})>0,$$

$$\forall m\in\mathcal{M},\quad \forall a_m,a'_m\in\mathcal{N},a_m\neq a'_m,0<P_m<1. \tag{2.21}$$

Therefore, the function $\Phi_2 : \boldsymbol{a} = (a_1, \cdots, a_M) \rightarrow \mathbb{R}$ is also an ordinal potential function and the game $\mathcal{G}_{OSA}$ with $0 < P_m < 1$, $\forall m \in \mathcal{M}$, is an ordinal potential game. □

From [98], it can be concluded that there exists at least one pure strategy NE in the formulated noncooperative game $\mathcal{G}_{OSA}$. However, deriving such NEs is not straightforward, especially for scenario (II). In next sections, how to achieve the pure strategy NE of $\mathcal{G}_{OSA}$ will be investigated.

2.3 BR based Algorithm

In this section, a BR based algorithm is proposed to find the pure-strategy NEs of $\mathcal{G}_{OSA}$, by assuming that there exists a coordinator for SUs to work in a round-robin fashion and a common control channel for SUs to broadcast their individual information, e.g., the updated actions and the transmission probabilities.

The proposed BR based algorithm is divided into two stages: in stage (1), each SU distributively learns the channel availability statistics $(\theta_1, \theta_2, \cdots, \theta_N)$, by adopting such as the upper-confidence-bound 1 (UCB1) algorithm in [100–102]; Based on these results, in stage (2), SUs select primary channels one by one, and in each round, one SU chooses the best response to the strategies of SUs who have already made decisions beforehand. Specifically, given $\mathcal{U}_n(t)$ $(\forall n \in \mathcal{N})$ in time slot t, the SU chooses the best channel $n^*(t)$ satisfying

$$n^*(t) = \arg_{n \in \mathcal{N}} \max[\omega_n(\mathcal{U}_n(t))] \tag{2.22}$$

where

$$\omega_n(\mathcal{U}_n(t)) = \begin{cases} \upsilon_n(|\mathcal{U}_n(t)|), & \text{if } m \in \mathcal{U}_n(t) \\ \upsilon_n(|\mathcal{U}_n(t)|+1), & \text{otherwise} \end{cases} \tag{2.23}$$

in scenario (I), or chooses $n^*(t)$ satisfying

$$n^*(t) = \arg_{n\in\mathcal{N}} \max[\theta_n P_m \kappa_n(\mathcal{U}_n(t))] \tag{2.24}$$

where

$$\kappa_n(\mathcal{U}_n(t)) = \begin{cases} 1, & \text{if } \mathcal{U}_n(t) = \{m\} \text{ or } \mathcal{U}_n(t) = \emptyset \\ \prod\limits_{l\in\mathcal{U}_n(t), l\neq m} (1-P_l), & \text{otherwise} \end{cases} \tag{2.25}$$

in scenario (II). The proposed BR based algorithm is summarized in Algorithm 1.

Theorem 2.2. *The proposed BR based algorithm converges to a pure-strategy NE of $\mathcal{G}_{OSA}$, starting from any point.*

The proof of Theorem 2.2 is shown in Appendix A.1.

Remark 2.1. *In the proposed BR based algorithm, the information exchange among SUs is indispensable, which may result in high network overhead, and may not be feasible in some practical communication networks. Therefore, designing a fully distributed online-adaptive algorithm is required, so that each SU can independently and adaptively adjust its own strategies based on its individual experienced action-reward without the coordinator or any information exchange.*

2.4 SLA based algorithm

In this section, a fully distributed algorithm based on SLA is proposed and its convergence to the pure-strategy NEs of $\mathcal{G}_{OSA}$ in an unknown and dynamic environment

Algorithm 1: The Best Response based Algorithm

Stage (1): Each SU estimates the channel availability statistics $(\theta_1, \theta_2, \cdots, \theta_N)$.
Stage (2):
Initialize $t = 0$, $a_m(0) = 0, \forall m \in \mathcal{M}$, and $\mathcal{U}_n(t) = \emptyset$, $\forall n \in \mathcal{N}$
Repeat:
Set $\mathcal{M}_1 = \mathcal{M}$, $\mathcal{M}_2 = \emptyset$.
for $t = 0, 1, \cdots,$ **do**
  While $\mathcal{M}_1 \neq \emptyset$,
  The coordinator randomly selects a SU $m \in \mathcal{M}_1$.
    if in scenario (I), i.e., $P_m = 1, \forall m \in \mathcal{M}$,
    the SU chooses $n^*(t)$ according to (2.22).
    else if in scenario (II), i.e., $0 < P_m < 1$, $\forall m \in \mathcal{M}$,
  the SU chooses $n^*(t)$ according to (2.24).
    end if
  The SU m broadcasts the selected channel $n^*(t)$ through the common control channel.
  Each SU updates $\{\mathcal{U}_n(t)\}_{n\in\mathcal{N}}$ according to the following rule:

$$\mathcal{U}_n(t+1) = \begin{cases} \mathcal{U}_n(t) \backslash m & \text{if } n = a_m(t-1), n \neq a_m(t) \\ \mathcal{U}_n(t) \bigcup \{m\} & \text{if } n = a_m(t), n \neq a_m(t-1) \\ \mathcal{U}_n(t) & \text{otherwise} \end{cases}$$

  The SU m tunes to $n^*(t)$ for sensing, and transmits with probability P_m if available.
  Exclude m from $\mathcal{M}_1$ and include it in $\mathcal{M}_2$.
  Update $t = t + 1$.
  end While
Until convergence.

is investigated. Since the game $\mathcal{G}_{OSA}$ in scenario (I) is proved to be an exact potential game, the existing SLA based algorithms for achieving the pure-strategy NEs (e.g., [57] and [62]) can be applied directly. However, investigating the convergence property of the SLA based algorithms for the ordinal potential game in scenario (II) is challenging. To my best knowledge, there is no SLA based algorithm available in literature for ordinal potential games. Thus, in this section, the discussion is focused

on the scenario (II). A modified noncooperative game $\tilde{\mathcal{G}}_{OSA}$ will be first defined, which is shown to have the same NE set as the original game $\mathcal{G}_{OSA}$. Then, by proving $\tilde{\mathcal{G}}_{OSA}$ to be a weighted potential game, a new SLA (N-SLA) algorithm is proposed to achieve the pure-strategy NEs of $\tilde{\mathcal{G}}_{OSA}$, which equivalently proves the convergence towards the NEs of $\mathcal{G}_{OSA}$.

2.4.1 A Modified Noncooperative Game

A modified noncooperative game from $\mathcal{G}_{OSA}$ is defined as follows.

Definition 2.3. A modified noncooperative game is defined as $\tilde{\mathcal{G}}_{OSA} = \{\mathcal{M}, \mathcal{N}, \{U_m (a_m, \boldsymbol{a}_{-m})\}_{m\in\mathcal{M}}\}$, where $\mathcal{M}$ and $\mathcal{N}$ are the same as in the game $\mathcal{G}_{OSA}$. $U_m(a_m, \boldsymbol{a}_{-m}) = \log_2(\theta_{a_m} P_m \prod_{l\in\mathcal{U}_{a_m}, l\neq m} (1 - P_l)) + \varpi$ $(\forall m \in \mathcal{M})$, where $\varpi > 0$ is a predefined constant to guarantee the utility $U_m(a_m, \boldsymbol{a}_{-m})$ is nonnegative.

Proposition 2.1. The modified game $\tilde{\mathcal{G}}_{OSA}$ has the same NE set as the original game $\mathcal{G}_{OSA}$.

Proof. In the modified game $\tilde{\mathcal{G}}_{OSA}$, each player m $(m \in \mathcal{M})$ tries to maximize its utility $U_m(a_m, \boldsymbol{a}_{-m})$, i.e.,

$$\max_{a_m\in\mathcal{N}} U_m(a_m, \boldsymbol{a}_{-m}), \qquad \forall m \in \mathcal{M} \tag{2.26}$$

Using the monotonicity of the logarithm function,

$$\begin{aligned} a_m^* &= \arg\max_{a_m\in\mathcal{N}} U_m(a_m, \boldsymbol{a}_{-m}) \\ &= \arg\max_{a_m\in\mathcal{N}} \log_2(\theta_{a_m} P_m \prod_{l\in\mathcal{U}_{a_m}, l\neq m} (1 - P_l)) \end{aligned}$$

$$= \arg\max_{a_m \in \mathcal{N}} \theta_{a_m} P_m \prod_{l \in \mathcal{U}_{a_m}, l \neq m} (1 - P_l)$$

$$= \arg\max_{a_m \in \mathcal{N}} \hat{R}_m(a_m, \boldsymbol{a}_{-m}), \qquad \forall m \in \mathcal{M} \tag{2.27}$$

Thus, the optimization problems (2.4) and (2.26) have the same solution set, or in other words, the modified game $\tilde{\mathcal{G}}_{OSA}$ and the original game $\mathcal{G}_{OSA}$ have the same NE set. □

Proposition 2.2. The modified game $\tilde{\mathcal{G}}_{OSA}$ is a weighted potential game.

Proof. According to (2.20), for $\forall m \in \mathcal{M}$, and $\forall a_m, a'_m \in \mathcal{N}$, $a_m \neq a'_m$,

$$\begin{aligned}
&\Phi_2(a'_m, \boldsymbol{a}_{-m}) - \Phi_2(a_m, \boldsymbol{a}_{-m}) \\
&= 2\beta_m(\log_2(\hat{R}_m(a_m, \boldsymbol{a}_{-m})) - \log_2(\hat{R}_m(a'_m, \boldsymbol{a}_{-m}))) \\
&= 2\beta_m(\tilde{R}_m(a_m, \boldsymbol{a}_{-m}) + \varpi - \tilde{R}_m(a'_m, \boldsymbol{a}_{-m}) - \varpi) \\
&= -2\beta_m(U_m(a'_m, \boldsymbol{a}_{-m}) - U_m(a_m, \boldsymbol{a}_{-m}))
\end{aligned} \tag{2.28}$$

According to [94], the modified game $\tilde{\mathcal{G}}_{OSA}$ is a weighted potential game, where the weighted potential function is $\Phi_2 : \boldsymbol{a} = (a_1, \cdots, a_M) \rightarrow \mathbb{R}$ and the weight vector is $(-2\beta_1, \cdots, -2\beta_M)$. Since (2.28) complies with the property (2.7), the weighted potential game is also an ordinal potential game, so that there exists at least one pure-strategy NE in $\tilde{\mathcal{G}}_{OSA}$. □

In the next subsection, an N-SLA algorithm is proposed to find the pure-strategy NEs of the modified game $\tilde{\mathcal{G}}_{OSA}$, which are also the NEs of the original game $\mathcal{G}_{OSA}$.

2.4.2 N-SLA Algorithm

Algorithm Description

At first, define $\boldsymbol{P}_m(j) = (P_{m1}(j), P_{m2}(j), \cdots, P_{mN}(j))$ as the mixed strategy of SU m ($m \in \mathcal{M}$) in the jth iteration, where $P_{mn}(j)$ ($n \in \mathcal{N}$) is the probability for SU m to choose channel n in the jth iteration and $\sum_{n=1}^{N} P_{mn}(j) = 1, \forall j \geq 0, \forall m \in \mathcal{M}$. The key ideas of the proposed N-SLA algorithm are as follows: 1) each SU m ($m \in \mathcal{M}$) chooses one primary channel for sensing and access in each iteration according to the current mixed strategy $\boldsymbol{P}_m(j)$; 2) the mixed strategy $\boldsymbol{P}_m(j)$ of each SU m is updated based on $\bar{Rw}_m(a_m(j), \boldsymbol{a}_{-m}(j))$, which is defined as

$$\bar{Rw}_m(a_m(j), \boldsymbol{a}_{-m}(j)) = \frac{\log_2(\hat{R}_m(a_m(j), \boldsymbol{a}_{-m}(j))) + \varpi}{\varpi} \tag{2.29}$$

and converges to a pure-strategy NE at the end. The details of the proposed N-SLA algorithm is described in Algorithm 2.

Remark 2.2. *Note that in the proposed N-SLA algorithm, SUs do not need to compute $\hat{R}_m(a_m(j), \boldsymbol{a}_{-m}(j))$ according to (2.5), by requesting the information of both the channel statistics and other SUs including their channel selection strategies and their transmission probabilities. Instead, each SU m ($m \in \mathcal{M}$) can estimate $\hat{R}_m(a_m(j), \boldsymbol{a}_{-m}(j))$ by only measuring the successful access it achieves within T' slots on the selected channel $a_m(j)$. Since the mixed strategy of each SU is updated only based on the individual experienced action-reward, there is no need of a coordinator for managing the sequential access of SUs, and each SU independently and adaptively updates its strategy without any information exchange. Therefore, the proposed N-SLA algorithm is fully distributed.*

Algorithm 2: N-SLA Algorithm

Initialization: Set $j = 0$ and $P_{mn}(j) = (\frac{1}{N}, \frac{1}{N}, \cdots, \frac{1}{N})$, $\forall m \in \mathcal{M}$.
While there exists a $\boldsymbol{P}_m(j)$ ($m \in \mathcal{M}$), in which the maximum $P_{mn}(j)$ ($n \in \mathcal{N}$) is less than 0.99,
each SU m selects a channel $a_m(j)$ according to its current mixed strategy $\boldsymbol{P}_m(j)$.
  for $t = jT' : T' - 1 + jT'$,
  in time slot t, each SU m ($m \in \mathcal{M}$) performs channel sensing and channel contention on the selected channel $a_m(j)$. At the end of the tth slot, each SU m receives the random reward $\hat{r}_m(a_m(j), \boldsymbol{a}_{-m}(j), t)$ specified by

$$\hat{r}_m(a_m(j), \boldsymbol{a}_{-m}(j), t) = X_{a_m(j)}(t) I_m(\mathcal{U}_{a_m(j)}, t)$$

  end for
  for $m = 1 : M$,
  SU m estimates $\hat{R}_m(a_m(j), \boldsymbol{a}_{-m}(j))$ according to

$$\hat{R}_m(a_m(j), \boldsymbol{a}_{-m}(j)) = \hat{R}_m^{est}(a_m(j), \boldsymbol{a}_{-m}(j)) + \xi$$
$$\hat{R}_m^{est}(a_m(j), \boldsymbol{a}_{-m}(j)) = \frac{\sum_{t=jT'}^{T'-1+jT'} \hat{r}_m(a_m(j), \boldsymbol{a}_{-m}(j), t)}{T'}$$

  where ξ is the estimation error.
  end for
Each SU m ($m \in \mathcal{M}$) updates its $\boldsymbol{P}_m(j)$ according to the following rule:

$$P_{mn}(j+1) = P_{mn}(j) + b\bar{R}w_m(a_m(j), \boldsymbol{a}_{-m}(j))(1 - P_{mn}(j)), \quad n = a_m(j)$$
$$P_{mn}(j+1) = P_{mn}(j) - b\bar{R}w_m(a_m(j), \boldsymbol{a}_{-m}(j))P_{mn}(j), \quad n \neq a_m(j) \qquad (2.30)$$

  where $0 < b < 1$ is the step size.
Update $j = j + 1$.
end While

Remark 2.3. *According to the existing SLA based algorithms ([57–62]), the mixed strategy $\boldsymbol{P}_m(j)$ of each SU m is updated based on the received instantaneous reward $r_m(a_m, \boldsymbol{a}_{-m}, t)$. However, such updating rule is not feasible in this problem since the sufficient condition specified by Theorem* 3.3 *in [56] for $\{\boldsymbol{P}_m(j)\}_m$ converging to the pure-strategy NEs no longer holds in ordinal potential games. In the implementation*

of the proposed N-SLA algorithm, the mixed strategy of each SU is updated based on $\bar{Rw}_m(a_m(j), \boldsymbol{a}_{-m}(j))$. *In the following, it will be shown that such updating rule guarantees the convergence towards the pure-strategy NEs of the formulated ordinal potential game.*

Convergence of the proposed N-SLA algorithm

In this subsection, it is shown that under the proposed N-SLA algorithm, the channel selection probability vector $\boldsymbol{P}_m(j)$ ($\forall m \in \mathcal{M}$) converges to the pure strategy NEs of the modified game $\tilde{\mathcal{G}}_{OSA}$. The proof consists of two stages. In the first stage, an ordinary differential equation (ODE) is derived, whose solution approximates the asymptotic behavior of the channel selection probability matrix $\boldsymbol{P}(j) = [\boldsymbol{P}_1^H(j), \cdots, \boldsymbol{P}_M^H(j)]$ if the parameter b used in (2.30) is sufficiently small, where $(\cdot)^H$ denotes the conjugate transpose operation. In the second stage, the solutions of the ODE are characterized and the long term behavior of $\boldsymbol{P}(\cdot)$ is derived.

Ordinary differential equation (ODE): Define $\bar{\boldsymbol{Rw}}(j) = (\bar{Rw}_1(a_1(j), \boldsymbol{a}_{-1}(j)), \cdots, \bar{Rw}_M(a_M(j), \boldsymbol{a}_{-M}(j)))$ as the reward profile at the jth iteration, and $G(\cdot)$ as a function of $\boldsymbol{P}(j)$, $\boldsymbol{a}(j)$, and $\bar{\boldsymbol{Rw}}(j)$, which represents the updating rule specified by (2.30). Then, the matrix form of the updating rule given by (2.30) can be expressed as

$$\boldsymbol{P}(j+1) = \boldsymbol{P}(j) + bG(\boldsymbol{P}(j), \boldsymbol{a}(j), \bar{\boldsymbol{Rw}}(j)) \tag{2.31}$$

Define $Y(\boldsymbol{P})$ as the conditional expectation of $G(\boldsymbol{P}(j), \boldsymbol{a}(j), \bar{\boldsymbol{R}}\boldsymbol{w}(j))$ given $\boldsymbol{P}(j) = \boldsymbol{P}$, i.e.,

$$Y(\boldsymbol{P}) = \mathbb{E}[G(\boldsymbol{P}(j), \boldsymbol{a}(j), \bar{\boldsymbol{R}}\boldsymbol{w}(j))|\boldsymbol{P}(j) = \boldsymbol{P}] \tag{2.32}$$

Lemma 2.1. *Define a piecewise-constant interpolation of* $\boldsymbol{P}(j)$ *as*

$$\boldsymbol{P}^b(t') = \boldsymbol{P}(j), \quad t' \in [jb, (j+1)b) \tag{2.33}$$

As the step size $b \to 0$*, the sequence* $\{\boldsymbol{P}^b(\cdot)\}$ *converges weakly to* $\boldsymbol{Q}(\cdot)$*, which is the solution of the following ODE:*

$$\frac{d\boldsymbol{Q}}{dt'} = Y(\boldsymbol{Q}), \boldsymbol{Q}(0) = \boldsymbol{P}(0) \tag{2.34}$$

where $\boldsymbol{P}(0)$ *is the initial channel selection probability matrix.*

Proof. The proof follows the same procedure to prove the Theorem 3.1 in [56]. □

Solution to the ODE: Since $\boldsymbol{P}$ contains $N \times M$ components, denoted by P_{mn}, $m \in \mathcal{M}$, $n \in \mathcal{N}$, the component equations of (2.34) can be written as

$$\begin{aligned}
\frac{dP_{mn}}{dt'} &= P_{mn}(1 - P_{mn})\mathbb{E}[\bar{R}w_m|\boldsymbol{P}, a_m = n] + \sum_{n' \in \mathcal{N}, n' \neq n} P_{mn'}(-P_{mn})\mathbb{E}[\bar{R}w_m|\boldsymbol{P}, a_m = n'] \\
&= P_{mn} \sum_{n' \in \mathcal{N}, n' \neq n} P_{mn'}\mathbb{E}[\bar{R}w_m|\boldsymbol{P}, a_m = n] - P_{mn} \sum_{n' \in \mathcal{N}, n' \neq n} P_{mn'}\mathbb{E}[\bar{R}w_m|\boldsymbol{P}, a_m = n'] \\
&= P_{mn}\Big(\sum_{n' \in \mathcal{N}, n' \neq n} P_{mn'}(\mathbb{E}[\bar{R}w_m|\boldsymbol{P}, a_m = n] - \mathbb{E}[\bar{R}w_m|\boldsymbol{P}, a_m = n'])\Big) \\
&= P_{mn}\Big(\sum_{n' \in \mathcal{N}} P_{mn'}(\mathbb{E}[\bar{R}w_m|\boldsymbol{P}, a_m = n] - \mathbb{E}[\bar{R}w_m|\boldsymbol{P}, a_m = n'])\Big), \\
&\forall m \in \mathcal{M}, \forall n \in \mathcal{N}
\end{aligned} \tag{2.35}$$

where $\mathbb{E}[\bar{Rw}_m|\boldsymbol{P}, a_m = n]$ denotes the expected reward of SU m if it employs the pure strategy n while any other SU m' ($\forall m' \in \mathcal{M}$, $m' \neq m$) employs the mixed strategy $\boldsymbol{P}_{m'}$. Specifically, $\mathbb{E}[\bar{Rw}_m|\boldsymbol{P}, a_m = n]$ can be represented as

$$\mathbb{E}[\bar{Rw}_m|\boldsymbol{P}, a_m = n] = \sum_{(a_1,\cdots,a_{m-1},a_{m+1},\cdots,a_M)} \bar{Rw}_m(n, \boldsymbol{a}_{-m}) \prod_{m'\in\mathcal{M},m'\neq m} P_{m'a_{m'}} \quad (2.36)$$

Theorem 2.3. *With a sufficiently small parameter b, $\boldsymbol{P}$ converges to a pure NE of the modified game $\tilde{\mathcal{G}}_{OSA}$.*

The proof of Theorem 2.3 is shown in Appendix A.2.

2.5 Simulation Results

In this section, numerical results are provided to demonstrate the performance of both the BR based and the N-SLA algorithms. The convergence of both algorithms is first illustrated, and then, the performance is evaluated in terms of sum log expected throughput, which is a commonly used metric to evaluate the tradeoff between efficiency and fairness among multiple users [103, 104]. In simulations, set $b = 0.1$, $T' = 100$ slots, and $\bar{r}_m^f = 1$, $\forall m \in \mathcal{M}$. Note that this simplification may affect the absolute performance values, but not the optimization. In addition, both the mean channel availability parameters θ_n ($\forall n \in \mathcal{N}$) and the transmission probabilities P_m ($0 < P_m \leq 1$, $\forall m \in \mathcal{M}$) are simulated following the uniform distribution, i.e, $\theta_n \sim U[0, 1]$, $\forall n \in \mathcal{N}$, and $P_m \sim U[0, 1]$, $\forall m \in \mathcal{M}$.

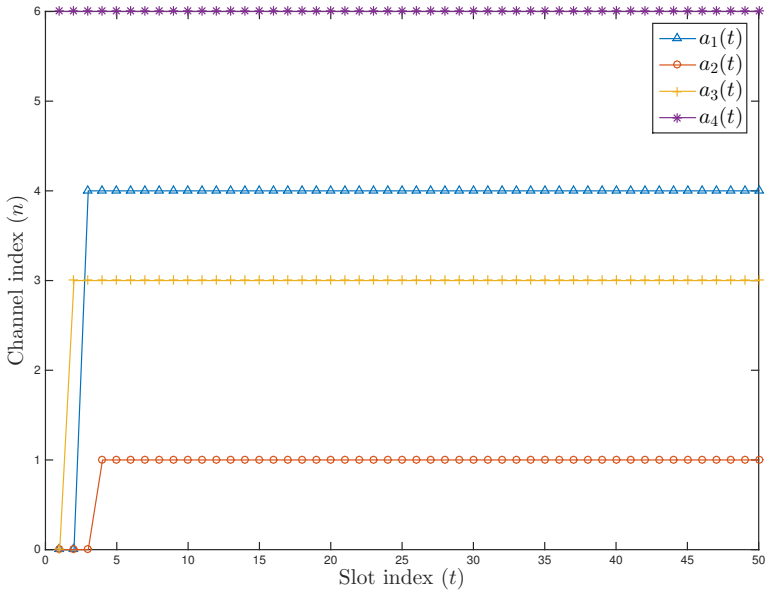

Figure 2.2: Evolution of the channel selection actions by the Algorithm 1 for the unconstrained transmission case

2.5.1 Convergence behavior of the proposed BR based and N-SLA algorithms

Figs. 2.2 and 2.3 show the convergence behaviors of $a_m(t)$ $(\forall m \in \mathcal{M})$ over time t by using Algorithm 1, with respect to the scenario (I) and the scenario (II), respectively. In Fig. 2.2, $M = 4$, $N = 8$, $(\theta_1, \cdots, \theta_8) = (0.5, 0.3, 0.7, 0.6, 0.2, 0.8, 0.3, 0.2)$, and in Fig.2.3, $M = 8$, $N = 4$, $(\theta_1, \cdots, \theta_4) = (0.8, 0.6, 0.5, 0.3)$, $(P_1, \cdots, P_8) = (0.2, 0.3, 0.6, 0.5, 0.4, 0.8, 0.6, 0.5)$. From these two figures, it can be seen that under the scenario (I) (i.e., Fig.2.2), the network converges after M time slots and the selected channels are the M most available primary channels $(\theta_1, \theta_3, \theta_4, \theta_6) = (0.5, 0.7, 0.6, 0.8)$, while under the scenario (II) (i.e., Fig. 2.3), due to the heteroge-

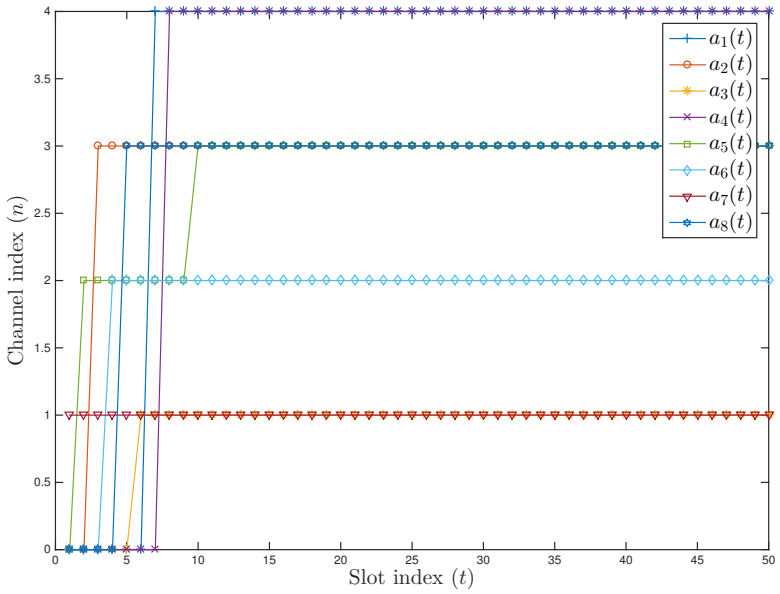

Figure 2.3: Evolution of the channel selection actions by the Algorithm 1 for the constrained transmission case

neous ties of SUs who have different transmission probabilities, the network convergence consumes more time than that in scenario (I).

Figs. 2.4 to 2.8 show the convergence behavior of the Algorithm 2, by setting $\varpi = 32$ and all other parameters to be the same as in Fig. 2.3. Fig. 2.4 plots the evolution of the channel selection probabilities $P_{mn}(j)$ $(\forall n \in \mathcal{N})$ for an arbitrary SU. It can be seen that the channel selection probability vector $\boldsymbol{P}_m(j)$ evolves from a mixed strategy $(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$ to a pure-strategy $(0, 0, 0, 1)$ within about 450 iterations. The evolution of the channel selection actions $a_m(j)$ for each SU m $(m \in \mathcal{M})$ is further shown in Figs. 2.5 to 2.8. It can be seen that after the network converges, SUs $1, 2, 5, 8$ will always select channel 1 to access (as shown in Fig.2.5), SUs $3, 7$ will always select channel 2 to access (as shown in Fig. 2.6), while SU 6 and SU 4

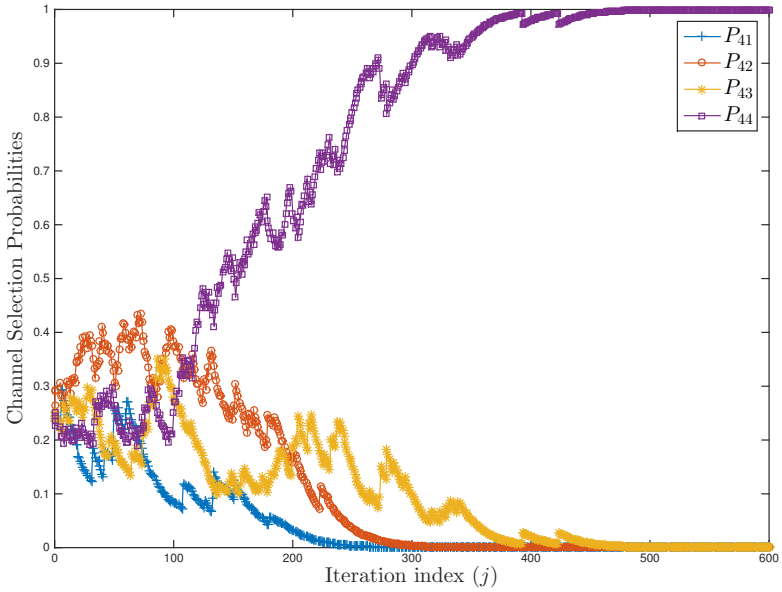

Figure 2.4: Evolution of the channel selection probabilities for SU 4 by the Algorithm 2

will always select channel 3 and channel 4 to access, respectively (as shown in Figs. 2.7 and 2.8). This channel selection result can be easily justified to be an NE of the formulated game.

Fig. 2.9 compares the proposed BR based and the N-SLA algorithms in terms of the values of the ordinal potential function at convergence. In this simulation, three scenarios with respect to the transmission probability vector $(P_1, \cdots, P_8)$ are considered. In scenario 1, the transmission probabilities of SUs are dissimilar, i.e., $(P_1, \cdots, P_8) = (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8)$; in scenario 2, the transmission probabilities across different SUs are randomly selected as $(P_1, \cdots, P_8) = (0.2, 0.3, 0.6, 0.5, 0.4, 0.8, 0.6, 0.5$ and in scenario 3, the transmission probabilities of all SUs are identical, i.e., $P_m = 0.5$, $\forall m \in \mathcal{M}$. Other parameter settings are $(\theta_1, \cdots, \theta_4) = (0.8, 0.6, 0.5, 0.3)$, ϖ is 60 for

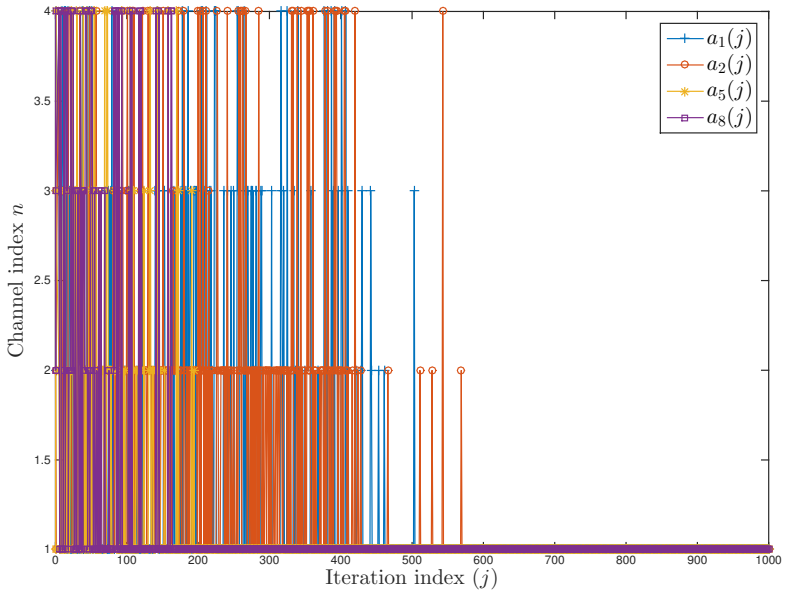

Figure 2.5: Evolution of the channel selection actions for SUs $1, 2, 5, 8$, by the Algorithm 2

scenario 1, 32 for scenario 2, and 80 for scenario 3. For the two proposed algorithms, under each scenario, the maximum and the minimum values of the ordinal potential function at convergence are compared by independently simulating 500 trials. From Fig.2.9, it can be seen that in all three scenarios, the two proposed algorithms achieve same maximum values, but different minimum values. The reason is explained as follows. As shown in the Theorem 2.3, the Algorithm 2 can converge to the NEs only when $b \to 0$. However, b cannot be too small in practice since smaller b leads to slower convergence speed. Thus, for a practical value of b, the solutions found by the Algorithm 2 may be non-equilibrium points, which lead to lower values of the ordinal potential function at convergence. However, the same maximum values achieved by the two proposed algorithms illustrate that Algorithm 2 can still converge to the NEs

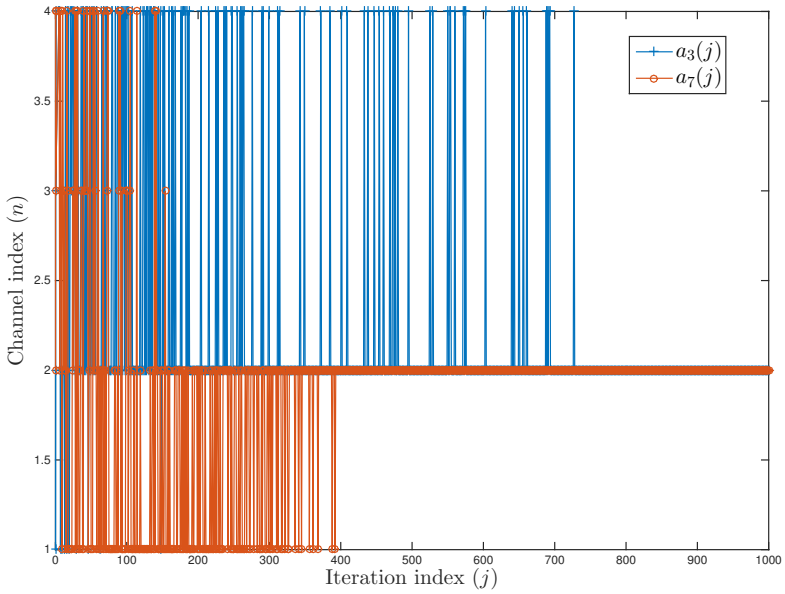

Figure 2.6: Evolution of the channel selection actions for SUs $3, 7$, by the Algorithm 2

under the current parameter settings. The percentages that Algorithm 2 converges to NEs in all three scenarios are further shown in Table 2.1. From the table, it can be seen that Algorithm 2 converges to the NEs most of the time, which clearly indicates the effectiveness of the proposed fully distributed algorithm.

2.5.2 Sum log expected throughput of the proposed algorithms

In this subsection, the system performance in terms of sum log expected throughput achieved by the NE solutions is evaluated. For comparison purpose, a random selection scheme and a globally optimal solution are also simulated. In the random selection scheme, each SU randomly chooses a channel in each time slot, while the globally optimal solution tries to maximize the sum log expected throughput in a

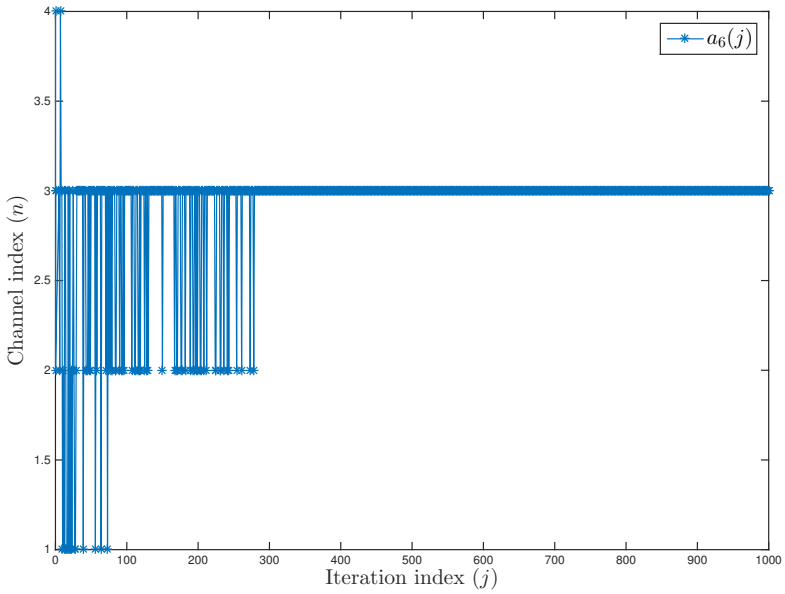

Figure 2.7: Evolution of the channel selection actions for SU 6 by the Algorithm 2

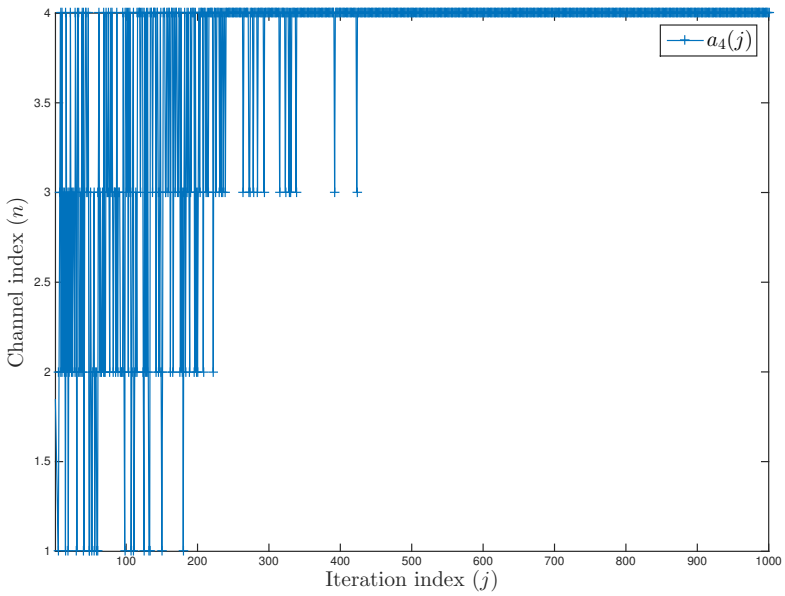

Figure 2.8: Evolution of the channel selection actions for SU 4 by the Algorithm 2

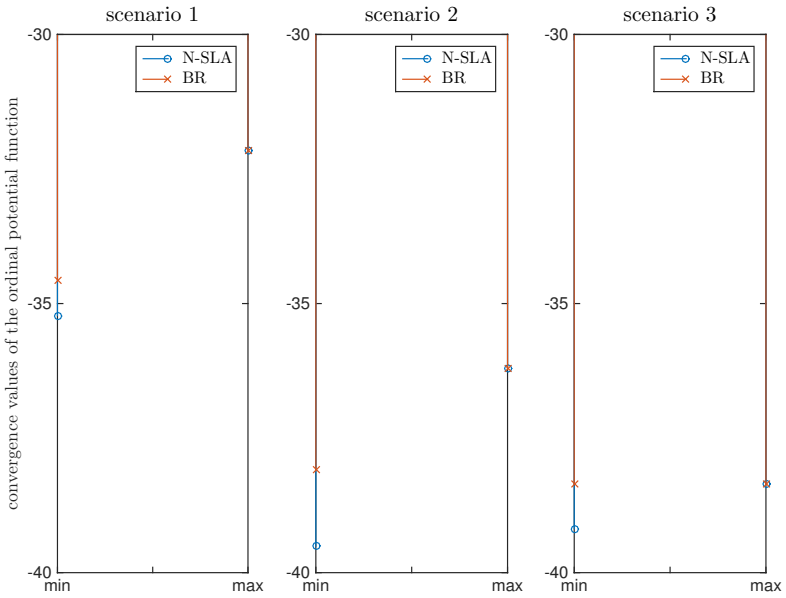

Figure 2.9: Comparison of the proposed two algorithms in terms of the convergence values of the ordinal potential function

centralized manner by assuming all the information including the primary channel statistics and the SUs' transmission probabilities is known *a priori*. The presented results are average values over 1000 independent trials.

Table 2.1: The equilibrium percentage achieved by Algorithm 2 in all three scenarios

scenario 1	scenario 2	scenario 3
94.6%	92.2%	93.4%

The performance comparison is carried out under three different cases, as shown in Figs. 2.10-2.12. Case I as shown in Fig. 2.10 considers a homogeneous OSA network where the availability statistics of primary channels and the transmission probabilities of SUs are identical. The parameter settings are $\theta_n = \theta = 0.7$, $\forall n \in \mathcal{N}$,

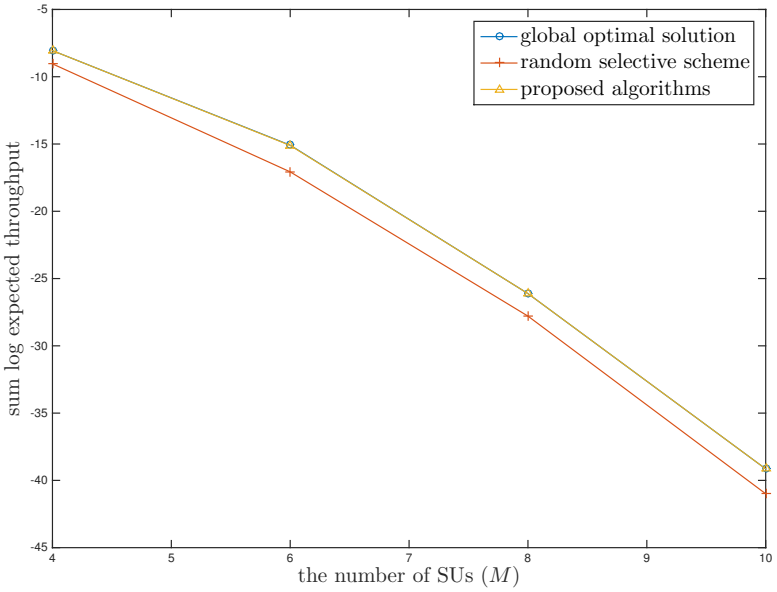

Figure 2.10: Performance comparison of different approaches in case I

and $P_m = P = 0.5$, $\forall m \in \mathcal{M}$. From Fig. 2.10, it can be seen that the proposed algorithms achieve the same performance as the globally optimal solution. It is because that in case I, the ordinal potential function specified by (2.11) can be rewritten as $\Phi_2(a_m, \boldsymbol{a}_{-m}) = (-\log_2(1-P))(\sum_{m\in\mathcal{M}} \log_2(\hat{R}_m(a_m, \boldsymbol{a}_{-m})) + M\log_2(\theta) + M\log_2(\frac{P}{1-P}))$. As discussed before, the proposed algorithms try to maximize the ordinal potential function $\Phi_2(a_m, \boldsymbol{a}_{-m})$. Thus, the obtained NE solution $(a_m^*, \boldsymbol{a}_{-m}^*)$ satisfies $(a_m^*, \boldsymbol{a}_{-m}^*) = \arg\max_{(a_m, \boldsymbol{a}_{-m})} (-\log_2(1-P))(\sum_{m\in\mathcal{M}} \log_2(\hat{R}_m(a_m, \boldsymbol{a}_{-m})) + M\log_2(\theta) + M\log_2(\frac{P}{1-P})) = \arg\max_{(a_m, \boldsymbol{a}_{-m})} \sum_{m\in\mathcal{M}} \log_2(\hat{R}_m(a_m, \boldsymbol{a}_{-m})) = \arg\min_{(a_m, \boldsymbol{a}_{-m})} \sum_{m\in\mathcal{M}} |\mathcal{U}_{a_m}|$. According to the procedures in Algorithm 1, all the pure-strategy NEs will achieve the same $\sum_{m\in\mathcal{M}} |\mathcal{U}_{a_m}|$. Thus, the proposed algorithms can achieve globally optimal performance.

Case II considers a CR network with different availability statistics across the

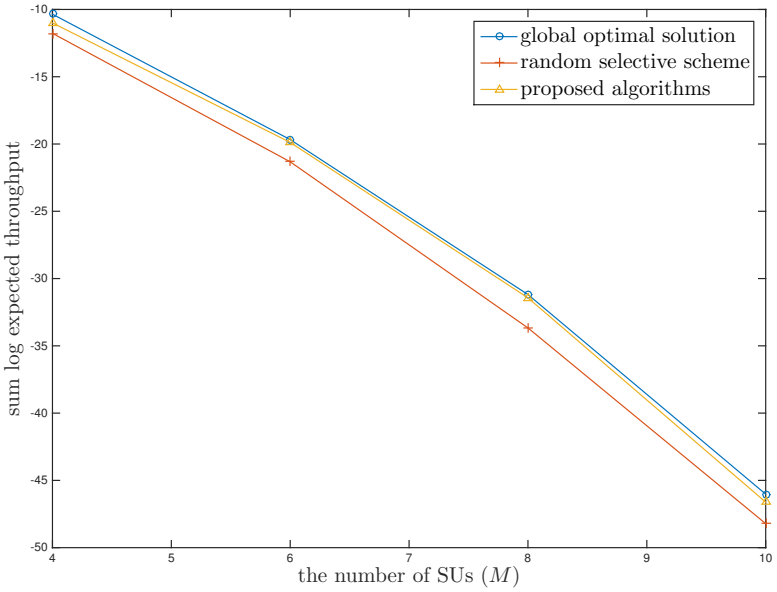

Figure 2.11: Performance comparison of different approaches in case II

primary channels, but identical transmission probabilities across the SUs, as shown in Fig.2.11. The parameter settings are $(\theta_1, \theta_2, \theta_3) = (0.7, 0.5, 0.2)$, and $P_m = P = 0.5$, $\forall m \in \mathcal{M}$. From this figure, the performance achieved by the proposed algorithms is very close to the globally optimal solution. This is because in case II, $\Phi_2(a_m, \boldsymbol{a}_{-m})$ can be rewritten as $\Phi_2(a_m, \boldsymbol{a}_{-m}) = (-\log_2(1-P))(\sum_{m\in\mathcal{M}} \log_2(\hat{R}_m(a_m, \boldsymbol{a}_{-m})) + \sum_{m\in\mathcal{M}} \log_2(\theta_{a_m}) + M \log_2(\frac{P}{1-P}))$, and the NEP solution $(a_m^*, \boldsymbol{a}_{-m}^*)$ achieved by the proposed algorithms satisfies $(a_m^*, \boldsymbol{a}_{-m}^*) = \arg \max_{(a_m, \boldsymbol{a}_{-m})} (\sum_{m\in\mathcal{M}} \log_2 (\hat{R}_m(a_m, \boldsymbol{a}_{-m})) + \sum_{m\in\mathcal{M}} \log_2 (\theta_{a_m}))$. Thus, due to the existence of the term $\sum_{m\in\mathcal{M}} \log_2(\theta_{a_m})$, the proposed algorithms can not find the globally optimal solution.

In case III as shown in Fig. 2.12, the simulated CR network has different availability statistics across the primary channels and different transmission probabilities across the SUs. Specifically, the simulation parameters are set as: $(\theta_1, \theta_2, \theta_3) =$

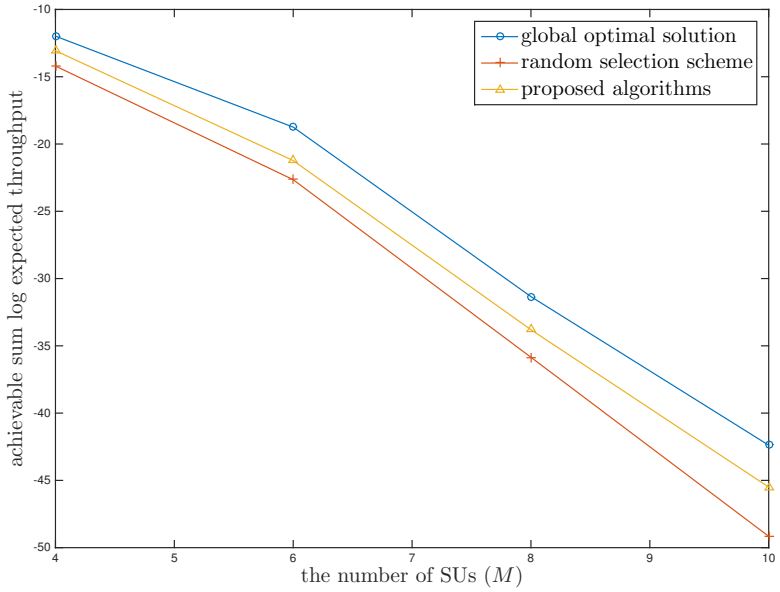

Figure 2.12: Performance comparison of different approaches in case III

$(0.7, 0.5, 0.2)$, $(P_1, \cdots, P_M) = (0.2, 0.7, 0.5, 0.1)$ for $M = 4$, $(P_1, \cdots, P_M) = (0.2, 0.6, 0.3, 0.8, 0.5, 0.3)$ for $M = 6$, $(P_1, \cdots, P_M) = (0.4, 0.2, 0.6, 0.3, 0.8, 0.1, 0.5, 0.6)$ for $M = 8$, and $(P_1, \cdots, P_M) = (0.2, 0.4, 0.4, 0.5, 0.2, 0.8, 0.1, 0.6, 0.3, 0.1)$ for $M = 10$. Since in case III, the ordinal potential function $\Phi_2(a_m, \boldsymbol{a}_{-m})$ specified by (2.11) is totally different from the global optimization $\sum_{m \in \mathcal{M}} \log_2(\hat{R}_m(a_m, \boldsymbol{a}_{-m}))$, an obvious gap exists between the proposed algorithms and the globally optimal solution.

However, in all these three figures, it can be seen that the proposed algorithms always outperform the random selection approach. Thus, it can be concluded that for distributed channel access, the proposed algorithms can not only converge to the pure-strategy NEs, but also achieve better network performance in terms of sum log expected throughput than the random selection approach.

Chapter 3

Distributed Multi-Mobile-Device Computation Offloading for Cloudlet-based Mobile Cloud Computing: A Game-Theoretic Machine Learning Approach

In this chapter, the problem of multi-mobile-device computation offloading for cloudlet-based MCC in a multi-channel wireless contention environment is investigated. The studied network is fully distributed so that each mobile device makes the offloading decisions based only on its individual information, and without information exchange. This multi-mobile-device computation offloading decision making problem is first formulated as a noncooperative game. After analyzing the structural property

of the formulated game, it is shown to be an exact potential game, which has at least one pure-strategy NE. To achieve the NEs in a fully distributed environment, a fully distributed computation offloading (FDCO) algorithm based on machine learning technology is proposed. Then, the performance of the proposed FDCO algorithm in terms of the number of beneficial cloudlet computing mobile devices and the network-wide execution cost is theoretically analyzed. Finally, simulation results validate the effectiveness of the proposed algorithm by comparing it with counterparts.

3.1 System Model and Game Formulation

3.1.1 System Model

A cloudlet-based MCC network consisting of a set of $\mathcal{M} = \{1, 2, \cdots, M\}$ mobile devices and one cloudlet that could be located in a cellular base station or a Wi-Fi access point [71] is considered. There are N wireless channels and the set of channels is denoted as $\mathcal{N} = \{1, 2, \cdots, N\}$. Each mobile device has a computationally intensive task to be completed, which consists of B equal-size data packets. Similar to many previous studies on MCC (e.g. [105–107]), a quasi-static scenario is considered where the set of mobile devices $\mathcal{M}$ remains unchanged during a computation offloading period (e.g., hundreds of milliseconds or several seconds), while may change across different periods. The mobile devices can process their tasks locally or offload them to the cloudlet via a wireless channel. Since both the communication and computation aspects play critical roles in the cloudlet-based MCC network, the communication and computation models are introduced next in details.

Communication Model

Without loss of generality, assume that a computation task requires N_{cyc} CPU cycles for processing. Denote $a_m \in \{0\} \bigcup \mathcal{N}$ as the computation offloading action of the mobile device m. $a_m = 0$, if the mobile device m decides to compute its task locally on its own mobile device. Otherwise, the mobile device m will offload its task via the wireless channel $a_m \in \mathcal{N}$. A slotted time structure is considered and the time slot length and the time slot index are denoted by τ and $t \in \mathcal{T} = \{1, 2, \cdots, \}$, respectively. In each time slot, if more than one mobile device select the same channel for offloading, a carrier sense multiple access (CSMA) mechanism is used to handle the potential collisions [108]. For any mobile device which has contended the channel successfully, it will receive one unit throughput in terms of one packet per slot. Thus, given the decision profile of all mobile devices, which is denoted as $\boldsymbol{a} = (a_1, \cdots, a_M)$, in each time slot t, the random transmission rate of a mobile device m that chooses to offload the computation to the cloudlet via the wireless channel $a_m \in \mathcal{N}$ can be computed as:

$$r_m(\boldsymbol{a}, t) = J_m(\boldsymbol{a}, t) \tag{3.1}$$

where $J_m(\boldsymbol{a}, t)$ indicates whether mobile device m successfully contends the channel in slot t and can be represented as

$$J_m(\boldsymbol{a}, t) = \begin{cases} 1, & \text{if mobile device } m \text{ contends channel } a_m \text{ successfully in slot } t \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

According to the principle of CSMA, $J_m(\boldsymbol{a}, t)$ is a Bernoulli random variable with a probability mass function (PMF) of

$$Pr(J_m(\boldsymbol{a}, t)) = \begin{cases} \frac{1}{s_{a_m}}, & \text{if } J_m(\boldsymbol{a}, t) = 1 \\ 1 - \frac{1}{s_{a_m}}, & \text{otherwise} \end{cases} \tag{3.3}$$

where s_{a_m} denotes the number of mobile devices that choose channel a_m for offloading.

Based on (3.1-3.3), the expected rate achieved by the mobile device m with the decision profile $\boldsymbol{a}$ can be calculated as

$$R_m(\boldsymbol{a}) = \frac{1}{s_{a_m}} \tag{3.4}$$

From (3.4), the average transmission time and energy consumption of the mobile device m for offloading the task of size B packets can be further calculated respectively as

$$\begin{aligned} t_{m,off}(\boldsymbol{a}) &= \frac{B}{R_m(\boldsymbol{a})} \\ &= B s_{a_m} \quad (slots) \end{aligned} \tag{3.5}$$

$$\begin{aligned} e_{m,off}(\boldsymbol{a}) &= \frac{pB}{R_m(\boldsymbol{a})}\tau \\ &= pB s_{a_m}\tau \quad (J) \end{aligned} \tag{3.6}$$

where p is the transmit power at each mobile device.

Computation Model

In the following, the execution overhead in terms of energy consumption and time cost for both local and cloudlet computing will be discussed.

Local Computing: For local computing, the computation tasks are executed on mobile devices. Let f_l be the computation capability (in terms of the CPU cycles per second) of a mobile device. Then, the time consumption for locally computing one task can be computed as

$$t_{l,exe} = \lceil \frac{N_{cyc}}{f_l \tau} \rceil \quad (slots) \tag{3.7}$$

and the required energy consumption is

$$e_{l,exe} = \nu_l N_{cyc} \quad (J) \tag{3.8}$$

where ν_l is a coefficient denoting the consumed energy per CPU cycle at a mobile device, which can be obtained by measurements as in [107].

Cloudlet Computing: For cloudlet computing, a mobile device first offloads its computation task to the cloudlet via a wireless channel, and then the cloudlet will execute the computation task on behalf of the mobile device. Specifically, for the computation offloading, the mobile device would incur an overhead for transmitting the computation task to the cloudlet. According to the communication model introduced before, the transmission time and energy consumption of each mobile device for offloading one task have been derived by (3.5) and (3.6), respectively.

After receiving the data, the cloudlet will execute the requested computation task. Let f_c be the computation capability (in terms of the CPU cycles per second) assigned to each mobile device by the cloudlet, which can be determined according to the cloud computing service contract subscribed by the mobile devices with the telecom operator [109]. Ordinarily, $f_c \geq f_l$. Then, the execution time of one task at

the cloudlet equals

$$t_{c,exe} = \lceil \frac{N_{cyc}}{f_c \tau} \rceil \quad (slots) \tag{3.9}$$

Combining (3.5), (3.6), and (3.9), the overall time (including transmission time and execution time) and energy consumption of mobile device m using cloudlet computing becomes

$$\begin{aligned} t_{m,c}(\boldsymbol{a}) &= t_{m,off}(\boldsymbol{a}) + t_{c,exe} \\ &= (Bs_{a_m} + \lceil \frac{N_{cyc}}{f_c \tau} \rceil) \quad (slots) \end{aligned} \tag{3.10}$$

$$\begin{aligned} e_{m,c}(\boldsymbol{a}) &= e_{m,off}(\boldsymbol{a}) \\ &= pBs_{a_m}\tau \quad (J) \end{aligned} \tag{3.11}$$

Actually, the mobile device should pay the cloud for computing service, if it selects to offload the computation task to the cloud. However, as shown in most related works in the literature [109–111], such cost was commonly assumed to be pre-paid. This chapter followed the same convention.

3.1.2 Game Formulation

By considering the potential contention among mobile devices and the fully distributed manner in decision making, the objective is to minimize each mobile device's execution overhead individually, which can be formulated as

$$\min_{a_m \in \{0\} \bigcup \mathcal{N}} O_m(a_m, \boldsymbol{a}_{-m}) \quad \forall m \in \mathcal{M} \tag{3.12}$$

where

$$O_m(a_m, \boldsymbol{a}_{-m}) = \begin{cases} \lambda_1 t_{l,exe} + \lambda_2 e_{l,exe}, & \text{if } a_m = 0 \\ \lambda_1 t_{m,c}(a_m, \boldsymbol{a}_{-m}) + \lambda_2 e_{m,c}(a_m, \boldsymbol{a}_{-m}), & \text{if } a_m \in \mathcal{N} \end{cases} \tag{3.13}$$

$\boldsymbol{a}_{-m} = (a_1, \cdots, a_{m-1}, a_{m+1}, \cdots, a_M)$ denotes the set of computation offloading actions by all mobile devices except the m^{th} one, and $\lambda_1, \lambda_2 \in (0, 1)$ denote the weighting parameters of time and energy consumption, respectively [1].

Substituting (3.7), (3.8), (3.10), and (3.11) into (3.13),

$$O_m(a_m, \boldsymbol{a}_{-m}) = \begin{cases} F_{l,1}, & \text{if } a_m = 0 \\ F_{c,1}(s_{a_m}), & \text{if } a_m \in \mathcal{N} \end{cases} \tag{3.14}$$

where

$$F_{l,1} = \lambda_1 \lceil \frac{N_{cyc}}{f_l \tau} \rceil + \lambda_2 \nu_l N_{cyc} \tag{3.15}$$

$$F_{c,1}(s_{a_m}) = \lambda_1 (B s_{a_m} + \lceil \frac{N_{cyc}}{f_c \tau} \rceil) + \lambda_2 p B \tau s_{a_m} \tag{3.16}$$

Obviously, $F_{c,1}(s_{a_m})$ is an increasing function of s_{a_m}. Since $1 \leq s_{a_m} \leq M$, $F_{c,1}(1) \leq F_{c,1}(s_{a_m}) \leq F_{c,1}(M)$. If $F_{l,1} < F_{c,1}(1)$, according to the optimization problem defined in (3.12), the mobile device will always choose to execute the computation task on its own device. Similarly, if $F_{c,1}(M) < F_{l,1}$, the mobile device will always choose to offload the computation task to the cloudlet. Thus, in this chapter, these two trivial situations will be ignored and the discussion will be focused on the case that $F_{c,1}(1) \leq F_{l,1} \leq F_{c,1}(M)$.

Theorem 3.1. *The distributed execution overhead minimization problem (3.12) is*

[1] In practice, the proper weights can be determined by applying the multiple criteria decision making theory [112].

equivalent to the following distributed utility maximization problem:

$$\max_{a_m \in \{0\} \bigcup \mathcal{N}} U_m(a_m, \boldsymbol{a}_{-m}) \quad \forall m \in \mathcal{M} \tag{3.17}$$

where the utility $U_m(a_m, \boldsymbol{a}_{-m})$ *of mobile device* m *is defined as*

$$U_m(a_m, \boldsymbol{a}_{-m}) = \begin{cases} \frac{1}{\lambda_2 \nu_l N_{cyc} + \lambda_1 (\lceil \frac{N_{cyc}}{f_l \tau} \rceil - \lceil \frac{N_{cyc}}{f_c \tau} \rceil)}, & \text{if } a_m = 0 \\ \frac{1}{\lambda_1 B + \lambda_2 B p \tau} \frac{1}{s_{a_m}}, & \text{if } a_m \in \mathcal{N} \end{cases} \tag{3.18}$$

Proof. Subtracting $\lambda_1 \lceil \frac{N_{cyc}}{f_c \tau} \rceil$ on both sides of (3.14) results in

$$O_m(a_m, \boldsymbol{a}_{-m}) - \lambda_1 \lceil \frac{N_{cyc}}{f_c \tau} \rceil = \begin{cases} \lambda_1 (\lceil \frac{N_{cyc}}{f_l \tau} \rceil - \lceil \frac{N_{cyc}}{f_c \tau} \rceil) + \lambda_2 \nu_l N_{cyc}, & \text{if } a_m = 0 \\ (\lambda_1 B + \lambda_2 p B \tau) s_{a_m}, & \text{if } a_m \in \mathcal{N} \end{cases} \tag{3.19}$$

Since $\lceil \frac{N_{cyc}}{f_l \tau} \rceil - \lceil \frac{N_{cyc}}{f_c \tau} \rceil \geq 0$, $O_m(a_m, \boldsymbol{a}_{-m}) - \lambda_1 \lceil \frac{N_{cyc}}{f_c \tau} \rceil > 0$. Thus,

$$\begin{aligned} &\min_{a_m \in \{0\} \bigcup \mathcal{N}} O_m(a_m, \boldsymbol{a}_{-m}) \iff \\ &\min_{a_m \in \{0\} \bigcup \mathcal{N}} (O_m(a_m, \boldsymbol{a}_{-m}) - \lambda_1 \lceil \frac{N_{cyc}}{f_c \tau} \rceil) \iff \\ &\max_{a_m \in \{0\} \bigcup \mathcal{N}} \frac{1}{O_m(a_m, \boldsymbol{a}_{-m}) - \lambda_1 \lceil \frac{N_{cyc}}{f_c \tau} \rceil} \iff \\ &\max_{a_m \in \{0\} \bigcup \mathcal{N}} U_m(a_m, \boldsymbol{a}_{-m}) \end{aligned} \tag{3.20}$$

□

According to (3.20), each mobile device tries to maximize its own utility function U_m. Thus, this distributed utility maximization problem can be formulated into a noncooperative game, which is denoted by $\mathcal{G}_{off} = \{\mathcal{M}, \mathcal{A}, \{U_m(a_m, \boldsymbol{a}_{-m})\}_{m \in \mathcal{M}}\}$. Here, $\mathcal{M}$ is the set of players (i.e., mobile devices), $\mathcal{A} = \{0\} \bigcup \mathcal{N}$ is the set of actions that each player can take, and $U_m(a_m, \boldsymbol{a}_{-m})$ is the utility of player m ($m \in \mathcal{M}$) upon taking action $a_m \in \mathcal{A}$ while other players taking $\boldsymbol{a}_{-m}$. Each player independently and selfishly adjusts its strategy to maximize its individual utility $U_m(a_m, \boldsymbol{a}_{-m})$.

Definition 3.1. A strategy profile $\boldsymbol{a}^* = (a_1^*, \cdots, a_M^*)$ is a pure strategy NE of the noncooperative game $\mathcal{G}_{off}$ if and only if no mobile device can improve its utility function by deviating unilaterally, i.e.,

$$U_m(a_m^*, \boldsymbol{a}_{-m}^*) \geq U_m(a_m, \boldsymbol{a}_{-m}^*), \quad \forall m \in \mathcal{M}, \forall a_m \in \mathcal{A} \tag{3.21}$$

In the following, an FDCO algorithm will be proposed to derive the pure strategy NEs of the formulated game $\mathcal{G}_{off}$ in a fully distributed way.

3.2 FDCO Algorithm

In this section, the existence of NEs in the formulated game $\mathcal{G}_{off}$ is first discussed by analyzing its structural properties, and then the proposed FDCO algorithm is presented in details.

3.2.1 Existence of NEs in the Formulated Game $\mathcal{G}_{off}$

Before discussing the existence of NEs in the formulated game $\mathcal{G}_{off}$, the definition of exact potential games is first given as follows.

Definition 3.2. A game is called an exact potential game if the deviation in the utility of an arbitrary player m is exactly reflected by the deviation in a function $\psi : \boldsymbol{a} = (a_1, \cdots, a_M) \rightarrow \mathbb{R}$, which is called an exact potential function, i.e.,

$$\begin{aligned} &U_m(a_m, \boldsymbol{a}_{-m}) - U_m(a'_m, \boldsymbol{a}_{-m}) = \psi(a_m, \boldsymbol{a}_{-m}) - \psi(a'_m, \boldsymbol{a}_{-m}), \\ &\forall m \in \mathcal{M}, \forall a_m, a'_m \in \mathcal{A}, a_m \neq a'_m \end{aligned} \tag{3.22}$$

Based on the definition of exact potential games, the following theorem is obtained.

Theorem 3.2. *The formulated game $\mathcal{G}_{off}$ has at least one pure strategy NE.*

Proof. Define a bounded function $\psi : \boldsymbol{a} = (a_1, \cdots, a_M) \rightarrow \mathbb{R}$ as

$$\psi(a_m, \boldsymbol{a}_{-m}) = \sum_{n=1}^{N} \sum_{v=1}^{s_n} F_{c,2}(v) + F_{l,2} \sum_{m=1}^{M} \mathbf{1}_{\{a_m=0\}} \tag{3.23}$$

where

$$F_{c,2}(v) = \frac{1}{\lambda_1 B + \lambda_2 B p \tau} \frac{1}{v} \tag{3.24}$$

$$F_{l,2} = \frac{1}{\lambda_2 \nu_l N_{cyc} + \lambda_1 (\lceil \frac{N_{cyc}}{f_l \tau} \rceil - \lceil \frac{N_{cyc}}{f_c \tau} \rceil)} \tag{3.25}$$

$$\mathbf{1}_{\{a_m=0\}} = \begin{cases} 1, & \text{if } a_m = 0 \\ 0, & \text{otherwise} \end{cases} \tag{3.26}$$

For a computation offloading decision profile $\boldsymbol{a}$, the number of players selecting channel n is denoted as s_n, $\forall n \in \mathcal{N}$. Suppose that an arbitrary player m ($m \in \mathcal{M}$) unilaterally changes its action from a_m to a'_m. Since a_m, $a'_m \in \mathcal{N} \cup \{0\}$, the following three cases are considered: 1) $a_m, a'_m \in \mathcal{N}$ and $a_m \neq a'_m$; 2) $a_m = 0$ and $a'_m \in \mathcal{N}$; 3) $a_m \in \mathcal{N}$ and $a'_m = 0$.

Case 1): Since $a_m, a'_m \in \mathcal{N}$ and $a_m \neq a'_m$,

$$U_m(a_m, \boldsymbol{a}_{-m}) = \frac{1}{\lambda_1 B + \lambda_2 B p \tau} \frac{1}{s_{a_m}} = F_{c,2}(s_{a_m}) \tag{3.27}$$

$$U_m(a'_m, \boldsymbol{a}_{-m}) = \frac{1}{\lambda_1 B + \lambda_2 B p \tau} \frac{1}{s_{a'_m} + 1} = F_{c,2}(s_{a'_m} + 1) \tag{3.28}$$

According to (3.23),

$$\psi(a_m, \boldsymbol{a}_{-m}) = \sum_{v=1}^{s_{a_m}} F_{c,2}(v) + \sum_{v=1}^{s_{a'_m}} F_{c,2}(v) + \sum_{n=1, n\neq a_m, a'_m}^{N} \sum_{v=1}^{s_n} F_{c,2}(v) + F_{l,2} \sum_{m=1}^{M} \mathbf{1}_{\{a_m=0\}} \tag{3.29}$$

$$\psi(a'_m, \boldsymbol{a}_{-m}) = \sum_{v=1}^{s_{a_m}-1} F_{c,2}(v) + \sum_{v=1}^{s_{a'_m}+1} F_{c,2}(v) + \sum_{n=1, n\neq a_m, a'_m}^{N} \sum_{v=1}^{s_n} F_{c,2}(v) + F_{l,2} \sum_{m=1}^{M} \mathbf{1}_{\{a_m=0\}} \tag{3.30}$$

Then,

$$\begin{aligned} \psi(a_m, \boldsymbol{a}_{-m}) - \psi(a'_m, \boldsymbol{a}_{-m}) &= F_{c,2}(s_{a_m}) - F_{c,2}(s_{a'_m}+1) \\ &= U_m(a_m, \boldsymbol{a}_{-m}) - U_m(a'_m, \boldsymbol{a}_{-m}) \end{aligned} \tag{3.31}$$

Case 2): Since $a_m = 0$ and $a'_m \in \mathcal{N}$,

$$U_m(a_m, \boldsymbol{a}_{-m}) = \frac{1}{\lambda_2 \nu_l N_{cyc} + \lambda_1 (\lceil \frac{N_{cyc}}{f_l \tau} \rceil - \lceil \frac{N_{cyc}}{f_c \tau} \rceil)} = F_{l,2} \tag{3.32}$$

$$U_m(a'_m, \boldsymbol{a}_{-m}) = \frac{1}{\lambda_1 B + \lambda_2 B p \tau} \frac{1}{s_{a'_m}+1} = F_{c,2}(s_{a'_m}+1) \tag{3.33}$$

and

$$\psi(a_m, \boldsymbol{a}_{-m}) = \sum_{v=1}^{s_{a'_m}} F_{c,2}(v) + \sum_{n=1, n\neq a'_m}^{N} \sum_{v=1}^{s_n} F_{c,2}(v) + F_{l,2} + F_{l,2} \sum_{i=1, i\neq m}^{M} \mathbf{1}_{\{a_i=0\}} \tag{3.34}$$

$$\psi(a'_m, \boldsymbol{a}_{-m}) = \sum_{v=1}^{s_{a'_m}+1} F_{c,2}(v) + \sum_{n=1, n\neq a'_m}^{N} \sum_{v=1}^{s_n} F_{c,2}(v) + F_{l,2} \sum_{i=1, i\neq m}^{M} \mathbf{1}_{\{a_i=0\}} \tag{3.35}$$

Obviously,

$$\begin{aligned} \psi(a_m, \boldsymbol{a}_{-m}) - \psi(a'_m, \boldsymbol{a}_{-m}) &= -F_{c,2}(s_{a'_m}+1) + F_{l,2} \\ &= U_m(a_m, \boldsymbol{a}_{-m}) - U_m(a'_m, \boldsymbol{a}_{-m}) \end{aligned} \tag{3.36}$$

Case 3): Following the similar procedure as in Case 2), it is easy to show that $\psi(a_m, \boldsymbol{a}_{-m}) - \psi(a'_m, \boldsymbol{a}_{-m}) = U_m(a_m, \boldsymbol{a}_{-m}) - U_m(a'_m, \boldsymbol{a}_{-m})$.

In summary, the formulated game $\mathcal{G}_{off}$ is an exact potential game with the exact potential function $\psi(a_m, \boldsymbol{a}_{-m})$ specified by (3.23). According to [98], the formulated game $\mathcal{G}_{off}$ has at least one pure strategy NE. □

3.2.2 FDCO Algorithm

The FDCO algorithm starts with a mixed strategy, which is defined as $\boldsymbol{P}_m(t) = (P_{m0}(t), P_{m1}(t), P_{m2}(t), \cdots, P_{mN}(t))$ for player m ($m \in \mathcal{M}$) in the tth time slot. Here, $P_{mi}(t)$ ($i \in \mathcal{A}$) is the probability for player m to choose the action i in the tth time slot and $\sum_{i=0}^{N} P_{mi}(t) = 1$, $\forall t \geq 0, \forall m \in \mathcal{M}$. The key idea of the proposed FDCO algorithm is as follows: Each player m ($m \in \mathcal{M}$) takes an action in each time slot according to the current mixed strategy $\boldsymbol{P}_m(t)$. Then, the environment responds with a random reward $\hat{Rw}_m(a_m(t), \boldsymbol{a}_{-m}(t))$, which is defined as

$$\hat{Rw}_m(a_m(t), \boldsymbol{a}_{-m}(t)) = \frac{Rw_m(a_m(t), \boldsymbol{a}_{-m}(t))}{F_{c,2}(1)} \tag{3.37}$$

where

$$Rw_m(a_m(t), \boldsymbol{a}_{-m}(t)) = \begin{cases} F_{l,2}, & \text{if } a_m(t) = 0 \\ \frac{1}{\lambda_1 B + \lambda_2 B p \tau} r_m(a_m(t), \boldsymbol{a}_{-m}(t)), & \text{if } a_m(t) \in \mathcal{N} \end{cases} \tag{3.38}$$

The mobile device m uses this reaction to update its strategy $\boldsymbol{P}_m(t)$ based on the SLA technology [56] as follows:

$$P_{mi}(t+1) = \begin{cases} P_{mi}(t) + b\hat{Rw}_m(a_m(t), \boldsymbol{a}_{-m}(t))(1 - P_{mi}(t)), & \text{if } i = a_m(t) \\ P_{mi}(t) - b\hat{Rw}_m(a_m(t), \boldsymbol{a}_{-m}(t))P_{mi}(t), & \text{if } i \neq a_m(t) \end{cases} \tag{3.39}$$

This cycle repeats until a pure strategy NE is converged. It is easy to see from (3.39) that each mobile device m updates its own strategy $\boldsymbol{P}_m(t)$ based on the local information, thus, the proposed FDCO algorithm allows each mobile device to learn the optimal strategy through interactions with an unknown and dynamic environment independently and automatically. The details of the proposed FDCO algorithm is described in Algorithm 3.

Algorithm 3: FDCO Algorithm

1 **Initialization**: Set $t = 0$ and $P_{mi}(t) = (\frac{1}{N+1}, \frac{1}{N+1}, \cdots, \frac{1}{N+1})$, $\forall m \in \mathcal{M}$.
2 **While** there exists a $\boldsymbol{P}_m(t)$ ($m \in \mathcal{M}$), in which the maximum $P_{mi}(t)$ ($i \in \mathcal{A}$) is less than 0.99,
3 each player m takes an action $a_m(t)$ according to its current mixed strategy $\boldsymbol{P}_m(t)$.
4 At the end of the tth slot, each player m receives the random reward $Rw_m(a_m(t), \boldsymbol{a}_{-m}(t))$ specified by (3.38).
5 Each player m ($m \in \mathcal{M}$) updates its $\boldsymbol{P}_m(t)$ according to the following rule:

$$P_{mi}(t+1) = \begin{cases} P_{mi}(t) + b\hat{Rw}_m(a_m(t), \boldsymbol{a}_{-m}(t))(1 - P_{mi}(t)), & \text{if } i = a_m(t) \\ P_{mi}(t) - b\hat{Rw}_m(a_m(t), \boldsymbol{a}_{-m}(t))P_{mi}(t), & \text{if } i \neq a_m(t) \end{cases}$$

where $0 < b < 1$ is the step size.
6 Update $t = t + 1$.
7 **end While**

3.2.3 Convergence Analysis

In this subsection, the convergence of the proposed FDCO algorithm towards the pure strategy NEs of the formulated game $\mathcal{G}_{off}$ is investigated. Let $\boldsymbol{P}(t) = [\boldsymbol{P}_1^H(t), \cdots, \boldsymbol{P}_M^H(t)]$. Then, according to [56], the following two lemmas are obtained.

Lemma 3.1. *Define a piecewise-constant interpolation of $\boldsymbol{P}(t)$ as*

$$\boldsymbol{P}^b(t') = \boldsymbol{P}(t), \quad t' \in [tb, (t+1)b) \tag{3.40}$$

As the step size $b \to 0$, the sequence $\{\boldsymbol{P}^b(\cdot)\}$ converges weakly to $\boldsymbol{Q}(\cdot)$, which is the solution of the following ordinary differential equation (ODE):

$$\frac{d\boldsymbol{Q}}{dt'} = Y(\boldsymbol{Q}), \boldsymbol{Q}(0) = \boldsymbol{P}(0) \tag{3.41}$$

where $\boldsymbol{P}(0)$ is the initial decision selection probability matrix and $Y(\boldsymbol{Q})$ is the conditional expected function defined as

$$Y(\boldsymbol{P}) = \mathbb{E}[G(\boldsymbol{P}(t), \boldsymbol{a}(t), \hat{\boldsymbol{Rw}}(t))|\boldsymbol{P}(t) = \boldsymbol{P}] \tag{3.42}$$

In (3.42), $\hat{\boldsymbol{Rw}}(t) = (\hat{Rw}_1(a_1(t), \boldsymbol{a}_{-1}(t)), \cdots, \hat{Rw}_M(a_M(t), \boldsymbol{a}_{-M}(t)))$ *denotes the reward profile, and* $G(\cdot)$ *is a function of* $\boldsymbol{P}(t)$, $\boldsymbol{a}(t)$, *and* $\hat{\boldsymbol{Rw}}(t)$, *which is defined by (3.39).*

Lemma 3.2. *If the parameter* b *is sufficiently small, all the stable stationary points of (3.41) are the NEs of the formulated game* $\mathcal{G}_{off}$.

Based on Lemmas 3.1 and 3.2, the following theorem can be derived, which provides a sufficient condition for $\boldsymbol{P}(t)$ to converge towards the NE of the formulated game $\mathcal{G}_{off}$.

Theorem 3.3. *If there exists a bounded differentiable function* $K(\boldsymbol{P}) : \boldsymbol{P} \to \mathbb{R}$ *such that for some positive constant* c,

$$\frac{\partial K(\boldsymbol{P})}{\partial P_{mi}} - \frac{\partial K(\boldsymbol{P})}{\partial P_{mi'}} = c(h_{mi}(\boldsymbol{P}) - h_{mi'}(\boldsymbol{P})) \tag{3.43}$$

where $h_{mi}(\boldsymbol{P})$ *denotes the expected reward function of player* m *if it employs the pure strategy* i *while other players* m' $(\forall m' \in \mathcal{M}, m' \neq m)$ *employs the mixed strategy* $\boldsymbol{P}_{m'}$, *i.e.,*

$$h_{mi}(\boldsymbol{P}) = \sum_{\boldsymbol{a}_{-m}(t)=(a_1(t),\cdots,a_{m-1}(t),a_{m+1}(t),\cdots,a_M(t))} \mathbb{E}[\hat{Rw}_m(i, \boldsymbol{a}_{-m}(t))] \prod_{m' \in \mathcal{M}, m' \neq m} P_{m' a_{m'}(t)} \tag{3.44}$$

the proposed FDCO algorithm converges to a pure strategy NE of the formulated game $\mathcal{G}_{off}$.

The proof of Theorem 3.3 is shown in Appendix B.1.

Next, it will be shown that such sufficient condition is satisfied in the formulated game $\mathcal{G}_{off}$. Define $K(\boldsymbol{P})$ as

$$K(\boldsymbol{P}) = \mathbb{E}[\psi(a_m(t), \boldsymbol{a}_{-m}(t))|\boldsymbol{P}]$$

$$= \sum_{i=1}^{N+1} P_{mi} \mathbb{E}[\psi(i, \boldsymbol{a}_{-m}(t))|\boldsymbol{P}] \tag{3.45}$$

Then,

$$\frac{\partial K(\boldsymbol{P})}{\partial P_{mi}} = \mathbb{E}[\psi(i, \boldsymbol{a}_{-m}(t))|\boldsymbol{P}] \tag{3.46}$$

and

$$\begin{aligned}
&\frac{\partial K(\boldsymbol{P})}{\partial P_{mi}} - \frac{\partial K(\boldsymbol{P})}{\partial P_{mi'}} \\
&= \mathbb{E}[\psi(i, \boldsymbol{a}_{-m}(t))|\boldsymbol{P}] - \mathbb{E}[\psi(i', \boldsymbol{a}_{-m}(t))|\boldsymbol{P}] \\
&= \sum_{\boldsymbol{a}_{-m}(t)=(a_1(t),\cdots,a_{m-1}(t),a_{m+1}(t),\cdots,a_M(t))} (\psi(i, \boldsymbol{a}_{-m}(t)) - \psi(i', \boldsymbol{a}_{-m}(t))) \prod_{m' \in \mathcal{M}, m' \neq m} P_{m' a_{m'}(t)} \\
&= \sum_{\boldsymbol{a}_{-m}(t)=(a_1(t),\cdots,a_{m-1}(t),a_{m+1}(t),\cdots,a_M(t))} (U_m(i, \boldsymbol{a}_{-m}(t)) - U_m(i', \boldsymbol{a}_{-m}(t))) \prod_{n' \in \mathcal{M}, m' \neq m} P_{m' a_{m'}(t)}
\end{aligned} \tag{3.47}$$

According to (3.38),

$$\begin{aligned}
\mathbb{E}[Rw_m(i, \boldsymbol{a}_{-m}(t))] &= \begin{cases} F_{l,2}, & \text{if } i = 0 \\ \frac{1}{\lambda_1 B + \lambda_2 B p \tau} \frac{1}{s_i}, & \text{if } i \in \mathcal{N} \end{cases} \\
&= U_m(i, \boldsymbol{a}_{-m}(t))
\end{aligned} \tag{3.48}$$

Combining (3.48) and (3.44),

$$\begin{aligned}
&h_{mi}(\boldsymbol{P}) - h_{mi'}(\boldsymbol{P}) \\
&= \sum_{\boldsymbol{a}_{-m}(t)=(a_1(t),\cdots,a_{m-1}(t),a_{m+1}(t),\cdots,a_M(t))} (\mathbb{E}[\hat{Rw}_m(i, \boldsymbol{a}_{-m}(t))] - \mathbb{E}[\hat{Rw}_m(i', \boldsymbol{a}_{-m}(t))]) \\
&\prod_{m' \in \mathcal{M}, m' \neq m} P_{m' a_{m'}(t)} \\
&= \sum_{\boldsymbol{a}_{-m}(t)=(a_1(t),\cdots,a_{m-1}(t),a_{m+1}(t),\cdots,a_M(t))} (\lambda_1 B + \lambda_2 B p \tau)(U_m(i, \boldsymbol{a}_{-m}(t)) - U_m(i', \boldsymbol{a}_{-m}(t)))
\end{aligned}$$

$$\prod_{m' \in \mathcal{M}, m' \neq m} P_{m' a_{m'}(t)}$$
$$= (\lambda_1 B + \lambda_2 B p \tau)(\frac{\partial K(\boldsymbol{P})}{\partial P_{mi}} - \frac{\partial K(\boldsymbol{P})}{\partial P_{mi'}}) \tag{3.49}$$

Therefore, the sufficient condition specified in Theorem 3.3 is satisfied, or in other words, the NEs of the formulated game $\mathcal{G}_{off}$ can be achieved by using the proposed FDCO algorithm.

Note that the proposed scheme actually consists of two stages: Learning stage and Offloading stage. At the Learning stage, following the FDCO algorithm, each mobile device m learns to make the optimal computation offloading decision by transmitting pilot signals. In particular, each mobile device m which selects decision $a_m \in \mathcal{N}$ according to the current $\boldsymbol{P}_m(t)$ will contend on channel a_m based on the CSMA mechanism. If the m^{th} mobile device contends a_m successfully, it will transmit pilot signal on it and receive a non-zero rate in the current slot. Otherwise, it will keep silent and receives rate 0. Based on this result, each mobile device m updates its $\boldsymbol{P}_m(t)$ according to (3.39), and repeats this process in the next slot till convergence. After the convergence, at the Offloading stage, each mobile device m executes its computation task following its optimal computation offloading decision from the previous stage. Thus, the efficiency of the proposed scheme is mainly determined by the length of the Learning stage. Since the slot length during the Learning stage depends on the size of pilot signal, and such pilot signal can be set to be arbitrarily small, the slot length at the Learning stage can be very small (e.g., in unit of millisecond as shown in [99]), so that the proposed scheme can converge within a very short time and can achieve a high efficiency.

3.3 Performance Analysis

In this section, the performance of the proposed FDCO algorithm is analyzed in terms of the number of beneficial cloudlet computing mobile devices and the network-wide execution cost.

3.3.1 The Number of Beneficial Cloudlet Computing Mobile Devices

Definition 3.3. Given a random computation offloading decision profile $\boldsymbol{a}$, the m^{th} mobile device that chooses cloudlet computing (i.e., $a_m \in \mathcal{N}$) is **beneficial** if the cloudlet computing does not incur higher execution cost than the local computing (i.e., $O_m(a_m, \boldsymbol{a}_{-m}) \leq O_m(0, \boldsymbol{a}_{-m})$).

Since a larger number of beneficial cloudlet computing mobile devices implies a higher utilization of the cloudlet resources and thus a higher revenue for providing the cloudlet computing service, the number of beneficial cloudlet computing mobile devices has been widely used as an important performance metric for cloudlet-based MCC networks [113].

According to the concept of NEs specified by (3.21), the following Corollary is obtained.

Corollary 3.1. *At the NE $\boldsymbol{a}^*$ of the formulated game $\mathcal{G}_{off}$, if a mobile device m chooses channel a_m^* ($a_m^* \in \mathcal{N}$) for offloading, this mobile device must be beneficial. Therefore, the number of beneficial cloudlet computing mobile devices at NE $\boldsymbol{a}^*$ equals $\sum_{m=1}^{M} \mathbf{1}_{\{a_m^*>0\}}$.*

Theorem 3.4. *For the formulated game $\mathcal{G}_{off}$, the proposed FDCO algorithm maximizes the number of beneficial cloudlet computing mobile devices.*

The proof of Theorem 3.4 is shown in Appendix B.2.

3.3.2 Network-wide Execution Cost

Under the FDCO algorithm, let C_{FDCO} be the total execution cost of all the mobile devices, and NUM_{FDCO} be the number of beneficial cloudlet computing mobile devices. Then, C_{FDCO} is bounded based on the following theorem.

Theorem 3.5. *For the formulated game $\mathcal{G}_{off}$, the proposed FDCO algorithm does not incur a larger network-wide execution cost than computing locally by all mobile devices. Moreover, if $NUM_{FDCO} = M$, C_{FDCO} is upper-bounded by*

$$C_{FDCO} \leq (\min(F_{l,1}, \frac{Z}{N}) - F_{l,1})M + MF_{l,1} \tag{3.50}$$

Otherwise, if $0 < NUM_{FDCO} < M$, C_{FDCO} is upper-bounded by

$$C_{FDCO} \leq (\min(F_{l,1}, \frac{Z}{N}) - F_{l,1})N(\frac{F_{c,2}(1)}{F_{l,2}} - 1) + NF_{l,1} \tag{3.51}$$

where

$$Z = \frac{1}{F_{c,2}(1)}M + N\lambda_1\lceil\frac{N_{cyc}}{f_c\tau}\rceil + (N-1)\frac{1}{F_{c,2}(1)} \tag{3.52}$$

The proof of Theorem 3.5 is shown in Appendix B.3.

3.4 Simulation Results

In this section, the performance of the proposed FDCO algorithm will be demonstrated through simulations. The convergence behavior of the proposed algorithm is

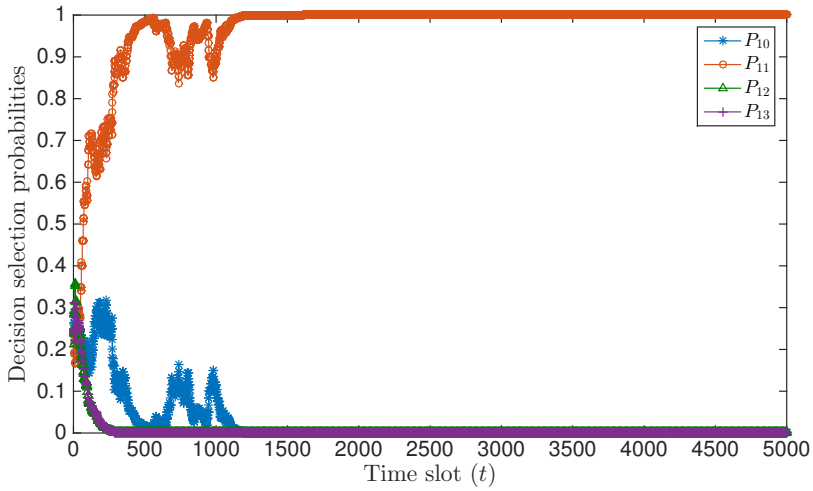

Figure 3.1: Evolution of the decision selection probabilities $P_{1i}(t)$ by the FDCO algorithm

first verified, and then its performance in terms of the number of beneficial cloudlet computing mobile devices and the network-wide execution cost is evaluated. In all simulations, set $M = 20$, $b = 0.1$, $\lambda_1 = \lambda_2 = 0.5$, $N_{cyc} = 1000$ Megacycles, $B = 10$, $f_m = 1$GHz, $f_c = 10$GHz, $\nu_l = 10W/GHz$, $p = 100$mW, $\tau = 20$ms. The parameter selection is based on the previous works in literature [61, 74, 114–117]. Note that the observations of this work are independent of the specific parameter settings.

3.4.1 Convergence Behavior of the FDCO Algorithm

Fig. 3.1 plots the evolution of decision selection probabilities $P_{mi}(t)$ $(i = 0, 1, 2, 3)$ for an arbitrary mobile device m $(m \in \mathcal{M})$. It can be seen that the decision selection probability vector $\boldsymbol{P}_m(t)$ evolves from a mixed strategy $(\frac{1}{N+1}, \frac{1}{N+1}, \cdots, \frac{1}{N+1})$ to a pure strategy $(0, 1, 0, 0)$ at the end, which illustrates the convergence of the proposed FDCO algorithm. To further verify the convergence towards a pure strategy NE by

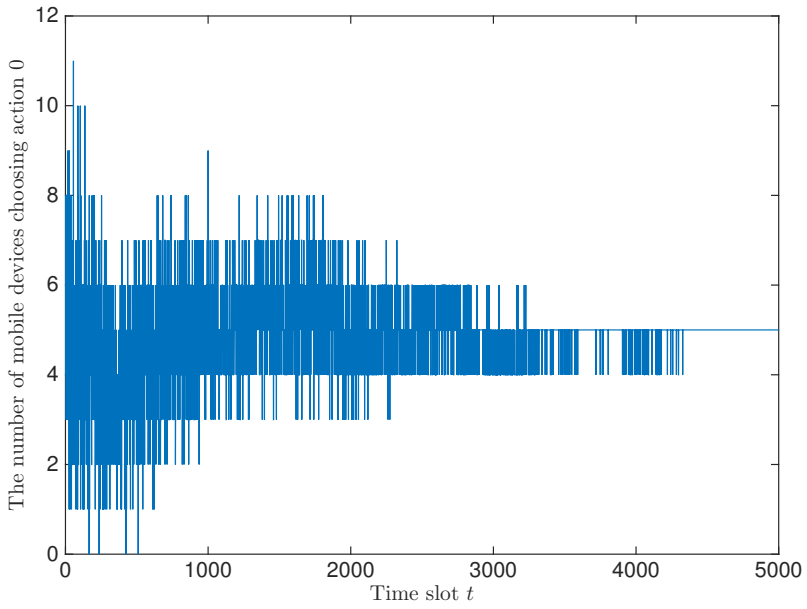

Figure 3.2: Evolution of the number of mobile devices choosing action 0 by the FDCO algorithm

the proposed FDCO algorithm, Figs.3.2-3.5 are plotted to show the evolution of the number of mobile devices that choose each action i ($i \in \{0, 1, 2, 3\}$). Note that $i = 0$ denotes the local computing, while $i = n$ ($1 \leq n \leq 3$) denotes the cloudlet computing via channel n. From these figures, it can be seen that after the system converges, there are 5 mobile devices choosing local computing, and on each channel, there are 5 mobile devices contending to offload their tasks to the cloudlet, which can be easily justified as a NE of the formulated game $\mathcal{G}_{off}$.

3.4.2 Performance Evaluation of the FDCO algorithm

In this section, the performance of the proposed FDCO algorithm in terms of the number of beneficial cloudlet computing mobile devices and the network-wide execu-

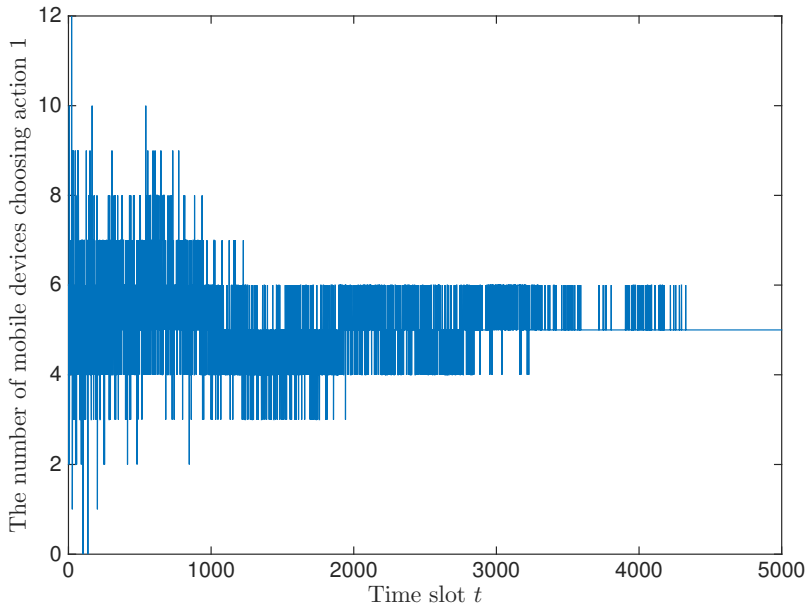

Figure 3.3: Evolution of the number of mobile devices choosing action 1 by the FDCO algorithm

tion cost is evaluated. Three ordinary greedy heuristic algorithms are also simulated for comparison. They are: 1) local computing, where all mobile devices choose to compute the tasks on their own devices; 2) cloudlet computing, where all mobile devices choose to offload the tasks to the cloudlet via a channel selected randomly with equal probability; and 3) random computing, where each mobile device chooses an action from $\{0\} \cup \mathcal{N}$ randomly with equal probability. If a mobile device chooses action 0, it will compute the task locally. Otherwise, it will offload its task to the cloudlet via channel n ($n \in \mathcal{N}$). Run experiments with different number of channels $N = 1, 2, \cdots, 5$, and repeat each experiment over 100 times.

Fig. 3.6 shows the comparison among the proposed FDCO algorithm, the cloudlet computing and the random computing algorithm in terms of the average number of

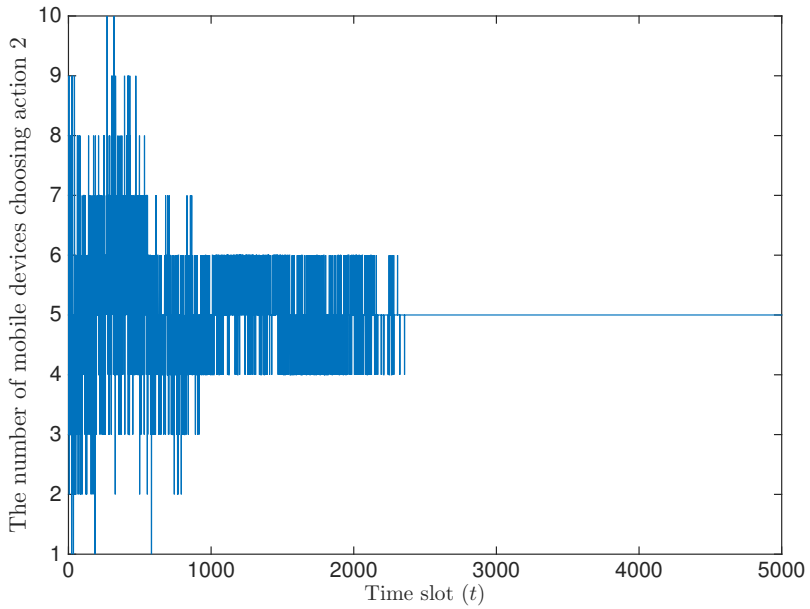

Figure 3.4: Evolution of the number of mobile devices choosing action 2 by the FDCO algorithm

beneficial cloudlet computing mobile devices. Note that the local computing algorithm is not included since its number of beneficial cloudlet computing mobile devices equals zero. From the figure, it can be seen that when $N \geq 4$, the number of beneficial cloudlet computing mobile devices achieved by the proposed FDCO algorithm equals M. That means if the spectrum resource is sufficient (i.e., $N \geq 4$ in this simulation), under the proposed FDCO algorithm, all mobile devices will automatically choose one channel to offload their tasks to the cloudlet. Then, according to the Corollary 3.1, all of them become beneficial cloudlet computing mobile devices. In all simulation scenarios, the proposed FDCO algorithm outperforms the other two. Even when $N = 5$, 42% performance gain can be achieved. This is because the proposed FDCO algorithm achieves an efficient wireless access coordination among multiple mobile de-

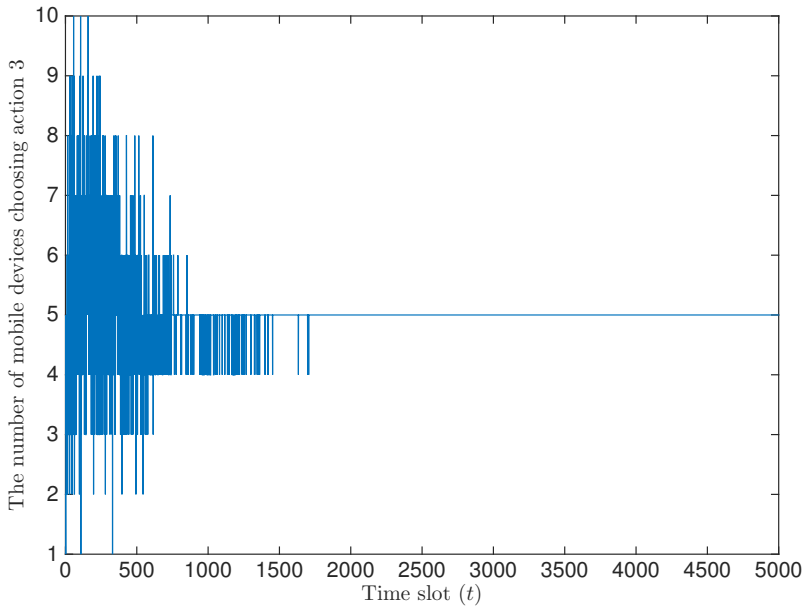

Figure 3.5: Evolution of the number of mobile devices choosing action 3 by the FDCO algorithm

vices, so as to maximize the number of beneficial cloudlet computing mobile devices.

Fig. 3.7 describes the comparison of the proposed FDCO algorithm with the three algorithms in terms of the average network-wide execution cost. From the figure, it can be seen that when the spectrum resource is small (e.g., $N \leq 3$ in this simulation), local computing can achieve less network-wide execution cost than cloudlet computing and random computing. It is because under this scenario, both cloudlet and random computing may cause severe congestion on wireless channels, and lead to high network-wide execution costs, nevertheless, such potential performance degradation due to the transmission collisions is avoided by using the local computing. However, with the increase of the number of available channels, the congestion issue on wireless channels can be alleviated, so that both cloudlet and random comput-

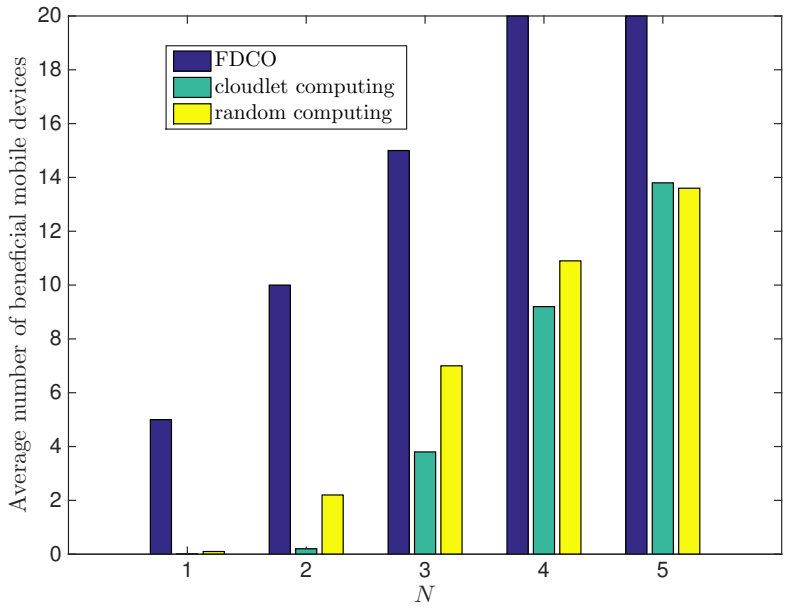

Figure 3.6: The comparison of different schemes in terms of the average number of beneficial cloudlet computing mobile device

ing methods outperform the local computing by adopting the low execution cost at the cloudlet. Under all simulation scenarios, the proposed FDCO algorithm always achieves the least network-wide execution cost by balancing the transmission and computation costs.

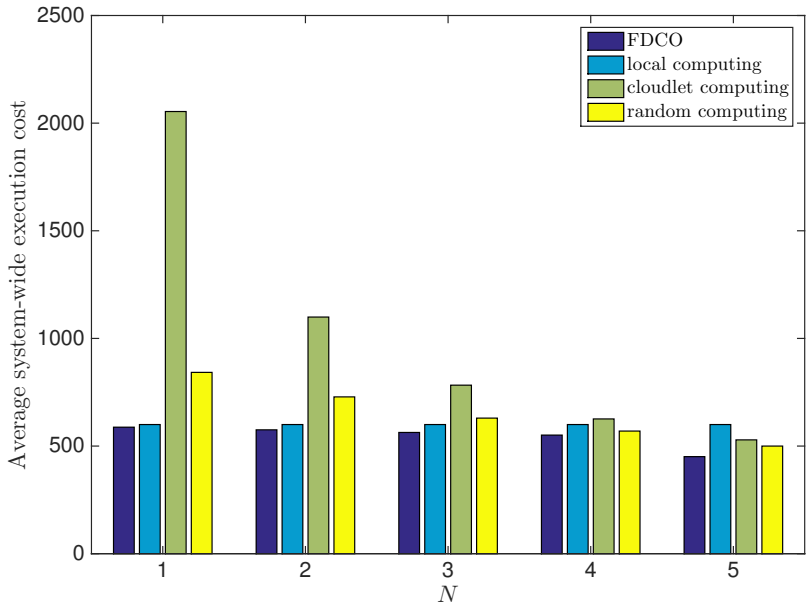

Figure 3.7: The comparison of different schemes in terms of the average network-wide execution cost

Chapter 4

Learning based Online Context-Aware Proactive Caching for an Energy Harvesting based Network

In this chapter, an energy harvesting based network consisting of a single MBS with grid power supply and multiple small-cells with energy harvesting is considered. The objective is to maximize the service ratio at the SBSs for both energy conservation and environmental protection. To address the mismatching issue between the random renewable energy generation and the random user request arrivals so as to fully reap the benefits of energy harvesting, a context-aware proactive caching problem is investigated. It is first formulated as an MDP framework. Then, to address the incomplete stochastic information about the network dynamics and the "curse of dimensionality" issue of the formulated MDP, an PDS-ARL algorithm is proposed, which learns on-the-fly the optimal context-aware proactive caching policy with a

high learning efficiency. The simulation results validate the efficacy of this algorithm by comparing it with baselines in terms of both the learning rate and the service ratio at the SBSs.

4.1 System Model and Problem Formulation

4.1.1 System Architecture

Consider an energy harvesting based network which comprises a single MBS and multiple non-overlapping small-cells. Each small-cell further consists of a SBS and several associated users, as shown in Fig. 4.1. Following [118, 119], assume SBSs operate in disjoint sub-channels with the MBS and the interference among neighbouring SBSs can be effectively eliminated by techniques such as enhanced inter-cell interference coordination (eICIC) or/and orthogonal multiple access [120, 121]. Hence, this chapter focuses on only one SBS and all other SBSs can work in the same way.

The time-slotted structure is considered and a set $\mathcal{M} = \{1, \cdots, M\}$ of M users is associated with the SBS. The MBS is powered by the grid power, while the SBS is powered through energy harvesting. The harvested energy is stored in a battery with a finite capacity of $e_{b_{max}}$. The SBS is connected with the MBS via a wired/wireless backhaul link to fetch contents and store them in its cache. Based on the local content popularity analysis [122–124], the SBS can proactively cache one content from each of L content categories $CC_1, \cdots, CC_L$ that the associated users may be interested in. Since contents may be outdated after a while, the content at the SBS is updated every T'' time slots, called one period [125, 126]. As in [118], since the focus here is on

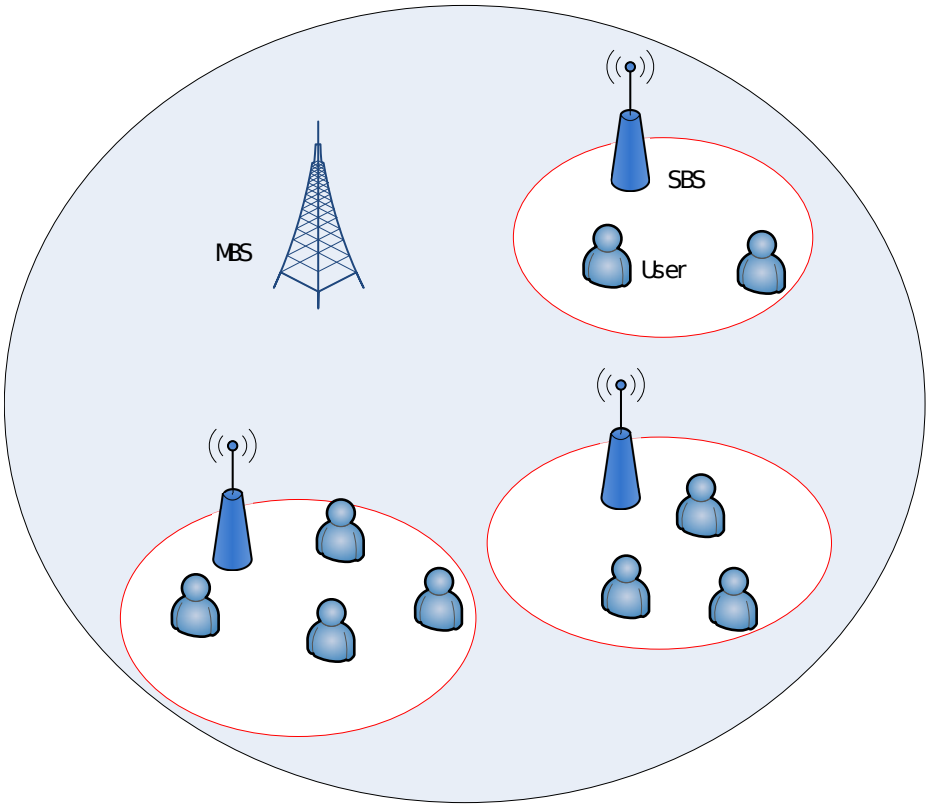

Figure 4.1: System Architecture.

the procedure of pushing contents from the SBS to its associated users, the time and energy consumption to fetch contents from the MBS is ignored. At each slot t, each user m ($m \in \mathcal{M}$) maintains a list of pushed contents $\mathcal{C}_{m,t}$ in its local cache. Notice that since users do not usually watch the same content twice, they will only cache the contents that haven't been watched yet.

4.1.2 Content Request Model

Along the time, each user m ($m \in \mathcal{M}$) generates an independent sequence of requests $\{V_{m,t}\}_t$, where $V_{m,t} \in \{0, 1\}$ is an indicator of a request in slot t. $V_{m,t} = 1$ if user m generates a request in slot t. Otherwise, $V_{m,t} = 0$. Define $\pi_m = Pr(V_{m,t} = 1)$,

which denotes the time-average activity probability for user m. In slot t, if $V_{m,t} = 1$, user m sends a request targeting content from one of L content categories according to its specific preference. A content preference profile (CPP) is associated with each individual user, which is defined as the probability distribution that each user requests a content from each category. Specifically, denote $\boldsymbol{CPP}_{m,t} = (P_{m,t,1}, \cdots, P_{m,t,L})$ as the CPP of user m in slot t, where $P_{m,t,l}$ $(l = 1, \cdots, L)$ denotes the probability that user m requests CC_l. By considering the facts that i) users will not request the content they already watched, and ii) the content of each category is updated every T'' slots, $\boldsymbol{CPP}_{m,t}$ evolves following a Markov process as follows:

- According to some measurement studies in [122–124], the initial CPP of each user can be obtained as $\boldsymbol{CPP}_m^0 = [P_{m,1}^0, \cdots, P_{m,L}^0]$, $m \in \mathcal{M}$. Thus, in the first slot of each period, indexed by $t = iT''$, $i = 0, 1, \cdots$, $\boldsymbol{CPP}_{m,t} = \boldsymbol{CPP}_m^0$.

- Within each period, user m ($m \in \mathcal{M}$) requests the content following the current profile $\boldsymbol{CPP}_{m,t}$. If user m is inactive or the request can not be satisfied in slot t, let

$$\boldsymbol{CPP}_{m,t+1} = \boldsymbol{CPP}_{m,t} \tag{4.1}$$

  Otherwise, if the request of CC_l ($l \in \{1, \cdots, L\}$) is satisfied, then, $\boldsymbol{CPP}_{m,t+1}$ is updated following [124]

$$P_{m,t+1,l'} = \begin{cases} 0, & l' = l; \\ \frac{P_{m,t,l'}}{\sum_{l'' \neq l} P_{m,t,l''}}, & l' \neq l. \end{cases} \tag{4.2}$$

4.1.3 Content Transmission Model

For the content transmission to user m ($m \in \mathcal{M}$), the required transmit power p_m satisfies

$$\mathbb{E}_{\xi}[W \log_2(1 + \frac{p_m|\xi|^2 d_m^{-\alpha}}{\sigma^2})] = r_0 \tag{4.3}$$

where r_0 is the target average data rate in each slot, W is the bandwidth of the SBS, ξ is the small-scale fast fading coefficient, d_m is the distance of user m from the SBS, α represents pathloss exponent, and σ^2 denotes noise power. Note that the expectation in (4.3) results from the fact that the slot duration, ranging from minutes to hours depending on the nature of services [87,90], is much longer than the fast fading at a timescale of milliseconds. Based on the channel model, the required transmit power p_m can be derived, and the corresponding energy consumption e_m can then be calculated as $e_m = p_m\tau$, where τ denotes the slot duration.

4.1.4 Working Modes and Battery Model

In each slot, the SBS has three possible working modes: sleep, reactive transmission, and proactive caching. In the sleep mode, the SBS does nothing, and all requests from the users will be blocked. In the reactive transmission mode, the SBS reactively satisfies one request initiated by the users in this slot. Since there may exist multiple requests, the SBS has to determine which request will be satisfied. Notice that although the SBS only targets at one user for transmission, the users with closer distances to the SBS can also receive the transmitted content, and will cache it if it has neither been watched nor cached yet, or just consume it if it is exactly what they

request currently. For this sense, more than one request can be satisfied by the SBS through reactive transmission at one time. In the proactive caching mode, the SBS proactively pushes an unrequested content to the users who may be interested in the future. In this mode, the SBS has to determine which content should be pushed to which users, and how much transmit power needed so that all the selected users can receive the content successfully. Obviously, in this case, the energy consumption is determined by the user with the worst channel condition.

Denote $e_{b_t} \leq e_{b_{max}}$ as the battery state at the beginning of slot t, and e_{c_t} as the energy consumption in slot t. $e_{c_t} = 0$ if the SBS is in the sleep mode and $e_{c_t} = e_m$ ($m \in \mathcal{M}$) if the SBS transmits to user m via either reactive transmission or proactive caching (where m should be the user with the worst channel condition). Since e_{c_t} is always constrained by the available battery energy, $e_{c_t} \leq e_{b_t}$. Let e_{h_t} be the harvested energy in slot t and be ergodic [118]. Then, the battery energy state evolves as

$$e_{b_{t+1}} = min\{e_{b_{max}}, e_{b_t} - e_{c_t} + e_{h_t}\} \tag{4.4}$$

4.1.5 Objective

When a user requests a content that is neither in its local cache nor satisfied by the SBS transmission, the request is then handled by the MBS. This chapter aims at maximizing the service ratio SR at the SBS, defined as the ratio between the number of requests served by the SBS and the number of total arrived requests. Mathematically, the objective can be formulated as

$$\max_{a_1,\cdots,a_T} SR = \lim_{T\to\infty} \frac{\sum_{t=0}^{T-1} \Omega(t)}{\sum_{t=0}^{T-1} \sum_{m=1}^{M} V_{m,t}} \tag{4.5}$$

where T is the total number of slots, $\Omega(t)$ is the number of requests satisfied by the SBS in slot t, $\sum_{t=0}^{T}\sum_{m=1}^{M} V_{m,t}$ is the total number of arrived requests within T slots from all users, and the decision variables $\{a_1, \cdots, a_T\}$ represent the actions taken by the SBS in each slot, denoting which content is transmitted to which user.

However, obtaining the optimal $\{a_1, \cdots, a_T\}$ in problem (4.5) is quite challenging due to the following reasons:

- In each slot t, the number of feasible actions can be huge. It is because there may be multiple requests from the users that satisfy $e_{c_t} \leq e_{b_t}$, so that any of these users can be the transmission target of the SBS. Moreover, for each feasible target, multiple choices may exist on determining the transmission content under the proactive caching mode. Such large number of feasible actions in each slot makes decision making very difficult.

- Due to the temporal correlation of both the battery states and the content preference profiles at each user, the current action not only affects the immediate reward, but also influences the network evolution in the future. Hence, in each slot, with the observed instantaneous network information, it is challenging to determine the optimal action so as to optimize the long term performance.

To address these issues, in the following, an PDS-ARL algorithm is proposed to solve this problem effectively and efficiently.

4.2 PDS-ARL algorithm

Due to the involved Markov process (both the battery energy evolution and the CPP evolution of each user), MDP [14] becomes an effective mathematical tool for solving problem (4.5). Next, problem (4.5) is first formulated as an MDP framework.

4.2.1 MDP Formulation

Since an standard MDP framework contains four elements: states, actions, reward function, and state transition, each of them will be described as follows.

States

In each slot t, to make action control, the SBS has to observe

- e_{b_t}: the battery energy state in slot t. Only if there is sufficient energy, the SBS can make the transmission action, either reactive transmission or proactive caching. Otherwise, it enters sleep.
- ζ_t: If each period is defined to consist of T'' stages, ζ_t denotes the index of the stage within the current period, and $\zeta_t \in \{0, 1, \cdots, T''-1\}$. If $\zeta_t = T''-1$, i.e., the last stage of a period, since the content of each category will be updated in the next slot, it is meaningless to do proactive caching in the current time slot.
- $\boldsymbol{\mathcal{C}}_t = (\mathcal{C}_{1,t}, \cdots, \mathcal{C}_{M,t})$: the pushed content lists at each user's local cache in slot t. If $\zeta_t = 0$, $\mathcal{C}_{m,t} = \emptyset$. Otherwise, $\mathcal{C}_{m,t} \subseteq \{CC_1, \cdots, CC_L\}$, $m \in \mathcal{M}$.
- $\boldsymbol{CPP}_t = (\boldsymbol{CPP}_{1,t}, \cdots, \boldsymbol{CPP}_{M,t})$: the content preference profiles for each user in slot t. If $\zeta_t = 0$, $\boldsymbol{CPP}_{m,t} = \boldsymbol{CPP}_m^0$.

- $\boldsymbol{Re}_t = (Re_{1,t}, \cdots, Re_{M,t})$: the content requests from each user in slot t. $Re_{m,t} = CC_l$ ($l \in \{1, \cdots, L\}$) denotes that user m requests the content from category CC_l, while $Re_{m,t} = 0$ denotes that user m is inactive.

In summary, the system state in each slot t can be denoted as $\boldsymbol{S}_t = (e_{b_t}, \zeta_t, \boldsymbol{\mathcal{C}}_t, \boldsymbol{CPP}_t, \boldsymbol{Re}_t)$.

To make the action control tractable, the battery energy e_{b_t} is discretized into a finite set. Similar assumption has also been adopted in the literature, e.g., [127] and [128]. Specifically, the battery energy is discretized with minimum energy unit of e_{unit}. Then, $e_{b_t} \in \{0, 1, \cdots, \tilde{e}_{b_{max}}\}$, where $e_{b_t} = i$ ($i \in \{0, 1, \cdots, \tilde{e}_{b_{max}}\}$) corresponds to ie_{unit} amount of energy in the battery and $e_{b_{max}} = \tilde{e}_{b_{max}} e_{unit}$. Similarly, the amount of energy arrived in each slot t is also discretized with $e_{h_t} = i$ representing there are ie_{unit} energy arrival. To discretize e_{c_t}, a series of distances $0 = D_1 < D_2 < \cdots < D_w = radii$ are selected, where $radii$ denotes the small-cell radius. If user m's ($m \in \mathcal{M}$) distance d_m from the SBS satisfies $D_{i-1} < d_m \leq D_i$, $i \in \{2, \cdots, w\}$, d_m is then discretized as D_i. $\{D_1, \cdots, D_w\}$ are selected so that after discretization, $e_m = p_m \tau = int_m e_{unit}$, where int_m is a positive integer.

Denote $\mathcal{S}$ as the system state space. Since $\boldsymbol{\mathcal{C}}_t$, $\boldsymbol{CPP}_t$ and $\boldsymbol{Re}_t$ are dependent on ζ_t, $\mathcal{S}$ can be divided into several subspaces with respect to different ζ_t. Specifically,

- if $\zeta_t = 0$, $\boldsymbol{\mathcal{C}}_t = \emptyset$, $\boldsymbol{CPP}_t = [\boldsymbol{CPP}_1^0, \cdots, \boldsymbol{CPP}_M^0]$, and $Re_{m,t} \in \{0, CC_1, \cdots, CC_L\}$, $m = 1, \cdots, M$. Thus, the associated subspace $\mathcal{S}_{\zeta_t=0}$ has a size of

$$|\mathcal{S}_{\zeta_t=0}| = (\tilde{e}_{b_{max}} + 1)(L+1)^M \tag{4.6}$$

- if $\zeta_t = i$, $i \in \{1, \cdots, T'' - 1\}$, it means within the current period, at most i

contents have been transmitted by the SBS, and for each user, it has cached at most i contents, and has requested at most i contents. Thus, the associated subspace $\mathcal{S}_{\zeta_t=i}$ has a size of

$$|\mathcal{S}_{\zeta_t=i}| = (\tilde{e}_{b_{max}} + 1)C_L^i[f(i,0) + f(i,1) + \cdots + f(i,i)]^M \tag{4.7}$$

where $f(i,j)$, $j \in \{0, 1, \cdots, i\}$, represents the number of all possible combinations $(\mathcal{C}_{m,t}, \boldsymbol{CPP}_{m,t}, Re_{m,t})$ for each user m ($m \in \mathcal{M}$) when it has cached j contents from the i contents transmitted by the SBS. $f(i,j)$ can be formulated as

$$f(i,j) = C_i^j(C_{L-j}^0(L+1) + C_{L-j}^1(C_{L-1}^1 + 1) + \cdots + C_{L-j}^{x_{i,j}}(C_{L-x_{i,j}}^1 + 1)) \tag{4.8}$$

Since users won't cache the contents they have already viewed, $C_{L-j}^y(C_{L-y}^1 + 1)$, $y = 0, 1, \cdots, x_{i,j}$, denotes the number of all possible combinations $(\boldsymbol{CPP}_{m,t}, Re_{m,t})$ for each user m ($m \in \mathcal{M}$) when it has watched y contents from the $(L-j)$ non-cached contents, where $x_{i,j}$ satisfies $x_{i,j} + j \leq L$ and $x_{i,j} \leq i$.

In summary, the overall state space size $|\mathcal{S}|$ can be calculated as

$$\begin{aligned} |\mathcal{S}| &= |\mathcal{S}_{\zeta_t=0}| + \sum_{i=1}^{T''-1} |\mathcal{S}_{\zeta_t=i}| \\ &= (\tilde{B}_{max} + 1)\Upsilon(L, M, T'') \end{aligned} \tag{4.9}$$

where

$$\Upsilon(L, M, T'') = (L+1)^M + \sum_{i=1}^{T''-1} C_L^i(f(i,0) + f(i,1) + \cdots + f(i,i))^M$$

From (4.9), it can be seen that the overall state space size can be quite huge, even with small values of L, M, and T''. For example, by setting $L = M = T'' = 3$, $\Upsilon(L, M, T'')$ already becomes 411565.

Actions

As modelled before, in each slot, the SBS has three possible working modes: sleep, reactive transmission, and proactive caching. However, these three modes may not always be available at the SBS. For example, if $\zeta_t = T'' - 1$, proactive caching is unavailable. Thus, the available actions are state-dependent. In the following, the action space, denoted as $\mathcal{A}(\boldsymbol{S}_t)$, will be described with respect to different system states.

- If $\zeta_t = T'' - 1$ and $\boldsymbol{Re}_t = [0, \cdots, 0]$, it means at the last stage of a period, there is no request from the users. In this case, the SBS does nothing, no matter how much energy available in the battery, i.e., $a_t \in \mathcal{A}(\boldsymbol{S}_t) = \{0\}$, where 0 denotes the SBS enters sleep.

- If $\zeta_t = T'' - 1$ and $\boldsymbol{Re}_t \neq [0, \cdots, 0]$, it means at the last stage of a period, there exist requests from the users. Denote $\mathcal{M}_t^r$ as a user set consisting of all users who request contents that have not been locally cached yet and can be satisfied by the battery energy. If $\mathcal{M}_t^r \neq \emptyset$, the reactive transmission mode is available, and the SBS can select one user from the set $\mathcal{M}_t^r$ for reactive transmission. Thus, in this case, $a_t \in \mathcal{A}(\boldsymbol{S}_t) = \{0, \mathcal{M}_t^r\}$, and the number of total possible actions equals $1 + |\mathcal{M}_t^r|$.

- If $\zeta_t \neq T'' - 1$ and $\boldsymbol{Re}_t = [0, \cdots, 0]$, it means there is no request from the users and proactive caching can be available. Denote $\mathcal{M}_{l,t}^p$ ($l \in \{1, \cdots, L\}$) as a set of users who have neither watched nor cached CC_l yet within the current period and satisfy $e_m \leq e_{b_t}$. If there exists $\mathcal{M}_{l,t}^p \neq \emptyset$, $l \in \{1, \cdots, L\}$, the proactive

caching mode becomes available, and the SBS can select one user from the set $\mathcal{M}_{l,t}^{p}$ as the user with the worst channel condition for proactively pushing CC_l. Thus, in this case, $a_t \in \mathcal{A}(\boldsymbol{S}_t) = \{0, \mathcal{M}_{1,t}^{p}, \cdots, \mathcal{M}_{L,t}^{p}\}$, and the number of total possible actions equals $1 + \sum_{l=1}^{L} |\mathcal{M}_{l,t}^{p}|$.

- If $\zeta_t \neq T''-1$ and $\boldsymbol{Re}_t \neq [0, \cdots, 0]$, all three working modes can be available, and $a_t \in \mathcal{A}(\boldsymbol{S}_t) = \{0, \mathcal{M}_{1,t}^{p}, \cdots, \mathcal{M}_{L,t}^{p}\}$. Since $\mathcal{M}_t^r \subseteq \{\mathcal{M}_{l,t}^{p}\}_{l\in\{1,\cdots,L\}}$, $\mathcal{A}(\boldsymbol{S}_t) = \{0, \{\mathcal{M}_{l,t}^{p}\}_{l\in\{1,\cdots,L}\}$, so that the number of total possible actions in this case equals $1 + \sum_{l=1}^{L} |\mathcal{M}_{l,t}^{p}|$.

Based on all above analysis, it is concluded that the action space size can be up to $1 + ML$.

Reward Function

The per-slot reward for each user m ($m \in \mathcal{M}$) is defined as $rw_{m,t}$, which denotes whether the current request is satisfied by the SBS or not. Mathematically, it can be expressed as

$$rw_{m,t} = \begin{cases} 0, \quad V_{m,t} = 0 \\ 0, \quad V_{m,t} = 1, Re_{m,t} \notin \mathcal{C}_{m,t}, \text{and } Re_{m,t} \text{ is not satisfied by the SBS transmission} \\ 1, \quad V_{m,t} = 1, Re_{m,t} \notin \mathcal{C}_{m,t}, \text{and } Re_{m,t} \text{ is satisfied by the SBS transmission} \\ 1, \quad V_{m,t} = 1, Re_{m,t} \in \mathcal{C}_{m,t} \end{cases} \tag{4.10}$$

Notice that in each slot t, if user m is inactive, i.e., $V_{m,t} = 0$, $rw_{m,t} = 0$. If user m is active, i.e., $V_{m,t} = 1$, $rw_{m,t} = 0$ when the content request can neither be found

in the local cache nor be satisfied by the SBS transmission. Otherwise, $rw_{m,t} = 1$. By summing up the rewards of all users, the overall reward rw_t is obtained, which denotes the number of total requests satisfied by the SBS in slot t and is formulated as

$$rw_t(\boldsymbol{S}_t, a_t) = \sum_{m=1}^{M} rw_{m,t} \tag{4.11}$$

State Transition

The state transition can be expressed as the following conditional probability:

$$\begin{aligned} &Pr(\boldsymbol{S}_{t+1}|\boldsymbol{S}_t, a_t) \\ &= Pr((e_{b_{t+1}}, \zeta_{t+1}, \boldsymbol{\mathcal{C}}_{t+1}, \boldsymbol{CPP}_{t+1}, \boldsymbol{Re}_{t+1})|(e_{b_t}, \zeta_t, \boldsymbol{\mathcal{C}}_t, \boldsymbol{CPP}_t, \boldsymbol{Re}_t), a_t) \\ &= Pr(e_{b_{t+1}}|e_{b_t}, a_t) Pr(\zeta_{t+1}|\zeta_t) Pr(\boldsymbol{\mathcal{C}}_{t+1}|\boldsymbol{\mathcal{C}}_t, \boldsymbol{Re}_t, \zeta_t, a_t) Pr(\boldsymbol{CPP}_{t+1}|\boldsymbol{CPP_t}, \boldsymbol{Re}_t, \zeta_t) \\ &\cdot Pr(\boldsymbol{Re}_{t+1}|\boldsymbol{CPP}_{t+1}) \end{aligned} \tag{4.12}$$

In the following, all conditional probabilities in (4.12) are calculated.

$Pr(e_{b_{t+1}}|e_{b_t}, a_t)$: Since $e_{b_{t+1}} = \min(e_{b_{max}}, e_{b_t} - e_{c_t} + e_{h_t})$, i.e.,

$$e_{b_{t+1}} = \begin{cases} e_{b_t} - e_{c_t} + e_{h_t}, & \text{if} \quad e_{h_t} < e_{b_{max}} + e_{c_t} - e_{b_t} \\ e_{b_{max}}, & \text{otherwise} \end{cases} \tag{4.13}$$

then,

$$Pr(e_{b_{t+1}}|e_{b_t}, a_t) = \begin{cases} P_{e_h}(e_{h_t} = e_{b_{t+1}} - e_{b_t} + e_{c_t}), & \text{if} \quad e_{b_{t+1}} < e_{b_{max}} \\ \sum_{e_{h_t} \geq e_{b_{max}} + e_{c_t} - e_{b_t}} P_{e_h}(e_{h_t}), & \text{if} \quad e_{b_{t+1}} = e_{b_{max}} \end{cases} \tag{4.14}$$

where $P_{e_h}(\cdot)$ denotes the stochastic model of the energy harvesting process. Notice that due to the high unpredictability of renewable energy, such a stochastic model may not be always available *a priori*.

$Pr(\zeta_{t+1}|\zeta_t)$: If $0 \leq \zeta_t < T'' - 1$, $\zeta_{t+1} = \zeta_t + 1$, otherwise, $\zeta_{t+1} = 0$. Thus,

$$Pr(\zeta_{t+1}|\zeta_t) = \begin{cases} 1, & \text{if} \quad 0 \leq \zeta_t < T'' - 1, \zeta_{t+1} = \zeta_t + 1 \\ 1, & \text{if} \quad \zeta_t = T'' - 1, \zeta_{t+1} = 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.15}$$

$Pr(\boldsymbol{\mathcal{C}}_{t+1}|\boldsymbol{\mathcal{C}}_t, \boldsymbol{Re}_t, \zeta_t, a_t)$: Since $Pr(\boldsymbol{\mathcal{C}}_{t+1}|\boldsymbol{\mathcal{C}}_t, \boldsymbol{Re}_t, \zeta_t, a_t) = Pr(\mathcal{C}_{1,t+1}|\mathcal{C}_{1,t}, Re_{1,t}, \zeta_t, a_t) \cdots Pr(\mathcal{C}_{M,t+1}|\mathcal{C}_{M,t}, Re_{M,t}, \zeta_t, a_t)$, it only needs to compute $Pr(\mathcal{C}_{m,t+1}|\mathcal{C}_{m,t}, Re_{m,t}, \zeta_t, a_t)$ for an arbitrary user m, $m \in \mathcal{M}$.

When $\zeta_t = T'' - 1$, since the content of each category will be updated in the next slot, the pushed list in the local cache will be reset to empty at slot $t + 1$, i.e., $\mathcal{C}_{m,t+1} = \emptyset$, while when $\zeta_t < T'' - 1$, $\mathcal{C}_{m,t+1}$ will be updated based on the current $\mathcal{C}_{m,t}$, $Re_{m,t}$, and a_t. Specifically, in slot t, if user m's request $Re_{m,t}$ ($Re_{m,t} \neq 0$) can be satisfied by its local cache, i.e., $Re_{m,t} \in \mathcal{C}_{m,t}$, $Re_{m,t}$ will be removed from its local cache in slot $t + 1$. If the SBS transmits CC_l ($CC_l \neq Re_{m,t}, l \in \{1, \cdots, L\}$) which has neither been viewed nor cached by user m yet with energy $e_{c_t} \geq e_m$, CC_l will be then added to user m's local cache in slot $t + 1$. Otherwise, the pushed list in user m's local cache will keep unchanged in slot $t + 1$. Mathematically, $\mathcal{C}_{m,t+1}$ can be formulated as:

- If $\zeta_t = T'' - 1$,

$$\mathcal{C}_{m,t+1} = \emptyset \tag{4.16}$$

- If $\zeta_t < T'' - 1$,

– when $Re_{m,t} \neq 0$, and $Re_{m,t} \notin \mathcal{C}_{m,t}$,

$$\mathcal{C}_{m,t+1} = \begin{cases} \mathcal{C}_{m,t}, & \text{if } e_{c_t} < e_m \\ \mathcal{C}_{m,t}, & \text{if the SBS transmits } Re_{m,t} \text{ with } e_{c_t} \geq e_m \\ \mathcal{C}_{m,t}, & \text{if the SBS transmits a content that has already been} \\ & \text{viewed or cached by user } m \text{ with } e_{c_t} \geq e_m \\ \mathcal{C}_{m,t} \bigcup \{CC_l\}, & \text{if the SBS transmits } CC_l \ (CC_l \neq Re_{m,t}, l \in \\ & \{1, \cdots, L\}) \text{ that has neither been viewed nor cached by} \\ & \text{user } m \text{ yet with } e_{c_t} \geq e_m \end{cases} \tag{4.17}$$

– When $Re_{m,t} \neq 0$, and $Re_{m,t} \in \mathcal{C}_{m,t}$,

$$\mathcal{C}_{m,t+1} = \begin{cases} \mathcal{C}_{m,t} \setminus Re_{m,l}, & \text{if } e_{c_t} < e_m \\ \mathcal{C}_{m,t} \setminus Re_{m,l}, & \text{if the SBS transmits a content that has already} \\ & \text{been viewed or cached by user } m \text{ with } e_{c_t} \geq e_m \\ \{\mathcal{C}_{m,t} \setminus Re_{m,l}\} \bigcup \{CC_l\}, & \text{if the SBS transmits } CC_l \ (l \in \{1, \\ & \cdots, L\}) \text{ that has neither been viewed nor cached by user} \\ & m \text{ yet with } e_{c_t} \geq e_m \end{cases} \tag{4.18}$$

– When $Re_{m,t} = 0$,

$$\mathcal{C}_{m,t+1} = \begin{cases} \mathcal{C}_{m,t}, & \text{if } e_{c_t} < e_m \\ \mathcal{C}_{m,t}, & \text{if the SBS transmits a content that has already been viewed or cached by user } m \text{ with } e_{c_t} \geq e_m \\ \mathcal{C}_{m,t} \bigcup \{CC_l\}, & \text{if the SBS transmits } CC_l \ (l \in \{1, \cdots, L\}) \text{ that has neither been viewed nor cached by user } m \text{ yet with } e_{c_t} \geq e_m \end{cases} \tag{4.19}$$

Correspondingly, the transition probability of $\mathcal{C}_{m,t}$ can be calculated as

- If $\zeta_t = T'' - 1$,

$$Pr(\mathcal{C}_{m,t+1} | \mathcal{C}_{m,t}, Re_{m,t}, \zeta_t, a_t) = \begin{cases} 1, & \text{if } \mathcal{C}_{m,t+1} = \emptyset \\ 0, & \text{otherwise} \end{cases} \tag{4.20}$$

- If $\zeta_t < T'' - 1$,

  – when $Re_{m,t} \neq 0$, and $Re_{m,t} \notin \mathcal{C}_{m,t}$,

  $$Pr(\mathcal{C}_{m,t+1} | \mathcal{C}_{m,t}, Re_{m,t}, \zeta_t, a_t)$$

$$= \begin{cases} 1, & \text{if } e_{c_t} < e_m \text{ and } \mathcal{C}_{m,t+1} = \mathcal{C}_{m,t} \\ 1, & \text{if the SBS transmits } Re_{m,t} \text{ with } e_{c_t} \geq e_m \text{ and } \mathcal{C}_{m,t+1} = \mathcal{C}_{m,t} \\ 1, & \text{if the SBS transmits a content that has already been viewed} \\ & \text{or cached by user } m \text{ and } \mathcal{C}_{m,t+1} = \mathcal{C}_{m,t} \\ 1, & \text{if the SBS transmits } CC_l \ (CC_l \neq Re_{m,t}, l \in \{1, \cdots, L\}) \text{ that} \\ & \text{has neither been viewed nor cached by user } m \text{ yet with } e_{c_t} \geq e_m, \\ & \text{and } \mathcal{C}_{m,t+1} = \mathcal{C}_{m,t} \bigcup \{CC_l\} \\ 0, & \text{otherwise} \end{cases} \tag{4.21}$$

– When $Re_{m,t} \neq 0$, and $Re_{m,t} \in \mathcal{C}_{m,t}$,

$$Pr(\mathcal{C}_{m,t+1} | \mathcal{C}_{m,t}, Re_{m,t}, \zeta_t, a_t)$$

$$= \begin{cases} 1, & \text{if } e_{c_t} < e_m \text{ and } \mathcal{C}_{m,t+1} = \mathcal{C}_{m,t} \setminus Re_{m,l} \\ 1, & \text{if the SBS transmits a content that has already been viewed} \\ & \text{or cached by user } m \text{ with } e_{c_t} \geq e_m \text{ and } \mathcal{C}_{m,t+1} = \mathcal{C}_{m,t} \setminus Re_{m,l} \\ 1, & \text{if the SBS transmits } CC_l \ (l \in \{1, \cdots, L\}) \text{ that has neither been} \\ & \text{viewed nor cached by user } m \text{ yet with } e_{c_t} \geq e_m, \text{ and } \mathcal{C}_{m,t+1} = \\ & \{\mathcal{C}_{m,t} \setminus Re_{m,l}\} \bigcup \{CC_l\} \\ 0, & \text{otherwise} \end{cases} \tag{4.22}$$

– When $Re_{m,t} = 0$,

$$Pr(\mathcal{C}_{m,t+1} | \mathcal{C}_{m,t}, Re_{m,t}, \zeta_t, a_t)$$

$$= \begin{cases} 1, & \text{if } e_{c_t} < e_m \text{ and } \mathcal{C}_{m,t+1} = \mathcal{C}_{m,t} \\ 1, & \text{if the SBS transmits a content that has already been viewed} \\ & \text{or cached by user } m \text{ with } e_{c_t} \geq e_m \text{ and } \mathcal{C}_{m,t+1} = \mathcal{C}_{m,t} \\ 1, & \text{if the SBS transmits } CC_l \ (l \in \{1, \cdots, L\}) \text{ that has neither} \\ & \text{been viewed nor cached by user } m \text{ yet, and } \mathcal{C}_{m,t+1} = \mathcal{C}_{m,t} \bigcup \{CC_l\} \\ 0, & \text{otherwise} \end{cases} \tag{4.23}$$

$Pr(\boldsymbol{CPP}_{t+1}|\boldsymbol{CPP_t}, \boldsymbol{Re}_t, \zeta_t)$: $Pr(\boldsymbol{CPP}_{t+1}|\boldsymbol{CPP}_t, \boldsymbol{Re}_t, \zeta_t) = Pr(\boldsymbol{CPP}_{1,t+1}|\boldsymbol{CPP}_{1,t}, Re_{1,t}, \zeta_t) \cdots Pr(\boldsymbol{CPP}_{M,t+1}|\boldsymbol{CPP}_{M,t}, Re_{M,t}, \zeta_t)$. Similarly, I only focus on computing $Pr(\boldsymbol{CPP}_{m,t+1}|\ \boldsymbol{CPP}_{m,t}, Re_{m,t}, \zeta_t)$ for an arbitrary user m, $\forall m \in \mathcal{M}$.

If $\zeta_t = T'' - 1$, the content preference profile for user m will be reset to the initial value $\boldsymbol{CPP}_m^0$ in slot $t+1$, i.e., $\boldsymbol{CPP}_{m,t+1} = \boldsymbol{CPP}_m^0$. Otherwise, $\boldsymbol{CPP}_{m,t+1}$ will be updated based on $\boldsymbol{CPP}_{m,t}$ and $Re_{m,t}$. Specifically, if user m is inactive in slot t, i.e., $Re_{m,t} = 0$, its content preference profile will keep unchanged in slot $t+1$, i.e., $\boldsymbol{CPP}_{m,t+1} = \boldsymbol{CPP}_{m,t}$. If user m requests $Re_{m,t}$ ($Re_{m,t} \neq 0$) in slot t, since such a request can always be satisfied by either the SBS via proactive caching or reactive transmission, or the MBS when it is blocked by the SBS, the probability that user m requests $Re_{m,t}$ will become 0 in slot $t+1$ and $\boldsymbol{CPP}_{m,t+1}$ is updated following:

$$P_{m,t+1,l} = \begin{cases} 0, & \text{if } CC_l = Re_{m,t} \\ \frac{P_{m,t,l}}{\sum_{CC_{l'} \neq Re_{m,t}} P_{m,t,l'}}, & \text{if } CC_l \neq Re_{m,t} \end{cases} \tag{4.24}$$

The transition probability of $\boldsymbol{CPP}_{m,t}$ can then be calculated as

$Pr(\boldsymbol{CPP}_{m,t+1}|\boldsymbol{CPP}_{m,t}, Re_{m,t}, \zeta_t)$

$$= \begin{cases} 1, & \text{if } \zeta_t = T'' - 1 \text{ and } \boldsymbol{CPP}_{m,t+1} = \boldsymbol{CPP}_m^0 \\ 1, & \text{if } \zeta_t < T'' - 1,\ Re_{m,t} = 0, \text{ and } \boldsymbol{CPP}_{m,t+1} = \boldsymbol{CPP}_{m,t} \\ 1, & \text{if } \zeta_t < T'' - 1,\ Re_{m,t} \neq 0, \text{ and } \boldsymbol{CPP}_{m,t+1} \text{ complies with (4.24)} \\ 0, & \text{otherwise} \end{cases} \tag{4.25}$$

$Pr(\boldsymbol{Re}_{t+1}|\boldsymbol{CPP}_{t+1})$: For each user m ($m \in \mathcal{M}$), it generates a content request in each time slot following the average activity probability π_m. Thus,

$$Pr(Re_{m,t+1}|\boldsymbol{CPP}_{m,t+1}) = \begin{cases} \pi_m P_{m,t+1,l}, & \text{if } Re_{m,t+1} = CC_l \\ 1 - \pi_m, & \text{if } Re_{m,t+1} = 0 \end{cases} \tag{4.26}$$

Based on the MDP framework, the original optimization problem (4.5) can be rewritten as

$$\max_{\mu(\boldsymbol{S}_t)} SR = \frac{\lim_{T\to\infty} \frac{1}{T}\mathbb{E}[\sum_{t=0}^{T-1} rw_t(\boldsymbol{S}_t, \mu(\boldsymbol{S}_t))]}{\sum_{m=1}^{M} \pi_m} \tag{4.27}$$

which is equivalent to

$$\max_{\mu(\boldsymbol{S}_t)} r\bar{w} = \lim_{T\to\infty} \frac{1}{T}\mathbb{E}[\sum_{t=0}^{T-1} rw_t(\boldsymbol{S}_t, \mu(\boldsymbol{S}_t))] \tag{4.28}$$

where the expectation operation is taken over all random parameters and the optimization is taken over any possible stationary policy $\mu : \boldsymbol{S}_t \in \mathcal{S} \to a_t \in \mathcal{A}(\boldsymbol{S}_t)$ [14].

According to [14], the optimal average reward $r\bar{w}^*$, together with the optimal differential value vector $\boldsymbol{dv}^* = [dv^*(\boldsymbol{S}_1), \cdots, dv^*(\boldsymbol{S}_{|\mathcal{S}|})]$, satisfies the following Bellman's equation:

$$r\bar{w}^* + dv^*(\boldsymbol{S}) = \max_{a\in\mathcal{A}(\boldsymbol{S})} [rw(\boldsymbol{S}, a) + \sum_{\boldsymbol{S}'\in\mathcal{S}} Pr(\boldsymbol{S}'|\boldsymbol{S}, a) dv^*(\boldsymbol{S}')], \quad \forall \boldsymbol{S} \in \mathcal{S} \tag{4.29}$$

If $a = \mu^*(\boldsymbol{S})$ attains the maximum value of (4.29) for each $\boldsymbol{S} \in \mathcal{S}$, the stationary policy μ^* is optimal. From (4.29), if all the stochastic information is known *a priori*,

the optimal policy μ^* can be obtained by using the traditional DP methods such as the value iteration or the policy iteration [16] in an offline manner. However, in practice,$P_{e_h}(\cdot)$ may be unknown, which makes the traditional offline algorithms become infeasible. R-learning [15] is the most widely used approach for solving the average reward maximization in MDP framework without the knowledge of transition probabilities. However, in this problem, this classic tabular based method can not be adopted since the large size of table ($|\mathcal{S}| \times |\mathcal{A}|$) requires enormous memories and very long time for convergence. To address this issue, in the next subsection, an PDS-ARL algorithm is proposed, which can learn the optimal policy on-the-fly with a high learning efficiency.

4.2.2 An PDS-ARL Algorithm

PDS based Learning Algorithm

Since in the state transition $Pr(\boldsymbol{S}_{t+1}|\boldsymbol{S}_t, a_t)$, all the transition probabilities except $Pr(e_{b_{t+1}}|e_{b_t}, a_t)$ can be derived *a priori*, this partially known information can be exploited to speed up the learning rate. Compared with the conventional R-learning algorithm, the proposed PDS based learning algorithm can significantly improve the convergence speed, without any performance degradation.

PDS: Define PDS as the intermediate system state after the SBS takes an action a_t but before the energy e_{h_t} is harvested. Specifically, the PDS in slot t, denoted by $\hat{\boldsymbol{S}}_t = (\hat{e}_{b_t}, \hat{\zeta}_t, \hat{\boldsymbol{\mathcal{C}}}_t, \widehat{\boldsymbol{CPP}}_t, \hat{\boldsymbol{Re}}_t)$, is defined as

$$\hat{e}_{b_t} = e_{b_t} - e_{c_t} \tag{4.30}$$

$$\hat{\zeta}_t = \zeta_{t+1} \tag{4.31}$$

$$\hat{\boldsymbol{\mathcal{C}}}_t = \boldsymbol{\mathcal{C}}_{t+1} \tag{4.32}$$

$$\widehat{\boldsymbol{CPP}}_t = \boldsymbol{CPP}_{t+1} \tag{4.33}$$

$\hat{Re}_{m,t}$ is generated following the probability distribution in $\widehat{\boldsymbol{CPP}}_{m,t}$ if

$$\hat{Re}_{m,t} \neq 0, m = 1, \cdots, M \tag{4.34}$$

Note that PDS is actually a virtual state.

PDS based Learning Algorithm: Based on the definition of PDS, the differential value function for each PDS is formulated as:

$$dv^*(\hat{\boldsymbol{S}}_t) = \sum_{\boldsymbol{S}_{t+1} \in \mathcal{S}} Pr(\boldsymbol{S}_{t+1}|\hat{\boldsymbol{S}}_t) dv^*(\boldsymbol{S}_{t+1}), \quad \hat{\boldsymbol{S}}_t \in \mathcal{S} \tag{4.35}$$

where the transition $Pr(\boldsymbol{S}_{t+1}|\hat{\boldsymbol{S}}_t)$ is independent of the action and satisfies:

$$Pr(\boldsymbol{S}_{t+1}|\hat{\boldsymbol{S}}_t) = \begin{cases} P_{e_h}(e_{h_t} = e_{b_{t+1}} - e_{b_t} + e_{c_t}), & \text{if} \quad e_{b_{t+1}} < e_{b_{max}} \\ \sum_{e_{h_t} \geq e_{b_{max}} + e_{c_t} - e_{b_t}} P_{e_h}(e_{h_t}), & \text{if} \quad e_{b_{t+1}} = e_{b_{max}} \end{cases} \tag{4.36}$$

By comparing (4.29) with (4.35), it is obvious that there is a deterministic mapping from $dv^*(\boldsymbol{S}_t)$ to $dv^*(\hat{\boldsymbol{S}}_t)$ as

$$r\bar{w}^* + dv^*(\boldsymbol{S}_t) = \max_{a_t \in \mathcal{A}(\boldsymbol{S}_t)} (rw_t(\boldsymbol{S}_t, a_t) + \sum_{\hat{\boldsymbol{Re}}_t} Pr(\hat{\boldsymbol{Re}}_t|\widehat{\boldsymbol{CPP}}_t) dv^*(\hat{\boldsymbol{S}}_t)), \quad \forall \boldsymbol{S}_t \in \mathcal{S} \tag{4.37}$$

Note that since the problem (4.37) is essentially the same as the problem (4.29), the optimal policy $\mu^*(\boldsymbol{S}_t)$ $(\forall \boldsymbol{S}_t \in \mathcal{S})$ can be determined by

$$\mu^*(\boldsymbol{S}_t) = \arg \max_{a_t \in \mathcal{A}(\boldsymbol{S}_t)} (rw_t(\boldsymbol{S}_t, a_t) + \sum_{\hat{\boldsymbol{Re}}_t} Pr(\hat{\boldsymbol{Re}}_t|\widehat{\boldsymbol{CPP}}_t) dv^*(\hat{\boldsymbol{S}}_t)) \tag{4.38}$$

According to (4.37)-(4.38), if $dv^*(\hat{\boldsymbol{S}}_t)$ for $\forall \hat{\boldsymbol{S}}_t \in \mathcal{S}$ can be learnt and approximated, the optimal differential value function $dv^*(\boldsymbol{S}_t)$ for each state $\boldsymbol{S}_t \in \mathcal{S}$ and the optimal policy $\mu^*(\boldsymbol{S}_t)$ can be obtained. From (4.35)-(4.36), $dv^*(\hat{\boldsymbol{S}}_t)$ $(\forall \hat{\boldsymbol{S}}_t \in \mathcal{S})$ can be obtained by just learning the unknown dynamics P_{e_h} on-the-fly. Similar to the conventional R-learning algorithm, an iterative method can be employed to approximate $dv^*(\hat{\boldsymbol{S}}_t)$ for $\forall \hat{\boldsymbol{S}}_t \in \mathcal{S}$. Specifically, at slot t, $dv^*(\hat{\boldsymbol{S}}_t)$ is updated by

$$dv^*_{t+1}(\hat{\boldsymbol{S}}_t) = (1-\epsilon_t) dv^*_t(\hat{\boldsymbol{S}}_t) + \epsilon_t dv^*_t(\boldsymbol{S}_{t+1}) \tag{4.39}$$

where ϵ_t is the learning rate in the tth iteration for weighting the newly learned experience, and satisfies the following stochastic approximation conditions: $0 < \epsilon_t < 1$, $\sum_t \epsilon_t = \infty$, and $\sum_t \epsilon_t^2 < \infty$ [129]. Compared with the conventional R-learning algorithm, where all state-action pairs have to be visited infinitely, the proposed PDS based learning algorithm reduces the learning space to include only states. Furthermore, since (4.37) is equivalent to (4.29), there is no performance degradation in the proposed PDS based learning algorithm.

However, due to the huge size of the state space $|\mathcal{S}|$, the PDS based learning algorithm may still encounter the curse of dimensionality. To address this challenge, in the following, an ARL algorithm is proposed.

ARL Algorithm

In this part, a parameterized function is first defined to represent the differential value function dv^*. Then, a gradient-descent based iteration method is proposed to learn the defined parameters.

Parameterized Function Representation: For the state space $\mathcal{S}$, a feature-extraction function $\chi : \mathcal{S} \to \mathcal{F}$ is adopted to map states into features in the feature space $\mathcal{F}$. Specifically, corresponding to an arbitrary state $\boldsymbol{S}_t \in \mathcal{S}$ (or an arbitrary PDS $\hat{\boldsymbol{S}}_t \in \mathcal{S}$), there is a feature vector $\chi(\boldsymbol{S}_t) = (F_{e_b}(\boldsymbol{S}_t), F_\zeta(\boldsymbol{S}_t), \{F_{m,CC_1}(\boldsymbol{S}_t), \cdots, F_{m,CC_L}(\boldsymbol{S}_t)\}_{m\in\mathcal{M}})$, where

- $F_{e_b}(\boldsymbol{S}_t) = (F_{e_b,0}(\boldsymbol{S}_t), F_{e_b,1}(\boldsymbol{S}_t), \cdots, F_{e_b,\tilde{e}_{b_{max}}}(\boldsymbol{S}_t))$ is the battery feature vector with

$$F_{e_b,i}(\boldsymbol{S}_t) = \begin{cases} 1, & \text{if } e_{b_t} = i \\ 0, & \text{otherwise} \end{cases}, \quad \forall i \in \{0, 1, \cdots, \tilde{e}_{b_{max}}\} \tag{4.40}$$

- $F_\zeta(\boldsymbol{S}_t) = (F_{\zeta,0}(\boldsymbol{S}_t), \cdots, F_{\zeta,T''-1}(\boldsymbol{S}_t))$ is the stage feature vector with

$$F_{\zeta,i}(\boldsymbol{S}_t) = \begin{cases} 1, & \text{if } \zeta_t = i \\ 0, & \text{otherwise} \end{cases}, \quad \forall i \in \{0, 1, \cdots, T'' - 1\} \tag{4.41}$$

- $F_{m,CC_l}(\boldsymbol{S}_t) = (F_{m,l,1}(\boldsymbol{S}_t),\ F_{m,l,2}(\boldsymbol{S}_t), F_{m,l,3}(\boldsymbol{S}_t))$ is the content feature vector of CC_l for user m, $m = 1, \cdots, M, l = 1, \cdots, L$. Within the current period,
  - $F_{m,l,1}(\boldsymbol{S}_t) = 1$ if CC_l has been watched by user m, otherwise, it is 0;
  - $F_{m,l,2}(\boldsymbol{S}_t) = 1$ if CC_l has been cached by user m, otherwise, it is 0;
  - $F_{m,l,3}(\boldsymbol{S}_t) = 1$ if CC_l is currently requested by user m, otherwise, it is 0.

Hence, by using the feature-extraction function χ, each state can be represented with a binary vector of length $\tilde{e}_{b_{max}} + 1 + T'' + 3LM$.

After feature extraction, a linear approximate architecture is considered for the differential value function dv^* as

$$\tilde{dv}_t(\hat{\boldsymbol{S}}_t, \boldsymbol{\gamma}_t) = \chi(\hat{\boldsymbol{S}}_t)\boldsymbol{\gamma}_t$$

$$= \sum_{i=1}^{\tilde{e}_{b_{max}}+1+T''+3LM} \chi(\hat{\boldsymbol{S}}_t)(i)\boldsymbol{\gamma}_t(i) \quad \forall \hat{\boldsymbol{S}}_t \in \mathcal{S} \tag{4.42}$$

where $\boldsymbol{\gamma}_t$ denotes the adaptable parameter vector at slot t, with a length of $\tilde{e}_{b_{max}} + 1 + T'' + 3LM$. By representing dv^* in such a compact way, at each slot t, instead of learning the differential value function for each PDS, it only needs to learn and update the parameter vector $\boldsymbol{\gamma}_t$, which has a much more manageable size of $\tilde{e}_{b_{max}} + 1 + T'' + 3LM$. To this end, the learning space has been significantly reduced from $|\mathcal{S}|$ to $\tilde{e}_{b_{max}} + 1 + T'' + 3LM$.

Gradient-Descent based Iteration: Define an error function as

$$err = \frac{1}{2}[dv^*(\hat{\boldsymbol{S}}) - \chi(\hat{\boldsymbol{S}})\boldsymbol{\gamma}]^2, \quad \forall \hat{\boldsymbol{S}} \in \mathcal{S} \tag{4.43}$$

Then, the optimal parameter vector $\boldsymbol{\gamma}^*$ should satisfy

$$\boldsymbol{\gamma}^* = \arg\min_{\boldsymbol{\gamma}} err \tag{4.44}$$

Based on the gradient-descent method, $\boldsymbol{\gamma}^*$ is obtained by adjusting the parameter vector following

$$\begin{aligned} \boldsymbol{\gamma}_{t+1} &= \boldsymbol{\gamma}_t - \frac{1}{2}\rho_t \bigtriangledown_{\boldsymbol{\gamma}_t} [dv^*(\hat{\boldsymbol{S}}_t) - \chi(\hat{\boldsymbol{S}}_t)\boldsymbol{\gamma}_t]^2 \\ &= \boldsymbol{\gamma}_t + \rho_t(dv^*(\hat{\boldsymbol{S}}_t) - \chi(\hat{\boldsymbol{S}}_t)\boldsymbol{\gamma}_t)\chi(\hat{\boldsymbol{S}}_t) \end{aligned} \tag{4.45}$$

where ρ_t is a positive step-size parameter. It has been proved in [15] that if ρ_t decreases in such a way as to satisfy the standard stochastic approximation conditions as described above, the gradient-descent based iteration (4.45) is guaranteed to converge to a local optimum.

Since $dv^*(\hat{\boldsymbol{S}}_t)$ is unknown, the iteration (4.45) cannot be performed directly. However, $h^*(\hat{\boldsymbol{S}}_t)$ can be approximated by the training example ex_t in slot t, as

$$\boldsymbol{\gamma}_{t+1} = \boldsymbol{\gamma}_t + \rho_t(ex_t - \chi(\hat{\boldsymbol{S}}_t)\boldsymbol{\gamma}_t)\chi(\hat{\boldsymbol{S}}_t) \tag{4.46}$$

where

$$ex_t = \max_{a_{t+1}\in\mathcal{A}(\boldsymbol{S}_{t+1})}(rw_{t+1}(\boldsymbol{S}_{t+1}, a_{t+1}) - r\bar{w}_t + \sum_{\hat{\boldsymbol{R}}\boldsymbol{e}_{t+1}} Pr(\hat{\boldsymbol{R}}\boldsymbol{e}_{t+1}|\widehat{\boldsymbol{CPP}_{t+1}})\chi(\hat{\boldsymbol{S}}_{t+1})\boldsymbol{\gamma}_t) \tag{4.47}$$

and the average reward $r\bar{w}_t$ updates following

$$r\bar{w}_{t+1} = r\bar{w}_t(1-\eta_t) + \eta_t(rw_{t+1}(\boldsymbol{S}_{t+1}, a_{t+1}) + \chi(\hat{\boldsymbol{S}}_{t+1})\boldsymbol{\gamma}_t - \chi(\hat{\boldsymbol{S}}_t)\boldsymbol{\gamma}_t) \tag{4.48}$$

η_t is the learning rate for updating $r\bar{w}$. Since ex_t is an unbiased estimate, i.e., $\mathbb{E}[ex_t] = dv^*(\hat{\boldsymbol{S}}_t)$, the iteration (4.46) can still be guaranteed to converge to a local optimum [15].

PDS-ARL algorithm

The proposed PDS-ARL algorithm is summarized as follows. The feature vectors for all system states are first extracted. Then, at the beginning of each slot t, the greedy action in (4.38) is taken at the SBS, where $dv^*(\hat{\boldsymbol{S}}_t)$ is replaced by its approximate function $\chi(\hat{\boldsymbol{S}}_t)\boldsymbol{\gamma}_t$. After performing the action, the immediate reward $rw_t(\boldsymbol{S}_t, a_t)$, the PDS $\hat{\boldsymbol{S}}_t$, and the next state $\boldsymbol{S}_{t+1}$ can be observed, based on which, the parameter vector $\boldsymbol{\gamma}_t$ and the average reward $r\bar{w}_t$ are updated following (4.46) and (4.48), respectively. This process continues until both $\boldsymbol{\gamma}_k$ and $\bar{r}_k$ converge. The details of the proposed algorithm is shown in Algorithm 4.

Algorithm 4: The Post-Decision State based Approximate Reinforcement Learning Algorithm

Input: the feature vectors for all system states, the learning rates ρ_t and η_t.
Output: the parameter vector $\boldsymbol{\gamma}$ and the average reward $r\bar{w}$.
Initialize $t = 0$, the parameter vector $\boldsymbol{\gamma}_t = \mathbf{0}$, the average reward $r\bar{w}_t = 0$, and the system state $\boldsymbol{S}_r \in \mathcal{S}$.
for $t = 0, \cdots, T-1$ **do**
set $\boldsymbol{S}_t = \boldsymbol{S}_r$, based on which, take the greedy action:

$$a_t = \arg \max_{a_t \in \mathcal{A}(\boldsymbol{S}_t)} (rw_t(\boldsymbol{S}_t, a_t) + \sum_{\hat{\boldsymbol{R}}\boldsymbol{e}_t} Pr(\hat{\boldsymbol{R}}\boldsymbol{e}_t | \widehat{\boldsymbol{CPP}}_t) \chi(\hat{\boldsymbol{S}}_t) \boldsymbol{\gamma_t})$$

Observe the immediate reward $rw_t(\boldsymbol{S}_t, a_t)$, the PDS $\hat{\boldsymbol{S}}_t$, and the next state $\boldsymbol{S}_{t+1}$.
At $\boldsymbol{S}_{t+1}$, take the greedy action:

$$a_{t+1} = \arg \max_{a_{t+1} \in \mathcal{A}(\boldsymbol{S}_{t+1})} (rw_{t+1}(\boldsymbol{S}_{t+1}, a_{t+1}) + \sum_{\hat{\boldsymbol{R}}\boldsymbol{e}_{t+1}} Pr(\hat{\boldsymbol{R}}\boldsymbol{e}_{t+1} | \widehat{\boldsymbol{CPP}_{t+1}}) \chi(\hat{\boldsymbol{S}}_{t+1}) \boldsymbol{\gamma}_t)$$

Observe the immediate reward $rw_{t+1}(\boldsymbol{S}_{t+1}, a_{t+1})$, the PDS $\hat{\boldsymbol{S}}_{t+1}$, and the state $\boldsymbol{S}_{t+2}$.
Update the parameter vector $\boldsymbol{\gamma}_t$ with equation (4.46), and the average reward $r\bar{w}_t$ with equation (4.48).
Update $\boldsymbol{S}_r = \boldsymbol{S}_{t+2}$.
end

4.3 Numerical Results

In this section, the effectiveness of the proposed PDS-ARL algorithm is evaluated. In the simulation, set $radii$ = 10m, $\{D_1, D_2, \cdots, D_w\} = \{0, 2, 4, 6, 8, 10\}$, $\frac{r_0}{W} = 1$ bps/Hz, and $\alpha = 2$. The maximum transmit power or equivalently the transmit power for cell-edge user is set as $p_{max} = 0.5$Watt. The channel coefficient ξ follows Rayleigh fading. The mean value of the channel fading and the noise power σ^2 are set so that (4.3) holds for $p_m = p_{max}$ and $d_m = radii$. For each user m, both the mean activity probability π_m and the distance d_m to the SBS follows uniform distributions,

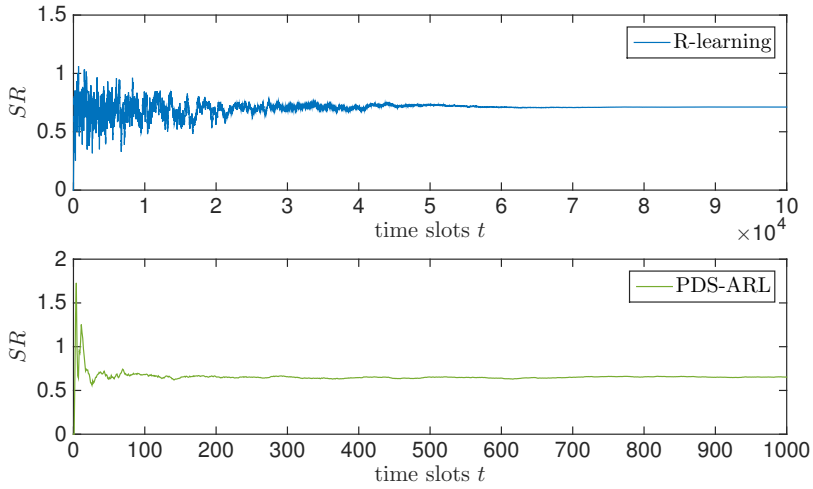

Figure 4.2: Comparison between the PDS-ARL algorithm and the R-learning algorithm

i.e, $\pi_m \sim U[0,1]$, $d_m \sim radii \times U[0,1]$, $\forall m$. For simplicity, the slot length τ is normalized as 1. The battery energy is quantized with unit $e_{unit} = 0.02$ so that the transmit power for a distance D_i, $i \in \{1, \cdots, 6\}$, equals $(i-1)^2 e_{unit}$. For the learning rates, I set $\rho_t = \frac{1}{t^{0.9}}$ and $\eta_t = \frac{1}{t}$. Assume that users' content preference distribution follows Zipf distribution with a skew parameter 3 and the energy arrival process follows a Poisson distribution with mean $\bar{e}_h$. Notice that the analytical results do not depend on any specific stochastic model, and hence can be applied to more general cases. Other parameters are defined for each simulation scenario separately.

Firstly, the effectiveness of the proposed PDS-ARL algorithm in terms of the convergence rate is evaluated by comparing it with the conventional R-learning algorithm. As analyzed before, R-learning algorithm has the severe "curse of dimensionality" issue, even with a small number of users. Thus, to facilitate the simulation, only one user is considered here, i.e. $M = 1$, with an activity probability 0.5 and a distance

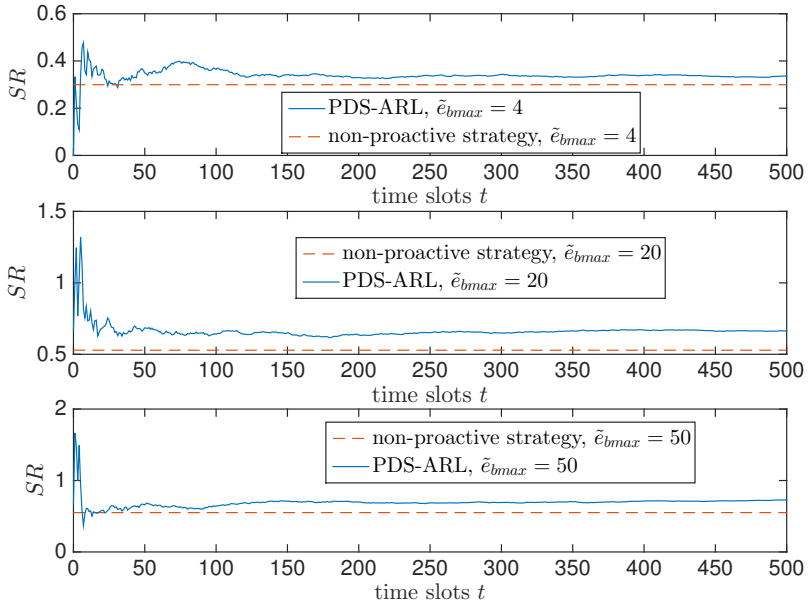

Figure 4.3: The influence of battery capacity $\tilde{e}_{b_{max}}$

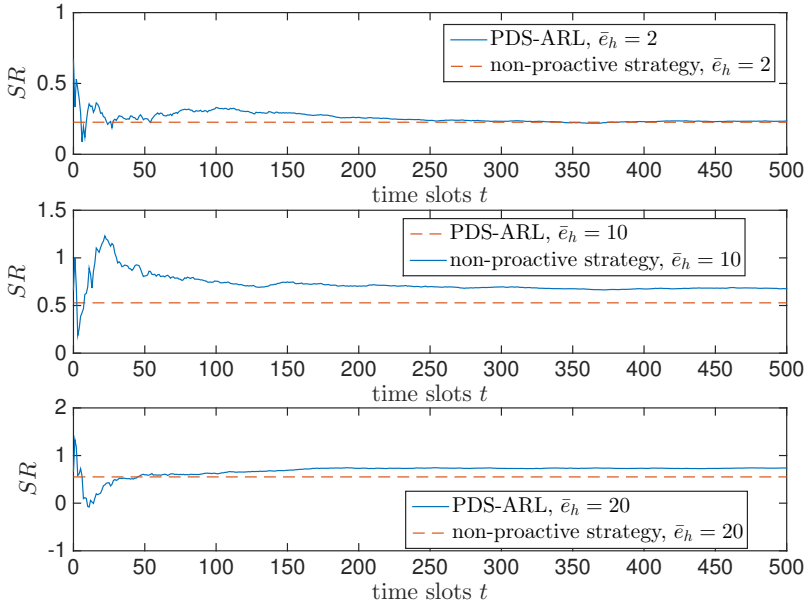

Figure 4.4: The influence of average energy arrival $\bar{e}_h$

6m to the SBS. Other parameters are set as follows: $T'' = 3$, $L = 3$, $e_{b_{max}} = 12$, and $\bar{e}_h = 5$. From Fig. 4.2, it can be seen that even with $M = 1$, R-learning algorithm takes more than 5×10^4 slots before convergence. On the contrary, the proposed PDS-ARL algorithm converges to a steady-state value within 200 slots. Furthermore, the performance of the proposed PDS-ARL algorithm is very similar to that of the R-learning algorithm. Namely, the proposed PDS-ARL algorithm can achieve a similar performance as the R-learning algorithm but with a significantly reduced convergence time. Thus, the proposed PDS-ARL algorithm is more effective and practical than the conventional algorithm.

Next, the performance of the proposed PDS-ARL algorithm in terms of SR is evaluated by comparing it with a non-proactive strategy, in which only reactive transmission is available. In all these simulations, unless specified, set $M = 3$, $T'' = 3$, $L = 3$, $\tilde{e}_{b_{max}} = 20$, $\bar{e}_h = 10$, $\pi_m = 0.5$, $\forall m \in \mathcal{M}$, $[d_1, d_2, d_3] = [4, 8, 6]$, and the initial content preference rankings for each user are randomly generated as $[CC_1, CC_2, CC_3]$, $[CC_3, CC_1, CC_2]$, and $[CC_2, CC_3, CC_1]$, respectively.

Figs. 4.3 and 4.4 evaluate the influence of the battery capacity $\tilde{e}_{b_{max}}$ and the average energy arrival $\bar{e}_h$, respectively. In Fig.4.3, set $\bar{e}_h = 10$ and vary $\tilde{e}_{b_{max}}$ from 4 to 50, while in Fig.4.4, set $\tilde{e}_{b_{max}} = 20$ and vary $\bar{e}_h$ from 2 to 20. From Fig.4.3, it can be seen that when the battery capacity is very limited (e.g., $\tilde{e}_{b_{max}} = 4$ in this simulation), the proposed PDS-ARL algorithm achieves very limited performance gains compared with the non-proactive strategy. However, with the increase of battery capacity, the proposed PDS-ARL algorithm achieves much better performance than the non-proactive strategy (e.g., yielding around 30% improvement when $\tilde{e}_{b_{max}} = 20$ and 50%

improvement when $\tilde{e}_{b_{max}} = 50$). This is because when the battery capacity $\tilde{e}_{b_{max}} = 4$, the SBS can only serve user 1, who requires 4 unit transmission power and is the least among all users. Thus, the benefit due to proactive caching is limited. However, when the battery capacity $\tilde{e}_{b_{max}} \geq 20$, all three users could be served by the SBS, which creates more opportunities for the SBS to perform proactive caching. By exploring the similarity of content preferences among users, the proposed PDS-ARL algorithm improves the energy utilization efficiency and hence outperforms the non-proactive strategy significantly. A similar explanation can be applied to the observations from Fig.4.4, where both two algorithms achieve almost the same performance when the average energy arrival $\bar{e}_h$ is very limited (e.g., $\bar{e}_h = 2$ in this simulation), while with the increase of $\bar{e}_h$, the proposed PDS-ARL algorithm achieves much better performance than the non-proactive strategy (e.g., yielding around 30% improvement when $\bar{e}_h = 10$ and 50% improvement when $\bar{e}_h = 20$).

Fig. 4.5 shows the performance comparison by varying user activity probability π_m, $m = 1, 2, 3$. From this figure, it can be seen that with a low user activity probability (e.g., $\pi_1 = \pi_2 = \pi_3 = 0.1$ in this simulation), the service ratio SR achieved by both two algorithms is close to 100%. With the increase of π_m, $m = 1, 2, 3$, both service ratios decrease, but the one with the non-proactive strategy drops much faster than that with the proposed PDS-ARL algorithm. This is because when $\pi_1 = \pi_2 = \pi_3 = 0.1$, the request generation rate is very low. By setting $\bar{e}_h = 10$ and $\tilde{e}_{b_{max}} = 20$, the battery energy will be always sufficient to support the content requests, which makes the advantage of proactive caching insignificant. With the increase of user activity probability, the request generation rate increases, which then leads to an increase of energy

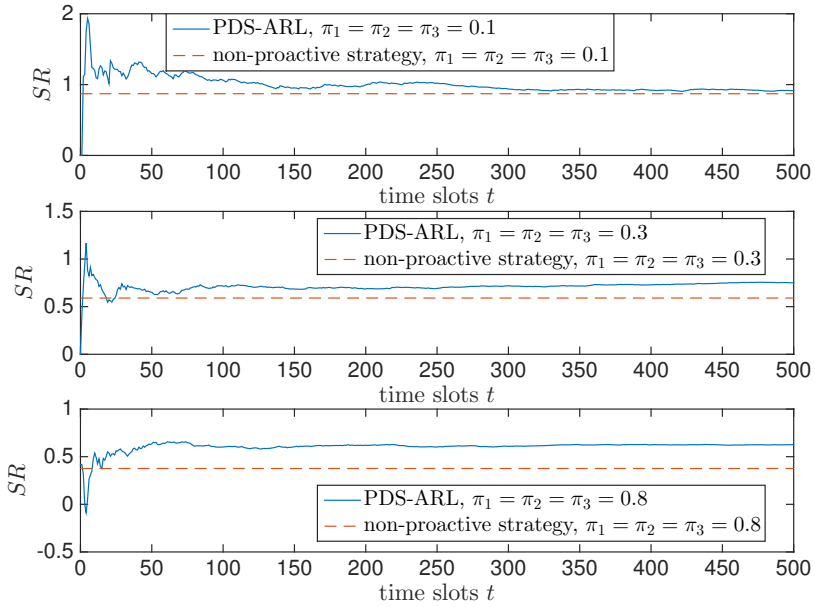

Figure 4.5: The influence of user activity probability π_m, $m = 1, 2, 3$.

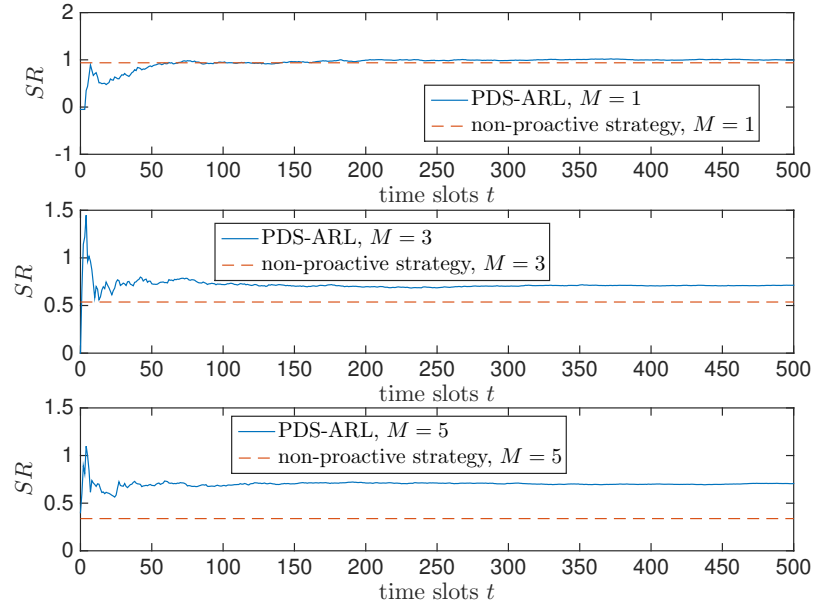

Figure 4.6: The influence of the number of users M.

consumption. Thus, the battery energy may not be always sufficient to support all users' content requests, which results in the decrease of the service ratios. However, with insufficient battery energy, optimizing energy utilization becomes important. Thus, by performing proactive caching to utilize the limited renewable energy in a more efficient manner, the proposed PDS-ARL algorithm achieves much better performance than the non-proactive strategy (e.g., yielding around 30% improvement when $\pi_m = 0.3$ and 65% improvement when $\pi_m = 0.8$).

Fig. 4.6 shows the performance comparison by varying the number of users M. In this simulation, I set $\tilde{e}_{b_{max}} = 50$, and $\pi_m = 0.5$, $\forall m$. In addition, for each user, the distance to the SBS and the initial content preference ranking are generated randomly. Specifically, for $M = 1$, $\{d_1\} = \{8\}$ and the user's initial content preference ranking is $\{CC_2, CC_1, CC_3\}$. For $M = 3$, $\{d_1, d_2, d_3\} = \{4, 8, 4\}$ and the users' initial content preference rankings are $\{CC_2, CC_3, CC_1\}$, $\{CC_3, CC_1, CC_2\}$, and $\{CC_1, CC_2, CC_3\}$. For $M = 5$, $\{d_1, d_2, d_3, d_4, d_5\} = \{2, 4, 6, 6, 8\}$ and the users' initial content preference rankings are $\{CC_1, CC_2, CC_3\}$, $\{CC_3, CC_1, CC_2\}$, $\{CC_2, CC_3, CC_1\}$, $\{CC_1, CC_3, CC_2\}$, and $\{CC_2, CC_1, CC_3\}$. From Fig. 4.6, it can be seen that with a single user in the small-cell, both algorithms achieve a similar performance, while with the increase of the number of users, the proposed PDS-ARL algorithm starts outperforming the non-proactive strategy and the performance gap increases with the number of users. The reason is similar to that in Fig. 4.5. With a small number of users, the battery energy is sufficient to support all content requests so that the proactive caching is not necessary. With the number of users increasing, more content requests can be generated and the deliberate energy utilization becomes important. By exploring

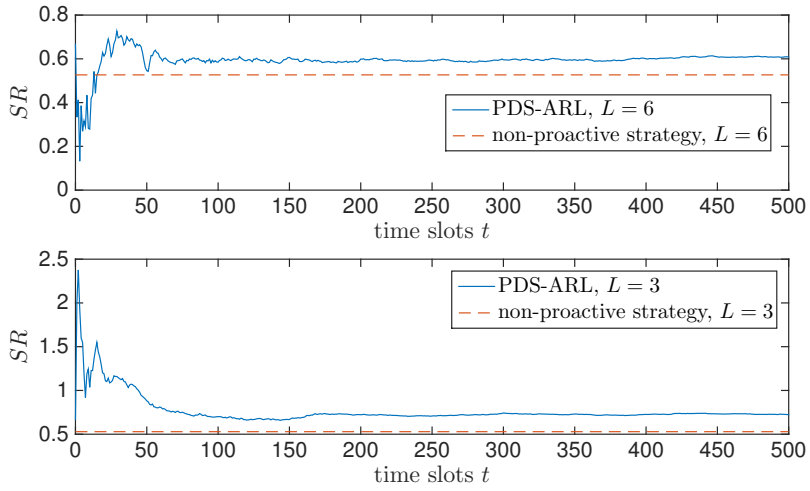

Figure 4.7: The influence of the number of content category L.

the similarity of content preferences among users, the proposed PDS-ARL algorithm improves the energy utilization significantly through effective proactive caching.

In order to demonstrate the impact of the CPP diversity among users, Fig. 4.7 presents the performance comparison by setting the period T'' as 3 and varying the number of content categories L. In this simulation, I randomly generate users' initial content preference rankings as follows: For $L = 3$, the initial content preference rankings are $\{CC_3, CC_1, CC_2\}$, $\{CC_3, CC_2, CC_1\}$, and $\{CC_1, CC_3, CC_2\}$; For $L = 6$, they are $\{CC_4, CC_3, CC_5, CC_2, CC_6, CC_1\}$, $\{CC_3, CC_1, CC_6, CC_4, CC_5, CC_2\}$, and $\{CC_6, CC_2, CC_4, CC_1, CC_5, CC_3\}$. With $T'' = 3$, each user will at most watch 3 preferable content categories within each period. If $L = 3$, all users have a common set of contents that will be potentially watched within each period, whereas if $L = 6$, since different users may have different content preferences, the 3 preferable content categories that will be most likely watched by each user can be quite different, and hence the similarity on users' CPPs can be very limited. Thus, as shown in Fig. 4.7,

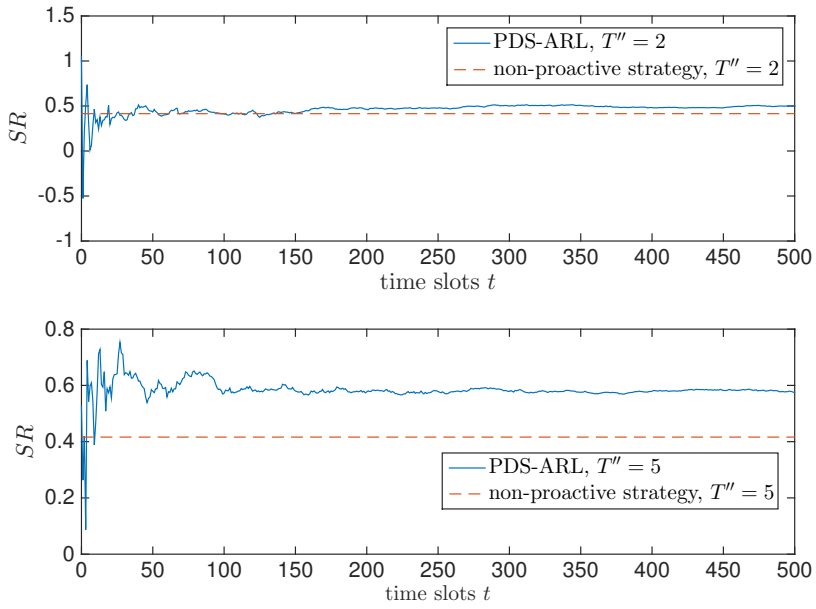

Figure 4.8: The influence of the period T''.

when $L = 3$, the proposed PDS-ARL algorithm attains 50% performance gains over the non-proactive strategy, while with $L = 6$, the performance gains arrive at 15%.

In Fig. 4.8, these two algorithms are compared by varying the period T''. The parameter settings are as follows: $\{\pi_1, \pi_2, \pi_3\} = \{0.5, 0.8, 0.6\}$; With $T'' = 2$, $L = 2$ and the initial content preference rankings are $\{CC_1, CC_2\}$, $\{CC_1, CC_2\}$, and $\{CC_2, CC_1\}$; With $T'' = 5$, $L = 5$ and the initial content preference rankings are $\{CC_2, CC_1, CC_3, CC_5, CC_4\}$, $\{CC_5, CC_2, CC_4, CC_3, CC_1\}$, and $\{CC_3, CC_5, CC_1, CC_4, CC_2\}$. When $T'' = 2$, the content of each category will be updated every a period of 2 time slots and within each period, only one time slot could be available for proactive caching. Thus, as shown in Fig. 4.8, due to the limited proactive caching opportunities, the performance gains of the proposed PDS-ARL algorithm over the non-proactive strategy is limited. Nevertheless, when $T'' = 5$, the content of each cat-

egory will be updated every 5 time slots and within each period, 4 time slots could be used for proactive caching. With the increase of the proactive caching opportunities, the performance gains of the proposed PDS-ARL algorithm over the non-proactive strategy reach 50% in Fig. 4.8.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this thesis, the applications of the model-free learning in three different wireless networks have been investigated. Specifically, distributed throughput maximization by a proposed N-SLA algorithm in an OSA network with multiple heterogeneous SUs and primary channels is studied in Chapter 2, followed by an analysis of the multi-mobile-device computation offloading in a multi-channel cloudlet-based MCC network through a proposed machine learning based FDCO algorithm in Chapter 3. In Chapter 4, the problem of context-aware proactive caching in an energy harvesting network consisting of a single MBS with grid power supply and multiple small-cells with energy harvesting is discussed, and an PDS-ARL algorithm which learns on-the-fly the optimal context-aware proactive caching policy with a high learning efficiency is proposed. Via effective and efficient model-free strategy learning mechanism designs, these networks can be adaptive to an unknown and dynamic environment and achieve

significant improvement of network performance when comparing with counterparts.

In Chapter 2, the distributed throughput maximization problem of the OSA network is first formulated as an ordinal potential game, which has at least one pure-strategy NE. Then, to achieve the pure-strategy NEs, two algorithms: a BR based algorithm and an N-SLA algorithm are proposed. The BR based algorithm is proved to converge towards NEs, but requiring indispensable information exchange among SUs, while the N-SLA algorithm is fully distributed and can converge towards NEs without any information exchange. Simulation results demonstrate that the proposed algorithms can not only guarantee the convergence to NEs, but also achieve a good network performance in terms of sum log expected throughput.

In Chapter 3, the problem of distributed computation offloading decision making among multiple mobile devices in the cloudlet-based MCC network is first formulated as a noncooperative game, which is further proved to be an exact potential game. Then, to achieve the pure strategy NEs, an FDCO algorithm based on machine learning technology is proposed, which can converge towards the pure strategy NEs without any information exchange. Furthermore, by conducting the performance analysis on the proposed FDCO algorithm, it shows that the proposed FDCO algorithm maximizes the number of beneficial cloudlet computing mobile devices and does not incur larger network-wide execution cost than computing locally by all mobile devices. Simulation results demonstrate that the proposed FDCO algorithm can not only converge to a pure strategy NE, but also achieve a good network performance in terms of both the number of beneficial cloudlet computing mobile devices and the network-wide execution cost.

In Chapter 4, the context-aware proactive caching problem in the energy harvesting based network is first formulated as an MDP framework. However, due to the incomplete stochastic information about the network dynamics and the "curse of dimensionality" issue of the formulated MDP, neither the conventional dynamic programming nor the standard reinforcement learning can be applicable. To address all these challenges, an PDS-ARL algorithm is designed to learn the optimal proactive caching policy on-the-fly. By adopting the PDS approach to reduce the action space and the ARL approach to reduce the state space, the proposed PDS-ARL algorithm achieves a remarkable improvement on the convergence rate compared with the conventional R-learning algorithm. Furthermore, by deliberate energy utilization via effective proactive caching, the proposed PDS-ARL algorithm improves the renewable energy utilization significantly compared with the non-proactive strategy. Through simulation, the effectiveness of the proposed PDS-ARL algorithm in terms of both the convergence rate and the service ratio has been demonstrated by comparing it with baselines.

From the results of this thesis, it can be seen that learning has great capacities and potential in solving network control with incomplete network evolution model or with limited information exchange. By effective and efficient specialized learning designs, the decision-making entities can adaptively choose their transmission strategies in a self-organized manner without much requirement for knowing the network conditions and achieve good performance. I hope this thesis will serve as an important guideline for future research directions to further understand model-free learning mechanisms and expand their applications in wireless networks.

5.2 Future Work

Some future research directions on the applications of model-free learning in wireless networks are outlined as follows.

- Obtaining the convergence condition for learning algorithms in more general multi-agent decision making processes: Generally speaking, the goal of a perfect self-organized learning mechanism for multi-agent decision making processes is to achieve self-play/autonomy, stability and optimality simultaneously. However, for multi-agent learning, improving network performance typically incurs more signalling and coordination, thus undermining the self-play structure. Especially, when learning is implemented under the framework of games, achieving any two goals is usually at the cost of undermining the third goal. As a result, most current studies focused on ensuring convergence to a stable operation point in self-play by allowing a limited control signal exchange. In the literature, the approaches to find the convergence condition of learning algorithms mainly fall into the following category. For learning processes that can be approximated with a linear system described as a set of ODEs in continuous time, the typical way of obtaining the convergence condition is to construct a Lyapunov function for the ODE-based dynamic and then prove that the strategy updating mechanism produces an asymptotic pseudo-trajectory of the flow defined by the ODE (see Chapters 2 and 3). Although there are a few already known conditions that ensure the convergence of a learning algorithm, most of them are limited within a small scope. One important case is the network control problem that is modelled as an exact potential game. However, the exact potential game

requires significant homogeneity among local players (or agents), which limits the applications for the exact potential game-based learning algorithms. Therefore, for most current studies with heterogeneous properties, whether a stability condition can be found for a learning algorithm remains an open issue and is yet to be addressed.

- Analyzing the convergence rate of learning algorithms: In addition to the issues associated with finding the convergence condition for a learning algorithm, another concern for applying model-free learning in wireless networks is to derive the convergence rate. Although analytical results for the convergence rate of learning algorithms are highly desired, most of the existing studies were only able to show empirical results for the learning convergence rate through numerical simulations. The reason for this is partly due to the asymptotic convergence condition (if there is any), which requires that the states and actions should be visited infinitely to ensure the convergence. Given such a limitation, one possible approach to analyze the convergence rate of a learning algorithm is to regard the learning process as a discrete time Markov chain. In this approach, the standard Markov chain analysis can be applied to obtain the expected learning time (number of iterations) before arriving at the chain's absorbing state (e.g., a NE).

- Designing efficient learning mechanisms in the scenarios of dramatical environmental changes: As described in Chapters 2, 3 and 4, the network environment is considered to be quasi-static, in which the MDP model of the network is invariant if the MDP-based learning (i.e., RL) is adopted, or the set of players

is invariant if the game-based learning (i.e., SLA) is adopted. However, when the network environment has dramatically changed in terms of the MDP model or the set of players, learners generally need to start the same learning process from the very beginning. As a result, when the decision-making agents are required to swiftly switch from an old situation to a new one, the existing learning algorithms will face great challenges if they can only restart the learning process in the new situation. In order to address such a challenge, a natural consideration is to utilize the acquired experience of strategy taking that is obtained from the old situations. Since the experience transferring paradigm, Transfer Learning (TL) [130], aims to transfer knowledge (i.e., experience) from the well-established learning processes to a newly-established learning process in a different situation, it is considered as a promising approach in the scenarios of dramatical environmental changes.

Bibliography

[1] P. Nicopolitidis, G. I. Papadimitriou, A. S. Pomportsis, P. Sarigiannidis, M. S. Obaidat, "Adaptive wireless networks using learning automata," *IEEE Wireless Commun.*, vol. 18, no. 2, pp. 75-81, Apr. 2011.

[2] L. P. Kaelbling, M. L. Littman, and A.W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, no. 1, pp. 237-285, May 1996.

[3] L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multi-agent reinforcement learning," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 2, pp. 156-172, Mar. 2008.

[4] W. Wang, A. Kwasinski, D. Niyato, and Z. Han, "A survey on applications of model-free strategy learning in cognitive wireless networks," *IEEE Commun. Surv. Tuts.*, vol. 18, no. 3, Third Quarter 2016.

[5] B. Wang and K. J. R. Liu, "Advances in cognitive radio networks: A survey," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 1, pp. 5-23, Feb. 2011.

[6] I. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, "A survey on spectrum management in cognitive radio networks," *IEEE Commun. Mag.*, vol. 46, no. 4, pp. 40-48, Apr 2008.

[7] M. Bkassiny, Y. Li, and S. Jayaweera, "A survey on machine-learning techniques in cognitive radios," *IEEE Commun. Surv. Tuts.*, vol. 15, no. 3, pp. 1136-1159, Third Quart. 2013.

[8] T. Yucek and H. Arslan, "A survey of spectrum sensing algorithms for cognitive radio applications," *IEEE Commun. Surv. Tuts.*, vol. 11, no. 1, pp. 116-130, First Quart. 2009.

[9] I. F.-Akyildiz, B. F. Lo, and R. Balakrishnan, "Cooperative spectrum sensing in cognitive radio networks: A survey," *Phys. Commun.*, vol. 4, no. 1, pp. 40-62, Mar. 2011.

[10] Y. Zeng, Y.-C. Liang, A. T. Hoang, and R. Zhang, "A review on spectrum sensing for cognitive radio: Challenges and solutions," *EURASIP J. Adv. Signal Process.*, vol. 2010, pp. 1-15, Jan. 2010.

[11] Z. Zhang, K. Long, and J. Wang, "Self-organization paradigms and optimization approaches for cognitive radio technologies: A survey," *IEEE Wireless Commun.*, vol. 20, no. 2, pp. 36-42, Apr. 2013.

[12] O. Aliu, A. Imran, M. Imran, and B. Evans, "A survey of self organization in future cellular networks," *IEEE Commun. Surv. Tuts.*, vol. 15, no. 1, pp. 336-361, First Quart. 2013.

[13] J. Mitola, "Cognitive radio-An integrated agent architecture for software defined radio," DTech thesis, Dept. Teleinformatics, Royal Inst. Technology (KTH), Kista, Sweden, May 2000.

[14] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York: John Wiley and Sons, 1994.

[15] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[16] M. Bkassiny, S. K. Jayaweera, and K. A. Avery, "Distributed reinforcement learning based MAC protocols for autonomous cognitive secondary users," *in 20th Annual Wireless and Optical Communications Conference (WOCC'11)*, Newark, NJ, Apr. 2011, pp. 1-6.

[17] A. Galindo-Serrano and L. Giupponi, "Distributed Q-learning for aggregated interference control in cognitive radio networks," *IEEE Trans. Veh. Technol.*, vol. 59, no. 4, pp. 1823 -1834, May 2010.

[18] C. Watkins, "Learning from delayed rewards," Ph.D. dissertation, University of Cambridge, United Kingdom, 1989.

[19] K.-L. A. Yau, P. Komisarczuk, and P. D. Teal, "Applications of reinforcement learning to cognitive radio networks," *in IEEE International Conference on Communications Workshops (ICC), 2010*, Cape Town, South Africa, May 2010, pp. 1-6.

[20] Z. Qu, R. Cui, Q. Song, S. Yin, "Predictive spectrum sensing strategy based on reinforcement learning," *China Commun.*, vol. 11, no. 10, pp. 117-125, 2014.

[21] P. Venkatraman, B. Hamdaoui, and M. Guizani, "Opportunistic bandwidth sharing through reinforcement learning," *IEEE Trans. Veh. technol.*, vol. 59, no. 6, pp. 3148-3153, Jul. 2010.

[22] H. Cao, H. Tian, J. Cai, A. S. Alfa, and S. Huang, "Dynamic load-balancing spectrum decision for heterogeneous services provisioning in multi-channel cognitive radio networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, Sep. 2017.

[23] V. Raj , I. Dias , T. Tholeti , and S. Kalyani, "Spectrum access in cognitive radio using a two-stage reinforcement learning approach," *IEEE J. Sel. topics in signal process.*, vol. 12, no. 1, Feb. 2018.

[24] I. Macaluso, D. Finn, B. Özgül, and L. A. DaSilva, "Complexity of spectrum activity and benefits of reinforcement learning for dynamic channel selection," *IEEE J. sel. areas in commun.*, vol. 31, no. 11, Nov. 2013.

[25] R. Sutton, D. Mcallester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *in Proc. 12th conference on Advances in Neural Information Processing Systems (NIPS '99)*. Denver, CO: MIT Press, 2001, pp. 1057-1063.

[26] J. Baxter and P. L. Bartlett, "Infinite-horizon policy-gradient estimation," *Journal of Artificial Intelligence Research*, vol. 15, pp. 319-350, 2001.

[27] G. D. Croon, M. F. V. Dartel, and E. O. Postma, "Evolutionary learning outperforms reinforcement learning on non-Markovian tasks," *in 8th European Conference on Artificial Life Workshop on Memory and Learning Mechanisms in Autonomous Robots*, Canterbury, Kent, UK, 2005.

[28] J. Peters and S. Schaal, "Policy gradient methods for robotics," *in IEEE/RSJ International Conference on Intelligent Robots and Systems (2006)*, Beijing, China, Oct. 2006, pp. 2219-2225.

[29] M. Riedmiller, J. Peters, and S. Schaal, "Evaluation of policy gradient methods and variants on the cart-pole benchmark," *in IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL ?07)*, Honolulu, HI, Apr. 2007, pp. 254 -261.

[30] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," *in Readings in Agents*, M. N. Huhns and M. P. Singh, Eds. San Mateo, CA, USA: Morgan Kaufmann, 1998, pp. 487-494.

[31] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," *Auton. Agents Multi-Agent Syst.*, vol. 11, no. 3, pp. 387-434, Nov. 2005.

[32] M. Ahmadabadi and M. Asadpour, "Expertness based cooperative q-learning," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 32, no. 1, pp. 66-76, Feb. 2002.

[33] S. Sen and M. Sekaran, "Individual learning of coordination knowledge," *J. Exp. Theor. Artif. Intell.*, vol. 10, no. 3, pp. 333-356, Jul. 1998.

[34] D. Fudenberg and J. Tirole, *Game Theory*. MIT Press, 1991.

[35] H. Li, Y. Liu, and D. Zhang, "Dynamic spectrum access for cognitive radio systems with repeated games," *in IEEE International Conference on Wireless Commun., Netw. and Inf. Security (WCNIS '10)*, Beijing, China, Jun. 2010, pp. 59 -62.

[36] H.-P. Shiang and M. van der Schaar, "Distributed resource management in multihop cognitive radio networks for delay-sensitive transmission," *IEEE Trans. Veh. Technol.*, vol. 58, no. 2, pp. 941-953, Feb. 2009.

[37] Q. Zhu, W. Saad, Z. Han, H. Poor, and T. Basar, "Eavesdropping and jamming in next-generation wireless networks: A game-theoretic approach," *in Proc. IEEE Mil. Commun. Conf.*, Baltimore, MD, USA, Nov. 2011, pp. 119-124.

[38] G. Arslan and J. Shamma, "Distributed convergence to Nash equilibria with local utility measurements," *in Proc. 43rd IEEE Conf. Decis. Control*, Los Angeles, CA, USA, Dec. 2004, vol. 2, pp. 1538-1543.

[39] T. Cui, L. Chen, and S. Low, "A game-theoretic framework for medium access control," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 7, pp. 1116-1127, Sep. 2008.

[40] J. Shamma and G. Arslan, "Dynamic fictitious play, dynamic gradient play, and distributed convergence to Nash equilibria," *IEEE Trans. Autom. Control*, vol. 50, no. 3, pp. 312-327, Mar. 2005.

[41] Y. Xu, J. Wang, Q. Wu, A. Anpalagan, and Y.-D. Yao, "Opportunistic spectrum access in unknown dynamic environment: A game-theoretic stochastic learning solution," *IEEE Trans. Wireless Commun.*, vol. 11, no. 4, pp. 1380-1391, Apr. 2012.

[42] J. Zheng, Y. Cai, N. Lu, Y. Xu, and X. Shen, "Stochastic game-theoretic spectrum access in distributed and dynamic environment," *IEEE Trans. Veh. Technol.*, vol. 64, no. 10, pp. 4807-4820, Oct. 2015.

[43] W. Zhong, Y. Xu, and M. Tao, "Precoding strategy selection for cognitive mimo multiple access channels using learning automata," *in Proc. IEEE Int. Conf. Commun.*, Cape Town, South Africa, May 2010, pp. 1-5.

[44] Z. Han, C. Pandana, and K. J. R. Liu, "Distributive opportunistic spectrum access for cognitive radio using correlated equilibrium and no-regret learning," *in Proc. IEEE Wireless Commun. Netw. Conf.*, Hong Kong, China, Mar. 2007, pp. 11-15.

[45] L. Chen, S. Iellamo, M. Coupechoux, and P. Godlewski, "An auction framework for spectrum allocation with interference constraint in cognitive radio networks," *in Proc. 29th Annu. Joint Conf. IEEE Comput. Commun. Soc.*, San Diego, CA, Mar. 2010, pp. 1-9.

[46] S. Maharjan, Y. Zhang, C. Yuen, and S. Gjessing, "Distributed spectrum sensing in cognitive radio networks with fairness consideration: Efficiency of correlated equilibrium," *in Proc. IEEE 8th Int. Conf. Mobile Adhoc Sens. Syst.*, Valencia, Spain, Oct. 2011, pp. 540-549.

[47] Z. Han, D. Niyato, W. Saad, T. Basar, and A. Hjorungnes, *Game Theory in Wireless and Communication Networks*. Cambridge, U.K.: Cambridge Univ. Press, 2012.

[48] B. Wang, Y. Wu, K. J. R Liu, and T. Clancy, "An anti-jamming stochastic game for cognitive radio networks," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 4, pp. 877-889, Apr. 2011.

[49] Y. Gwon, S. Dastangoo, C. Fossa, and H. Kung, "Competing mobile network game: Embracing antijamming and jamming strategies with reinforcement learning," *in Proc. IEEE Conf. Commun. Netw. Secur.*, Oct. 2013, pp. 28-36.

[50] J. Li, K. Ramachandran, and T. K. Das, "A reinforcement learning (Nash-R) algorithm for average reward irreducible stochastic games," *J. Mach. Learn. Res.*, 2007.

[51] A. Greenwald, M. Zinkevich, and P. Kaelbling, "Correlated q-learning," *in Proc. 20th Int. Conf. Mach. Learn.*, Aug. 2003, pp. 242-249.

[52] H. Tembine, *Distributed Strategic Learning for Wireless Engineers*. Boca Raton, FL, USA: CRC Press, 2012.

[53] Y. Su and M. van der Schaar, "Dynamic conjectures in random access networks using bio-inspired learning," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 4, pp. 587-601, May 2010.

[54] X. Chen, Z. Zhao, and H. Zhang, "Stochastic power adaptation with multiagent reinforcement learning for cognitive wireless mesh networks," *IEEE Trans. Mobile Comput.*, vol. 12, no. 11, pp. 2155-2166, Nov. 2013.

[55] Y. Cao, D. Duan, X. Cheng, L. Yang, and J.Wei, "QoS-oriented wireless routing for smart meter data collection: Stochastic learning on graph," *IEEE Trans. Wireless Commun.*, vol. 13, no. 8, pp. 4470-4482, Aug. 2014.

[56] P. Sastry, V. Phansalkar, and M. Thathachar, "Decentralized learning of Nash equilibria in multi-person stochastic games with incomplete information", *IEEE Trans. Syst., Man, Cybern. B*, vol. 24, no. 5, pp. 769-777, 1994.

[57] M. Zandi, M. Dong, and A. Grami, "Distributed stochastic learning and adaptation to primary traffic for dynamic spectrum access," *IEEE Trans. on Wireless Commun.*, vol. 15, no. 3, pp. 1675-1688, Mar. 2016.

[58] Y. Xu, Q. Wu, J. Wang, L. Shen, and A. Anpalagan, "Robust multiuser sequential channel sensing and access in dynamic cognitive radio networks: Potential games and stochastic learning," *IEEE Trans. on Veh. Technol.*, vol. 64, no. 8, pp. 3594-3607, Aug. 2015.

[59] Q. Wu et al., "Distributed channel selection in time-varying radio environment: Interference mitigation game with uncoupled stochastic learning," *IEEE Trans. Veh. Technol.*, vol. 62, no. 9, pp. 4524-4538, Nov. 2013.

[60] J. Zheng, Y. Cai, Y. Xu, and A. Anpalagan, "Distributed channel selection for interference mitigation in dynamic environment: A game-theoretic stochastic learning solution," *IEEE Trans. Veh. Technol.*, vol. 63, no. 9, pp. 4757-4762, Nov. 2014.

[61] J. Zheng, Y. Cai, N. Lu, Y. Xu, and X. Shen, "Stochastic game-theoretic spectrum access in distributed and dynamic environment," *IEEE Trans. on Veh. Technol.*, vol. 64, no. 10, pp. 4807-4820, Oct. 2015.

[62] R. W. Rosenthal, "A class of games possessing pure strategy Nash equilibria," *J. Game Theory*, vol. 2, pp. 65-67, 1973.

[63] K. Kumar and Y. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *IEEE Comput.*, vol. 43, no. 4, pp. 51-56, Apr. 2010.

[64] J. Cohen, "Embedded speech recognition applications in mobile phones: Status, trends, and challenges," *in Proc. IEEE Int. Con. on Acoustics, Speech and Signal Proc.*, Mar. 2008, pp. 5352-5355.

[65] J. Shen, H. Tan, J. Wang, J. Wang, and S. Lee, "A novel routing protocol providing good transmission reliability in underwater sensor networks," *J. Int. Technol.*, vol. 16, no. 1, pp. 171-178, Jan. 2015.

[66] Z. Pan, Y. Zhang, and S. Kwong, "Efficient motion and disparity estimation optimization for low complexity multiview video coding," *IEEE Trans. Broadcast.*, vol. 61, no. 2, pp. 166-176, Jun. 2015.

[67] N. Fernando, S. Loke, and W. Rahayu, "Mobile cloud computing: A survey," *Future Gener. Comput. Syst.*, vol. 29, pp. 84-106, Jan. 2013.

[68] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal and Inf. Proc. over Netw.*, vol. 1, no. 2, Jun. 2015.

[69] F. Liu et al., "Gearing resource-poor mobile devices with powerful clouds: Architectures, challenges, and applications," *IEEE Wireless Commun.*, vol. 20, no. 3, pp. 14-22, Jun. 2013.

[70] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervas. Comput.*, vol. 8, no. 4, pp. 14-23, Oct. 2009.

[71] Y. Jararweh, L. Tawalbeh, F. Ababneh, and F. Dosari, "Resource efficient mobile computing using cloudlet infrastructure," *in Proc. IEEE 9th Int. Conf. on Mobile Ad-hoc and Sensor Netw.*, Dec. 2013, pp. 373-377.

[72] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," *in Proc. IEEE ISIT*, Jul. 2016, pp. 1451-1455.

[73] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590-3605, Dec. 2016.

[74] Y. Wang, M. Sheng, X. Wang, L. Wang, J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Trans. Commu.*, vol. 64, no. 10, Oct. 2016.

[75] L. Yang, J. Cao, H. Cheng, and Y. Ji, "Multi-user computation partitioning for latency sensitive mobile cloud applications," *IEEE Trans. Computers*, vol. 64, no. 8, pp. 2253-2266, Aug. 2015.

[76] L. Zhou, H. Wang, "Toward blind scheduling in mobile media cloud: Fairness, simplicity, and asymptotic optimality," *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 735-746, Jun. 2013.

[77] J. Xu, L. Chen, and S. Ren, "Online learning for offloading and autoscaling in energy harvesting mobile edge computing," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 3, pp. 361-373, Sep. 2017.

[78] C. Li et al., "Towards sustainable in-situ server systems in the big data era," *ACM SIGARCH Comput. Archit. News*, vol. 43, no. 3, pp. 14-26, 2015.

[79] T. Han and N. Ansari, "A traffic load balancing framework for software- defined radio access networks powered by hybrid energy sources," *IEEE/ACM Trans. Netw.*, vol. 24, no. 2, pp. 1038-1051, Mar. 2015.

[80] G. Piro et al., "HetNets powered by renewable energy sources: Sustainable next-generation cellular networks," *IEEE Internet Comput.*, vol. 17, no. 1, pp. 32-39, Jan. 2013.

[81] Y. Mao, Y. Luo, J. Zhang, and K. B. Letaief, "Energy harvesting small cell networks: Feasibility, deployment, and operation," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 94-101, Jun. 2015.

[82] J. Yang and S. Ulukus, "Optimal packet scheduling in an energy harvesting communication system," *IEEE Trans. Commun.*, vol. 60, no. 1, pp. 220-230, Jan. 2012.

[83] J. Gong, S. Zhou, and Z. Niu, "Optimal power allocation for energy harvesting and power grid coexisting wireless communication systems," *IEEE Trans. Commun.*, vol. 61, no. 7, pp. 3040-3049, Jul. 2013.

[84] V. Sharma, U. Mukerji, V. Joseph, and S. Gupta, "Optimal energy management policies for energy harvesting sensor nodes," *IEEE Trans. Wireless Commun.*, vol. 9, no. 4, pp. 1326-1336, Apr. 2010.

[85] S. Zhou, J. Gong, Z. Zhou, W. Chen, and Z. Niu, "GreenDelivery: Proactive content caching and push with energy-harvesting-based small cells," *IEEE Commun. Mag.*, vol. 53, no. 4, pp. 142-149, Apr. 2015.

[86] J. Qiao, Y. He, and X. (Sherman) Shen, "Proactive caching for mobile video streaming in millimeter wave 5G networks,"*IEEE Trans. Wireless Commun.*, vol. 15, no. 10, pp. 7187-7198, Oct. 2016.

[87] J. Tadrous, A. Eryilmaz, and H. EI Gamal, "Joint smart pricing and proactive content caching for mobile services," *IEEE/ACM Trans. Netw.*, vol. 24, no. 4, pp. 2357-2371, Aug. 2016.

[88] J. Du, C. Jiang, Y. Qian, Z. Han, and Y. Ren, "Resource allocation with video traffic prediction in cloud-based space systems," *IEEE Trans. Multimedia*, vol. 18, no. 5, pp. 820-830, May 2016.

[89] P. Si, H. Yue, Y. Zhang, and Y. Fang, "Spectrum management for proactive video caching in information-centric cognitive radio networks," *IEEE J. Sel. areas Commun.*, vol. 34, no. 8, pp. 2247-2259, Aug. 2016.

[90] J. Tadrous, and A. Eryilmaz, "On optimal proactive caching for mobile networks with demand uncertainties," *IEEE/ACM Trans. New.*, vol. 24, no. 5, pp. 2715-2727, Oct. 2016.

[91] C. Yi, S. Huang and J. Cai, "An incentive mechanism integrating joint power, channel and link management for social-aware D2D content sharing and proactive caching," *IEEE Trans. Mobile Comput.*, vol. PP, no. 99, pp. 1-1, 2017.

[92] H. Cao, H. Tian, J. Cai, A. S. Alfa, and S. Huang, "Dynamic load-balancing spectrum decision for heterogeneous services provisioning in multi-channel cognitive radio networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, Sep. 2017.

[93] S. Mahadevan, "Average reward reinforcement learning: Foundations, algorithms, and empirical results," *Mach. Learn.*, vol. 22, no. 1, pp. 159-195, 1996.

[94] D. Monderer and L. Shapley, "Potential games," *Games Economic Behavior*, vol. 14, pp. 124-143, 1996.

[95] K. Cohen, A. Leshem, and E. Zehavi, "Game theoretic aspects of the multi-channel ALOHA protocol in cognitive radio networks," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 11, pp. 2276-2288, Nov. 2013.

[96] K. Cohen and A. Leshem, "Distributed game-theoretic optimization and management of multichannel ALOHA networks," *IEEE/ACM Trans. on Netw.*, vol. 24, no. 3, pp. 1718-1731, Jun. 2016.

[97] C. Singh, A. Kumar, and R. Sundaresan, "Combined base station association and power control in multichannel cellular networks," *IEEE/ACM Trans. Netw.*, vol. 24, no. 2, pp. 1065-1080, Apr. 2016.

[98] B. Vcking and R. Aachen, "Congestion games: Optimization in competition", *in Proc. Algorithms Complexity Durham Workshop*, Sep. 2006.

[99] Y. Xu, J. Wang, Q. Wu, A. Anpalagan, and Y. Yao, "Opportunistic spectrum access in unknown dynamic environment: A game-theoretic stochastic learning solution", *IEEE Trans. on Wireless Commun.*, vol. 11, no. 4, pp. 1380 - 1391, Apr. 2012.

[100] K. Liu and Q. Zhao, "Distributed learning in multi-armed bandit with multiple player", *IEEE Trans. Signal Process.*, vol. 58, no. 11, pp. 5665-5681, Nov. 2010.

[101] A. Anandkumar, N. Michael, K. Tang, and A. Swami, "Distributed algorithms for learning and cognitive medium access with logarithmic regret", *IEEE J. Sel. Areas Commun.*, vol. 29, no. 4, pp. 731-745, Apr. 2011.

[102] Y. Gai and B. Krishnamachari, "Decentralized online learning algorithms for opportunistic spectrum access", *IEEE GLOBECOM'11*, pp. 1-6.

[103] W. Yu, T. Kwon, and C. Shin, "Multicell coordination via joint scheduling, beamforming and power spectrum adaptation", *IEEE INFOCOM'11*, pp. 2570-2578.

[104] Y. Gai, H. Liu, and B. Krishnamachari, "A packet dropping-based incentive mechanism for M/M/1 queues with selfish users", *IEEE INFOCOM'11*, pp. 2687-2695.

[105] A. Rudenko, P. Reiher, G. J. Popek, and G. H. Kuenning, "Saving portable computer battery power through remote process execution," *Mobile Comput. Commun. Rev.*, vol. 2, no. 1, pp. 19-26, 1998.

[106] D. Huang, P. Wang, and D. Niyato, "A dynamic offloading algorithm for mobile computing," *IEEE Trans. Wireless Commun.*, vol. 11, no. 6, pp. 1991-1995, Jun. 2012.

[107] Y. Wen, W. Zhang, and H. Luo, "Energy-optimal mobile application execution: Taming resource-poor mobile devices with cloud clones," *in Proc. IEEE INFOCOM*, Mar. 2012, pp. 2716-2720.

[108] Q. Zhao, L. Tong, A. Swami, et al., "Decentralized cognitive MAC for opportunistic spectrum access in ad hoc networks: A POMDP framework," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 3, pp. 589-600, Apr. 2007.

[109] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, Oct. 2016.

[110] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Paral. Distrib. Syst.*, vol. 26, no. 4, pp. 974-983, Apr. 2014.

[111] M. V. Barbera, S. Kosta, A. Mei, and J. Stefa, "To offload or not to offload? The bandwidth and energy costs of mobile cloud computing," *in Proc. IEEE INFOCOM*, 2013, pp. 1285-1293.

[112] J. Wallenius et al., "Multiple criteria decision making, multiattribute utility theory: Recent accomplishments and what lies ahead," *Manage. Sci.*, vol. 54, no. 7, pp. 1336-1349, 2008.

[113] A. Jin, Wei Song, and W. Zhuang, "Auction-based resource allocation for sharing cloudlets in mobile cloud computing," *IEEE Trans. Emerging Topics in Comput.*, vol. 6, no. 1, pp. 45-57, 2018.

[114] 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 Proc. IEEE ISCC*, Jul. 2012, pp. 59-66.

[115] T. S. Rappaport, *Wireless Communications: Principles and Practice*. Upper Saddle River, NJ, USA: Prentice Hall, 1996.

[116] T. Q. Quek, G. de la Roche, I. Güvenç, and M. Kountouris, *Small Cell Networks: Deployment, PHY Techniques, and Resource Management*. Cambridge, U.K.: Cambridge Univ. Press, 2013.

[117] Q. Wu, Y. Xu, J. Wang, L. Shen, J. Zheng, and A. Anpalagan, "Distributed channel selection in time varying radio environment: Interference mitigation game with uncoupled stochastic learning," *IEEE Trans. Veh. Technol.*, vol. 62, no. 9, pp. 4524-4538, Nov. 2013.

[118] J. Gong, S. Zhou, Z. Zhou, Z. Niu, "Policy optimization for content push via energy harvesting small cells in heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 2, pp. 717-729, Feb. 2017.

[119] J. Liao, K.-K. Wong, Y. Zhang, Z. Zheng, and K. Yang, "Coding, multicast, and cooperation for cache-enabled heterogeneous small cell networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 10, pp. 6838-6853, Oct. 2017.

[120] K. Poularakis, G. Iosifidis, and L. Tassiulas, "Approximation algorithms for mobile data caching in small cell networks," *IEEE Trans. Commun.*, vol. 62, no. 10, pp. 3665-3677, Oct. 2014.

[121] D. Lopez-Perez, I. Guvenc, G. de la Roche, M. Kountouris, T. Q. S. Quek, and J. Zhang, "Enhanced intercell interference coordination challenges in heterogeneous networks," *IEEE Wireless Commun.*, vol. 18, no. 3, pp. 22-30, Jun. 2011.

[122] D. Karamshuk, N. Sastry, M. Al-Bassam, A. Secker, and J. Chandaria, "Take-away TV: Recharging work commutes with predictive preloading of catch-up TV content," *IEEE J. sel. areas Commun.*, vol. 34, no. 8, pp. 2091-2101, Aug. 2016.

[123] S. Müller, O. Atan, M. v. d. Schaar, and A. Klein, "Context-aware proactive content caching with service differentiation in wireless networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 2, pp. 1024-1036, Feb. 2017.

[124] H. Ahlehagh and S. Dey, "Video-aware scheduling and caching in the radio access network," *IEEE/ACM Trans. Netw.*, vol. 22, no. 5, pp. 1444-1462, Oct. 2014.

[125] W. Hu, Y. Jin, Y. Wen, Z. Wang, and L. Sun, "Towards Wi-Fi AP-assisted content prefetching for on-demand TV series: A learning-based approach," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. PP, no. 99, pp. 1-1, 2017.

[126] W. Chen, and H. Vincent Poor, "Content pushing with request delay information," *IEEE Trans. Commun.*, vol. 65, no. 3, pp. 1146-1161, 2017.

[127] B. Guenter, N. Jain, and C. Williams, "Managing cost, performance, and reliability tradeoffs for energy-aware server provisioning," *in Proc. IEEE INFOCOM*, Shanghai, China, 2011, pp. 1332-1340.

[128] Y. Zhang and M. van der Schaar, "Structure-aware stochastic storage management in smart grids," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 6, pp. 1098-1110, Dec. 2014.

[129] C. J. C. H. Watkins, "Learning from delayed rewards," Ph. D. dissertation, Dept. Comput. Sci., Cambridge Univ., Cambridge, U. K., 1989.

[130] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *J. Mach. Learn. Res.*, vol. 10, pp. 1633-1685, Dec. 2009.

Appendix A

Appendices of Chapter 2

A.1 Proof of Theorem 2.2

In time slot t ($t \geq 1$), suppose SU m ($m \in \mathcal{M}$) is selected, and the current action profile of all the other SUs is denoted as $\boldsymbol{a}_{-m}(t)$. Following (2.22) (under scenario (I)) or (2.24) (under scenario (II)), the SU m chooses $a_m(t) = n^*(t)$ so that

$$\hat{R}_m(n^*(t), \boldsymbol{a}_{-m}(t)) \geq \hat{R}_m(n, \boldsymbol{a}_{-m}(t)) \quad \forall n \in \mathcal{N} \tag{A.1}$$

Using the property of ordinal potential games (2.7),

$$\Phi(n^*(t), \boldsymbol{a}_{-m}(t)) \geq \Phi(n, \boldsymbol{a}_{-m}(t)) \quad \forall n \in \mathcal{N} \tag{A.2}$$

where

$$\Phi(n^*(t), \boldsymbol{a}_{-m}(t)) = \begin{cases} \Phi_1(n^*(t), \boldsymbol{a}_{-m}(t)), & \text{in scenario } (I) \\ \Phi_2(n^*(t), \boldsymbol{a}_{-m}(t)), & \text{in scenario } (II) \end{cases} \tag{A.3}$$

Since $\boldsymbol{a}_{-m}(t-1) = \boldsymbol{a}_{-m}(t)$ and $a_m(t-1) \in \mathcal{N}$, (A.2) can be rewritten as

$$\Phi(\boldsymbol{a}(t)) \geq \Phi(\boldsymbol{a}(t-1)) \tag{A.4}$$

Thus, in both scenarios, the ordinal potential function $\Phi(\boldsymbol{a}(t))$ is non-decreasing with time t. Due to the bounded property of both $\Phi_1(a_m, \boldsymbol{a}_{-m})$ and $\Phi_2(a_m, \boldsymbol{a}_{-m})$ ($\forall m \in \mathcal{M}, \forall a_m \in \mathcal{N}$), $\Phi(\boldsymbol{a}(t))$ will converge to a (local) maximum. Since any (local) maximum of the ordinal potential function is a NE [56], the Theorem 2.2 holds.

A.2 Proof of Theorem 2.3

Define a function $\iota(\boldsymbol{P})$ as

$$\iota(\boldsymbol{P}) = \mathbb{E}[\Phi_2(a_m, \boldsymbol{a}_{-m})|\boldsymbol{P}]$$

$$= \sum_{n=1}^{N} P_{mn} \mathbb{E}[\Phi_2(n, \boldsymbol{a}_{-m}) | a_m = n, \boldsymbol{P}]$$
$$= \sum_{n=1}^{N} P_{mn} \sum_{a_1, \cdots, a_{m-1}, a_{m+1}, \cdots, a_M} \Phi_2(n, \boldsymbol{a}_{-m}) \prod_{m' \in \mathcal{M}, m' \neq m} P_{m' a_{m'}} \quad \text{(A.5)}$$

Thus,

$$\frac{\partial \iota(\boldsymbol{P})}{\partial P_{mn}} = \sum_{a_1, \cdots, a_{m-1}, a_{m+1}, \cdots, a_M} \Phi_2(n, \boldsymbol{a}_{-m}) \prod_{m' \in \mathcal{M}, m' \neq m} P_{m' a_{m'}}, \quad \forall m \in \mathcal{M}, \forall n \in \mathcal{N} \quad \text{(A.6)}$$

and

$$\frac{d\iota(\boldsymbol{P})}{dt'} = \sum_{m \in \mathcal{M}, n \in \mathcal{N}} \frac{\partial \iota(\boldsymbol{P})}{\partial P_{mn}} \frac{dP_{mn}}{dt'} \quad \text{(A.7)}$$

Substituting (2.35) and (A.6) into (A.7),

$$\frac{d\iota(\boldsymbol{P})}{dt'} = \sum_{m \in \mathcal{M}, n \in \mathcal{N}} \mathbb{E}[\Phi_2(n, \boldsymbol{a}_{-m}) | \boldsymbol{P}, a_m = n] P_{mn} \Big(\sum_{n' \in \mathcal{N}} P_{mn'} (\mathbb{E}[\bar{Rw}_m | \boldsymbol{P}, a_m = n] -$$
$$\mathbb{E}[\bar{Rw}_m | \boldsymbol{P}, a_m = n']) \Big)$$
$$= -\varpi \sum_{m \in \mathcal{M}, n \in \mathcal{N}, n' \in \mathcal{N}} \beta_m P_{mn} P_{mn'} (\mathbb{E}[\bar{Rw}_m | \boldsymbol{P}, a_m = n] - \mathbb{E}[\bar{Rw}_m | \boldsymbol{P}, a_m = n'])^2 \quad \text{(A.8)}$$

According to (2.12), $\beta_m < 0$, $\forall m \in \mathcal{M}$. Thus, $\iota(\boldsymbol{P})$ is non-decreasing along time. In addition, since $\iota(\boldsymbol{P})$ is bounded as specified in (A.5), $\boldsymbol{P}$ converges to $\boldsymbol{P}^*$ such that $\frac{d\iota(\boldsymbol{P}^*)}{dt'} = 0$. Thus, from (A.8) and (2.35),

$$\frac{d\iota(\boldsymbol{P}^*)}{dt'} = 0,$$
$$\Rightarrow P^*_{mn} P^*_{mn'} (\mathbb{E}[\bar{Rw}_m | \boldsymbol{P}^*, a_m = n] - \mathbb{E}[\bar{Rw}_m | \boldsymbol{P}^*, a_m = n'])^2 = 0, \quad \forall m \in \mathcal{M}, \forall n, n' \in \mathcal{N}$$
$$\Rightarrow \frac{dP^*_{mn}}{dt'} = 0, \quad \forall m \in \mathcal{M}, \forall n \in \mathcal{N}$$
$$\Rightarrow \boldsymbol{P}^* \text{ is a stable stationary point of the ODE (2.34)} \quad \text{(A.9)}$$

Following the Theorem 3.2 in [56] that all stable stationary points of the ODE are the NEs, Theorem 2.3 holds.

Appendix B

Appendices of Chapter 3

B.1 Proof of Theorem 3.3

Since $\boldsymbol{P}$ contains $(N+1) \times M$ components, denoted by P_{mi}, $m \in \mathcal{M}$, $i \in \mathcal{A}$, the component equations of (3.41) can be written as

$$\frac{dP_{mi}}{dt'} = P_{mi}(1-P_{mi})\mathbb{E}[\hat{Rw}_m(a_m(t), \boldsymbol{a}_{-m}(t))|\boldsymbol{P}(t) = \boldsymbol{P}, a_m(t) = i] + \sum_{i' \in \mathcal{A}, i' \neq i} P_{mi'}(-P_{mi})$$

$$\begin{aligned}
&\mathbb{E}[\hat{Rw}_m(a_m(t), \boldsymbol{a}_{-m}(t))|\boldsymbol{P}(t) = \boldsymbol{P}, a_m(t) = i'] \\
&= P_{mi} \sum_{i' \in \mathcal{A}, i' \neq i} P_{mi'} h_{mi}(\boldsymbol{P}) - P_{mi} \sum_{i' \in \mathcal{A}, i' \neq j} P_{mi'} h_{mi'}(\boldsymbol{P}) \\
&= P_{mi}(\sum_{i' \in \mathcal{A}, i' \neq i} P_{mi'}(h_{mi}(\boldsymbol{P}) - h_{mi'}(\boldsymbol{P}))) \\
&= P_{mi}(\sum_{i' \in \mathcal{A}} P_{mi'}(h_{mi}(\boldsymbol{P}) - h_{mi'}(\boldsymbol{P}))), \quad \forall m \in \mathcal{M}, \forall i \in \mathcal{A}
\end{aligned} \tag{B.1}$$

Then,

$$\begin{aligned}
\frac{dK(\boldsymbol{P})}{dt'} &= \sum_{m \in \mathcal{M}, i \in \mathcal{A}} \frac{\partial K(\boldsymbol{P})}{\partial P_{mi}} \frac{dP_{mi}}{dt'} \\
&= \sum_{m \in \mathcal{M}, i \in \mathcal{A}} \frac{\partial K(\boldsymbol{P})}{\partial P_{mi}} P_{mi}(\sum_{i' \in \mathcal{A}} P_{mi'}(h_{mi}(\boldsymbol{P}) - h_{mi'}(\boldsymbol{P}))) \\
&= \frac{1}{2} \sum_{m \in \mathcal{M}, i, i' \in \mathcal{A}} P_{mi} P_{mi'} (\frac{\partial K(\boldsymbol{P})}{\partial P_{mi}} - \frac{\partial K(\boldsymbol{P})}{\partial P_{mi'}})(h_{mi}(\boldsymbol{P}) - h_{mi'}(\boldsymbol{P}))
\end{aligned} \tag{B.2}$$

Substituting (3.43) into (B.2),

$$\frac{dK(\boldsymbol{P})}{dt'} = \frac{1}{2} c \sum_{m \in \mathcal{M}, i, i' \in \mathcal{A}} P_{mi} P_{mi'} (h_{mi}(\boldsymbol{P}) - h_{mi'}(\boldsymbol{P}))^2 \geq 0 \tag{B.3}$$

Thus, $K(\boldsymbol{P})$ is non-decreasing along time. In addition, since $K(\boldsymbol{P})$ is bounded, $\boldsymbol{P}$ converges to $\boldsymbol{P}^*$ such that $\frac{dK(\boldsymbol{P}^*)}{dt'} = 0$. From (B.3) and (B.1),

$$\begin{aligned}
&\frac{dK(\boldsymbol{P}^*)}{dt'} = 0 \\
&\Rightarrow P^*_{mj} P^*_{mi'} (h_{mi}(\boldsymbol{P}^*) - h_{mi'}(\boldsymbol{P}^*))^2 = 0, \forall m \in \mathcal{M}, \forall i, i' \in \mathcal{A} \\
&\Rightarrow \frac{dP^*_{mi}}{dt'} = 0, \quad \forall m \in \mathcal{M}, \forall i \in \mathcal{A} \\
&\Rightarrow \boldsymbol{P}^* \text{ is a stable stationary point of the ODE (3.41).}
\end{aligned} \tag{B.4}$$

According to Lemma 3.2, Theorem 3.3 holds.

B.2 Proof of Theorem 3.4

According to Theorem 3.3, the proposed FDCO algorithm converges to a pure strategy NE $\boldsymbol{a}^*$ that maximizes $K(\boldsymbol{P})$, i.e.,

$$\boldsymbol{a}^* = \arg\max_{\boldsymbol{P}} K(\boldsymbol{P}) = \arg\max_{\boldsymbol{a}} \psi(\boldsymbol{a}) \tag{B.5}$$

From (3.23),

$$\begin{aligned}
\psi(\boldsymbol{a}) &= \sum_{n=1}^{N} \sum_{v=1}^{s_n} F_{c,2}(v) + F_{l,2} \sum_{m=1}^{M} \mathbf{1}_{\{a_m=0\}} \\
&= \sum_{n=1}^{N} \sum_{v=1}^{s_n} F_{c,2}(v) + F_{l,2}(M - \sum_{n=1}^{N} s_n) \\
&= \sum_{n=1}^{N} (F_{c,2}(1) - F_{l,2} + F_{c,2}(2) - F_{l,2} + \cdots + F_{c,2}(s_n) - F_{l,2}) + F_{l,2} M
\end{aligned} \tag{B.6}$$

Since $F_{c,2}(v)$ specified by (3.24) is decreasing with v, it is easy to prove

$$\begin{aligned}
\max_{\boldsymbol{a}} \psi(\boldsymbol{a}) \iff & \max_{\boldsymbol{a}} \sum_{n=1}^{N} s_n \\
& \text{s.t.} F_{c,2}(s_n) \geq F_{l,2} \quad \forall n \in \mathcal{N} \\
\iff & \max_{\boldsymbol{a}} \sum_{m=1}^{M} \mathbf{1}_{\{a_m>0\}} \\
& \text{s.t.} F_{c,1}(s_{a_m}) \leq F_{l,1} \quad \forall a_m \in \mathcal{N}, m \in \mathcal{M}
\end{aligned} \tag{B.7}$$

Combining with (B.5),

$$\boldsymbol{a}^* = \arg\max_{\boldsymbol{a}} \psi(\boldsymbol{a})$$

$$= \begin{cases} \arg\max_{\boldsymbol{a}} \sum_{m=1}^{M} \mathbf{1}_{\{a_m>0\}} \\ \text{s.t.} F_{c,1}(s_{a_m}) \leq F_{l,1} \quad \forall a_m \in \mathcal{N}, m \in \mathcal{M} \end{cases} \tag{B.8}$$

B.3 Proof of Theorem 3.5

Let $\bar{\boldsymbol{a}}$ be an arbitrary pure strategy NE achieved by the FDCO algorithm, and $\bar{s}_n$ be the number of mobile devices selecting channel n ($\forall n \in \mathcal{N}$) at NE $\bar{\boldsymbol{a}}$. Given $F_{c,1}(1) \leq F_{l,1}$, $\sum_{m=1}^{M} O_m(\bar{\boldsymbol{a}}) = MF_{l,1}$ if $F_{c,1}(1) = F_{l,1}$. In the following, I focus on the case that $F_{c,1}(1) < F_{l,1}$, i.e., there exists at least one mobile device $\bar{m}$ that chooses cloudlet computing via channel $\bar{a}_{\bar{m}} \in \mathcal{N}$. According to the concept of NEs,

$$O_{\bar{m}}(\bar{a}_{\bar{m}}, \bar{\boldsymbol{a}}_{-\bar{m}}) \leq F_{l,1} \tag{B.9}$$

$$O_{\bar{m}}(\bar{a}_{\bar{m}}, \bar{\boldsymbol{a}}_{-\bar{m}}) \leq O_{\bar{m}}(i, \bar{\boldsymbol{a}}_{-\bar{m}}) \quad \forall i \in \mathcal{N} \tag{B.10}$$

Summing up both sides of (B.10) over i yields

$$\begin{aligned} &\sum_{i\in\mathcal{N}} O_{\bar{m}}(\bar{a}_{\bar{m}}, \bar{\boldsymbol{a}}_{-\bar{m}}) \leq \sum_{i\in\mathcal{N}} O_{\bar{m}}(i, \bar{\boldsymbol{a}}_{-\bar{m}}) \\ &\Rightarrow O_{\bar{m}}(\bar{a}_{\bar{m}}, \bar{\boldsymbol{a}}_{-\bar{m}}) \leq \frac{\sum_{i\in\mathcal{N}} O_{\bar{m}}(i, \bar{\boldsymbol{a}}_{-\bar{m}})}{N} \end{aligned} \tag{B.11}$$

Since

$$\begin{aligned} \sum_{i\in\mathcal{N}} O_{\bar{m}}(i, \bar{\boldsymbol{a}}_{-\bar{m}}) &= F_{c,1}(\bar{s}_{\bar{a}_{\bar{m}}}) + \sum_{i\in\mathcal{N}, i\neq\bar{a}_{\bar{m}}} F_{c,1}(\bar{s}_i + 1) \\ &= \frac{1}{F_{c,2}(1)} \sum_{i\in\mathcal{N}} \bar{s}_i + N\lambda_1 \lceil \frac{N_{cyc}}{f_c\tau} \rceil + (N-1)\frac{1}{F_{c,2}(1)} \\ &\leq \frac{1}{F_{c,2}(1)} M + N\lambda_1 \lceil \frac{N_{cyc}}{f_c\tau} \rceil + (N-1)\frac{1}{F_{c,2}(1)} \end{aligned} \tag{B.12}$$

equation (B.11) can be rewritten as

$$\begin{aligned} O_{\bar{m}}(\bar{a}_{\bar{m}}, \bar{\boldsymbol{a}}_{-\bar{m}}) &\leq \frac{1}{NF_{c,2}(1)} M + \lambda_1 \lceil \frac{N_{cyc}}{f_c\tau} \rceil + \frac{N-1}{N}\frac{1}{F_{c,2}(1)} \\ &= \frac{Z}{N} \end{aligned} \tag{B.13}$$

Combining (B.9) and (B.13),

$$O_{\bar{m}}(\bar{a}_{\bar{m}}, \bar{\boldsymbol{a}}_{-\bar{m}}) \leq \min(F_{l,1}, \frac{Z}{N}) \tag{B.14}$$

Since

$$C_{FDCO} = \sum_{m=1}^{M} O_m(\bar{a}_m, \bar{\boldsymbol{a}}_{-m})$$

$$= \sum_{m\in\mathcal{M},\bar{a}_m>0} O_m(\bar{a}_m, \bar{\boldsymbol{a}}_{-m}) + \sum_{m\in\mathcal{M},\bar{a}_m=0} F_{l,1}$$
$$= \sum_{m\in\mathcal{M},\bar{a}_m>0} O_m(\bar{a}_m, \bar{\boldsymbol{a}}_{-m}) + (M - \sum_{m\in\mathcal{M}} \mathbf{1}_{\{\bar{a}_m>0\}})F_{l,1} \quad \text{(B.15)}$$

according to (B.14), (B.15) can then be rewritten as

$$C_{FDCO} \leq \min(F_{l,1}, \frac{Z}{N}) \sum_{m\in\mathcal{M}} \mathbf{1}_{\{\bar{a}_m>0\}} + F_{l,1}(M - \sum_{m\in\mathcal{M}} \mathbf{1}_{\{\bar{a}_m>0\}})$$
$$= (\min(F_{l,1}, \frac{Z}{N}) - F_{l,1}) \sum_{m\in\mathcal{M}} \mathbf{1}_{\{\bar{a}_m>0\}} + MF_{l,1} \quad \text{(B.16)}$$

Obviously, at the NE $\bar{\boldsymbol{a}}$, the number of beneficial cloudlet computing mobile devices is no more than M, i.e., $\sum_{m\in\mathcal{M}} \mathbf{1}_{\{\bar{a}_m>0\}} \leq M$. If $\sum_{m\in\mathcal{M}} \mathbf{1}_{\{\bar{a}_m>0\}} = M$, from (B.16),

$$C_{FDCO} \leq (\min(F_{l,1}, \frac{Z}{N}) - F_{l,1})M + MF_{l,1} \quad \text{(B.17)}$$

If $\sum_{m\in\mathcal{M}} \mathbf{1}_{\{\bar{a}_m>0\}} < M$, there exists at least one mobile device $\hat{m}$ that takes an action $\bar{a}_{\hat{m}} = 0$. Since at the NE $\bar{\boldsymbol{a}}$, the mobile device $\hat{m}$ cannot reduce its execution cost by choosing cloudlet computing via any channel $n \in \mathcal{N}$, i.e.,

$$F_{c,1}(\bar{s}_n + 1) \geq F_{l,1} \quad \forall n \in \mathcal{N}$$
$$\Rightarrow \bar{s}_n \geq \frac{F_{c,2}(1)}{F_{l,2}} - 1 \quad \forall n \in \mathcal{N} \quad \text{(B.18)}$$

then,

$$\sum_{m\in\mathcal{M}} \mathbf{1}_{\{\bar{a}_m>0\}} = \sum_{n=1}^{N} \bar{s}_n \geq N(\frac{F_{c,2}(1)}{F_{l,2}} - 1) \quad \text{(B.19)}$$

Since $(\min(F_{l,1}, \frac{Z}{N}) - F_{l,1}) \leq 0$, substituting (B.19) into (B.16),

$$C_{FDCO} \leq (\min(F_{l,1}, \frac{Z}{N}) - F_{l,1})N(\frac{F_{c,2}(1)}{F_{l,2}} - 1) + MF_{l,1} \quad \text{(B.20)}$$

Thus, the Theorem 3.5 holds.

List of Publications

[1] **Huijin Cao**, Jun Cai, "Online adaptive transmission strategy for buffer-aided cooperative NOMA systems," accepted by *IEEE Transactions on Mobile Computing*.

[2] **Huijin Cao** and Jun Cai, "Distributed multiuser computation offloading for cloudlet-based mobile cloud computing: A game-theoretic machine learning approach," *IEEE Transactions on Vehicular Technology*, vol. 67, no.1, Jan. 2018.

[3] **Huijin Cao**, Jun Cai, "Distributed opportunistic spectrum access in an unknown and dynamic environment: A stochastic learning approach, *IEEE Transactions on Vehicular Technology*, vol. PP, no. 99, 2018.

[4] **Huijin Cao**, Hongqiao Tian, Jun Cai, Attahiru S. Alfa, "Dynamic load-balancing spectrum decision for heterogeneous services provisioning in multi-channel cognitive radio networks, *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, Sep. 2017.

[5] **Huijin Cao**, Jun Cai, "Context-Aware Proactive Caching for Heterogeneous Networks with Energy Harvesting: An Online Learning Approach," *Greencom*, Halifax, Canada, Jul. 2018.

[6] Hongqiao Tian, Jun Cai, Attahiru S. Alfa, Shiwei Huang and **Huijin Cao**, "Dynamic Load-Balancing Spectrum Decision for Cognitive Radio Networks with Multi-Class Services," in *proc. of 2015 International Conference on Wireless Communications and Signal Processing (WCSP)*, Nanjing, China, pp. 1–5, Oct. 2015.

[7] **Huijin Cao**, Jun Cai, Attahiru Alfa, Zhen Zhao, "Efficient resource allocation scheduling for MIMO-OFDMA-CR downlink systems, *2016 8th International Conference on Wireless Communications and Signal Processing (WCSP)*.

[8] Zhen Zhao, Changyan Yi, Jun Cai, **Huijin Cao**, Queueing Analysis for Medical Data Transmissions with Delay-Dependent Packet Priorities in WBANs, *2016*

8th International Conference on Wireless Communications and Signal Processing (WCSP).