On Optimal Radio Access Technology Selection in Heterogeneous Wireless Networks

A thesis submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

by

Arghyadip Roy
(Roll No. 134076018)

Under the guidance of:
Prof. Abhay Karandikar
and
Prof. Prasanna Chaporkar

Department of Electrical Engineering
Indian Institute of Technology Bombay
Powai, Mumbai 400076
2019

To my parents (Mr. Saktipada Roy and Mrs. Sujata Roy) and brother (Aikyadip)

Thesis Approval

The thesis titled

On Optimal Radio Access Technology Selection in Heterogeneous Wireless Networks

by

Arghyadip Roy

(Roll No. 134076018)

is approved for the degree of

Doctor of Philosophy

[illegible] 15/7/19

Prof. Neelesh Mehta (Examiner)

[illegible]

Prof. Gaurav Kasbekar (Examiner)

[illegible]

Prof. Abhay Karandikar (Advisor)

[illegible]

Prof. Prasanna Chaporkar (Co-Advisor)

M. Ravikanth

Prof. M. Ravikanth (Chairperson)

Date: 15/07/2019

Place: Mumbai

Declaration

I, Arghyadip Roy, declare that this written submission represents my ideas in my own words and wherever others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/ data/ fact/ source in my submission. I understand that any violation of the above will be cause for disciplinary action by IIT Bombay and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Date:	July 15, 2019
Place:	Mumbai

[illegible]

Arghyadip Roy

Abstract

In a heterogeneous wireless network, a user can associate with any Radio Access Technology (RAT) when there are multiple RATs available and can move seamlessly among them. To handle the explosive growth of traffic in cellular network, the idea of mobile data traffic offloading to Wireless Fidelity (WiFi) has been proposed. Today's wireless networks consist of a multitude of Radio Access Technologies (RATs), each being controlled by individual controllers. The unprecedented growth of data traffic is driving academia and standardization organizations towards the inter-working of various RATs which can circumvent the problem of suboptimal utilization of network-wide resources. Application of Software Defined Networking (SDN) principles enables the control and management of various RATs in a unified way.

In this thesis, we focus on mobile data offload assisted optimal association problem in a heterogeneous network. The problem where we aim to maximize the total system throughput is formulated within the framework of Markov Decision Process (MDP). Another problem where we aim to maximize the total system throughput subject to a constraint on the blocking probability of voice users is formulated as a Constrained Markov Decision Process (CMDP). Relative Value Iteration Algorithm (RVIA) and gradient descent algorithms are used to determine the optimal policy. In our model, we consider the possibility of mobile data user offload from one RAT to another during the association or the departure of a user. Optimality of the threshold policy is derived. Based on the threshold based optimal policies, we propose two computationally efficient algorithms for RAT selection in a Fourth Generation (4G) Long Term Evaluation (LTE)-WiFi Heterogeneous Network (HetNet) and conduct extensive simulations in Network Simulator-3 (ns-3) to compare their performance with existing RAT selection algorithms. Simulation results indicate that algorithms proposed by us perform better than the other schemes in improving different system metrics like the total system throughput and the blocking probability of voice users.

However, these algorithms require the knowledge of the statistics of the arrival processes of voice and data users. To address this, an online algorithm for optimal RAT selection is proposed based on a RVIA centric Q-learning approach. The proposed algorithm can be implemented without any explicit knowledge of the arrival processes of voice and data users. Simulation results are presented to exhibit the convergence behavior of the proposed scheme to the optimal policy.

Although the convergence to optimality is guaranteed, this learning scheme need to iteratively learn the value functions for all state-action pairs, thus possessing large memory requirement. Additionally, due to the associated exploration mechanism, the convergence rate is slow, especially under a large state space. To address this issue, we propose a Post-Decision State (PDS) learning algorithm which speeds up the learning process by removing the requirement of action exploration mechanism in the Q-learning algorithm. Furthermore, the PDS learning algorithm has a lower space complexity than that of the Q-learning algorithm because instead of the state-action pair values we need to store only the value function associated with the states. We prove that the PDS learning algorithm indeed converges to the optimality. ns-3 simulation results are presented to demonstrate the convergence behaviors of Q-learning and PDS learning algorithms.

However, the convergence speed and storage complexity of Q-learning and PDS learning algorithms can be further improved if the underlying threshold structure can be exploited while learning. Therefore, we propose a structure-aware online learning algorithm for RAT selection which reduces the feasible policy space, thereby offering lesser storage and computational complexity and faster convergence. We prove that the proposed algorithm converges to the optimal policy. We present simulation results to demonstrate how the knowledge of structural properties affect the convergence speed, when compared to traditional learning schemes.

RAT selection solutions lack the consideration of practical network parameters such as channel states of users, control signaling overheads associated with mobile data offloading. We consider a network where users of different priorities are present. To take into account the practical network aspects, we consider the problem of optimal RAT selection to maximize the total system throughput subject to constraints on the high priority user blocking probability and the low priority user offloading probability and formulate the problem as a CMDP. We reduce the dimensionality of the action space by eliminating the provably sub-optimal actions. Moreover, to address the curse of dimensionality and the curse of modeling associated with the computation of optimal policy, we propose two low-complexity online heuristic algorithms for RAT selection.

To conduct experiments, we develop an ns-3 based evaluation platform based on an SDN based network architecture for the control and management of LTE and WiFi networks. We restructure the existing modules of ns-3 towards this purpose. Experimental results demonstrate that the proposed algorithms provide near-optimal performance and outperform traditional RAT selection algorithms under realistic network scenarios including user mobility.

Contents

Acknowledgments

The work presented in this thesis would not have been possible without the support of many people who have helped me through my journey as a doctoral student at IIT Bombay. I take this opportunity to express my sincere gratitude and appreciation to all who have made this Ph.D. thesis possible.

First and foremost, I would like to thank my advisor Prof. Abhay Karandikar for his invaluable guidance and encouragement. His mentoring in the conceptualization phase, the research work phase and the paper and thesis writing phases has been tremendously useful. His enthusiasm, positive outlook and unparalleled energy have always been a source of inspiration. Each discussion with him has provided me new insights and learnings which will always remain valuable to me. He has been extremely supportive whenever I needed any help. I have learned from him how to approach a problem in a systematic way and how to handle critical situations with patience. I am really proud and privileged to have a fulfilling experience of working under his guidance.

My special thanks also goes to my co-advisor Prof. Prasanna Chaporkar for his continuous support, guidance and encouragement. From him, I have learned how to keep patience in tough situations and handle them in a calm way. I owe a lot of gratitude for him for always being there whenever I needed any help. Countless discussions with Prof. Karandikar and Prof. Chaporkar has been extremely useful for me and I am proud that I will be known as their students.

I am very grateful to Prof. Vivek Borkar for his technical inputs during the course of my Ph.D. I consider myself very fortunate to have the opportunity to interact with him. His technical genius coupled with a great sense of humor has always been a pleasant experience to me.

I would also like to thank Mr. Pranav Jha for his guidance and help during the journey of my Ph.D. Working with him was really a great experience. His enthusiasm coupled with extreme patience made every discussion and conversation between us interactive and interesting.

I would like to thank my Research Progress Committee (RPC) members, Prof. Gaurav Kasbekar and Prof. Sharayu Moharir for their valuable suggestions, comments and feedback regarding my work. Their perspectives have brought greater clarity about my work to me. I thank them for their suggestions which have helped me improving the overall quality of the thesis.

I appreciate the assistance provided by Sangeeta Ma'am, Sonal, Margaret, Beena, Aditya Ji and Rajesh Ji in all official works. I acknowledge all the help provided by EE Office staff, in particular Mrs. Madhu and Mr. Santosh. Their prompt responses made the official work smooth.

It is always a pleasure to thank my labmates at IIT Bombay. I want to thank Akshatha, Indu, Sadaf, Pragnya, Shashi, Meghna and Meera for the wonderful times we had. Also, I want to thank them for hundreds of “tea” discussions in the GG canteen over the last five years. I would also like to thank my friends including Sudip, Bithin, Anjan, Manna, Amitrajit, Sanjay, Shahid and Sonia who made my life at IIT Bombay memorable. Many thanks to friends who may not be physically close to me but has always been close to my heart. I want to thank Rik, Amit, Debayan, Abhishek, Aritra, Nabanita, Amrita, Sumagna, Nilanjan, Meghdoot and Asta-da for their unconditional support throughout this journey.

The journey would have been impossible without the support of my family and relatives. I thank my family for encouraging me to follow my dreams. I am very grateful to my parents, Mr. Saktipada Roy and Mrs. Sujata Roy and my brother Aikyadip for their emotional support. This thesis is dedicated to my wonderful family.

Arghyadip Roy
March 2019

List of Acronyms

2G	Second Generation
3G	Third Generation
4G	Fourth Generation
5G	Fifth Generation
3GPP	Third Generation Partnership Project
ANDSF	Access Network Discovery and Selection Function
AP	Access Point
APN	Access Point Name
a.s.	almost surely
BS	Base Station
BSS	Basic Service Set
CAA	Constrained Markov Decision Process based Association Algorithm
CBR	Constant Bit Rate
CMDP	Constrained Markov Decision Process
CQI	Channel Quality Indicator
CSMA/CA	Carrier Sense Multiple Access/ Collision Avoidance
CTS	Clear to Send
DCF	Distributed Coordination Function
DHCP	Dynamic Host Configuration Protocol
DIFS	Distributed Coordination Function Inter-Frame Space
DNS	Domain Name System
DP	Dynamic Programming
DPI	Deep Packet Inspection
DTMC	Discrete Time Markov Chain

EB	ExaByte
eNodeB	Evolved NodeB
EPC	Evolved Packet Core
GBR	Guaranteed Bit Rate
GSM	Global System for Mobile Communications
HetNet	Heterogeneous Network
IFOM	Internet Protocol Flow Mobility
IMSI	International Mobile Subscriber Identity
IP	Internet Protocol
I-WLAN	Interworking-Wireless Local Area Network
LM	Lagrange Multiplier
LTE	Long Term Evolution
MAC	Medium Access Control
MAPCON	Multi-Access Packet Data Network Connectivity
MCC	Mobile Country Code
MCSA	Myopic with Constraint Satisfaction Algorithm
MDP	Markov Decision Process
MME	Mobility Management Entity
MNC	Mobile Network Code
ODE	Ordinary Differential Equation
OFDM	Orthogonal Frequency Division Multiplexing
OFDMA	Orthogonal Frequency Division Multiple Access
PDCP	Packet Data Convergence Protocol
PDN	Packet Data Network
PDS	Post-Decision State
PDU	Protocol Data Unit
PLMN	Public Land Mobile Network
PRB	Physical Resource Block
QoS	Quality of Service
RAT	Radio Access Technology
RE	Resource Element

R.H.S.	Right Hand Side
RL	Reinforcement Learning
RRC	Radio Resource Control
RRM	Radio Resource Management
RTS	Request To Send
RVIA	Relative Value Iteration Algorithm
SA	Stochastic Approximation
SAL	Structure-Aware Learning
SDN	Software Define Networking
SIFS	Short Inter-Frame Space
SMCSA	State-aware Myopic with Constraint Satisfaction Algorithm
SMDP	Semi-Markov Decision Process
SNR	Signal to Noise Ratio
STA	Wireless Station
TCP	Transmission Control Protocol
TTI	Transmission Time Interval
UDP	User Datagram Protocol
UE	User Equipment
UMAA	Unconstrained Markov Decision Process-based Association Algorithm
UMTS	Universal Mobile Telecommunications System
VIA	Value Iteration Algorithm
VoIP	Voice over Internet Protocol
WBA	Wireless Broadband Alliance
WiFi	Wireless Fidelity
WLAN	Wireless Local Area Network
WPA2	Wireless Fidelity Protected Access 2

List of Symbols

β	Lagrange multiplier
$B_{\max}$	Constraint on the blocking probability
C	Number of common resource blocks reserved in LTE for voice and data users
C_L	Number of resource blocks in LTE
λ_d	Average arrival rate of data users
λ_H	Average arrival rate of high priority users
λ_L	Average arrival rate of low priority users
λ_v	Average arrival rate of voice users
μ_d	Average service rate of data users
μ_H	Average service rate of high priority users
μ_L	Average service rate of low priority users
μ_v	Average service rate of voice users
$O_{\max}$	Constraint on the offloading probability
p_g	Probability that the channel state of the arriving user in LTE is good
$R_{L,D}$	Bit rate of a single data user in LTE
$R_{L,H}$	Bit rate of a high prioirty user in LTE
$R_{L,L}$	Data rate for a single resource block in LTE for low priority data users
$R_{L,V}$	Bit rate of a single voice user in LTE
$R_{W,D}(k)$	Per-user throughput of k users in WiFi
θ	Threshold vector
W	Maximum number of WiFi users so that per-user throughput is greater than a threshold

List of Tables

List of Figures

Chapter 1

Introduction

Recent years have witnessed an ever-increasing demand for high data rate in cellular mobile networks. The key factors affecting the demand are increase in the number of always-connected devices such as smartphones and tablets, large scale deployment of machine-to-machine devices and widespread use of entertainment services like YouTube, Apples iTunes. Worldwide monthly mobile data traffic stood at 19 ExaBytes (EBs) by the end of 2018. However, according to the industry forecast, monthly mobile data traffic is expected to exceed 49 EBs by 2021 [1, 2]. Figure in [2] also reveals that the monthly consumption of data is expected to increase consistently across the globe over the years. As a result of this increase in data traffic, the data user density is continuously growing. This is one of the biggest challenges to be handled in upcoming next generation wireless networks.

High data user density in small geographical regions may produce temporal and spatial variations in traffic pattern . The role of the current Fourth Generation (4G) network is to meet the requirement of spectral efficiency for such a dense geographical region. This necessitates additional spectral efficiency improvement mechanisms for mobile data traffic. However, the Quality of Service (QoS) requirements of different applications to ensure satisfactory performances are diverse. Real time applications such as Voice over Internet Protocol (VoIP), video streaming generally have strict delay and rate requirements [3]. On the other hand, non-real time applications such as file downloads do not have any strict data rate or delay requirement. However, file transfer applications cannot tolerate any packet loss. The goal of the next generation wireless network is to meet these diverse

QoS requirements of different applications while improving the spectral efficiency.

Several techniques have been suggested in the literature to mitigate the issue of handling an enormous amount of mobile data traffic, while satisfying QoS requirements of users in the 4G network. Due to the scarcity of the spectrum and the gradual saturation of physical layer based solutions, network layer based approaches for improving the spectral efficiency of cellular networks are of more interest. The idea of Heterogeneous Network (HetNet) is a network layer based concept, where an overlay of low power small cells (pico cells and femto cells), Wireless Local Area Network (WLAN) (popularly known as Wireless Fidelity (WiFi) network) Access Points (APs) is performed on existing Third Generation Partnership Project (3GPP) [4] 4G Long Term Evolution (LTE) [5] macro Base Stations (BSs). As demonstrated in Fig. 1.1, while the macro BSs ensure wide coverage area, small cells and WiFi APs are intended to serve high density geographical regions. Upcoming Fifth Generation (5G) [6, 7] cellular system aims to provide high data rate, reduced latency and low energy consumption along with a support for higher system capacity. It also targets to provide connectivity to a large number of devices. To cater to these various needs, it is expected that the future wireless system will be a conglomeration of a large number of Radio Access Technologies (RATs). Therefore, with the advent of future 5G system, HetNets are envisioned to contain a wide variety of RATs with different characteristics.

In a typical HetNet comprising multiple RATs, users can be associated with any RAT. Moreover, after association, based on certain conditions, some users' data can be steered to any other RAT, alleviating the cellular network load and improving the overall user experience. WiFi is a suitable candidate for steering traffic in a 4G LTE HetNet because it operates on the unlicensed spectrum, and the deployment cum operating cost of WiFi is low. The mechanism of steering the LTE data traffic to WiFi is known as LTE-WiFi mobile data offloading. The idea of mobile data offloading from LTE to WiFi has been standardized in 3GPP Release 12 specifications [8].

We consider a 3GPP LTE cellular network with an overlay of IEEE 802.11 based WiFi APs for our analysis. In Sections 1.1 and 1.2, we provide overview of LTE and WiFi, respectively. A description of architectures and mechanisms for the interworking between LTE and WiFi networks in an LTE-WiFi HetNet is provided in Section 1.3. A discussion

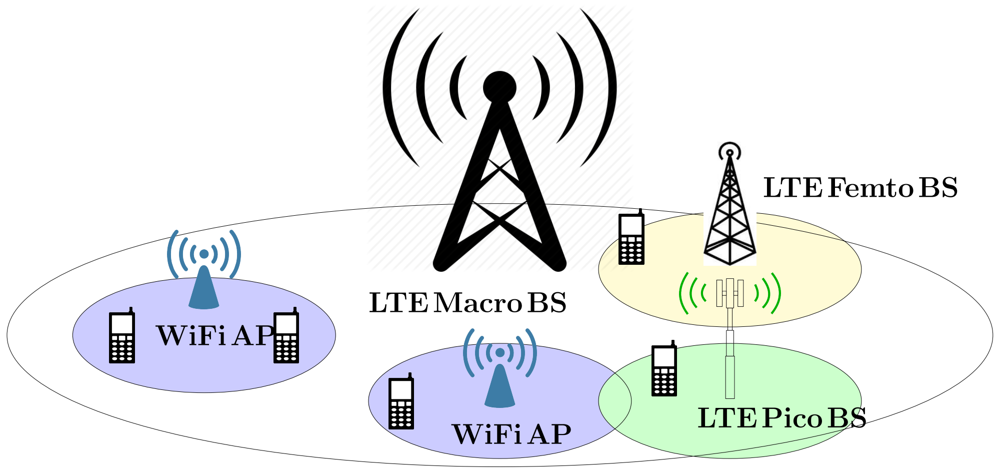


Figure 1.1: A heterogeneous network comprising various RATs.

on the main challenges involved in the RAT selection in HetNets is provided in Section 1.4. In the subsequent sections, we highlight the motivation and main contributions of the thesis followed by the organization of the thesis.

1.1 Overview of Long Term Evolution (LTE)

LTE [5] technology emerges as a part of 4G wireless communication which succeeds the older Second Generation (2G) and Third Generation (3G) technologies. Contrary to the older generations of wireless technologies, LTE introduces an all Internet Protocol (IP) packet core network for voice and data services. Furthermore, LTE provides significantly higher data rate and better coverage compared to 2G and 3G technologies. LTE also facilitates seamless mobility of users, reduced latency and flexible network deployment.

The physical layer of LTE uses Orthogonal Frequency Division Multiplexing (OFDM) to counteract the frequency selective fading. OFDM divides the wide-band frequency-selective fading channel into a number of narrow-band subchannels (known as *sub-carriers*). Since each sub-carrier is narrow-band, the transmission in each sub-carrier experiences flat fading. Thus, the system is able to counteract multipath fading [9], thereby greatly enhancing the performance compared to older generation networks.

Using OFDM, multiple users can be assigned different groups of sub-carriers in a single time slot. This multiple access scheme where multiple users can be scheduled

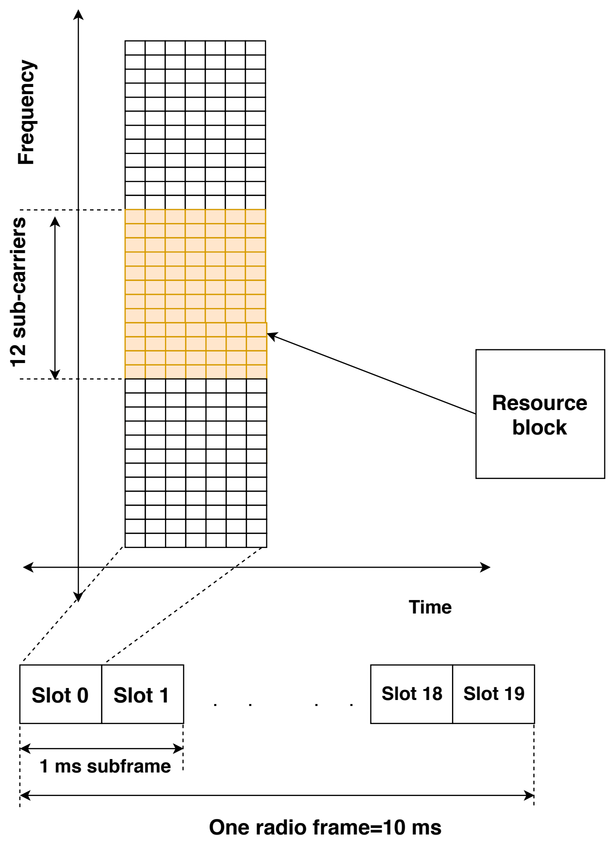


Figure 1.2: Resource block structure in LTE.

using OFDM is known as Orthogonal Frequency-Division Multiple Access (OFDMA). Due to the time varying and frequency-selective nature of the wireless channels, the fading characteristics of different sub-carriers are different. Therefore, in OFDMA, each user is allocated the set of sub-carriers which provides the highest throughput. This results in multi-user diversity which improves the spectral efficiency of the system [9, 10].

In LTE, the allocated radio resources may be defined in terms of time, space, frequency or a combination of them. The space dimension comes into picture when multiple antennas are present in the system. For a given antenna, resources can be defined in terms of time and frequency. The time-frequency resource structure is illustrated in Fig. 1.2. The length of a radio frame is 10 ms. A radio frame is divided into ten subframes. Each subframe consists of 2 time slots, each of which has a duration of 0.5 ms. The number of OFDM symbols belonging to each slot depends on the sub-carrier spacing and the length of a cyclic prefix [9]. Each slot contains six or seven symbols. A *Physical Resource Block* (PRB) is defined as a group of 12 sub-carriers, each having a bandwidth of 180

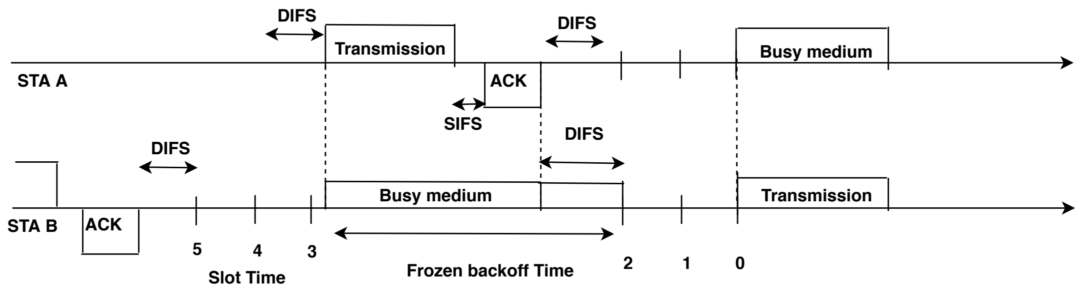


Figure 1.3: Basic access mechanism in WiFi.

KHz with a sub-carrier spacing of 15 KHz over a single slot. The smallest resource unit in LTE is known as a *Resource Element* (RE). An RE comprises one sub-carrier in one slot. The unit of data transmission in LTE is known as *transport block* which refers to the Protocol Data Unit (PDU) of the Medium Access Control (MAC) layer. Transport blocks are moved from the MAC layer to the physical layer once every subframe (1 ms) which is known as Transmission Time Interval (TTI).

1.2 Overview of Wireless Fidelity (WiFi)

WiFi is a non-cellular broadband wireless technology which has become popular due to its low operating cost, higher data rate and easy extensibility [11, 12]. Developments in WiFi technology are driven by IEEE 802.11 family of standards [13]. Table 1.1 provides a comparison of various WiFi standards. The major components of 802.11 networks are

Table 1.1: Comparison of various WiFi standards.

WiFi Standard	**Maximum data rate**	**Frequency of operation**
802.11a	54 Mbps	5.8 GHz
802.11b	11 Mbps	2.4 GHz
802.11g	54 Mbps	2.4 GHz
802.11n	600 Mbps	2.4 GHz/ 5 GHz
802.11ac	3.4 Gbps	5 GHz

AP and Wireless Stations (STAs) which communicate with each other over the wireless medium. The basic building block of a WiFi network is the *Basic Service Set* (BSS) which consists of a group of STAs communicating among each other over a *basic service area* [11]. In this thesis, we focus on an *infrastructure* BSS system [12] where all communications between the STAs happen through the APs. Before the delivery of data to STAs, STAs need to register or associate with the APs. Although association is a mandatory function, IEEE 802.11 does not specify any particular implementation strategy.

IEEE 802.11 MAC uses a *Carrier Sense Multiple Access/ Collision Avoidance* (CSMA/CA) scheme to control the access to the wireless medium. The basic mechanism for accessing the medium in CSMA/CA is Distributed Coordination Function (DCF). Before sending the data, the STAs and the APs sense the channel. If they detect that the channel is idle, then they start sending the data. Retransmission of collided packets are handled using exponential back-off rules. However, due to collision, wireless capacity is wasted. DCF introduces two techniques for packet transmission, viz., two-way handshaking (also known as *basic access*) and four-way handshaking techniques. In the basic access technique, the receiver sends a positive acknowledgement after successfully receiving the packet sent by the sender. In four-way handshaking, IEEE 802.11 uses the concept of Request to Send (RTS) and Clear to Send (CTS) signals. The RTS/CTS mechanism is designed to counteract the *hidden terminal* problem [11]. In this process, a wireless node which intends to send data, initially sends an RTS signal. When the target node receives the RTS signal, it sends a CTS signal to the source node. Other nodes which hear the RTS/CTS signal remain silent. After the exchange of RTS and CTS signals is over, the sender node can start transmitting data without any possibility of collision. However, the RTS/CTS mechanism consumes a certain amount of wireless capacity and incurs an additional latency before the data transfer starts. When the data frames are received by the receiver, they must acknowledge the reception. For acknowledgement and RTS/CTS signals, transmission can start after the Short Interframe Space (SIFS) [11] has elapsed. Similarly, there is another timer called DCF Interframe Space (DIFS) which is the minimum idle time for contention-based transmissions. If an STA is free for a period which is longer than DIFS, it may immediately access the wireless medium. The basic access mechanism is demonstrated in Fig. 1.3 where two STAs (STA A and STA B) contend for accessing the channel to transmit their respective data.

In the next section, we discuss the existing architectures and interworking aspects of an LTE-WiFi HetNet. Subsequently, we describe the prevalent issues in RAT selection or user association in an LTE-WiFi HetNet. This will set the background for the rest of the chapters in the thesis where we address the RAT selection problem in an LTE-WiFi HetNet. Note that throughout the thesis, the terminologies "RAT selection" and "association" are used interchangeably.

1.3 LTE-WiFi Heterogeneous Networks

In an LTE based HetNet, apart from LTE macro BSs, LTE small cells like picocells and femtocells may be present. Moreover, as shown in Fig. 1.4, in an LTE-WiFi HetNet, WiFi APs may be deployed in hotspot regions to support high data rate. LTE BSs and WiFi APs are rolled out with different objectives in mind. While the cellular network deployment is done primarily to handle mobility of User Equipments (UEs) and provide QoS guarantees to users, WiFi APs are deployed with an objective of providing high data rate to a set of static users without any QoS guarantee. Most of the modern smartphones have a dedicated WiFi interface which motivates the operators to focus on the interworking between LTE and WiFi. In an LTE-WiFi HetNet, based on certain conditions, data users can be offloaded from LTE to WiFi and vice versa. This offers an interesting way to achieve load balancing and is expected to improve the network performance as well as the user satisfaction.

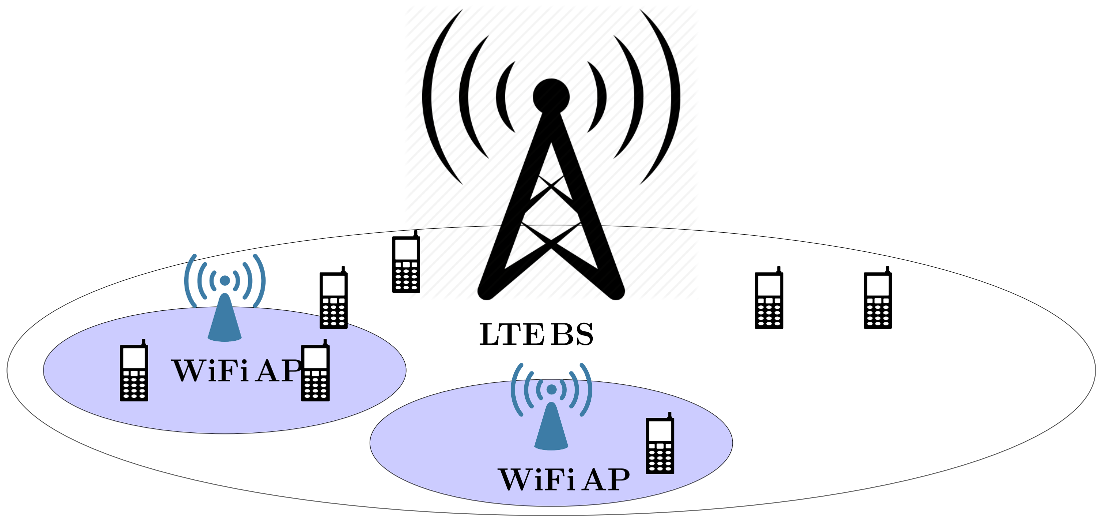


Figure 1.4: An LTE-WiFi HetNet.

1.3.1 LTE-WiFi Interworking Architecture

3GPP technologies (including LTE) and WiFi are embedded in smartphones as two separate interfaces. Specific IP flows can be routed to the WiFi path without forwarding them through the 3GPP core network. Different architectures have been proposed in the literature for the integration of WiFi and LTE networks. The Interworking WLAN (I-WLAN) architecture is defined in [8] as a part of 3GPP Release 6 [14] specifications. From the 3GPP perspective, the relationship between LTE and WiFi is coined as a trusted or a non-trusted network [15]. The 3GPP Evolved Packet Core (EPC) specifies the interworking between 3GPP and non-3GPP RATs for both trusted and non-trusted networks and supports inter-RAT handover. Interworking functionalities are defined in terms of network discovery, RAT selection, traffic prioritization, user authentication, roaming capabilities, QoS and seamless handover.

To facilitate the interworking between 3GPP and non-3GPP RATs, 3GPP proposes an optional entity in EPC known as Access Network Discovery and Selection Function (ANDSF). The main objective of ANDSF is to provide the best-connected services to the mobile users. Based on the user preferences and the operator policies, ANDSF provides the best RAT to the mobile users. ANDSF is always available to the mobile users irrespective of which RAT it is associated with, provided the RATs are interconnected through the EPC. While in roaming state, a user can access the ANDSF server of its home operator and the ANDSF server of the visited network. If any conflict occurs, ANDSF server of the visited network takes the precedence. The notions of visiting and home Public Land Mobile Network (PLMN) are applicable only to cellular networks. Home PLMN is a PLMN where the Mobile Country Code (MCC) and the Mobile Network Code (MNC) of the PLMN identity match with the MCC and MNC of the International Mobile Subscriber Identity (IMSI). A PLMN which is different from the home PLMN is called a visited PLMN. The IP address of the ANDSF server can either be provided to the users by the home operator or can be discovered by Domain Name System (DNS) or Dynamic Host Configuration Protocol (DHCP) query.

In 3GPP Release 10, specifications are provided to support simultaneous connectivity of UEs to 3GPP and non-3GPP accesses. This can be supported by one of the following protocols:

1. **Multi-Access Packet Data Network (PDN) Connectivity (MAPCON)**: MAPCON provides the ability to possess two different Access Point Names (APNs) on cellular and non-3GPP accesses.

2. **IP Flow Mobility (IFOM)**: IFOM enables a user to select an appropriate RAT on a per-IP flow basis and move seamlessly between the RATs.

3. **Non-seamless Offload**: Non-seamless offload also allows us to select the RAT on a per-IP flow basis. However, the RAT selection is done under the constraint that the non-3GPP access does not go through the 3GPP EPC. Therefore, session continuity is not supported.

Among all these approaches, IFOM is the most preferred approach as seamless offloading can be supported without any service discontinuity.

1.3.2 Enhancements in WiFi

As a part of WiFi offloading, some enhancements have been made in WiFi as well to integrate the WiFi access network in the mobile core network. IEEE 802.11u [16] is an extension to the existing 802.11 standard which specifies the protocols for the interworking with other RATs. It defines the functionalities required for the support of network discovery and selection by UEs, information transfer between different RATs and mechanisms for the provision of emergency services. To this end, IEEE 802.11u provides several extensions to the legacy MAC layer.

Apart from 802.11u, IEEE 802.11i is another set of standards for establishing a secure and two-way authenticated channel between an STA and an AP before establishing the actual connection. Support of WiFi Protected Access 2 (WPA2) is mandatory in devices to enable this functionality.

WiFi Alliance and Wireless Broadband Alliance (WBA) introduce Hotspot 2.0 as a WiFi functionality enhancement protocol. Hotspot 2.0 [17] is an enhancement to IEEE 802.11u. Hotspot 2.0 enables the STAs to automatically discover and connect to APs having roaming agreement with the user's home network.

Existing efforts towards the interworking between LTE and WiFi reveal that LTE-WiFi interworking is a promising solution for handling the rapid growth of traffic in today's network. However, for an efficient interworking, it is necessary that both these networks are controlled and managed in an intelligent way so that the optimal performance can be obtained. The arrival of a new user in the LTE-WiFi system triggers the need for RAT selection. Also, the active users in different RATs may need to get offloaded to other RATs based on different criteria. Therefore, efficient RAT selection and offloading algorithms are required to improve the overall network performance.

1.4 Challenges in RAT Selection in HetNet

In a HetNet, a large number of different RATs can coexist. In such a setting, the choice of an appropriate RAT for the association of incoming users is an important issue. The selection of RAT plays an important role in improving different system-wide objective functions such as throughput, delay. RAT selection decisions can be taken based on the availability of resources and the QoS requirements of users. The key idea is to ensure that all users are associated with the appropriate RAT so that the desired QoS can be met. A RAT selection algorithm which ensures this, may need to take into account various factors such as loading conditions of the RATs, signal strengths received from individual RATs. In existing networks, RAT selection decisions are taken by individual network elements based on simple criteria such as received signal strength at the user. However, when we consider a HetNet where WiFi APs are overlaid on LTE BSs, the existing association scheme may not perform well. This is because the existing association scheme dictates that whenever a user is in the vicinity of a WiFi AP, it always gets associated with the WiFi AP. This may cause serious overloading at the WiFi AP, causing a degradation in the overall network performance. Therefore, while choosing association decisions in an LTE-WiFi HetNet, one needs to carefully consider various network parameters of both the RATs so that improved system performance can be obtained.

RAT selection decisions can be taken either at the user side [18–24] or at the network side [25–39]. In user-initiated RAT selection schemes, there is no cooperation between LTE and WiFi networks, and users decide which RAT should be selected based on certain criteria. As users individually perform RAT selection decisions, each user makes a selfish decision so as to maximize its own utility function. Hence, user-initiated RAT selection schemes may not provide the globally optimal solution. This necessitates the need of a network-initiated RAT selection algorithm which aims to optimize the overall network performance rather than optimizing individual user utilities. Furthermore, in terms of overall network performance, network-initiated approaches exhibit significant performance improvement over user-initiated solutions [40].

User-initiated solutions are essentially distributed in nature and therefore, require computing individual utilities which may have low computational burden. Since the network-initiated solutions need to take into account network parameters of various RATs, they may be computationally inefficient compared to user-initiated solutions. Moreover, centralized computations in network-initiated solutions may require the exchange of RAT-specific information with a centralized entity. This requires additional signaling exchanges between the LTE BS/WiFi AP and the centralized entity, thereby incurring an additional delay in RAT selection. On the other hand, since in distributed RAT selection mechanisms each user takes their individual decisions, information exchange across various RATs may not be required. However, unlike user-initiated approaches, network-initiated approaches for RAT selection provide a way for the overall system optimization. Therefore, in this thesis, we target to investigate different network-initiated RAT selection techniques for the optimization of system-wide QoS metrics like total system throughput, blocking probability.

For an efficient interworking between LTE and WiFi, it may be necessary to develop a unified framework which can control and manage these RATs together. With the upcoming 5G [6,7] network, future wireless networks are expected to be a mixture of a variety of RATs. Therefore, the need for a unified framework becomes even more pronounced. Presence of a unified framework allows us to possess a global view of different RATs. In the absence of a global view, as observed in today's network, the utilization of network resources becomes suboptimal. However, optimal performance can be achieved if the RAT selection functionality is controlled and managed within a unified framework. The concept of Software Defined Networking (SDN) [41–43] may be instrumental in achieving the unified control and management of various RATs. The basic idea behind SDN is the split of control and data plane elements and functionalities in a network. Using SDN, the control plane functionalities can be aggregated by decoupling them from the network elements of LTE and WiFi. Since the control plane has a global view of the network, this approach allows us to design the RAT selection algorithms in an optimal manner.

1.5 Motivation for the Thesis

Having introduced the challenges associated with the RAT selection in an LTE-WiFi HetNet, we now set out to study four different aspects which may be important in devising network-initiated RAT selection strategies. We consider an LTE-WiFi network where different types of users are present. Users can be classified into various types depending on several factors such as type of application, priority.

While designing association schemes, generally we target to optimize different system metrics depending on our requirements. Total system throughput is an important system parameter from a network operator's perspective since the throughput experienced by the data users may have a significant impact on the profit and customer base of the operator. Therefore, we consider the maximization of the total system throughput as the optimization goal. Existing literature on user association focuses on devising RAT selection strategies which specify which RAT should an incoming user be associated with. However, in practice, the system may be dynamic in nature. In other words, users may arrive and depart from the system at any point in time. While associating a user to a RAT, one may choose to offload an existing user from the RAT to another RAT. This may result in a better system performance than that of the association of the user without any consideration of offloading. Similarly, after the initial association, if a user departs from the system, it is necessary to investigate whether offloading of an existing user can result in a better system utilization.

The second aspect which influences the RAT selection strategy while optimizing the total system throughput is related to the blocking probability of users. Consider a system where voice and data users are present. Voice users generally require QoS guarantees which can be obtained by associating them with LTE. When we maximize the total system throughput, data users are preferably admitted in the system because usually their contribution in terms of throughput is more than that of voice users. Since voice and data users share resources in LTE, this may result in excessive blocking of voice users. This causes an increase in the blocking probability of voice users. Thus, when we aim to maximize the total system throughput, it is important to consider a constraint on the voice user blocking probability. We address this trade-off while designing association algorithms in an LTE-WiFi HetNet.

The third aspect that affects a RAT selection decision is the channel condition of the users which is an indicator of the quality of signals received from the LTE BS/ WiFi AP. The channel conditions of different users in a HetNet may be different. The number of resource blocks consumed by a user is a function of the channel condition of the user. If there is an arrival or a departure of user, then channel condition of the user may also play a role in the choice of a user for offloading. For example, if a user departs from LTE and we decide to offload a user from WiFi to LTE, it may be better to choose a user with 'good' channel condition in LTE rather than choosing one with 'bad' channel.

The fourth aspect which is important from the RAT selection perspective while aiming to maximize the total system throughput subject to a blocking probability constraint is the offloading probability of users (i.e., fraction of offloaded traffic). In the considered offloading-assisted association problem where we aim to maximize the total system throughput subject to a blocking probability constraint, it may happen that a user is offloaded repeatedly from one RAT to another within a short time interval. This behavior is undesirable as this may cause a lot of control signaling traffic in the *backhaul*, resulting in a congestion in the core network. Therefore, it is necessary to consider a constraint on the offloading probability of users so that the amount of backhaul traffic can be restricted. In this thesis, apart from the tradeoff between the total system throughput and the blocking probability, we also address the trade-off involving the total system throughput, the blocking probability of users and the offloading probability of users.

These problems can be formulated as control problems where we aim to obtain the optimal association strategy for users. A well-known approach of solving such problems is formulating the problem as a Markov Decision Process (MDP) and then solve them using Dynamic Programming (DP) methods. However, this approach is associated with the *curse of dimensionality*, i.e., the computation of the optimal policy becomes intractable, when the state space is large. It so turns out that, in our problem, the optimal policy possesses a provably threshold structure. To be precise, the optimal policy dictates that one RAT selection strategy is preferred over all other strategies upto a certain threshold on some system parameters. Beyond the threshold, some other strategy is preferred. Inspired by the existence of the threshold structure, we design computationally efficient and storage-efficient RAT selection algorithms.

There is another issue which is prevalent in DP based methods of computing the optimal policy. These methods require the knowledge of the transition probabilities associated with the underlying Markov chain in order to compute the optimal policy. The knowledge of the transition probabilities in turn requires the knowledge of the system model, i.e., statistical properties of the arrival processes of users, which may be difficult to obtain in reality. This is known as the *curse of modeling*. To overcome this difficulty, suitable assumptions on the arrival processes can be made. However, the performance of the modeled system may be far from optimality and is governed by the accuracy of modeling. This provides us the motivation to design RAT selection algorithms that do not need the knowledge of the statistics of the arrival processes of users. To this end, we design online RAT selection schemes which converge to the optimality without requiring the knowledge of the statistics of the arrival rates of users. This is achieved by using Reinforcement Learning (RL) and Stochastic Approximation (SA) frameworks.

Furthermore, if the knowledge of the threshold based optimal policy is incorporated in the RL framework, it may result in faster convergence to the optimal policy than traditional learning algorithms which do not possess the knowledge of the threshold structures. We propose a computationally efficient online RAT selection algorithm which exploits the knowledge regarding the existence of threshold based optimal policy while learning.

The RAT selection solutions proposed in this thesis are suitable for implementation within an SDN framework. We implement low complexity RAT selection algorithms within a practical SDN based evaluation platform developed using Network Simulator-3 (ns-3) [44].

Note that although the RAT selection solutions in this thesis are developed for an LTE-WiFi HetNet, the framework considered by us is generic and can be extended easily to include other RATs as well.

1.6 Contributions and Organization of the Thesis

In this section, we highlight some of the contributions of the thesis. We organize the thesis into eight chapters. In Chapter 2, we present a review of related literature. Chapters 3-7 describe our contributions in this thesis. The chapter-wise contributions are described below.

- In Chapter 3, we consider the optimal association problem in an LTE-WiFi HetNet comprising voice and data users. We determine the optimal RAT selection policy which targets to maximize the total throughput of the system. We also determine the optimal policy which maximizes the total throughput of the system subject to a constraint on the voice user blocking probability. Offloading of data users can be performed at the time of association and departure of users. While there are numerous approaches for RAT selection in a HetNet, no other work in the literature has considered the trade-off between the total system throughput and the blocking probability of voice users for an LTE-WiFi HetNet within an optimization framework. The considered problems are formulated as an MDP and a Constrained Markov Decision Process (CMDP) problem, respectively. The threshold structure of the optimal policy is derived analytically. We propose RAT selection algorithms based on the derived optimality of threshold policies. We demonstrate through analysis that the proposed algorithms provide significant reductions in computational and storage complexities compared to traditional DP methods. We conduct ns-3 simulations to compare the performance of the proposed algorithms with those of existing RAT selection algorithms in the literature.

- In Chapter 4, we consider an extension of the RAT selection framework developed in Chapter 3. Although the algorithms proposed in Chapter 3 provide reductions in computational and storage complexities compared to traditional algorithms, they are not free from the curse of modeling. To address this issue, in Chapter 4, we propose two online learning algorithms for RAT selection which iteratively compute the optimal policy. The proposed algorithms can be implemented without any explicit knowledge of the statistics of the arrival processes of voice and data users, utilizing the framework of SA. The first RAT selection algorithm is based on the traditional Q-learning based approach [45]. The proposed approach is based on an online implementation of Relative Value Iteration Algorithm (RVIA) which is a well-known algorithm for solving MDP problems. The second RAT selection algorithm is based on the reformulation of RVIA by introducing the notion of a virtual state called post-decision state. We prove that the proposed algorithm converges asymptotically to the optimal policy. Moreover, the proposed algorithm provides faster convergence and lower space complexity than Q-learning algorithm. ns-3 simulation results are presented to demonstrate the convergence behaviors of both the algorithms.

- Convergence speeds of the algorithms proposed in Chapter 4 can be further enhanced if the underlying threshold structure (as investigated in Chapter 3) of the optimal policy can be exploited. In Chapter 5, we propose a novel structure-aware learning algorithm for RAT selection. Although there are existing learning approaches for RAT selection in the literature, the idea of exploitation of the threshold structure in the learning framework has not been considered in the literature before. The proposed algorithm considers the threshold as a parameter and updates it iteratively in the RL framework. We prove that the proposed algorithm converges to the globally optimal threshold policy. The structure-aware learning algorithm provides lower computational and space complexities than those of the learning algorithms proposed in Chapter 4. Simulation results indicate that the proposed algorithm converges to a near-optimal solution after a reasonable number of iterations. Moreover, simulations under realistic scenarios demonstrate that the proposed algorithm outperforms state-of-the-art RAT selection algorithms.

- In Chapter 6, we consider an extension of the model considered in Chapters 3, 4 and 5. We consider an LTE-WiFi system where users of multiple priorities are present. Unlike the previous chapters, while maximizing the total system throughput subject to a blocking probability constraint, we consider another constraint on the offloading probability of low priority users. In Chapter 6, we also investigate the role of the channel states of users in RAT selection. The considered problem is modeled as a CMDP problem. We reduce the dimensionality of the action space by proving the suboptimality of different actions in various states. To address the curse of dimensionality and the curse of modeling, we propose two low-complexity online RAT selection algorithms. Although the proposed algorithms are greedy in nature, they satisfy the constraints associated with the problem. We also provide a comparison of storage and computational complexities of the association algorithms.

- In Chapter 7, we implement the association algorithms (proposed in Chapter 6) on an SDN-based evaluation platform. To conduct simulations, we propose an LTE-WiFi network architecture based on the paradigm of SDN. In the proposed architecture, the radio resource management related decisions are unified in an SDN controller. We develop the SDN-based evaluation platform according to the proposed LTE-WiFi network architecture by restructuring the existing modules of ns-3. The proposed algorithms are implemented on the platform. Simulation results demonstrate that the performances of the algorithms are close to optimal. We also propose modifications to the proposed algorithms so that they can function well in the presence of user mobility. Performances of the proposed algorithms are compared with other state-of-the-art algorithms under realistic network scenarios.

We conclude along with possible directions for future work in Chapter 8. We include three appendices in the thesis. In the interest of preserving the flow of the thesis, some of the proofs are presented in the first appendix. The second appendix contains a generalized framework of the structure-aware learning algorithm proposed in Chapter 5 which can be applied to any general MDP problem where the optimal policy has a threshold structure. To make the thesis self-contained, the third appendix presents existing approaches in the literature on Markov Decision Process, Reinforcement Learning and Stochastic Approximation.

Chapter 2

RAT Selection in HetNets: Perspectives and Open Research Challenges

As pointed out in the previous chapter, traditional RAT selection techniques, as present in today's network, do not perform well in HetNets. Therefore, one requires to carefully investigate how RAT-specific characteristics can be exploited to devise efficient RAT selection techniques in HetNets. In this chapter, we present some existing efforts in the literature towards RAT selection in HetNets. As discussed in the previous chapter, offloading of data users from LTE to WiFi may play a major role in improving the network performance, and it is important to exploit the offloading opportunities which arise due to arrivals and departures of users. Accordingly, we review various LTE-WiFi offloading solutions proposed in the literature. In this thesis, we focus on network-initiated RAT selection and offloading solutions as they provide a framework for system-wide optimization due to the presence of a global view of the network. With the emergence of the SDN paradigm, unification of control and management procedures of various RATs which offers a global view of the entire network, is facilitated. Hence, in this chapter, we also present state-of-the-art techniques in RAT selection and offloading in SDN-based LTE-WiFi networks. Towards the end of the chapter, we discuss some key open issues in the literature which form the motivation for our investigation in rest of the thesis.

2.1 RAT Selection in Heterogeneous Networks

Existing RAT selection solutions in HetNets can be broadly classified into two categories, viz., user-initiated [18–24, 46, 47] and network-initiated [25–39, 48] solutions. While a subset of these works focus on heuristic based approaches, rest of the works focus on optimization based approaches aiming to optimize different system metrics such as resource utilization, generated revenue and QoSs of users.

2.1.1 User-initiated RAT Selection

Existing user-initiated RAT selection schemes take into account various factors such as Signal-to-Noise Ratio (SNR), load information and resource utilization. In [24], a user-initiated RAT selection algorithm based on SNR and loading information of individual RATs with the adaptation of hysteresis mechanism, is considered for an LTE-WiFi HetNet. The performance of this scheme is compared against network-initiated optimized cell-range extension schemes [5] to steer users to other RATs. In [19], a distributed RAT selection algorithm is proposed for a hybrid WLAN based on the distance and the peak rate obtained from different APs. The problem where each user selfishly chooses the RAT with an objective of maximizing the individual throughput, is considered in [20] and formulated as a non-cooperative game. In [18], the authors propose a user-initiated RAT selection algorithm in an LTE-WiFi HetNet. The proposed algorithm addresses the trade-off between resource utilization and QoS. The authors in [46] propose a distributed association scheme which maximizes the network utility subject to constraints on user requirements. The proposed scheme is based on the utility obtained from past associations of the user. In [47], a low complexity RAT selection algorithm is proposed for an LTE network comprising macro, pico and femto cells. The proposed algorithm achieves a near-optimal performance with a theoretical guarantee on the performance. Moreover, the authors extend their proposed scheme to the cell range extension technique so that it can be implemented without much changes in the existing system.

In user-initiated schemes, since each user makes an individual selfish decision to maximize its own utility function, these approaches may not result in a globally optimum solution. Therefore, there is a need for a unified framework having a global view of both

LTE and WiFi networks so that RAT selection decisions can be taken by considering the parameters of both RATs. In a network-initiated approach, instead of optimizing individual utilities, total system performance can be optimized based on operators' requirements.

2.1.2 Network-initiated RAT Selection

In this subsection, we review some representative network-initiated RAT selection approaches. Few heuristic-based RAT selection schemes are considered in [28–31]. While the algorithm proposed in [30] prefers WiFi over cellular network regardless of the service type, the one proposed in [31] prefers cellular RAT for voice users and WiFi for data users. In [28], a simple admission control scheme is described inside a cellular-WiFi system to improve the support for voice and data services. The objective is to study how the resource utilization varies with the admission parameters and the impact of user mobility and traffic properties on admission parameters. The authors in [28] devise schemes where admission and rate control strategies are applied to limit the WiFi traffic, so that WiFi can work in the most efficient manner.

There are several network-initiated RAT selection schemes [25, 26, 32–39, 49] which follow the optimization approach. In [38], the optimal RAT selection problem is addressed in a Global System for Mobile Communications (GSM)- Universal Mobile Telecommunications System (UMTS) HetNet with aims of optimizing different objective functions like throughput, blocking probability using the formalism of MDP. Since the associated policy iteration algorithm to solve the problem scales exponentially as the system size, the authors propose a computationally efficient heuristic algorithm which prefers UMTS for voice users and GSM for data users. In [39], the authors consider the sum of logarithms of throughputs obtained from different associations, and the association which results in maximum value for this sum is chosen among STAs and APs. However, the authors do not take into account user arrivals and departures. In [32], an optimal client-AP association problem in a WLAN is considered within the framework of continuous time MDP, and user arrival/departure is considered as a feasible decision epoch. Another RAT selection problem in an LTE-WiFi HetNet where two user profiles are present, is considered in [34]. LTE is always preferred by the high-priority users. Only a portion of LTE resources can be shared with the low-priority users. The key objective is to maximize the generated revenue from an operator's perspective to obtain the optimal partitioning between the dedicated and the shared resources. However, the proposed algorithm scales exponentially with the system size. The authors in [25] propose distributed association algorithms by formulating the association problem as a non-cooperative game and compare the performance with the centralized globally optimal scheme. A context-aware RAT selection algorithm is proposed in [26]. The proposed algorithm, which can be implemented on the user side, albeit with network assistance, minimizes the signaling overhead as well base station computations. A joint user association and interference management problem in a two-tier HetNet is considered in [49]. The authors propose a computationally efficient method to maximize the network utility under proportional fairness. However, this approach is not adaptable to quick changes in network dynamics.

While the above solutions provide significant insight into RAT selection strategies, only a few of them focus on computational efficiency. Therefore, practical implementations of the proposed algorithms become infeasible. As discussed in Chapter 1, the curse of dimensionality and the curse of modeling are the two main driving factors behind this issue. In the first case, standard dynamic programming techniques to solve optimal RAT selection and offloading problems become computationally inefficient and thus difficult to implement when the state space is large. Although a few works propose computationally efficient heuristics for RAT selection, the performances achieved by them are usually suboptimal. Due to the curse of modeling, we need to know the statistics of the arrival processes of users which govern the transition probabilities between different states, in order to determine the optimal policy. In practice, this may be difficult to obtain. Recent studies [50–52] on the characteristics of cellular traffic reveal that although the voice traffic can be predicted accurately, state-of-the-art prediction schemes for data traffic are not very satisfactory. In the absence of knowledge regarding the statistics of system dynamics, RL [35, 53, 54] based schemes, which can learn the optimal policy by trial-and-error and can be implemented online, are adopted in the literature. The authors in [35] aim to improve the network performance and the user experience jointly and formulate the RAT selection problem as a Semi-Markov Decision Process (SMDP) problem. However, a Q-learning [45] based approach is also adopted since the network parameters may be difficult to obtain in reality. Although the asymptotic convergence to optimality is guaranteed, the scheme possesses a large memory requirement. Additionally, due to the *exploration* [45] mechanism, the convergence rate is slow, especially under large state spaces.

2.2 LTE-WiFi Offloading

As a part of interworking between LTE and WiFi, offloading of data users [55,56] from one RAT to another plays a vital role in the capacity improvement of the system. Due to the high availability of WiFi APs in shopping malls, residential buildings, offices etc., they are well-suited for offloading of high data rate services with limited mobility. Furthermore, as modern smartphones, laptops and tablets support WiFi features, offloading of mobile data traffic to WiFi APs has become a prominent and cost-effective solution to alleviate the heavily overloaded cellular network. Different LTE-WiFi offloading solutions [21–23, 27, 53, 57–63] have been proposed in the literature. Similar to the case of RAT selection, offloading decisions can be user-initiated or network-initiated in nature.

2.2.1 User-initiated Offloading Techniques

Based on the time of offloading, user-initiated offloading can be divided into two types, viz., *on-the-spot offloading* and *delayed offloading*. Performance improvement achieved by on-the-spot offloading [57] is analyzed in [21]. The basic idea behind on-the-spot offloading is to steer the data users to WiFi whenever WiFi network is available. On the contrary, delayed offloading [21] waits for a certain time interval before using the cellular network, once the user is out of WiFi AP coverage. This is done to avoid the "ping-pong" effect, where one data user repeatedly goes out and comes inside the coverage area of WiFi APs within a short span of time. In [22], offloading efficiencies, i.e., the ratios of the offloaded traffic to the total traffic, of on-the-spot and delayed offloading are derived. In [59], the authors consider the trade-off between the cost and the delay involved in opportunistic WiFi offloading and propose an adaptive strategy to maximize the user satisfaction. In [23], opportunistic offloading is considered in the context of vehicular users. The authors propose two game-based WiFi offloading schemes by predicting the offloading opportunities and the access cost in the proposed framework.

In [58], the offloading decision is user-initiated and based on the combined information of signal strength and network load of LTE/WiFi. However, this algorithm, being a greedy one, fails to converge to a globally optimum solution. Based on the locally available information at the users and following a Q-learning approach, [53] undertakes distributed traffic offloading decisions. Although this work addresses the curse of modeling associated with the DP based approaches, the convergence behavior of the proposed algorithm is slow. The work in [60] considers RAT selection and data offloading problem in an LTE-WiFi HetNet by incorporating user mobility, WiFi availability and various pricing mechanisms in the model. The considered problem which aims at providing congestion-aware network selection, is formulated as a non-cooperative game.

2.2.2 Network-initiated Offloading Techniques

In network-initiated offloading techniques, decisions concerning when to offload and which user(s) to offload, is taken at the network side. In [61], the authors propose a network-initiated LTE-WiFi solution where data users are offloaded from LTE to WiFi when the capacity requirement becomes more than the capacity of LTE. The proposed algorithm uses a combination of load and SNR information to distribute the traffic between LTE and WiFi. However, the proposed algorithm may not perform well when the traffic load in the LTE network is of the order of the load in WiFi APs. The work in [62] considers a generalized framework where the users in LTE can be offloaded to a small cell or a WiFi AP. The authors propose offloading algorithms that achieve an appropriate balance between the revenue obtained by serving the users and the fairness among the users. In [63], the authors investigate the effectiveness of ANDSF server which assists in data offloading in a 3GPP framework. The authors consider a scenario where the users follow a policy and then report the observed performance to the cellular operator. The cellular operator modifies the policy based on the report. The authors propose a practical offloading scheme and compare the performance of the scheme with those of other state-of-the-art algorithms. The network-initiated offloading approach described in [27] computes the optimal fraction of traffic to be offloaded to WiFi to maximize the per-user throughput of the system. The proposed offloading scheme in [27] performs better than the on-the-spot offloading [57] scheme. However, this offloading approach does not take into account a dynamic system where users arrive and depart at any point in time. Additionally, this model does not incorporate voice users inside an LTE network. In the presence of voice users, blocking probability of voice users may play an important role in choosing offloading decisions. We have addressed these limitations in our model in Chapter 3.

Contrary to user-initiated offloading solutions where each user aims to maximize its own utility, in network-based offloading mechanisms, overall network performance can be optimized. In [40], the interworking between LTE and WiFi is considered in an LTE-WiFi system. The authors consider user-centric and network-centric proportionally fair resource allocation strategies. The analysis presented in the paper reveals that the network-centric approach provides a better throughput than that of the user-initiated approach. However, the network-initiated algorithm may be computationally more complex than the user-initiated algorithm as the former needs to consider the parameters of various RATs while taking decisions. Moreover, the network-initiated scheme may incur additional delay because of the requirement of exchanging the RAT-specific information with a centralized entity. On the contrary, the user-initiated scheme is distributed in nature, and hence, no information exchange is needed between the RATs.

2.3 SDN-based RAT Selection and Offloading

A unified framework can facilitate the exchange of RAT-specific information with a centralized entity, thereby enabling network-initiated decisions based on the global view of various RATs. With the 5G [6, 7] standardization in progress, it is expected that the future wireless networks will contain a large number of RATs. Therefore, it is important to define a unified framework which can control and manage these multiple RATs together.

Since the SDN paradigm facilitates the split of control and data planes in a network, the control plane functionalities of different RATs can be unified in a centralized controller. RAT selection and offloading problems in SDN-based HetNets are proposed in recent literature [64–68]. The authors in [66] consider an SDN-enabled dynamic path selection problem in a multi-RAT system and propose an algorithm which chooses the path based on the rate obtained. The obtained rate takes into account various factors such as radio conditions of UEs, performance requirements of different flows and loading conditions of

RATs. In [67], the authors propose a QoS-aware RAT selection algorithm in an SDN-based HetNet. The authors in [67] perform RAT selection based on a metric which takes into account bit rate requirements of users and capabilities of different RATs. A user association heuristic which considers multiple traffic classes and scales well with the LTE/WiFi HetNet system, is proposed in [68]. The performance of the proposed algorithm is evaluated in an SDN-based testbed.

2.4 Discussions and Open Problems

In this chapter, we have provided a description of existing RAT selection techniques in HetNets. Furthermore, we have reviewed the existing efforts towards efficient LTE-WiFi offloading, followed by a literature review of initiatives towards RAT selection and offloading in SDN-based LTE-WiFi HetNets. Both user-initiated and network-initiated approaches have been adopted in the literature. Since network-initiated approaches provide a framework for overall system optimization, we focus on network-initiated approaches for RAT selection and offloading in an LTE-WiFi network.

Various network-initiated approaches in the literature focus on optimizing different system metrics. However, as discussed in Chapter 1, the trade-off between the total system throughput and the blocking probability of voice users in a dynamic system within an optimization framework has not been considered in the literature before. From the perspective of total system throughput maximization, admission of data users are more preferable as the throughput contribution of voice users is usually lower than that of data users. This may increase the blocking probability of voice users since voice and data users are allocated resources in LTE from a common resource pool. We address the resulting trade-off between the total system throughput and the blocking probability of voice users in this thesis. Furthermore, unlike others, we investigate the role of offloading in improving the system performance at the time of arrivals and departures of users in a dynamic LTE-WiFi system. We address the trade-off between the total system throughput and the blocking probability of voice users in an offload-assisted RAT selection problem in Chapter 3 within an MDP based framework. However, due to the curse of dimensionality associated with DP based methods to solve the problem, the optimal solution is difficult to compute when the state space is large. We shall demonstrate in Chapter 3 that the optimal RAT selection strategy obtained by us contains a threshold structure. By virtue of the threshold structure, we propose optimal RAT selection algorithms which bring significant reductions in the computational and storage complexities in comparison to traditional DP based methods.

The curse of modeling is another drawback associated with the DP based methods of computing the optimal solution. DP based methods require the knowledge of transition probabilities between different states associated with the underlying Markov chain which is governed by the statistics of the system dynamics. In practice, these may be difficult to obtain. Few works in the literature have proposed RL based RAT selection/ offloading schemes which can work without the knowledge of the statistics of the system dynamics. However, the convergence behaviors of traditional RL algorithms such as Q-learning [45] are slow which renders them difficult to implement in practice. In Chapter 5, we shall propose an RL algorithm which exploits the knowledge of the threshold structure of the optimal solution in the learning framework. It provides significant reductions in the computational and storage complexities as well as the convergence speed in comparison to traditional RL schemes. To the best of our knowledge, this is one of the first attempts of exploiting the knowledge of the threshold structure in the learning framework.

Another aspect which we focus in this thesis is the role of channel conditions of the users in RAT selection. As discussed in Chapter 1, channel conditions of users may play a major role in association decisions since users with different channel conditions may consume different number of resource blocks in LTE. Moreover, in the case of arrival or departure of users, channel states of users need to be considered for the choice of an appropriate user for offloading.

Apart from the trade-off between the total system throughput and the voice user blocking probability, the trade-off involving the total system throughput, the blocking probability and the offloading probability is also very important. This is because maximizing the total system throughput subject to a blocking probability constraint may increase the offloading probability of the users. This particular problem has not been studied in the literature before. We shall formulate the problem within the framework of CMDP. Furthermore, we shall propose low-complexity online algorithms which although are greedy in nature, satisfy the associated constraints. The proposed low-complexity algorithms are implemented on an ns-3 based evaluation platform which is built following the SDN principles.

In the next chapter, we begin our analysis by considering the throughput efficient RAT selection problem in an LTE-WiFi HetNet. Furthermore, we incorporate the trade-off between the total system throughput and the voice user blocking probability in the considered framework.

Chapter 3

Throughput Efficient RAT Selection

In this chapter, we investigate an optimal association policy for an LTE-WiFi HetNet. The objective is to obtain a RAT selection policy which maximizes the total system throughput. However, as described in Chapter 2, there exists a trade-off between the total system throughput and the blocking probability of voice users. Higher layers such as network and MAC layer may impose certain restrictions on the blocking probability of voice users due to QoS requirements. This motivates us to subsequently determine a RAT selection policy which maximizes the total system throughput subject to a constraint on the blocking probability of voice users. We formulate these problems as optimization problems within Markov Decision Process (MDP) and Constrained Markov Decision Process (CMDP) frameworks, respectively. Value Iteration Algorithm (VIA) and gradient descent are among the algorithms that are used traditionally to compute optimal policies for MDP and CMDP formulations. However, such techniques are computationally prohibitive due to the high computational complexity associated with VIA under large state spaces. This motivates looking for structural properties that reduce the search space for the optimal policy. Towards this end, we establish that the optimal policy has a threshold structure. It is proved that the optimal policies admit a threshold structure, where after a certain threshold on the number of WiFi data users, data users are served using LTE. The existence of a similar threshold for the blocking of voice users is also established. As a policy search over the entire policy space may become computationally inefficient, we propose algorithms that exploit the threshold structure for determining the optimal RAT selection policy. Our analysis reveals that the proposed algorithms provide significant improvements in computational and storage complexities over the traditional policy iteration algorithm [69]. We conduct extensive simulations in ns-3 to compare the performances of the proposed algorithms with existing RAT selection algorithms proposed in [57]. 3GPP recommended parameters are used in the simulations. Simulation results indicate that the algorithms proposed by us perform better than existing schemes in improving different system metrics.

The rest of the chapter is organized as follows. The system model is described in Section 3.1. In Section 3.2, the RAT selection problems are formulated within the frameworks of unconstrained and constrained continuous-time MDP, respectively. In Section 3.3, we derive the threshold structure of the optimal policy. Algorithms for the association of users in LTE-WiFi HetNet are proposed in Section 3.4. Section 3.5 presents simulation results. We conclude in Section 3.6.

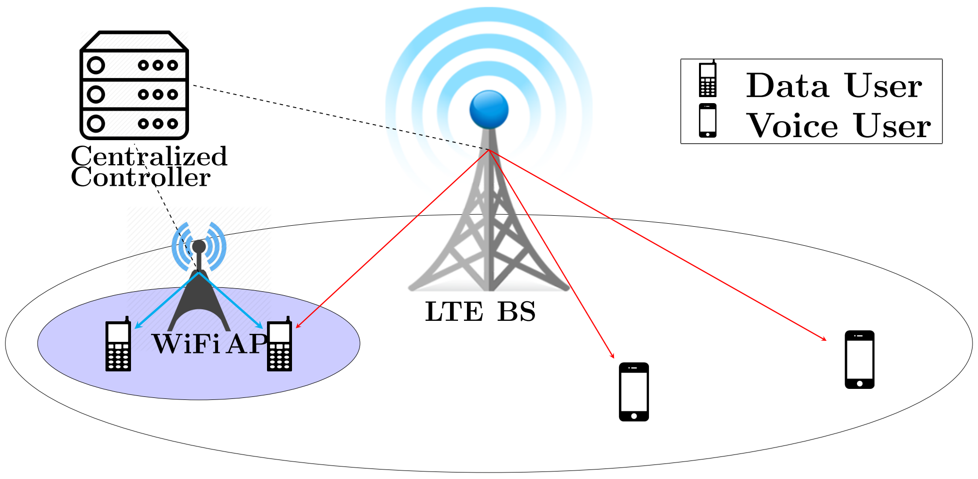


Figure 3.1: LTE-WiFi heterogeneous network architecture.

3.1 System Model

We consider a system where an LTE BS and a WiFi AP are present. As illustrated in Fig. 3.1, we assume that both the BS and the AP are connected to a centralized controller by high capacity lossless links. RAT selection and offloading decisions are taken by the centralized controller, possessing an overall view of the network. To be precise, the centralized controller has a knowledge of the load conditions of LTE and WiFi networks.

Moreover, whenever a user arrives in the system or departs from the system, the controller is notified by the LTE BS or the WiFi AP.

We assume that the voice and data users are geographically located at any point in the LTE BS coverage area. Data users outside the dual coverage area of the LTE BS and the WiFi AP always get associated with the LTE BS, and no decision is involved in this case. Therefore, without loss of generality, we take into consideration only those data users which are present inside the WiFi AP coverage area. We assume that there is a common resource pool in LTE for the voice users as well as the data users inside the WiFi AP coverage area. Data users inside the dual coverage area can be associated with the LTE BS or the WiFi AP. All users are assumed to be stationary. Voice and data user arrivals follow independent Poisson processes with means λ_v and λ_d, respectively. Service times for voice users are independent and identically distributed. Similarly, Service times for data users are independent and identically distributed. Moreover, service times for voice and data users are independent. We assume that service times for voice and data users are exponentially distributed with means $\frac{1}{\mu_v}$ and $\frac{1}{\mu_d}$, respectively. For justification behind these assumptions, see [70].

Remark 1. *Although for the brevity of notation, a single LTE BS and a single AP is considered, the system model can be generalized to a single LTE BS and multiple APs with non-overlapping coverage areas by considering the number of data users in different APs in the system model. Moreover, considering that each point in a geographical area is mapped to a single LTE BS (the LTE BS with highest average signal strength, say), multiple BSs can also be included in the system model with slight modifications. In the case of multiple overlapping APs inside the coverage area of an LTE BS, a one-to-one mapping between a user location and an AP using a decision criterion (based on highest average signal strength, say) reduces the problem to a single BS-multiple non-overlapping AP problem. The set where more than one AP have equal signal strength is non-generic in the sense that in the associated parameter space it has Lebesgue measure* 0.

Remark 2. *Most of the cellular network operators support seamless offloading of data users to trusted (operator-deployed) WiFi APs. The decision regarding when to offload is at the discretion of the service providers. We build our system model based on the architecture provided by the cellular operators for interworking with WiFi [8].*

3.1.1 State Space

We model the system as a controlled continuous time stochastic process $\{X(t)\}_{t\geq 0}$ defined on a state space $\mathcal{S}$. Any state $s \in \mathcal{S}$ is represented as a 3-tuple $s = (i, j, k)$, where i, j and k represent the number of voice users in LTE, the number of data users in LTE and the number of data users in WiFi, respectively. The system state remains unchanged unless an existing user departs or a new user arrives in the system. The arrivals and departures in the system are referred to as events. Let the set of all events be denoted by $\mathcal{E}$. The list of events in $\mathcal{E}$ are as follows:

$$\mathcal{E} = \begin{cases} E_1, & \text{Arrival of a voice user in the system,} \\ E_2, & \text{Arrival of a data user in the system,} \\ E_3, & \text{Departure of an existing voice user from LTE,} \\ E_4, & \text{Departure of an existing data user from LTE,} \\ E_5, & \text{Departure of an existing data user from WiFi.} \end{cases}$$

Whenever an event occurs, the centralized controller takes an action, and based on the type of event and the action chosen by the controller, a state transition happens. Note that the transitions of $\{X(t)\}_{t\geq 0}$ happen only at the event epochs and not otherwise. Since the associated continuous time Markov chain has a finite number of states, the Markov chain is regular (i.e., the rate of exponentially distributed sojourn time in every state is bounded) [71]. Thus, it suffices to observe the system state only at the event epochs to know the entire sample path. A finite amount of reward and cost rates are associated with each state-action pair. Detailed descriptions of the action space, state transitions, reward and cost are provided in subsequent subsections.

Now, we elaborate on the structure of $\mathcal{S}$. We say that $(i, j, k) \in \mathcal{S}$ if $(i + j) \leq C$ and $k \leq W$, where C is the total number of common resource blocks reserved in LTE for voice and data users, and W is the maximum number of users in WiFi so that the per-user throughput in WiFi is greater than a pre-defined threshold. The condition $(i + j) \leq C$ arises because we assume that in each LTE subframe, each admitted user is allocated one resource block. If this allocation is not possible, a new user is not admitted in the LTE system. Furthermore, note that WiFi throughput decays monotonically as the number of WiFi users increases [72]. We assume that each user should get more than a threshold value of average throughput (say 2 Mbps) for a meaningful connectivity, which leads to the bound W on the maximum number of users that can be accommodated in the WiFi system.

Remark 3. *Although the allocation of multiple resource blocks is closer to the practical scenario, this complicates the system model while the methodology and approach adopted do not change.*

3.1.2 Action Space

The set of actions defines a set of possible association and offloading strategies in the event of arrival or departure of a user. Let the action space be denoted by $\mathcal{A}$. Depending on the arrival or departure, we have a set of actions as stated below.

$$\mathcal{A} = \begin{cases} A_1, & \text{Block the arriving user or do nothing during departure,} \\ A_2, & \text{Accept voice/data user in LTE,} \\ A_3, & \text{Accept data user in WiFi,} \\ A_4, & \text{Accept voice user in LTE and offload one data user to WiFi,} \\ A_5, & \text{Move one data user to a RAT (from which departure has occurred).} \end{cases}$$

Remark 4. *Actions are chosen based on the system state and the event occurred. One way of representing this is embedding the event in the state space so that the action depends only on the system state. However, to avoid notational complications associated with this approach, we view the action as a function of the system state and the event.*

Let the set of states (subset of $\mathcal{S}$) in which action a is feasible given occurrence of event E_l be denoted by $\mathcal{S}_{E_l,a}$. Thus, in the case of voice user arrival, we have,

$$\mathcal{S}_{E_1,a} = \begin{cases} \mathcal{S} \setminus \{(0,0,0)\}, & a = A_1, \\ \mathcal{S} \setminus \{(i,j,k) : (i+j) = C\}, & a = A_2, \\ \mathcal{S} \setminus \{(i,j,k) : (j=0)||(k=W)\}, & a = A_4, \\ \{\emptyset\}, & \text{else.} \end{cases}$$

For data user arrival,

$$\mathcal{S}_{E_2,a} = \begin{cases} \{(i,j,W) : (i+j) = C\}, & a = A_1, \\ \mathcal{S} \setminus \{(i,j,k) : (i+j) = C\}, & a = A_2, \\ \mathcal{S} \setminus \{(i,j,k) : k = W\}, & a = A_3, \\ \{\emptyset\}, & \text{else.} \end{cases}$$

For voice user departure from LTE,

$$\mathcal{S}_{E_3,a} = \begin{cases} \mathcal{S} \setminus \{(i,j,k) : (i+j) = C\}, & a = A_1, \\ \mathcal{S} \setminus \{(i,j,k) : ((i+j) = C) || (k = 0)\}, & a = A_5, \\ \{\emptyset\}, & \text{else.} \end{cases}$$

For data user departure from LTE,

$$\mathcal{S}_{E_4,a} = \begin{cases} \mathcal{S} \setminus \{(i,j,k) : (i+j) = C\}, & a = A_1, \\ \mathcal{S} \setminus \{(i,j,k) : ((i+j) = C) || (k = 0)\}, & a = A_5, \\ \{\emptyset\}, & \text{else.} \end{cases}$$

For data user departure from WiFi,

$$\mathcal{S}_{E_5,a} = \begin{cases} \mathcal{S} \setminus \{(i,j,k) : k = W\}, & a = A_1, \\ \mathcal{S} \setminus \{(i,j,k) : (j = 0) || (k = W)\}, & a = A_5, \\ \{\emptyset\}, & \text{else.} \end{cases}$$

In the case of voice and data user arrivals, the sets of all possible actions are $\{A_1, A_2, A_4\}$ and $\{A_1, A_2, A_3\}$, respectively. However, when an event E_l occurs, action a is not feasible if the system state is not present in $\mathcal{S}_{E_l,a}$. In this chapter, voice user blocking (A_1) is considered to be a feasible action in all the states, provided the system is not empty. We consider blocking as a feasible action for data users, only when capacity is reached for both the RATs. When a user departs from LTE or WiFi, the controller can choose either A_1 or A_5. If after the departure of a user from LTE, A_5 is chosen, it offloads one data user from WiFi to LTE.

3.1.3 State Transitions

Let us assume that from each state s, under an action a, the system makes a transition to a state s' with a positive probability $p_{ss'}(a)$. In state $s = (i, j, k)$, let the sum of arrival and service rates of users be denoted by $v(i, j, k)$. Therefore, we have,

$$v(i, j, k) = \lambda_v + \lambda_d + i\mu_v + j\mu_d + k\mu_d.$$

Then,

$$p_{ss'}(a) = \begin{cases} \frac{\lambda_v}{v(i',j',k')}, & s' = (i', j', k'), \\ \frac{\lambda_d}{v(i',j',k')}, & s' = (i', j', k'), \\ \frac{i'\mu_v}{v(i',j',k')}, & s' = (i'-1, j', k'), \\ \frac{j'\mu_d}{v(i',j',k')}, & s' = (i', j'-1, k'), \\ \frac{k'\mu_d}{v(i',j',k')}, & s' = (i', j', k'-1). \end{cases}$$

Values of i', j' and k' as a function of different actions a (conditioned on events E_l) are summarized in Table 3.1, where $a|E_l$ denotes the conditional event that action a is chosen in response to event E_l and $E_l||E_k$ denotes the union of events E_l and E_k.

Table 3.1: Transition probability table.

$\boldsymbol{a\|E_l}$	$\boldsymbol{(i', j', k')}$
$A_1\|(E_1\|\|\ldots\|\|E_5)$	(i, j, k)
$A_2\|E_1$	$(i+1, j, k)$
$A_2\|E_2$	$(i, j+1, k)$
$A_3\|E_2$	$(i, j, k+1)$
$A_4\|E_1$	$(i+1, j-1, k+1)$
$A_5\|(E_3\|\|E_4)$	$(i, j+1, k-1)$
$A_5\|E_5$	$(i, j-1, k+1)$

Note that this table is exhaustive in all kinds of events and actions. However, in a state, we need to consider only those events and actions which are feasible in that state.

3.1.4 Rewards and Costs

Let the reward and cost functions per unit time corresponding to state s and action a be represented by $r(s, a)$ and $c(s, a)$, respectively. Let $R_{L,V}$ and $R_{L,D}$ denote the bit rates of voice and data users in LTE, respectively. To keep the model simple and computationally tractable, we assume that the bit rate of a data user in LTE is constant. In general, a voice user generates Constant Bit Rate (CBR) traffic, and hence we take $R_{L,V}$ to be a constant. $R_{W,D}(k)$ corresponds to the per-user data throughput of k users in WiFi. We assume full buffer traffic model [72] for WiFi. The calculation of $R_{W,D}(k)$ is based on the contention-driven medium access of WiFi users. It is a function of the probabilistic transmission attempts of the users, corresponding success and collision probabilities, and slot times for successful, idle and busy transmissions during collisions.

Remark 5. *Data users want to experience the maximum possible data rate. We assume that a Transmission Control Protocol (TCP) connection is established between the data user and the LTE BS (WiFi AP) before the data transfer begins. The throughput term signifies the maximum data rate provided by the TCP pipe for a single TCP connection.*

The reward per unit time in a state under the occurrence of an event and an action chosen is defined as the total system throughput in that state under that event and the chosen action. The complete description of reward rates for different events and actions in state s is provided in Table 3.2. Note that the reward rate is a monotone increasing function of i and j.

The cost function considered here is as follows. Whenever the centralized controller blocks an incoming voice user, one unit cost is incurred per unit time. Otherwise, it is zero. Thus,

$$c(s, a) = \begin{cases} 1, & \text{if voice user is blocked,} \\ 0, & \text{else.} \end{cases} \tag{3.1}$$

We consider the blocking of data users only when both LTE and WiFi systems are full. Hence, we do not consider any cost on the blocking of data users. Note that once a voice user is associated with LTE, QoSs in terms of delay and data rate are guaranteed by allocating dedicated bearers providing Guaranteed Bit Rate (GBR) in LTE. Therefore, we consider only the blocking probability as the QoS requiremet for voice users.

Table 3.2: Reward rate table.

$(\boldsymbol{a}\vert\boldsymbol{E_l})$	$\boldsymbol{r(s,a)}$
$(A_1\vert E_1)$	$iR_{L,V} + jR_{L,D} + kR_{W,D}(k)$
$(A_1\vert E_2)$	$iR_{L,V} + jR_{L,D} + kR_{W,D}(k)$
$(A_1\vert E_3)$	$(i-1)R_{L,V} + jR_{L,D} + kR_{W,D}(k)$
$(A_1\vert E_4)$	$iR_{L,V} + (j-1)R_{L,D} + kR_{W,D}(k)$
$(A_1\vert E_5)$	$iR_{L,V} + jR_{L,D} + (k-1)R_{W,D}(k-1)$
$(A_2\vert E_1)$	$(i+1)R_{L,V} + jR_{L,D} + kR_{W,D}(k)$
$(A_2\vert E_2)$	$iR_{L,V} + (j+1)R_{L,D} + kR_{W,D}(k)$
$(A_3\vert E_2)$	$iR_{L,V} + jR_{L,D} + (k+1)R_{W,D}(k+1)$
$(A_4\vert E_1)$	$(i+1)R_{L,V} + (j-1)R_{L,D} + (k+1)R_{W,D}(k+1)$
$(A_5\vert E_3)$	$(i-1)R_{L,V} + (j+1)R_{L,D} + (k-1)R_{W,D}(k-1)$
$(A_5\vert E_4)$	$iR_{L,V} + jR_{L,D} + (k-1)R_{W,D}(k)$
$(A_5\vert E_5)$	$iR_{L,V} + (j-1)R_{L,D} + kR_{W,D}(k)$

3.2 Problem Formulation & Solution Methodology

A *decision rule* describes the mapping from a state-event pair to an action at different states $s \in S$ and decision epochs t_n. An association *policy* is a sequence of decision rules $(\pi^{t_1}, \pi^{t_2}, \ldots, \pi^{t_n}, \ldots)$ taken at different decision epochs. Our goal is to determine an association policy which maximizes the total system throughput. This can be formulated as a continuous-time unconstrained MDP problem. In this case, there exists a *pure optimal policy* [69]. A pure policy is a stationary policy where the optimal action in every state is deterministic. Since the contribution of the data users to the total system throughput is more than that of the voice users, the optimal association policy may result in a high blocking probability of voice users. Hence, to address the trade-off between the total system throughput and the voice user blocking probability, we consider the CMDP problem, where we target to maximize the total system throughput subject to a constraint on the voice user blocking probability. In this case, a stationary randomized optimal policy exists [73]. A *randomized policy* is a mixture of two pure policies with associated probabilities. The arrivals and departures of users can occur at arbitrary points in time, which makes the problem continuous time in nature.

3.2.1 Problem Formulation

Let $\mathcal{M}$ be the set of all memoryless policies. To guarantee a unique stationary distribution, we assume that Markov chains associated with the policies are unichain. Following the policy $M \in \mathcal{M}$, let the average reward and cost of the system over infinite horizon be denoted by V^M and B^M, respectively. Let $R(t)$ and $C(t)$ be the total reward and cost of the system incurred up to time t, respectively. For the unconstrained MDP problem, our objective is to maximize the total system throughput which can be described as follows.

$$\text{Maximize:} \quad V^M = \lim_{t\to\infty} \frac{1}{t}\mathbb{E}_M[R(t)], \tag{3.2}$$

where $\mathbb{E}_M$ denotes the expectation operator under the policy M. However, for the CMDP problem, our objective is to maximize the total system throughput subject to a constraint on the blocking probability of voice users. This can be described as follows.

$$\begin{aligned} &\text{Maximize:} \quad V^M = \lim_{t\to\infty} \frac{1}{t}\mathbb{E}_M[R(t)], \\ &\text{subject to:} \quad B^M = \lim_{t\to\infty} \frac{1}{t}\mathbb{E}_M[C(t)] \le B_{\max}, \end{aligned} \tag{3.3}$$

where $B_{\max}$ denotes the constraint on the blocking probability of voice users. Our objective is to determine the optimal policy for both unconstrained and constrained MDP problems. Since the optimal policies are known to be stationary policies, the corresponding limits in Equation (3.2) and (3.3) exist.

3.2.2 Equivalent Discrete-time MDP and Lagrangian Approach

To obtain the optimal policy using RVIA [69], we need to employ the Lagrangian approach [73]. In this approach, for a fixed value of Lagrange Multiplier (LM) β, the reward function is given by

$$r(s,a;\beta) = r(s,a) - \beta c(s,a).$$

The dynamic programming equation described below provides the necessary condition for optimality in case of SMDP $\forall s \in \mathcal{S}$, where $s' \in \mathcal{S}$.

$$V(s) = \max_a [r(s,a;\beta) + \sum_{s'} p_{ss'}(a)V(s') - \rho\bar{t}(s,a)],$$

where $V(s), \rho, \bar{t}(s,a)$ denote the value function of state $s \in \mathcal{S}$, the optimal average reward and the mean transition time from state s under the action a, respectively.

Since the sojourn times are exponential, this is a special case of continuous time controlled Markov chain for which we have,

$$0 = \max_a [r(s,a;\beta) - \rho + \sum_{s'} q(s'|s,a)V(s')], \tag{3.4}$$

where $q(s'|s,a)$ are controlled transition rates satisfying $q(s'|s,a) \geq 0$ for $s' \neq s$ and $\sum_{s'} q(s'|s,a) = 0$. Note that Equation (3.4) follows directly from Poisson equation [74] for continuous time Markov chain. If we scale all the transition rates by a positive scalar, it amounts to time scaling which scales the average reward accordingly for every policy including the optimal, but does not change the optimal policy. Thus, without loss of generality, we assume that $-q(s|s,a) \in (0,1)\ \forall a$, implying in particular that $q(s'|s,a) \in [0,1]$ for $s' \neq s$. Adding $V(s)$ to both sides of Equation (3.4), we have,

$$V(s) = \max_a [r(s,a;\beta) - \rho + \sum_{s'} p_{ss'}(a)V(s')], \tag{3.5}$$

where $p_{ss'}(a) = q(s'|s,a)$ for $s' \neq s$ and $p_{ss'}(a) = 1 + q(s'|s,a)$ for $s' = s$ (recall that $q(s|s,a)$ is negative). This equation is the DP equation for a discrete time MDP (say $\{X_n\}$) with controlled transition probabilities $p_{ss'}(a)$. Here onwards we focus on discrete time setting as described in Equation (3.5).

For a fixed value of β, the following equation describes how RVIA can be used to solve the equivalent unconstrained maximization problem.

$$V_{n+1}(s) = \max_a [r(s,a;\beta) + \sum_{s'} p_{ss'}(a)V_n(s') - V_n(s^*)], \tag{3.6}$$

where $V_n(.)$ is an estimate of the value function after n^{th} iteration, and s^* is an arbitrary but fixed state. Next, we aim to determine the value of β ($= \beta^*$, say) which maximizes the average expected reward subject to the cost constraint. The following gradient descent algorithm describes the rule to update the value of β.

$$\beta_{k+1} = \beta_k + \frac{1}{k}(B^{\pi_{\beta_k}} - B_{\max}), \tag{3.7}$$

where β_k is the value of β, $\frac{1}{k}$ is the step size and $B^{\pi_{\beta_k}}$ is the voice user blocking probability, respectively, at k^{th} iteration. Once the value of β^* is determined, we obtain the optimal policy by employing a perturbation of β^* by a small amount ϵ in both directions (policies $\pi_{\beta^*-\epsilon}$ and $\pi_{\beta^*+\epsilon}$, say) with associated costs $B_{\beta^*-\epsilon}$ and $B_{\beta^*+\epsilon}$, respectively. Finally, we have a randomized optimal policy where the the policies $\pi_{\beta^*-\epsilon}$ and $\pi_{\beta^*+\epsilon}$ are chosen with probabilities p and $(1-p)$, such that

$$pB_{\beta^*-\epsilon} + (1-p)B_{\beta^*+\epsilon} = B_{\max}.$$

We know [75] that the optimal stationary policy can be randomized in at most one $s \in \mathcal{S}$ where the optimal action is randomized between two actions.

3.3 Structure of the Optimal Policy

The dynamic programming equations (Equations (3.5) and (3.6)) described in the previous section are exploited to establish the fact that the optimal policy is of threshold type. The optimality of threshold policy is established with the aid of some lemmas.

3.3.1 Optimal Policy for Data Users

In this section, we present the structural properties of the optimal policy for the association of data users along with their physical interpretations. Let us denote the throughput increment in WiFi when the number of WiFi users increases from k to $(k+1)$ by $\tilde{R}_{W,D}(k)$. Therefore, $\tilde{R}_{W,D}(k) = (k+1)R_{W,D}(k+1) - kR_{W,D}(k)$. We assume the following.

Assumption 1. *Let $R_{L,D}$ be such that $R_{L,D} \geq \tilde{R}_{W,D}(k)$, $\forall k \geq k_{th}$ and $R_{L,D} < \tilde{R}_{W,D}(k)$, $\forall k < k_{th}$, where k_{th} is a threshold such that if $k \geq k_{th}$, the data rate improvement provided by a single data user in LTE is more than the improvement in total WiFi throughput as the number of WiFi data users is increased from k to $(k+1)$.*

Remark 6. *Following the full buffer traffic model [72], $\tilde{R}_{W,D}(k)$ initially increases with k and then decreases. This behavior matches with Assumption 1. The value of k_{th} can be easily obtained by computing $\tilde{R}_{W,D}(k)$ (following [72]) for different values of k and comparing with $R_{L,D}$.*

The following two lemmas describe a threshold structure of the optimal policy for the association of data users. Specifically, up to a certain threshold on the total number of data users, data users are served using WiFi. After the threshold is crossed, data users are served using LTE.

Lemma 1. *For every i and j such that $(i+j) < C$, if the total number of data users in the system is $(j+k) \le k_{th}$, then the optimal policy is to serve all data users using WiFi. In other words, $(j+k) \le k_{th} \implies j = 0$.*

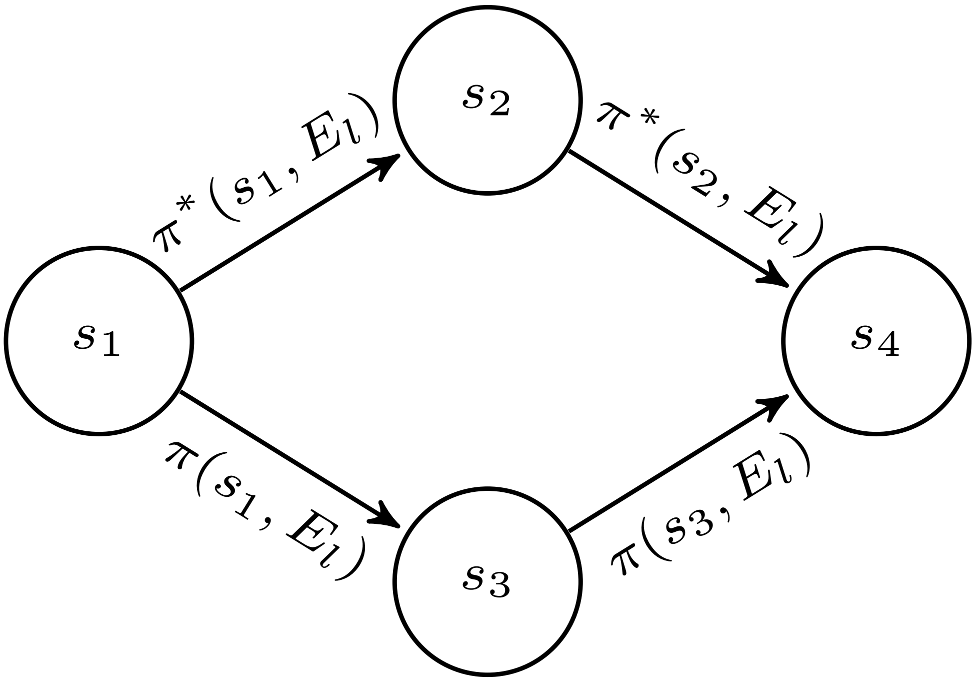


Figure 3.2: Sample path under different policies.

Proof. Since the decisions of association and offloading are involved during the arrival and the departure of users, proving this lemma is equivalent to proving the following statements.

(a) A_3 (Accept in WiFi) is optimal when there are less than k_{th} data users in the system, and a data user arrives.

(b) A_1 (Do nothing) is optimal when there are less than or equal to k_{th} data users in the system, and a voice user from LTE departs.

(c) A_1 (Do nothing) is optimal when there are less than or equal to k_{th} data users in the system, and a data user from WiFi departs.

We prove the statements by sample path arguments. Suppose the system starts at time $t = 0$.

Proof of (a): We consider the scenario when the system is in the state $s_1 = (i, 0, 0)$, when a data user arrival occurs (after a time t_1, say). Assume that the optimal policy π^* does not associate the incoming data user with WiFi. Therefore, the optimal action must be A_2 (accept in LTE). As the optimal policy is π^*, we have $V^{\pi^*}(s) \geq V^{\hat{\pi}}(s)$, $\forall \hat{\pi} \in \prod$ and $\forall s \in \mathcal{S}$, where $\prod$ is the set of all policies. Let us consider another policy π (non-stationary in general) which chooses A_3 in state $s_1 = (i, 0, 0)$. As illustrated in Fig. 3.2, let us assume that starting from the state s_1 and following the policy π^* and π, the system reaches the state $s_2 = (i, 1, 0)$ and $s_3 = (i, 0, 1)$, respectively. The inter-arrival times and service times are same for both the sample paths as we have considered a Markovian system. Assume that from the state s_2, based on the next event (after a time t_2) and the chosen action, the system makes a transition to the state s_4 according to the policy π^*. Before reaching the state s_2, the sample path followed by the policy π^* has one less WiFi data user and one more LTE data user than that of the policy π before it reaches the state s_3. Suppose, the policy π is such that in state s_3, it takes the same action as that of policy π^* and additionally offloads one data user from WiFi to LTE. Hence, sample path followed by both the policies end up in the same state s_4. We construct π in such a manner that from the state s_4 onwards, both the policies choose the same actions and follow the same sample path. Therefore, the difference of value functions of the state s_1 under the policy π^* and π is

$$V^{\pi^*}(s_1) - V^{\pi}(s_1) = R_{L,D} - R_{W,D}(1).$$

Since $R_{L,D} < \tilde{R}_{W,D}(k), \forall k < k_{th}$ and $\tilde{R}_{W,D}(1) = R_{W,D}(1)$, we have, $V^{\pi^*}(s_1) < V^{\pi}(s_1)$. Clearly, this contradicts the original claim that π^* is an optimal policy. Since the Markov chains induced by different policies are recurrent, the state $(i, 0, 0)$ is visited infinitely often. Upon each visit, the choice of action A_3 upon a data user arrival provides more reward than action A_2. Therefore, when there is no data user in the system, and one data user arrives, A_3 is optimal. In a similar manner, it can be proved that A_3 is optimal when a data user arrives and the system is in state $(i, 0, k)$, where $k < k_{th}$.

Proof of (b) and (c): These can be proved using a similar sample path argument.

$\square$

In Lemma 1, following Assumption 3, since for $k < k_{th}$, the data rate improvement is more if an additional data user is served using WiFi rather than using LTE, it is optimal to serve the data users using WiFi.

Lemma 2. *For every i and j such that $(i + j) < C$, if the total number of data users in the system is $(j + k) > k_{th}$, then the optimal policy is to serve k_{th} data users using WiFi and all other data users using LTE. In other words, $(j + k) > k_{th} \implies k = k_{th}$.*

Proof. Similar to Lemma 1, proving this lemma is equivalent to proving the following statements.

(a) A_2 (Accept in LTE) is optimal when there are more than or equal to k_{th} data users in the system, and one data user arrives.

(b) A_1 (Do nothing) is optimal when there are more than k_{th} data users in the system, and a voice/data user from LTE departs.

(c) A_5 (Data offload to a RAT from where a user has departed) is optimal when there are more than k_{th} data users in the system, and a data user from WiFi departs.

Proof of (a):

From Lemma 1, we have, $(j + k) \leq k_{th} \implies j = 0$. We consider the scenario when the system is in the state $(i, 0, k_{th})$, and a data user arrival occurs. Assume that the optimal policy π^* does not associate this incoming data user with LTE. Consequently, the optimal action must be A_3. As the optimal policy is π^*, we have $V^{\pi^*}(s) \geq V^{\hat{\pi}}(s)\ \forall \hat{\pi} \in \prod$ in every state s. Let us consider another policy π which chooses A_2 in state $(i, 0, k_{th})$. As illustrated in Fig. 3.2, starting from the state $(i, 0, k_{th})$ and following the policies π^* and π, the system reaches the states s_2 and s_3, respectively. From the state s_2, based on an event, the system reaches the state s_4. Suppose, in the state s_3, the action governed by the policy π is such that it chooses the same action as that of policy π^* and additionally, offloads one data user from LTE to WiFi. Clearly, paths followed by both the policies end up in the same state s_4. We construct π in such a way that from state s_4 onwards, both of them follow the same path. Similar to the previous lemma, the difference of value functions under the policies π^* and π is

$$V^{\pi^*}(s_1) - V^{\pi}(s_1) = (k + 1)R_{W,D}(k + 1) - kR_{W,D}(k) - R_{L,D}.$$

Since $R_{L,D} \geq \tilde{R}_{W,D}(k), \forall k \geq k_{th}$, we have, $V^{\pi^*}(s_1) < V^{\pi}(s_1)$. Clearly, this contradicts the original claim that π^* is an optimal policy. Thus, A_2 is optimal when there are k_{th} data users in WiFi, and one data user arrives. The same result can be extended for the case when there are k_{th} data users in WiFi, more than or equal to one data user in LTE, and one data user arrives.

Statements (b) and (c) can be proved in a similar way. □

The physical significance of Lemma 2 is that for $k \geq k_{th}$, the data rate improvement provided by a single data user in LTE is more than that in WiFi (following Assumption 1), and hence, it is optimal to serve up to k_{th} data users using WiFi and serve the additional data users using LTE.

Following lemma is a direct consequence of how the system is modeled.

Lemma 3. *For every i and j such that $(i + j) = C$, the optimal policy is to serve all the incoming data users using WiFi until $k = W$, where an incoming data user is blocked.*

Proof. Proof follows directly from the system model. □

3.3.2 Optimal Policy for Voice Users

In this section, we characterize the optimal policy for the arrival of voice users. We prove that the optimal policy is of threshold type. In this section, the terminologies "increasing" and "decreasing" are used in the weak sense of "non-decreasing" and "non-increasing", respectively. In each state, let the sum of arrival and service rates be denoted by $v(i, j, k)$. Thus, we have,

$$v(i, j, k) = \lambda_v + \lambda_d + i\mu_v + (j + k)\mu_d.$$

Let us define $f(i, j, k) = iR_{L,V} + jR_{L,D} + kR_{W,D}(k)$. The subsequent lemma describes the superiority of one action over the other for the association of incoming voice users. Specifically, up to a certain threshold on the total number of data users, A_4 (accept voice user in LTE with data user offload to WiFi) is better than A_2 (accept voice user in LTE). After the threshold is crossed, A_2 becomes better.

Lemma 4. *In the case of a voice user arrival in state (i, j, k), where $(i + j) < C$,*

(i) A_4 is always better than A_2 if $k < k_{th}$,

(ii) A_2 is always better than A_4 if $k \geq k_{th}$.

Proof. Proof is similar to the proof of Lemma 1. □

Similar to Lemmas 1 and 2, following Assumption 1, since for $k < k_{th}$, the data rate improvement is more if an additional data user is served in WiFi rather than in LTE, A_4 is better than A_2. Hence, when $k < k_{th}$, the choice of optimal action is between A_4 (accept voice user in LTE with data user offload to WiFi) and A_1 (blocking). Similarly, for $k \geq k_{th}$, the optimal action is either A_2 (accept voice user in LTE) or A_1.

The following lemma describes that when capacity is not reached in LTE and a voice user arrives, a threshold structure is observed. Until a threshold on the number of voice and data users in LTE, A_2 (for $k \geq k_{th}$) or A_4 (for $k < k_{th}$) is preferred. After the threshold, A_1 becomes optimal.

Lemma 5. *For every i and j such that $(i + j) < C$ and a voice user arrival,*

(i) if the optimal action in state (i, j, k) is A_1, then the optimal actions in states $(i + 1, j, k)$ and $(i, j + 1, k)$ are also A_1,

(ii) if the optimal action in state (i, j, k) is A_2 (A_4), then the optimal actions in states $(i - 1, j, k)$ and $(i, j - 1, k)$ are also A_2 (A_4).

Proof. Proof is provided in Appendix A.1. □

When the number of voice/data users in LTE is less, A_2 or A_4 is chosen as the optimal action in the event of a voice user arrival. When i or j crosses a certain threshold, the number of free resources for incoming voice users decreases. Therefore, the blocking probability of voice users increases. Thus, after a threshold on i or j, A_1 becomes optimal.

However, when $(i + j) = C$, since A_2 is infeasible, optimal action is either A_1 or A_4. The lemma presented next discusses the threshold nature of the optimal policy for voice user arrivals when $(i + j) = C$.

Lemma 6. *For every i and j such that $(i + j) = C$ and a voice user arrival,*

(i) if the optimal action in state (i, j, k) is A_1, then the optimal action in state $(i + 1, j - 1, k)$ is also A_1,

(ii) if the optimal action in state (i, j, k) is A_4, then the optimal action in state $(i - 1, j + 1, k)$ is also A_4.

Proof. Proof is provided in Appendix A.2. □

The physical interpretation of this lemma is that for the states with $(i+j) = C$, when i is small, A_4 is preferred. However, when i crosses a threshold, since j becomes small, and consequently, the total system throughput is small, A_4 may further lower the total system throughput. Therefore, blocking of voice users is chosen as the optimal action.

3.4 Proposed Network-Initiated Association Algorithms

Based on the threshold structures of the optimal policies for the unconstrained MDP and CMDP problems, in this section, we propose two network-initiated association algorithms for an LTE-WiFi HetNet.

3.4.1 Unconstrained MDP-based Association Algorithm

The details of the Unconstrained MDP-based Association Algorithm (UMAA) is presented in Algorithm 1. As discussed in Section 3.3, the threshold for the service of data users, k_{th}, can be computed easily without the knowledge of the arrival and service rates of users. The value of k_{th} completely specifies the optimal policy for the arrival of data users (event E_2) and the departure of voice and data users (events E_3, E_4 and E_5). Since this is an unconstrained problem, we set $\beta = 0$.

The procedure THRESHOLD–POLICY–SEARCH of UMAA computes the optimal policy for the arrival of voice users by solving an unconstrained MDP problem. The proposed procedure is motivated from the well-known policy iteration algorithm [69]. However, the threshold nature of the optimal policy (as derived in Lemmas 5 and 6) is exploited which offers a significant reduction in computational complexity over policy iteration. We initially choose an arbitrary threshold th_0. In the POLICY–EVALUATION procedure, the system of value function equations is solved to determine the value function vector corresponding to thresholds th_n, $th_n + 1$ and $th_n - 1$. In the POLICY–IMPROVEMENT procedure, if the policy corresponding to the threshold $th_n + 1$ (or $th_n - 1$) improves the value function vector with at least one strict inequality (denoted by $\succeq$), the threshold is updated to $th_n + 1$ (or $th_n - 1$). In other words, compared to th_n, the policy corresponding to the threshold th_{n+1} (or $th_n - 1$) has lesser number of states where the action is suboptimal. The procedure stops when no further improvement in value function vector

Algorithm 1 Unconstrained MDP-based Association Algorithm (UMAA).

```
Input: $\lambda_v, \lambda_d, \mu_v, \mu_d, R_{L,V}, R_{L,D}, R_{W,D}(.)$.
 1: Compute $k_{th}$ for data users (See Remark 6). Set $\beta \leftarrow 0$.
 2: procedure THRESHOLD–POLICY–SEARCH
 3:     Initialize threshold $th_0 \leftarrow th$. Set $n \leftarrow 0$.
 4:     procedure POLICY–EVALUATION
 5:         Solve the linear system for policies corresponding to thresholds $th_n$,$(th_n - 1)$ and $(th_n + 1)$.
 6:         $V(s) = \sum_{E_l} \hat{p}(s, E_l)[\hat{r}(s, E_l, M(s); \beta) + V\big(s'(E_l, M(s))\big)] + \big(1 - \sum_{E_l} \hat{p}(s, E_l)\big)V(s)$, $\forall s \in \mathcal{S}$.
 7:     end procedure
 8:     procedure POLICY–IMPROVEMENT
 9:         If $V_{th_n+1} \succeq V_{th_n}$, set $th_{n+1} \leftarrow th_n + 1$.
10:         Elseif $V_{th_n-1} \succeq V_{th_n}$, set $th_{n+1} \leftarrow th_n - 1$.
11:         Else set $th_{n+1} \leftarrow th_n$.
12:         If $th_{n+1} == th_n$ stop,
13:         Else $n \leftarrow n + 1$ and go to Line 4.
14:     end procedure
15: end procedure
Output: Deterministic optimal policy.
16: Store thresholds $va_c(j, k)$ and $va_{lc}(j, k)$ for the association of voice users for $(i+j) = C$ and $(i+j) < C$,
    respectively.
17: procedure POLICY–IMPL
18:     for each arrival of voice users do
19:         if $(i + j) < C$ then
20:             Choose $A_1$ if $i \geq va_{lc}(j, k)$.
21:             Choose $A_2$ if $i < va_{lc}(j, k)$ and $k \geq k_{th}$.
22:             Choose $A_4$ otherwise.
23:         else
24:             Choose $A_4$ if $i < va_c(j, k)$, $A_1$ otherwise.
25:         end if
26:     end for
27:     for each arrival of data users do
28:         if $(i + j) < C$ then
29:             Choose $A_3$ if $k < k_{th}$, $A_2$ otherwise.
30:         else
31:             Choose $A_3$ if $k < W$, $A_1$ otherwise.
32:         end if
33:     end for
34:     for each departure of users from LTE (WiFi) do
35:         Choose $A_1$ $(A_5)$ if $k \leq k_{th}$, $A_5(A_1)$ otherwise.
36:     end for
37: end procedure
```

Algorithm 2 CMDP-based Association Algorithm (CAA).

```
Input: $\lambda_v, \lambda_d, \mu_v, \mu_d, R_{L,V}, R_{L,D}, R_{W,D}(.), B_{\max}$.
 1: Compute threshold $k_{th}$ for the association of data users.
 2: procedure CALC–OPT–POLICY
 3:     Initialize $\beta$.
 4:     while $|B^{\pi_\beta} - B_{\max}| > \theta$ do
 5:         procedure THRESHOLD–POLICY–SEARCH
 6:             See Algorithm 1.
 7:         end procedure
 8:         Update $\beta$ using Equation (3.7).
 9:     end while
10: end procedure
Output: Randomized optimal policy.
11: Compute thresholds $va_c(j,k)$ and $va_{lc}(j,k)$ for association of voice users for $(i+j) = C$ and $(i+j) < C$,
    respectively.
12: procedure POLICY–IMPL
13:     As discussed in Algorithm 1.
14: end procedure
```

is possible. Since in every iteration the policy improves and remains within the set of threshold policies, UMAA converges to the optimal threshold policy.

The calculated thresholds for the association of voice/data users are made available to the centralized controller connected to both the LTE BS and the WiFi AP. Since the centralized controller has an overall view of the whole system, the information regarding the numbers of active voice and data users in LTE and WiFi is available to it. Whenever there is an arrival or a departure, the controller initiates the procedure POLICY–IMPL, as described in UMAA. This procedure determines the state of the system based on the number of active users in LTE and WiFi networks and then chooses an appropriate action based on the corresponding thresholds.

3.4.2 CMDP-based Association Algorithm

We describe the CMDP-based Association Algorithm (CAA) which addresses the issue of high blocking probability of voice users, which may be encountered in UMAA. The details of CAA are described in Algorithm 2. Apart from the same set of input parameters as required by UMAA, CAA requires $B_{\max}$ as an additional parameter to specify the constraint on the blocking probability of voice users. The procedure CALC–OPT–POLICY in CAA computes the randomized optimal policy for the considered CMDP problem. First, the optimal policy is determined for a fixed value of β using same methodologies as in UMAA. Then the value of β is updated until the difference of B^{π_β} and $B_{\max}$ becomes less than θ, where θ is a very small positive real number. All other procedures are similar to the procedures described in UMAA.

3.4.3 Complexity Analysis

In this subsection, we investigate the gain in computational complexities of the proposed algorithms in comparison to the traditional policy iteration algorithm [69], which performs the policy search over the entire policy space in order to determine the optimal policy. As discussed before, the computation of optimal policies for data user arrival, voice user departure and data user departure is equivalent to the calculation of k_{th} with a computational complexity of $O(1)$.

Now, we derive the computational complexity associated with obtaining the optimal policy for voice user arrivals. In UMAA, we evaluate three policies by solving a set of equations using Gaussian elimination with associated complexity $O(|\mathcal{S}|^3)$, where $|\mathcal{S}|$ is the dimensionality of the state space. As described in Section 3.1, a state (i, j, k) in the state space $\mathcal{S}$ has the constraints $(i + j) \leq C$ and $k \leq W$. Therefore, the dimensionality of the state space is $|\mathcal{S}| = O(C^2W)$. Hence, each iteration of the policy evaluation can be performed in a polynomial time as a function of C and W. In the policy improvement phase, the threshold structure of the optimal policies for the service of data user and voice user arrival are exploited to reduce the number of iterations in comparison to the policy iteration algorithm. Without the knowledge of the threshold properties, the number of feasible policies for voice user arrival is $O(|\mathcal{A}|^{|\mathcal{S}|})$, where $|\mathcal{A}|$ is the size of the action space. To analyze how the threshold nature of the optimal policy helps in reducing the number of feasible policies, we consider the following cases.

1. $0 \leq k < k_{th}$: As derived in Lemma 1, $k < k_{th} \implies j = 0$. According to Lemma 5, there exists a threshold on the value of i, where the optimal action changes to A_1. Since for every value of k, the threshold of blocking of voice users can be placed

anywhere on the line $j = 0$, the number of policies in this region is equal to $C^{k_{th}}$.

2. $k = k_{th}$: Using Lemma 1 and 2, $k = k_{th} \implies j \geq 0$. For every value of j, the threshold of blocking of voice users (see Lemma 5) can be placed in $(C - j)$ different ways. Thus, the total no of policies is $O(C!)$.

3. $W \geq k > k_{th}$: As established in Lemma 3, $k > k_{th} => (i + j) = C$. According to Lemma 6, there exists a threshold on the value of i, where the optimal action changes from A_4 to A_1. Thus, for every value of k, the threshold can be placed in C different ways. Thus, the total number of policies in this region is $C^{(W-k_{th})}$.

Therefore, the total number of feasible policies is $O(C!C^W)$ which is the number of iterations required for UMAA in the worst case. As a result, the worst case computational complexity of UMAA (which is $O(C!C^W)$) is less than the computational complexity of policy iteration (which is $O(|\mathcal{A}|^{C^2W})$). Thus, the knowledge of threshold properties helps in achieving faster convergence in comparison to policy iteration. Unike the policy iteration pahse, the policy improvement phase consists of a comparison of value functions, which reduces the computational complexity from $O(|\mathcal{S}|^2|\mathcal{A}|)$ to $O(|\mathcal{S}|)$.

The computational complexity of CAA for a single iteration of β is equal to the computational complexity of UMAA.

The threshold property also reduces the storage complexity of the policy. Without the knowledge of the threshold nature of the optimal policy, the number of bits required to store the optimal policy is $O(C^2W)$. By virtue of the threshold properties, we need to store the value of a single threshold in the case of data users and $(k_{th} + C + (W - k_{th}))$ thresholds in the case of voice users, which corresponds to storing only $(C + W)$ bits. This is a considerable reduction in the storage complexity as well.

3.5 Numerical and Simulation Results

In this section, the algorithms proposed in the last section are implemented in ns-3 to observe the performance of the proposed algorithms. Performance of the proposed algorithms in terms of the blocking probability of voice users and the total system throughput is compared to the performance of on-the-spot WiFi offloading algorithm [57].

In this algorithm [57], data user chooses LTE only when there is no WiFi coverage. Therefore, in the considered system model, with on-the-spot offloading, data users always get associated with WiFi until WiFi capacity is exhausted. Voice users always get associated with LTE BS, and when LTE capacity is full, they are blocked.

We also compare the performance of the proposed algorithms with the LTE-preferred scheme. In this scheme, voice and data users are associated with LTE until LTE reaches its capacity. When LTE reaches its capacity, and a voice user arrives, one existing LTE data user is offloaded to WiFi unless WiFi system also reaches its capacity. Otherwise, the incoming voice user is blocked. Additionally, when a voice user from LTE departs, if there is more than or equal to one data user in WiFi, an existing WiFi data user is moved to LTE.

3.5.1 Simulation Model and Evaluation Procedure

The simulated network model consists of a 3GPP LTE BS and an IEEE 802.11g [13] WiFi AP. All users are taken to be stationary. The AP is approximately 50 m away from the LTE BS, and data users are distributed uniformly within 30 m radius of the WiFi AP. The WiFi AP is assumed to be deployed by the same cellular operator and hence trusted from the point of view of interworking. LTE and WiFi network parameters used in the simulation, as summarized in Table 3.3 and 3.4, are based on 3GPP models [76, 77] and saturation throughput [72] 802.11g WiFi model. Propagation delay in WiFi network is assumed to be negligible. We consider CBR traffic for voice and data users in LTE. The generation of a fixed rate uplink flow is implemented in ns-3 using an application developed by us, which works similar to the ON/OFF application. This application creates sockets between the sender and the receiver, and fixed sized packets are transmitted from the sender to the receiver at a constant bit rate.

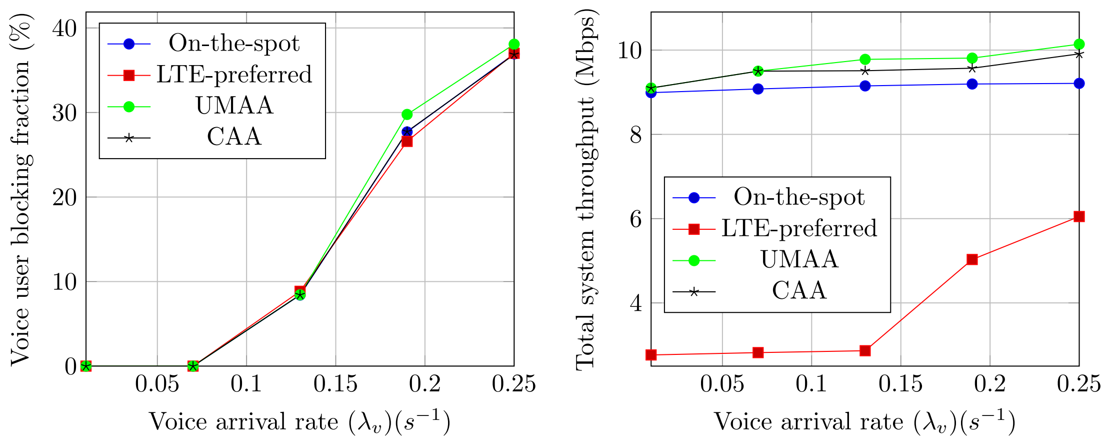
3.5.2 Voice User Arrival Rate Variation

3.5.2.1 Voice User Blocking Probability Performance

Fig. 3.3a illustrates the variation of voice user blocking percentage of on-the-spot offloading [57], LTE-preferred, UMAA and CAA as a function of λ_v. In on-the-spot offloading,

Table 3.3: LTE network model.

Parameter	**Value**
Maximum voice capacity	10 users
Maximum data capacity	10 users
Voice bit rate of a single user	20 kbps
Data bit rate of a single user	5 Mbps
Voice packet payload	50 bits
Data packet payload	600 bits
Tx power for BS and UE	46 dBm and 23 dBm
Noise figure for BS and UE	5 dB and 9 dB
Antenna height for BS and UE	32 m and 1.5 m
Antenna parameter for BS and UE	Isotropic antenna
Path loss	$128.1 + 37.6 \log(R)$, R in kms



(a) Voice user blocking percentage vs. λ_v ($\lambda_d = 1/20, \mu_v = 1/60$ and $\mu_d = 1/10$).

(b) Total system throughput vs. λ_v ($\lambda_d = 1/20, \mu_v = 1/60$ and $\mu_d = 1/10$).

Figure 3.3: Plot of blocking fraction of voice users and total system throughput for different algorithms under varying λ_v.

voice users are blocked when LTE reaches the capacity. When λ_v is small, the voice user blocking probability is small. However, as λ_v increases, the probability of approaching the LTE capacity and hence the voice user blocking probability increases. The voice user blocking probability in UMAA is small when λ_v is small. However, as λ_v increases, voice

Table 3.4: WiFi network model.

Parameter	**Value**
Channel bit rate	54 Mbps
User Datagram Protocol (UDP) header	224 bits
Packet payload	1500 bytes
Slot duration	20 μs
Short Inter-frame Space (SIFS)	10 μs
Distributed Coordination Function IFS (DIFS)	50 μs
Minimum acceptable per-user throughput	3.5 Mbps
Tx power for AP	23 dBm
Noise figure for AP	4 dB
Antenna height for AP	2.5 m
Antenna parameter	Isotropic antenna

user blocking probability values become marginally higher than the corresponding values for on-the-spot offloading. UMAA may introduce blocking of voice users even when LTE has not reached its capacity, i.e., for states with $(i + j) < C$. Voice users have very less contribution to the total system throughput. Hence, voice users are blocked to save resources for data users which contribute significantly to the total system throughput.

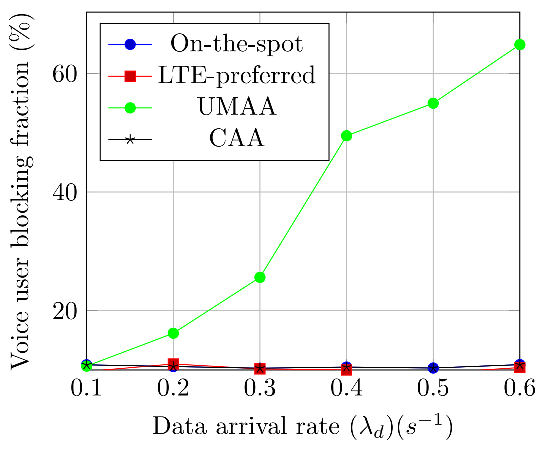
However, in CAA, the number of states with proactive blocking (blocking when $(i + j) < C$) is reduced due to the presence of a constraint on the voice user blocking probability. Additionally, when i is small, the optimal action in states with $(i + j) = C$ becomes A_4 (accept voice user in LTE and data offload to WiFi). Voice user blocking probability contribution comes mainly from the states with $(i + j) = C$, where i is large (say states $(C, 0, 0)$,$(C - 1, 1, 0)$ etc.). Since a major fraction of voice user blocking occurs when $(i + j) = C$ and i is large, the system becomes analogous to the on-the-spot offloading. Hence, the voice user blocking probability performance of CAA is almost similar to that of the on-the-spot offloading algorithm. Both on-the-spot offloading and LTE-preferred schemes block voice users only when LTE system is full with only voice users. Therefore, the blocking probability performances of these two schemes are similar.

3.5.2.2 Total System Throughput Performance

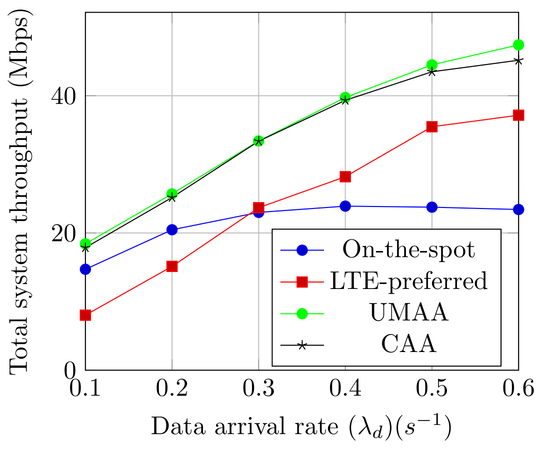
Total system throughput performance comparison of different algorithms is illustrated in Fig. 3.3b. In on-the-spot offloading, the average number of voice users in LTE increases with λ_v, while the average number of data users in WiFi remains constant. Thus, the total system throughput increases with λ_v. In the case of UMAA, with an increase in λ_v, the blocking probability of voice users increases. Therefore, the fraction of voice users in the system decreases, and the total system throughput increases. Besides, UMAA performs a significant amount of load balancing under actions A_4 (accept voice user in LTE with data user offload to WiFi) and A_5 (move data user to the RAT from where a user has departed). With higher λ_v, load balancing actions are chosen more frequently. Thus, with higher λ_v, UMAA exhibits a greater improvement over on-the-spot offloading algorithm. The improvement in total system throughput varies from 1.22% (for $\lambda_v = 0.01$) to 10.32% (for $\lambda_v = 0.25$). In Fig. 3.3b, we observe that CAA also performs better than on-the-spot offloading. However, due to the presence of a constraint on the voice user blocking probability, the performance improvement is lower than that of UMAA. For lower values of λ_v ($\lambda_v = 0.01, 0.07$), the total system throughput of CAA is same as that of UMAA as the optimal policy for the CMDP is same as that of the unconstrained MDP. On-the-spot offloading algorithm blocks the voice users only when LTE reaches capacity. Typically, in CAA also, blocking of voice users occurs when the LTE is full with a large number of voice users. However, due to load balancing of data users, CAA outperforms the on-the-spot offloading algorithm. With $\lambda_v = 0.01$, the improvement in total system throughput is only 1.22% and with $\lambda_v = 0.25$, it becomes 7.60%. LTE-preferred scheme associates both voice and data users to LTE. As λ_v increases, the probability that LTE reaches its capacity, increases. Therefore, to accommodate the incoming voice users, existing LTE data users are offloaded to WiFi. This increases the total system throughput when WiFi load is low. However, both UMAA and CAA outperform the LTE-preferred scheme.

3.5.3 Data User Arrival Rate Variation

In this section, performances of the considered algorithms are compared for varying data user arrival rates.



(a) Voice user blocking percentage vs. λ_d ($\lambda_v = 1/6, \mu_v = 1/60$ and $\mu_d = 1/10$).



(b) Total system throughput vs. λ_d ($\lambda_v = 1/6, \mu_v = 1/60$ and $\mu_d = 1/10$).

Figure 3.4: Plot of blocking fraction of voice users and total system throughput for different algorithms under varying λ_d.

3.5.3.1 Voice User Blocking Probability Performance

In on-the-spot offloading, voice and data users are accepted in LTE and WiFi, respectively. Consequently, as observed in Fig. 3.4a, changes in λ_d do not affect the blocking probability performance of voice users in LTE. In the case of UMAA, increase in λ_d associates more number of data users with LTE since the optimal policy for data users is to associate with LTE after the number of WiFi data users crosses a certain threshold. Therefore, the number of free LTE resources for voice users reduces, eventually increasing the blocking probability of voice users. The voice user blocking probability of UMAA is worse than that of on-the-spot offloading and increases with λ_d. The blocking probability performance of CAA and LTE-preferred scheme are similar to that of the on-the-spot offloading. Since the voice users are blocked in the states where the only feasible action is blocking (say state $(C, 0, 0)$), the decision epochs where voice users are blocked are almost same as that of the on-the-spot offloading and the LTE-preferred scheme.

3.5.3.2 Total System Throughput Performance

In Fig. 3.4b, total system throughput for different algorithms are plotted as a function of λ_d. In on-the-spot offloading, with an increase in λ_d, the number of WiFi data users increases, and this increases the total system throughput. However, for a high λ_d, the effect of contention among data users reduces the rate of increment of the total system throughput. In UMAA, as λ_d increases, more number of data users are served using LTE. Since the throughput contribution of data users is more than that of voice users, the blocking probability of voice users increases with λ_d. Thus, the fraction of voice users in the system reduces, effectively causing more improvement in the total system throughput. When $\lambda_d = 0.1$, the improvement in system metric is 25.22%, whereas for $\lambda_d = 0.6$, the system metric almost doubles. In Fig. 3.4b, the total system throughput values for CAA are smaller than the corresponding values for UMAA. The reduction in blocking probability of voice users comes at a price of the reduction in the total system throughput. Still, due to optimal association and load balancing decisions, CAA reduces the effect of contention among data users in WiFi and hence performs better than the on-the-spot offloading algorithm. For example, with $\lambda_d = 0.1$, the improvement in system metric is about 22.96% and with $\lambda_v = 0.6$, it becomes almost 93%. Similar to Fig. 3.3b, in case of the LTE-preferred scheme, as λ_d increases, more data users are offloaded to WiFi, resulting in an improvement in total system throughput. Since the effect of contention in the LTE-preferred scheme is lesser than that of the on-the-spot offloading, LTE-preferred scheme outperforms the on-the-spot offloading under large λ_d. However, the performance of LTE-preferred scheme is worse than both UMAA and CAA.

3.6 Conclusion

In this chapter, we have formulated the optimal association problem in an LTE-WiFi HetNet as an MDP problem with an objective of maximizing the total system throughput. Constrained MDP formulation has also been presented where maximizing the total system throughput is subject to a constraint on the blocking probability of voice users. Threshold structures on the optimal policy for the association of voice and data users have been derived. Based on the structure of the optimal policies, we have proposed two algorithms for the association and offloading of voice/data users in an LTE-WiFi HetNet. The analysis indicates that the knowledge of threshold structures reduces the number of feasible policies and hence reduces the computational complexity considerably (from $O(|\mathcal{A}|^{C^2W})$ to $O(C!C^W)$) in comparison to the policy iteration algorithm. Simulation results have demonstrated that although the voice user blocking probability performance of UMAA is worse than that of on-the-spot offloading, CAA performs as good as on-the-spot offloading and LTE-preferred scheme. Moreover, the proposed algorithms perform better than both on-the-spot offloading algorithm and LTE-preferred scheme in improving the total system throughput.

However, the proposed algorithms require the knowledge of the statistics of the arrival processes of voice and data users for the computation of the optimal policy. In practice, these statistics may be difficult to obtain. In the next chapter, we propose RAT selection algorithms that can operate without the knowledge of the statistics of the arrival processes and hence, are practically implementable.

Chapter 4

Online Algorithms for Throughput Efficient RAT Selection

In this chapter, an extension of the framework developed in Chapter 3 is considered. In Chapter 3, we have determined the optimal RAT selection policy which maximizes the total system throughput subject to a constraint on the blocking probability of voice users. Although the proposed algorithms provide significant reductions in computational complexity over the traditional policy iteration, they require the knowledge of the transition probabilities between different states associated with the underlying Markov chain. Transition probabilities are governed by the statistics of the arrival processes of voice and data users. Recent studies [50–52] on the characteristics of cellular traffic establish that although the voice traffic can be predicted accurately, the current prediction schemes for data traffic may not perform well. Therefore, obtaining real-time traffic statistics in today's cellular network is difficult. Performances achieved by different schemes are usually obtained under an assumed distribution of the arrival processes. However, the assumed model may be inaccurate since the parameters are difficult to obtain in practice. To handle this problem, we propose a model *unaware* online RAT selection algorithm which converges to the optimal policy in the long run. The proposed algorithm is based on Q-learning [45] which is a well-known RL algorithm for MDP problems. One of the main advantages of learning is that it avoids explicit estimation which may have high variance or may be computationally prohibitive. It replaces the conditional averaging in an iterative scheme for solving the DP equation by an actual evaluation at an observed transition and an incremental update which does the conditional averaging implicitly. The proposed algorithm can be implemented without the knowledge of the statistics of the arrival processes. Although [53] and [25] undertake Q-learning based approaches for RAT selection and offloading for an LTE-based HetNet, contrary to our network-initiated approach, these schemes are user-initiated in nature. The scheme in [25] uses network-provided information in taking RAT selection decisions without the possibility of offloading of data users. In [53], distributed traffic offloading decisions are taken by users based on the available local information, without any consideration of association strategies.

However, due to the associated exploration mechanism, Q-learning algorithm may take a large number of iterations to converge. Furthermore, Q-learning algorithm needs to store the value functions associated with every state-action pair, thus possessing a significant storage complexity. To address these issues, in this chapter, we also propose a Post-Decision State (PDS) learning algorithm which speeds up the learning process by removing the action exploration. This approach is based on a reformulation of the RVIA equation and can be implemented online in the SA framework. Furthermore, the PDS learning algorithm has a lower space complexity than that of Q-learning because instead of the state-action pair values, we need to store the value functions associated with the states. We also prove the convergence of the PDS learning algorithm for RAT selection to the optimality. Extensive simulations are conducted in ns-3 [44] to evaluate the performances and convergence behaviors of the proposed association algorithms.

The rest of the chapter is organized as follows. Section 4.1 describes the system model and the problem formulation within the CMDP framework. Section 4.2 discusses the RL implementation for the problem described in Section 4.1 and proposes an online Q-learning algorithm for RAT selection. We introduce the notion of PDS in Section 4.3. Section 4.4 proposes the PDS learning algorithm for RAT selection. Section 4.5 presents simulation results. We conclude in Section 4.6.

4.1 System Model & Problem Description

The system model and problem formulation adopted in this chapter has already been introduced in Sections 3.1 and 3.2 of Chapter 3, respectively. Therefore, we do not repeat the system model and problem formulation in this chapter. However, for the sake of completeness, we revisit the important aspects of our model and problem formulation.

4.1.1 System Model

We consider a system consisting of an LTE BS and a WiFi AP (as shown in Fig. 3.1 of Chapter 3) where voice and data users are present. There is a common resource pool in LTE for voice users and data users. All the users are stationary.

Voice and data user arrivals follow Poisson processes with means λ_v and λ_d, respectively. Service times for voice and data users are exponentially distributed with means $\frac{1}{\mu_v}$ and $\frac{1}{\mu_d}$, respectively. The state in the state-space $\mathcal{S}$ is denoted by a vector $s = (i, j, k)$, where i, j denote the number of voice and data users in LTE, and k denotes the number of data users in WiFi, respectively. The set of events in the event space $\mathcal{E}$ are arrival of a voice user in the system (E_1), arrival of a data user in the system (E_2), departure of an existing voice user from LTE (E_3), departure of an existing data user from LTE (E_4) and departure of an existing data user from WiFi (E_5). The action space is denoted by $\mathcal{A}$. For the purpose of readability, we restate the details of the actions in $\mathcal{A}$.

$$\mathcal{A} = \begin{cases} A_1, & \text{Block the arriving user or do nothing during departure,} \\ A_2, & \text{Accept voice/data user in LTE,} \\ A_3, & \text{Accept data user in WiFi,} \\ A_4, & \text{Accept voice user in LTE and offload one data user to WiFi,} \\ A_5, & \text{Move one data user to a RAT (from which departure has occurred).} \end{cases}$$

Action A_1 corresponds to blocking of an incoming user in the case of an arrival or doing nothing at departure. Action A_2 and A_3 correspond to accepting an user in LTE and WiFi, respectively. Under action A_4, a voice user is accepted in LTE, and one data user is offloaded to WiFi. Action A_5 offloads one data user to a RAT from which a departure has just occurred. From each state s, under an action a, the system makes a transition to a different state s' with a positive probability $p_{ss'}(a)$. Description of transition probabilities has been provided in Table 3.1 of Chapter 3. The reward and cost functions per unit time for state s and action a are denoted by $r(s, a)$ and $c(s, a)$, respectively. The descriptions of reward rates and cost rates in state $s = (i, j, k)$ for different events and actions have been provided in Table 3.2 and Equation (3.1) of Chapter 3, respectively.

4.1.2 Problem Formulation & Solution Technique

As already stated in Chapter 3, our objective is to determine an association policy which maximizes the total throughput of the system, subject to a constraint on the blocking probability of voice users. As described in Section 3.2 of Chapter 3, this problem has been formulated as a constrained continuous time MDP. The optimal policy is a mixture of two pure policies with associated probabilities [73]. The optimality equation has been described in Equation (3.5) of Chapter 3. Equations (3.6) and (3.7) of Chapter 3 have described how the optimality equation can be solved using a combination of RVIA and gradient descent approaches.

4.2 Reinforcement Learning

As discussed in the previous section, the optimal policy can be obtained by employing RVIA, provided the transition probabilities $p_{ss'}(a)$ are known beforehand. The knowledge of transition probability in turn requires the knowledge of the parameters of the arrival processes of voice and data users. In practice, these parameters may be difficult to obtain. RL based techniques are good candidates in such scenarios. Typically, RL based techniques learn which action to perform by trial-and-error and hence can work in a model-free manner. We choose Q-learning [45] for its simplicity and popularity. However, since the model considered in this chapter is a constrained continuous time MDP model involving average reward, traditional Q-learning algorithm needs some modifications. We describe the necessary modifications in the subsequent section.

4.2.1 Proposed Online Algorithm

The system moves through different states based on the arrival/departure of users and various actions taken in different states. Let the Q-value, the expected long-term average reward associated with a state s and an action $\pi(s)$ as specified by the policy π, be denoted by $Q_\pi(s, \pi(s))$. The objective is to determine an optimal policy π^*, which maximizes the

Q-value associated with a state, as described in the equation below.

$$\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q_\pi(s,a), \quad \forall s, \pi. \tag{4.1}$$

The theory of SA [78] enables us to remove the expectation operation in Equation (3.6) of Chapter 3 and achieve optimality in policy by doing averaging over time. Let $g(n)$ be an update sequence which possesses the following properties,

$$\sum_{n=1}^{\infty} g(n) = \infty; \sum_{n=1}^{\infty} (g(n))^2 < \infty. \tag{4.2}$$

Let $h(n)$ be another update sequence which possesses the same properties as described in Equation (4.2) along with the additional properties described below,

$$\sum_{n=1}^{\infty} (g(n) + h(n))^2 < \infty; \lim_{n\to\infty} \frac{h(n)}{g(n)} \to 0. \tag{4.3}$$

The key idea is to update the Q-value associated with one state-action pair at a time and keep the other values unchanged. This scheme translates into the following equation.

$$\begin{aligned} &Q_{n+1}(s,a) = (1-g(n))Q_n(s,a) + g(n)[r(s,a) - \beta c(s,a) + \max_{a' \in \mathcal{A}} Q_n(s',a') \\ &- Q_n(s^*,a^*)t(s,a,s')]; \\ &Q_{n+1}(\tilde{s},\tilde{a}) = Q_n(\tilde{s},\tilde{a}), \quad \forall (\tilde{s},\tilde{a}) \neq (s,a), \end{aligned} \tag{4.4}$$

where $t(s,a,s')$ denotes the random transition time to move from state s to s' under the action a, and (s^*,a^*) is a fixed state-action pair. However, this scheme works for a fixed value of LM β. To obtain the optimal value of β, β is to be iterated along the timescale $h(n)$, as specified below.

$$\beta_{n+1} = \Lambda[\beta_n + h(n)(B_n - B_{\max})], \tag{4.5}$$

where the projection operator Λ ensures that the value of LM remains bounded in the interval $[0, L]$ for a large $L > 0$. The assumptions on $g(n)$ and $h(n)$ as specified in Equations (4.2) and (4.3) guarantee that the two variables are updated on two different timescales. In addition, the update of LM is done on a slower timescale than the update of Q-value of state-action pair. From the slower LM timescale perspective, $Q(s,a)$ appears to be converged to optimality corresponding to the current LM value. From the faster timescale perspective, LM appears to be almost fixed. $g(n)$ and $h(n)$ are two different learning rates which specify how much importance is to be given to the old Q-value over the current reward.

At every decision epoch, i.e., upon every arrival or departure of users, the centralized controller (as shown in Fig. 3.1 of Chapter 3) chooses an action. If the network receives a high reward by selecting an action, it may prefer to *exploit* that action in future decision epochs. However, the network needs to *explore* other actions as well with time in order to observe whether they result in significant amounts of rewards. The aim of the proposed algorithm is to exploit the actions with high reward with a sufficient number of explorations. In this chapter, we adopt the ϵ-greedy [45] approach for exploration and exploitation. At every decision epoch, if the network is in state s, the proposed algorithm explores with a probability $\epsilon(s)$ and exploits the action having the highest Q-value with a probability $(1 - \epsilon(s))$. In the exploration phase, all the feasible actions in state s are chosen with equal probabilities, and exploration is gradually reduced over time.

The proposed two timescale Q-learning algorithm is described in Algorithm 3. As

Algorithm 3 Constrained SMDP based two timescale Q-learning algorithm

```
1: Initialize number of iterations $k \leftarrow 1$, Q-values of state-action pairs $Q(s,a) \leftarrow 0, \quad \forall s \in \mathcal{S}, a \in \mathcal{A}$ and
   LM $\beta \leftarrow 0$.
2: while TRUE do
3:     Determine the system state $s$.
4:     if exploration phase then
5:         Choose one of the feasible actions at random.
6:     else
7:         Choose action $a = \arg\max_{a} Q(s,a)$.
8:     end if
9:     Observe reward $r(s,a;\beta) = r(s,a) - \beta c(s,a)$.
10:    Go to next state $s'$.
11:    Observe transition time to next state $t(s,a,s')$.
12:    Update $Q(s,a)$ according to Equation (4.4).
13:    Update the LM according to Equation (4.5).
14:    Update $s \leftarrow s'$ and $k \leftarrow k+1$.
15: end while
```

described in the algorithm, Q-values associated with different state-action pairs, the LM and the number of iterations are initialized at the beginning. Based on a random event (arrival or departure), the system state is initialized. When the system is in state s, it chooses exploration and exploitation with finite probabilities. In the exploration phase, a random action is selected, while in the exploitation phase, the system chooses the action with the highest Q-value. Based on the observed reward in that state and the transition time to next state s', $Q(s,a)$ is updated along with the LM. This process is thus continued for all decision epochs. Simulation results demonstrate that this algorithm indeed converges in a reasonable number of iterations, provided the number of visits to each state-action pair is sufficiently large, and the learning rates are slowly reduced to zero, as specified in Equation (4.2).

However, due to the associated exploration mechanism, Q-learning algorithm may take a large number of iterations to converge. Furthermore, Q-learning algorithm needs to store the value functions associated with every state-action pair, thus possessing a significant storage complexity. To address these issues, in this the next section, we propose a PDS learning algorithm which speeds up the learning process by removing the action exploration. This approach is based on a reformulation of the RVIA optimality equation and can be implemented online in the SA framework. Furthermore, the storage complexity of the PDS learning algorithm is lower than that of Q-learning.

Remark 7. *If the assumptions on service times for voice and data users are relaxed, i.e., these quantities can have any general distribution, then the problem converts into a semi-Markov decision process problem. In this case also, a Q-learning based approach can be devised. If the sojourn times in different states are bounded, then a scheme similar to Equation (4.4) where the observed transition time is multiplied to the current Q-value of a specified state-action pair, can be chosen. Note that the continuous time MDP considered in this chapter is a special case of semi-Markov decision process where the service times are exponentially distributed.*

4.3 Post-Decision State Framework

In the next section, we propose a PDS learning algorithm which can operate without the knowledge of the statistics of the arrival processes of voice and data users and can still converge to the optimal solution. However, before describing the online algorithm, we introduce the notion of the PDS in this section.

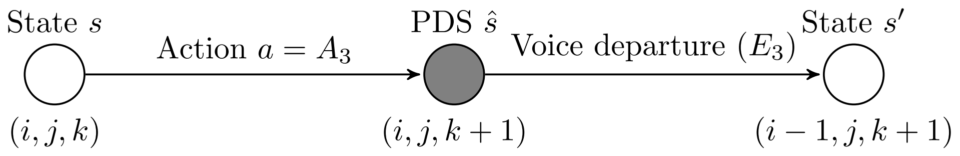


Figure 4.1: Transition among PDSs and pre-decision states.

A PDS is defined to be an imaginary state of the system just after an action is chosen and before the unknown system dynamics (noise) adds into the system. The idea behind PDS is to factor the transition from one state to another into known and unknown components. The known component consists of the effect of the action taken in a state, whereas the unknown component comprises the unknown random dynamics of the system (viz., the arrival and departure of voice and data users). Let us assume that the state of the system is $s = (i, j, k) \in \mathcal{S}$ at some decision epoch. Based on the chosen action, the system moves to the post-decision state $\hat{s} = (\hat{i}, \hat{j}, \hat{k}) \in \mathcal{S}$. Based on the next event, the system moves to the actual pre-decision state $s' = (i', j', k') \in \mathcal{S}$. Throughout this chapter, whenever we refer to a "state", we always refer to a pre-decision state. An example transition involving pre-decision states and PDSs is illustrated in Fig. 4.1. Under action A_3, the system makes a transition from state $s = (i, j, k)$ to PDS $\hat{s} = (i, j, k + 1)$. Under the next event E_3, the system moves from the PDS $\hat{s}$ to the pre-decision state $s' = (i - 1, j, k + 1)$. In other words, the known information regarding the transition from state s to s' is incorporated in PDS $\hat{s}$. On the other hand, transition from PDS $\hat{s}$ to state s' consists only of the unknown system dynamics which is not included in the PDS. Let $\hat{V}(\hat{s})$ be the value function associated with the PDS $\hat{s} \in \mathcal{S}$. Thus, we have,

$$\hat{V}(\hat{s}) = \mathbb{E}_{s'}[V(s')],$$

where the expectation $\mathbb{E}_{s'}$ is taken over all the pre-decision states which are reachable from the post-decision state $\hat{s}$. Let the transition probability from PDS $\hat{s}$ to pre-decision state s' be denoted by $p(\hat{s}, s')$. The post-decision Bellman equation for the post-decision state $\hat{s} = (\hat{x}, \hat{y}, \hat{z}) \in \mathcal{S}$ is

$$\hat{V}(\hat{s}) = \sum_{s'} p(\hat{s}, s') \max_{a}[r(s', a; \beta) + \hat{V}(\hat{s'})] - \rho, \tag{4.6}$$

where $\hat{s'}$ is the post-decision state when action a in chosen in pre-decision state s'. Using

Equation (4.6), the RVIA based update rule is as follows.

$$\begin{aligned} \hat{V}_{n+1}(\hat{s}) =& \sum_{s'} p(\hat{s}, s') \max_a [r(s', a; \beta) + \hat{V}_n(\hat{s'})] - \hat{V}_n(\hat{s^*}), \\ \hat{V}_{n+1}(\hat{s''}) =& \hat{V}_n(\hat{s''}) \quad \forall \hat{s''} \neq \hat{s}, \end{aligned} \tag{4.7}$$

where $\hat{s}$ is the PDS associated with the n^{th} decision epoch, and $\hat{s^*}$ is a fixed PDS. The idea is to update one component at a time and keep the others unchanged. This idea is translated into an online algorithm which updates the value function of the PDS associated with the current decision epoch.

4.4 Online RAT Selection Algorithm

The system changes states based on different events, i.e., the arrival/departure of users and various actions chosen in different states. Since we do not know the statistics of arrival rates of voice and data users and the max operator occurs outside the averaging operation with respect to the transition probabilities of the underlying Markov chain in Equation (3.6) of Chapter 3, online implementation of the same is not feasible. However, in Equation (4.7), the expectation operation which resides outside the max operation can be replaced by averaging over time (which is same as the corresponding empirical measure) to estimate the optimal value function of the PDSs. Using the theory of SA [78], we can remove the expectation operation in Equation (4.7) and still converge to the optimality in policy by doing averaging over time.

The key idea is to update the value function associated with one PDS at a time and keep the other PDS values unchanged. Let Y_n be the PDS which is updated at n^{th} iteration. Also, define $\gamma(\hat{s}, n) = \sum_{m=0}^{n} I\{\hat{s} = Y_n\}$, i.e., number of times PDS $\hat{s}$ is updated till n^{th} iteration. The scheme is as follows.

$$\begin{aligned} \hat{V}_{n+1}(\hat{s}) = (1 - g(\gamma(\hat{s}, n)))\hat{V}_n(\hat{s}) + g(\gamma(\hat{s}, n))\{\max_a [r(s', a; \beta) + \hat{V}_n(\hat{s'})] - \hat{V}_n(\hat{s^*})\}, \\ \hat{V}_{n+1}(\hat{s''}) = \hat{V}_n(\hat{s''}) \quad \forall \hat{s''} \neq \hat{s}. \end{aligned} \tag{4.8}$$

However, the scheme (4.8) is a primal RVIA algorithm which solves a dynamic programming equation for a fixed value of LM β. To obtain optimality in β, β is to be iterated along the timescale $h(n)$, as described below.

$$\beta_{n+1} = \Lambda[\beta_n + h(n)(B_n - B_{\max})]. \tag{4.9}$$

Therefore, the primal-dual RVIA can be described as follows.

If the system is at PDS $\hat{s}$ at the n^{th} iteration, then do the following.

$$\begin{aligned}\hat{V}_{n+1}(\hat{s}) &= (1-g(\gamma(\hat{s},n)))\hat{V}_n(\hat{s}) + g(\gamma(\hat{s},n))\{\max_a[r(s',a;\beta) + \hat{V}_n(\hat{s'})] - \hat{V}_n(\hat{s^*})\}, \\ \hat{V}_{n+1}(\hat{s''}) &= \hat{V}_n(\hat{s''}) \quad \forall \hat{s''} \neq \hat{s},\end{aligned} \tag{4.10}$$

$$\beta_{n+1} = \Lambda[\beta_n + h(n)(B_n - B_{\max})]. \tag{4.11}$$

The assumptions on $g(n)$ and $h(n)$ (Equations (4.2) and (4.3)) ensure that two quantities are updated on two different timescales. The value of LM is updated on a slower timescale than that of the value function. From the slower LM timescale point of view, $\hat{V}(\hat{s})$ appears to be equilibrated in accordance with the current LM value, and from the faster timescale view, LM appears to be almost constant. This two-timescale scheme induces a "leader-follower" behavior. The slow (fast) timescale iterate does not interfere in the convergence of the fast (slow) timescale iterate.

Theorem 1. *The schemes (4.10)-(4.11) converge to $(\hat{V}, \beta^*)$ almost surely (a.s.).*

Proof. The proof is presented in Appendix A.3. □

Based on the analysis presented above, the two timescale PDS online learning algorithm is described in Algorithm 4. As described in the algorithm, value functions

Algorithm 4 PDS learning algorithm

```
1: Initialize number of iterations $k \leftarrow 1$, value function vector $\hat{V}(\hat{s}) \leftarrow 0, \quad \forall \hat{s} \in \mathcal{S}$ and the LM $\beta \leftarrow 0$.
2: while TRUE do
3:     Determine the event (arrival/departure) in the current decision epoch.
4:     Choose action $a$ which maximizes the R.H.S expression in Equation (4.8).
5:     Update the value function of PDS $\hat{s}$ using (4.8).
6:     Update the LM according to Equation (4.9).
7:     Update $\hat{s} \leftarrow \hat{s'}$ and $k \leftarrow k + 1$.
8: end while
```

associated with different states, the LM and the number of iterations are initialized at the beginning. Based on a random event (arrival or departure of voice/data user), the system state is initialized. When the current PDS of the system is $\hat{s}$, the system chooses an action which maximizes the R.H.S expression in Equation (4.8). Based on the observed reward in the current PDS $\hat{s'}$, $\hat{V}(\hat{s})$ is updated along with the LM. This process is repeated for every decision epoch.

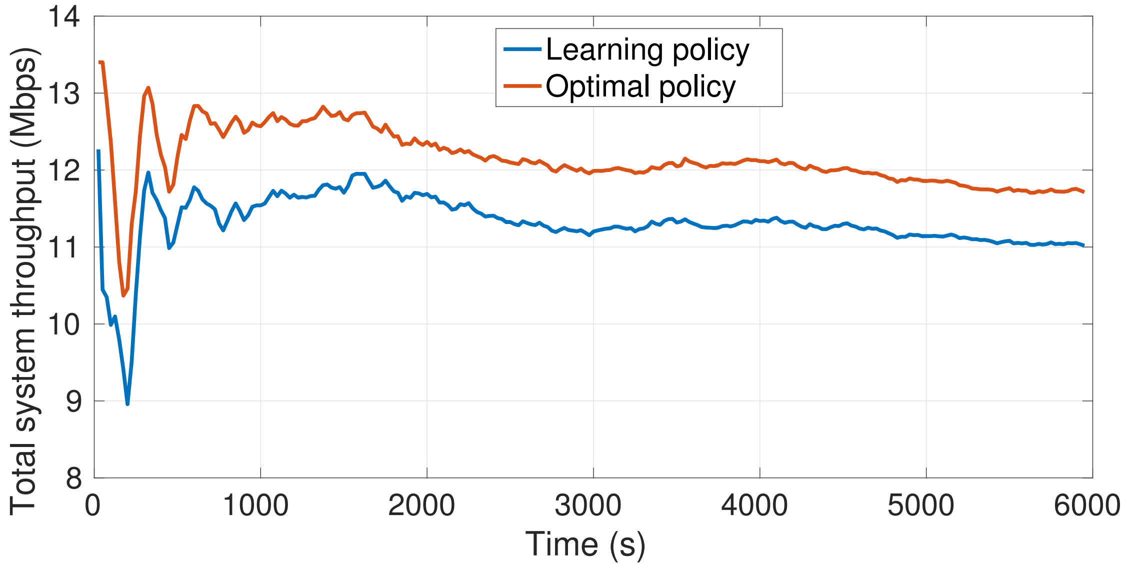
Remark 8. *Note that the proposed Q-learning algorithm stores the value functions for every state-action pair, i.e., $|\mathcal{S}| \times |\mathcal{A}|$ values and updates the value function of one state-action pair at a time. The PDS learning algorithm (see Equation (4.10)) requires storing $|\mathcal{S}|$ PDS value functions and feasible actions in every state, i.e., $|\mathcal{S}|$ values. Clearly, there is a significant reduction in storage complexity from $O(|\mathcal{S}| \times |\mathcal{A}|)$ to $O(|\mathcal{S}|)$.*

4.5 Simulation Results

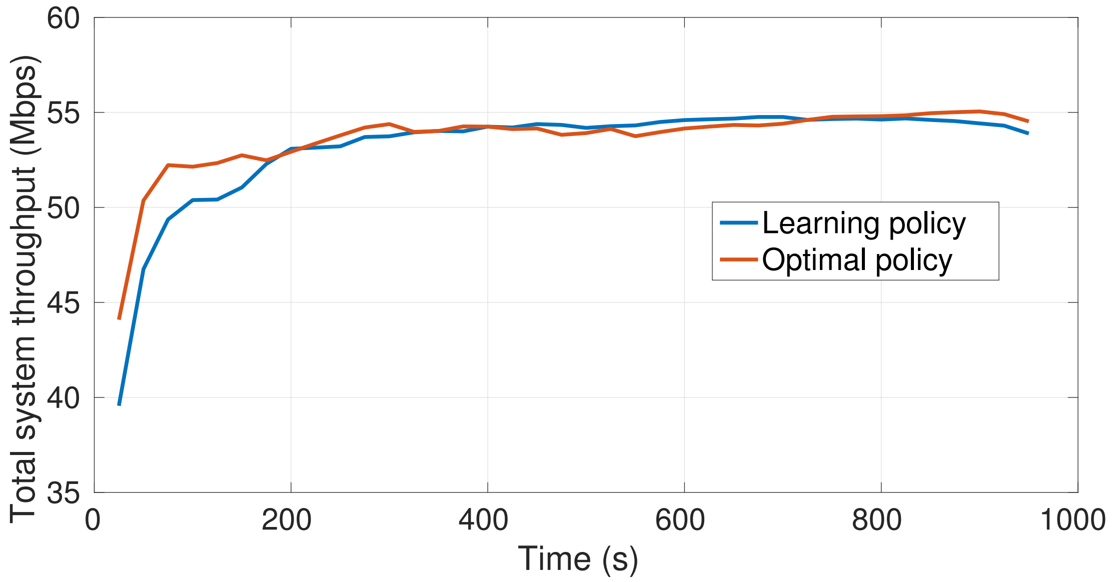
In this section, the proposed Q-learning and PDS learning algorithms are simulated in ns-3 to observe the performance. Optimal policy obtained by solving the continuous time MDP problem using RVIA, when the statistics of the arrival rates of voice and data users are known, is also simulated in ns-3. It is observed that the Q-learning algorithm converges to the optimal policy as the time progresses. Moreover, the performance of Q-learning algorithm in terms of total system throughput is compared to the performance of the optimal policy. We also provide a performance comparison of Q-learning algorithm with PDS learning algorithm in terms of convergence speed to optimality.

4.5.1 Simulation Model and Evaluation Procedure

The simulation setup considered in this chapter is identical to the setup considered in Section 3.5 of Chapter 3. The network model is simulated with a 3GPP LTE BS and an IEEE 802.11g [13] WiFi AP. All users are assumed to be stationary. The AP is trusted from the point of view of interworking and approximately 50 m away from the LTE BS. Data users are assumed to be distributed uniformly within 30 m radius of the WiFi AP. LTE and WiFi network parameters used in our simulation are based on 3GPP [76]- [77] models and saturation throughput [72] of IEEE 802.11g WiFi model, respectively. They have been summarized in Table 3.3 and 3.4 of Chapter 3. Although the system model does not have any assumption on the scheduling strategy, for simulation purposes, we consider proportional fair scheduling for the LTE BS. For the Q-value and LM updates, we consider $g(n) = \frac{1}{n^{0.6}}$ and $h(n) = \frac{1}{n}$.



(a) $\lambda_v = \lambda_d = 0.5s^{-1}$ and $\mu_v = \mu_d = 1s^{-1}$.



(b) $\lambda_v = \lambda_d = 1.0s^{-1}$ and $\mu_v = \mu_d = 0.1s^{-1}$.

Figure 4.2: Comparison of performance of the optimal policy and the Q-learning based policy.

4.5.2 Convergence Analysis of Q-learning Algorithm

Fig.(4.2a) and (4.2b) illustrate how the proposed Q-learning algorithm converges to the optimal policy in terms of the total system throughput. It is evident that as the time progresses, the total throughput of the system gradually becomes closer to the total throughput under the optimal policy. However, in Fig.(4.2a), since the users are served at rates which are faster than the corresponding arrival rates ($\lambda_v < \mu_v$ and $\lambda_d < \mu_d$) and the simulations start with an initially empty system, the average number of voice and data users in the system at any point in time is less. As a result, all states in the state space are not visited often. Even after a sufficient number of iterations, occasionally some new states are visited, and the algorithm starts to learn the optimal policy in those states by trial-and-error. Therefore, even after a sufficient amount of time, the total throughput provided by the Q-learning algorithm is marginally lower (around 0.6 Mbps) than that of the optimal policy. Nevertheless, the performance offered by the Q-learning algorithm is very close to the optimal one in the long run.

On the contrary, in case of $\lambda_v = \lambda_d = 1.0s^{-1}$ and $\mu_v = \mu_d = 0.1s^{-1}$ (see Fig.(4.2b)), since the arrival rates are higher than the service rates, the probability of visiting all states in the state space is relatively higher. Hence, after a sufficient amount of time, almost all the state-action pairs are visited reasonable number of times, resulting in the convergence to the optimal policy. Fig.(4.2b) demonstrates that indeed optimality in total system throughput is achieved after almost $200s$. Except this small interval at the beginning of the simulation, in most of the states, the action with the highest Q-value matches with the action governed by the optimal policy.

In Fig.4.3, we demonstrate how the value of LM converges with the number of iterations under different values of $\lambda_v, \lambda_d, \mu_v$ and μ_d. Upon every arrival and departure, the value of LM is updated as specified in Algorithm 3, depending on the cost incurred. It is observed that for the two cases that we consider, the LM converges approximately after 500 and 1500 iterations, respectively. However, the actual time taken to achieve convergence depends on the chosen values of arrival and service rates. Due to similar reasons as observed in Fig.(4.2a) and (4.2b), the convergence behavior in the second case is better than that of the first case. In the first case, since the users are served at rates which are faster than the corresponding arrival rates, even after a sufficient number of

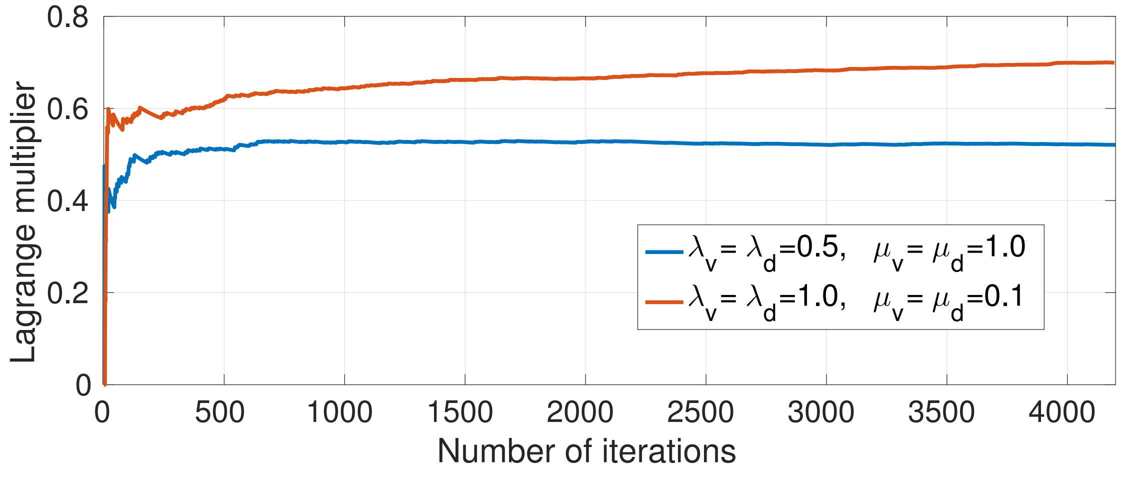
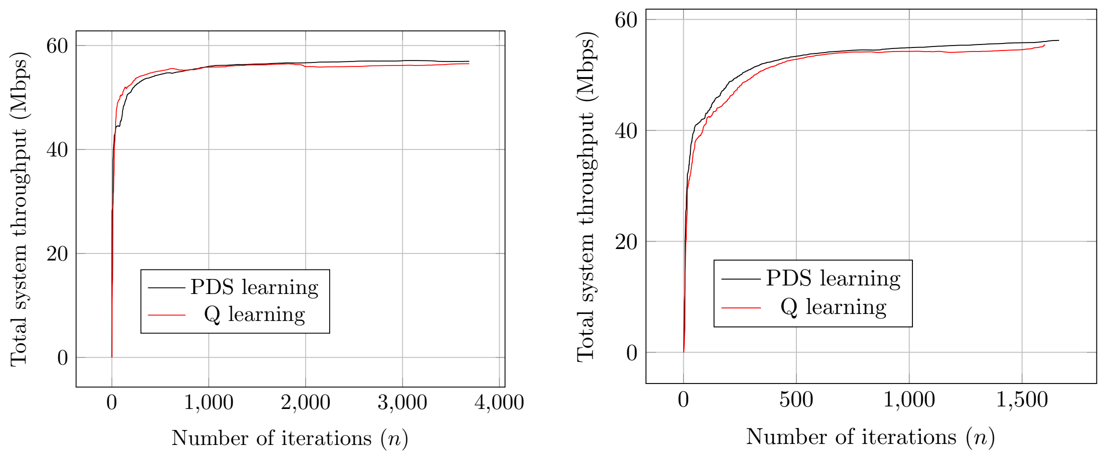


Figure 4.3: Convergence of Lagrange multiplier.



(a) ($\mu_v = \mu_d = 1/20$).

(b) ($\mu_v = \mu_d = 1/30$).

Figure 4.4: Plot of total system throughput vs. number of iterations (n) for different algorithms.

iterations, occasionally some new states are visited, and the algorithm starts to learn the optimal policy in those states by trial-and-error.

4.5.3 Performance Analysis of PDS learning Algorithm

In this subsection, we analyze the performance of PDS learning algorithm and compare the performance with that of Q-learning. For the update of the PDS value functions and LM, we consider $g(n) = \frac{1}{(\lfloor \frac{n}{1000} \rfloor + 2)^{0.6}}$ and $h(n) = \frac{10}{n}$.

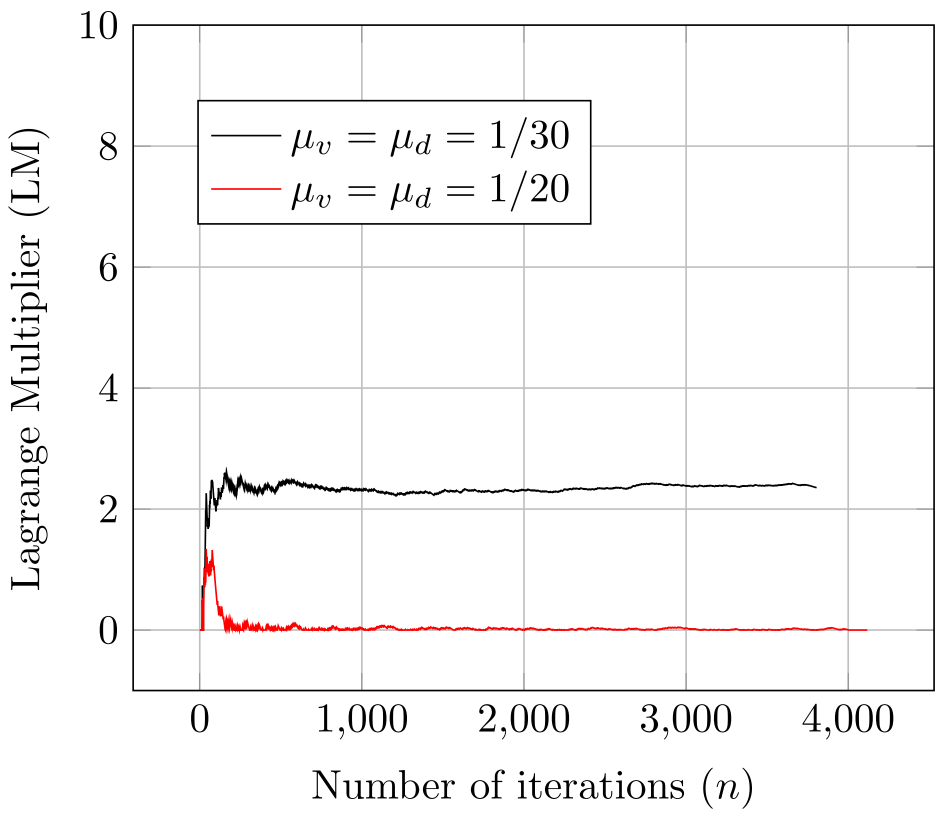


Figure 4.5: Plot of LM vs. number of iterations (n) for PDS learning algorithms.

Fig.(4.4a) and (4.4b) illustrate how the Q-learning and PDS learning algorithms converge with increasing number of iterations (n). We keep $\lambda_v = \lambda_d = 1$. It is evident that PDS learning algorithm outperforms Q-learning in terms of convergence speed. Contrary to PDS learning, even after a considerable amount of iterations, Q-learning explores different actions with finite probabilities, thereby reducing its convergence speed. Fig.4.5 depicts the convergence of LM as n increases. It is evident that as the number of iteration increases, LM converges.

4.6 Conclusions

In this chapter, we have considered the optimal RAT selection problem in an LTE-WiFi HetNet where the statistics of arrival processes of voice and data users are not known. We have proposed an online learning algorithm based on the paradigm of Q-learning. The proposed algorithm does not require the knowledge of the statistics of arrival processes of data and voice users and hence fits well for an online implementation under the paradigm of SA. To address the problem of slow convergence and high storage complexity associated with Q-learning, we have also proposed a PDS learning algorithm which can be implemented online without the knowledge of the statistics of the arrival processes. It has been proved that the PDS learning algorithm converges to the optimal policy. Simulation results have demonstrated that both the algorithms indeed converge to optimality after a reasonable number of iterations. Moreover, it has been observed that the convergence speed of PDS learning algorithm is better than that of Q-learning.

However, both the proposed algorithms do not exploit the optimality of threshold policies derived in Chapter 3. If this result can be exploited in the learning framework, intuitively, faster convergence may be achieved. This is due to the fact that the policies which are not threshold in nature, can be eliminated from the search space while learning. Furthermore, the storage and computational complexity associated with the learning process may also be reduced. In the next chapter, we consider the problem of exploitation of the knowledge regarding the structure of the optimal policy in the learning framework to achieve faster convergence to optimality.

Chapter 5

Structure-aware Learning for RAT Selection

In this chapter, we consider an extension of the online learning framework developed in Chapter 4. The algorithms proposed in Chapter 4 can operate without the knowledge of the arrival processes of voice and data users and hence, can be implemented online. The issue of slow convergence and high storage complexity associated with Q-learning has been addressed in the PDS learning framework. However, the convergence speed, the storage complexity and the computational complexity can be further improved if we exploit the knowledge of the threshold structure of the optimal policy in the learning process. In this chapter, we propose an online learning algorithm for RAT selection which exploits the known threshold structure of the optimal policy to further reduce the computational complexity and is one of the first attempts to do so.

We have shown in Chapter 3 that the optimal policy has a threshold structure, wherein after a certain threshold on the number of WiFi data users, data users are served using LTE. A similar property exists for the admission of voice users, where after a certain threshold on the number of LTE data and voice users, voice users are blocked. In this chapter, we exploit the threshold properties in Chapter 3 and propose a structure-aware learning algorithm which, instead of the entire policy space, searches the optimal policy only from the set of threshold policies. This reduces the convergence time as well as the computational and the storage complexities in comparison to Q-learning and PDS learning algorithms proposed in the last chapter. We prove that the threshold vector iterates in the proposed structure-aware learning algorithm indeed converge to the globally optimal solution. Note that the analytical methodologies presented in this chapter to learn the optimal threshold policy are developed independently and can be applied to any learning problem where the optimal policy is threshold in nature. We present a generalized framework of the proposed algorithm in Appendix B.

Although we make some simplifying assumptions to facilitate the analysis, performance of the proposed scheme is studied in realistic conditions without the simplifying assumptions. Extensive simulations are conducted in ns-3 [44] to characterize the performance of the proposed algorithm and compare the performance with those of Q-learning and PDS learning algorithms proposed in Chapter 4. Furthermore, we observe that the proposed algorithm outperforms other benchmark algorithms under realistic network scenarios like channel fading, dynamic resource scheduling and user mobility.

The rest of the chapter is organized as follows. Section 5.1 describes the system model and the problem formulation. Section 5.2 proposes a structure-aware learning algorithm for RAT selection in an LTE-WiFi HetNet along with the proof of convergence. A comparison of computational and storage complexities of the proposed algorithm with those of Q-learning and PDS learning algorithms proposed in Chapter 4 is provided in Section 5.3. Simulation results are presented in Section 5.4, followed by conclusions in Section 5.5.

5.1 System Model & Problem Formulation

The system model and problem formulation are analogous to those of Chapters 3 and 4. We have considered a system where an LTE BS and a WiFi AP are present. Voice and data user arrivals follow Poisson processes with means λ_v and λ_d, respectively. Service times for voice and data users are exponentially distributed with means $\frac{1}{\mu_v}$ and $\frac{1}{\mu_d}$, respectively. Details of the state space, action space, state transitions, reward and cost are described in Section 3.1 of Chapter 3.

The problem formulation is similar to the problem formulation described in Section 3.2 of Chapter 3. We aim to determine an association policy which maximizes the total system throughput subject to a constraint on the blocking probability of voice users.

The problem has been formulated as a CMDP. We obtain the optimal policy for the CMDP problem using the approach of obtaining the equivalent discrete-time MDP and Lagrangian method, as described in Section 3.2 of Chapter 3.

5.2 Structure-aware Online RAT selection Algorithm

The optimal policy can be obtained by employing RVIA, provided the transition probabilities $p_{ss'}(a)$ associated with the state-action pairs are known beforehand. The knowledge of transition probability in turn requires the knowledge of the statistics of the arrival processes of voice and data users. These parameters may be difficult to obtain in reality. In Chapter 4, we have proposed Q-learning and PDS learning algorithms which learn the optimal policy by trial and error.

In this section, we propose a learning algorithm exploiting the threshold properties of the optimal policy. The Q-learning and PDS learning algorithms proposed in Chapter 4 do not take into account the threshold nature of the optimal policy (derived in Chapter 3) and hence optimizes over the entire policy space. However, utilizing the threshold nature of the optimal policy, the feasible policy space can be reduced significantly. To this end, we propose a structure-aware online learning algorithm which searches the optimal policy only from the set of threshold policies, providing faster convergence than Q-learning and PDS learning algorithms.

5.2.1 Gradient Based Online Algorithm

Before proceeding, we revisit the symbols which will be used in this section. As defined in Chapter 3, i denotes the number of voice users in LTE. j and k denote the number of data users in LTE and WiFi, respectively. Let the throughput increment in WiFi when the number of WiFi users increases from k to $(k+1)$ be denoted by $\tilde{R}_{W,D}(k)$. Therefore, $\tilde{R}_{W,D}(k) = (k+1)R_{W,D}(k+1) - kR_{W,D}(k)$. We assume the following.

Assumption 2. *$\tilde{R}_{W,D}(k)$ is a non-increasing function of k. This assumption is in line with the full buffer traffic model [72].*

Summary of the structural properties of the optimal policy is as follows. Detailed proofs of the structural properties can be found in Chapter 3.

1. Upto a threshold on the number of WiFi data users (say k_{th}), serve data users in WiFi (A_3) and then serve them using LTE (A_2) until LTE is full. When LTE is full, i.e., $(i+j) = C$, the optimal policy is to serve all data users using WiFi until $k = W$, where an incoming data user is blocked.

2. For every state (i, j, k) such that $k < k_{th}(k \geq k_{th})$ and (i, j) satisfies $(i+j) < C$, upon a voice user arrival, $A_4(A_2)$ is better than $A_2(A_4)$.

3. For every state (i, j, k) such that (i, j) satisfies $(i+j) < C$, upon a voice user arrival, if the optimal action in state (i, j, k) is blocking, then the optimal action in state $(i+1, j, k)$ is also blocking.

4. For every state (i, j, k) such that (i, j) satisfies $(i+j) = C$, upon a voice user arrival, if the optimal action in state (i, j, k) is blocking, then the optimal action in state $(i+1, j-1, k)$ is also blocking.

Using the first two properties, we can eliminate a number of suboptimal actions. In the case of data user arrival (event E_2) and departure of voice and data users (events E_3, E_4 and E_5), a single decision is involved. This may provide improved convergence because contrary to an online algorithm without any knowledge of the structural property, we no longer need to learn optimal actions in some states. The only event where multiple decisions are involved is the voice user arrival (event E_1). As stated in Properties 3 and 4, the value of the threshold on i, where the optimal action changes to blocking, is a function of j and k. Thus, if we have the knowledge of the values of thresholds, we can characterize the policy completely. The idea is to optimize over the threshold vector (say θ) using an update rule, so that the value of the threshold vector θ converges to the optimal value. Before proceeding further, we determine the dimension of θ using the succeeding analysis.

Using Properties 1 and 2, we can identify three regions.

1. $0 \leq k < k_{th}$: Using Property 1, we have $j = 0$. For each value of k, we need to know the value of the threshold which belongs to the set $\{0, 1, \dots, C\}$.

2. $k = k_{th}$: Using Property 1, $k = k_{th} \implies j \geq 0$. Thus, it boils down to computing a single threshold which belongs to the set $\{0, 1, \dots, C - j\}$ (Property 3), for each

value of j $(0 \leq j < C)$. Also, we need to compute a single threshold for $(i+j) = C$ (Property 4).

3. $W > k > k_{th}$: Using Property 1, $k > k_{th} \implies (i+j) = C$. Thus, using Property 4, we need to obtain the threshold of blocking for $(W - k_{th} - 1)$ values of k.

Therefore the dimension of $\theta = (k_{th} + C + W - k_{th})$=$(C+W)$.

Thus, $\theta(0)$ corresponds to the value of threshold when $k = 0$. $\theta(0)$ belongs to the set $\{0, 1, \ldots, C\}$. Similarly, $\theta(1)$ corresponds to the value of threshold when $k = 1$ and so on.

Remark 9. *When the state space becomes too large, then it becomes cumbersome to represent a policy since this requires tabulating actions corresponding to each state. Due to the threshold nature of the optimal policy, the representation using the threshold vector becomes computationally efficient. Instead of storing the optimal action corresponding to each state, we just need to store $(C+W)$ individual thresholds.*

As assumed in Chapter 4, $g(n)$ and $h(n)$ are positive step-size sequences possessing the properties described in Equations (4.2) and (4.3) of Chapter 4.

We consider a class of threshold policies which can be described in terms of the threshold vector θ. The main idea behind the online algorithm is to compute the gradient of the system metric, i.e., the average reward of the system, with respect to θ and improve the policy by updating the value of θ in the direction of the gradient. Therefore, following [74], one needs to compute the gradient of the system metric. To express the dependence of the parameters associated with the underlying Markov chain on θ explicitly, we need to redefine the notations. Let the transition probability associated with the Markov chain $\{X_n\}$ as a function of θ be given by

$$P_{ss'}(\theta) = P(X_{n+1} = s' | X_n = s, \theta).$$

Note that a given θ corresponds to a fixed threshold policy. In this case, the underlying system dynamics becomes a Markov chain. The transition probabilities of the Markov chain for a given θ are parameterized by the equation described above.

Assumption 3. *We assume that for every $s, s' \in \mathcal{S}$, $P_{ss'}(\theta)$ is a bounded, twice differentiable function, and the first and second derivative of $P_{ss'}(\theta)$ is bounded.*

Let the average reward of the Markov chain, steady state stationary probability of state s, value function of state s (as a function of θ) be denoted by $\rho(\theta)$, $\pi(s,\theta)$ and $V(s,\theta)$, respectively. The following proposition provides a closed-form expression for the gradient of the average reward of the system. A proof for the same can be found in [74]. Although [74] considers a generalized case where the reward function depends on θ, in our case the same proof holds with the exception that the gradient of the reward function is zero.

Proposition 1. *Under assumptions on $P_{ss'}(\theta)$ as stated before, we have,*

$$\nabla\rho(\theta) = \sum_{s\in\mathcal{S}} \pi(s,\theta) \sum_{s'\in\mathcal{S}} \nabla P_{ss'}(\theta) V(s',\theta). \tag{5.1}$$

Hence, we can compute the value of $\nabla\rho(\theta)$ (or $\nabla P_{ss'}(\theta)$) to construct an incremental scheme similar to a stochastic gradient algorithm for the threshold values, of the form

$$\theta_{n+1} = \theta_n + h(n)\nabla\rho(\theta_n), \tag{5.2}$$

where θ_n represents the value of threshold vector in n^{th} iteration on the slower timescale $h(n)$. Given a threshold θ, we assume that the state transition in state $s = (i,j,k)$ is given by $P_0(s'|s)$, if $i < \theta(T)$ and $P_1(s'|s)$, otherwise, where $\theta(T)$ denotes the component of θ which corresponds to state s. Specifically,

$$T = \begin{cases} k+j, & (i+j) \neq C, \\ C+k, & (i+j) = C. \end{cases} \tag{5.3}$$

As an example, if the state of the system is $s = (i,0,0)$, then since $k = 0$, we have $T = 0$. Hence, we need to update the value of $\theta(0)$. Therefore, according to the two-timescale gradient based learning framework, on the faster timescale, we have,

$$\begin{aligned} V_{n+1}(s,\theta) &= (1-g(\gamma(s,n)))V_n(s,\theta) + g(\gamma(s,n))[r(s,a;\beta) + V_n(s',\theta) - V_n(s^*,\theta)], \\ V_{n+1}(s'',\theta) &= \hat{V}_n(s'',\theta), \quad \forall s'' \neq s. \end{aligned} \tag{5.4}$$

For example, if the current state is $s = (i,0,0)$ and $(i < \theta_n(0))$, then state transition is determined by $P_0(s'|s)$ (accept in LTE (A_2)), i.e., $s' = (i+1,0,0)$, else, s' is determined by $P_1(s'|s)$ (blocking (A_1)), i.e., $s' = (i,0,0)$. However, value functions corresponding to other states are kept unchanged.

Note that, the above scheme works for a fixed value of the threshold vector θ and the LM β. To obtain the optimal value of θ, θ is to be iterated along the slower timescale $h(n)$. Note that, although individual components of the threshold take discrete values, we interpolate them to the continuous domain to be able to apply the online update rule. Since the threshold policy is a step function (governed by $P_0(s'|s)$ up to a threshold and $P_1(s'|s)$, thereafter) defined at discrete points, Assumption 3 is not satisfied at every point. Therefore we approximate the threshold policy in state s by a randomized policy which is a function of θ ($f(s, \theta)$, say). We define

$$P_{ss'}(\theta) \approx P_1(s'|s)f(s,\theta) + P_0(s'|s)(1 - f(s,\theta)),$$

where $f(s, \theta(T)) = \dfrac{e^{(i-\theta(T)-0.5)}}{1 + e^{(i-\theta(T)-0.5)}}$ in state $s = (i, j, k)$, provides a convenient approximation to the step function.

Remark 10. *The rationale behind the choice of this function is the fact that it is continuously differentiable, and the derivative is nonzero everywhere.*

Remark 11. *Another choice of $f(s, \theta(T))$ could be the following.*

$$f(s, \theta(T)) = 0.I\{i \leq \theta(T)\} + 1.I\{i \geq \theta(T) + 1\} + (i - \theta(T)).I\{\theta(T) < i < \theta(T) + 1\}.$$

This function uses approximation only in the interval $(\theta(T), \theta(T) + 1)$. Therefore, the approximation error in this case is lesser than that of the sigmoid function considered earlier. However, the convergence behavior may be worse because the derivative of the function is nonzero only in the interval $(\theta(T), \theta(T) + 1)$. Therefore, if the initial guesses of the iterates are outside this interval, gradient becomes zero and the scheme fails to converge. Therefore, better approximation may be flatter away from the threshold and hence may spoil the convergence behavior. However, in general, this may not hold, and one needs to analyze the convergence behavior in a case-by-case manner.

While designing an online update scheme for θ, instead of $\nabla\rho(\theta_n)$ (See Equation (5.2)), we can evaluate $\nabla P_{ss'}(\theta)$. The steady-state stationary probabilities inside the summation inside Equation (5.1) can be omitted by performing averaging over time. We have,

$$\nabla P_{ss'}(\theta) = (P_1(s'|s) - P_0(s'|s))\nabla f(s, \theta). \tag{5.5}$$

In the right hand side of Equation (5.5), we incorporate a multiplication factor of $\frac{1}{2}$ since multiplication by a constant term does not alter the online scheme. The physical significance of this operation is that at any iteration, we have state transitions according to $P_0(.|.)$ and $P_1(.|.)$ with equal probabilities. This operation is done to associate the notion of a randomized policy with Equation (5.5). The update of θ in the slower timescale $h(n)$ is as follows.

$$\begin{aligned} \theta_{n+1}(T) &= \Delta_T[\theta_n(T) + h(n)\nabla f(s, \theta_n(T))(-1)^{\alpha_n} V_n(s', \theta_n)], \\ \theta_{n+1}(T') &= \theta_n(T') \quad \forall T' \neq T, \end{aligned} \tag{5.6}$$

where α_n is a random variable which takes values 0 and 1 with equal probabilities. When it takes the value 0, then s' is determined by $P_1(s'|s)$, otherwise by $P_0(s'|s)$. The averaging property of SA then leads to the effective drift (5.5). Depending on the state visited, the T th component of the vector θ is updated as shown in Equation (5.6). For example, if the current state is $(1, 0, 0)$, then $\theta_n(0)$ is updated (see Equation (5.3)), and other components are kept unchanged. The projection operator Δ_T is a function which ensures that the iterates remain bounded in the interval $[0, M(T)]$, where

$$M(T) = \begin{cases} C - (T - k_{th}), & \text{if } (k_{th} + C) \geq T \geq k_{th}, \\ C, & \text{else.} \end{cases} \tag{5.7}$$

Therefore,

$$\Delta_T(x) = \begin{cases} x, & \text{if } 0 \geq x \geq M(T), \\ 0, & \text{if } 0 < x, \\ M(T), & \text{else.} \end{cases}$$

Similar to Algorithm 4, to obtain the optimal value of β, β is to be iterated along the same timescale $h(n)$, as specified below,

$$\beta_{n+1} = \Lambda[\beta_n + h(n)(B_n - B_{\max})], \tag{5.8}$$

where the definition of Λ is provided alongside Equation (4.5). For the ease of the reader, we restate the definition as follows:

$$\Lambda(x) = \begin{cases} x, & \text{if } 0 \geq x \geq L, \\ 0, & \text{if } 0 < x, \\ L, & \text{else.} \end{cases}$$

Remark 12. *The dynamics of the LM and the threshold vector are not dependent on each other directly. However, both β and θ iterates depend on the value functions in the faster timescale. Therefore θ is updated in the same timescale as that of β, without requiring a third timescale.*

Lemma 7. *$\hat{V}_N(i+1,j,k) - \hat{V}_N(i,j,k)$ is non-increasing in N.*

Proof. The proof is provided in Appendix A.4. □

Lemma 8. *If (i,j) satisfies $(i+j) = C$, $\hat{V}_N(i+1,j-1,k+1) - \hat{V}_N(i,j,k)$ is non-increasing in N.*

Proof. The proof is provided in Appendix A.5. □

Lemma 9. *$\rho(\theta)$ is unimodal in θ, where $\theta(T) \in [0, M(T)]$ for $T = 0, \dots, (C+W-1)$.*

Proof. The proof is provided in Appendix A.6. □

Theorem 2. *The iterates in Equations (5.4), (5.6) and (5.8) converge to optimal policy a.s.*

Proof. The proof is provided in Appendix A.7. □

Note that the convergence result holds for the approximated system under the assumption that the derivative of the approximation function is non-zero everywhere, as stated before. Based on the analysis described above, the structure-aware online learning algorithm is stated in Algorithm 5. As described in the algorithm, value functions asso-

Algorithm 5 Structure-aware learning algorithm

```
1: Initialize number of iterations $k \leftarrow 1$, value function $V(s) \leftarrow 0$, $\forall s \in \mathcal{S}$, the LM $\beta \leftarrow 0$ and the
   threshold vector $\theta \leftarrow 0$.
2: while TRUE do
3:     Choose action $a$ given by the current value of threshold vector $\theta$.
4:     Update the value function of states $s$ using Equation (5.4).
5:     Update the LM according to Equation (5.8).
6:     Update the threshold vector according to Equation (5.6).
7:     Update $s \leftarrow s'$ and $k \leftarrow k+1$.
8: end while
```

ciated with different states, the LM, the threshold vector and the number of iterations are initialized at the beginning. When the current state of the system is s, the system chooses the action which is given by the current value of the threshold vector. Based on the observed reward, $V(s)$ and θ are updated along with the LM. This process is repeated for every decision epoch.

5.3 Comparison of Complexities of Learning Algorithms

In this section, we provide a comparison of storage and computational complexities of Q-learning and PDS learning algorithms proposed in Chapter 4 and structure-aware learning algorithm proposed in this chapter. We summarize the storage and computational complexities of these schemes in Table 5.1.

As discussed in Remark 8 of Chapter 4, the storage complexities of Q-learning and PDS learning are $O(|\mathcal{S}| \times |\mathcal{A}|)$ and $O(|\mathcal{S}|)$, respectively. Q-learning updates the value function of one state-action pair at a time. While updating the value function, Q-learning evaluates $|\mathcal{A}|$ functions. While updating the PDS value function, PDS learning algorithm also evaluates $|\mathcal{A}|$ functions, resulting in a per-iteration complexity of $O(|\mathcal{A}|)$.

In the case of structure-aware learning algorithm, we no longer need to store $|\mathcal{S}|$ value functions. Rather, by virtue of the threshold nature of optimal policy, we consider three cases.

1. $0 \leq k < k_{th}$: Since we have $j = 0$, for each value of k, we need to store $(C + 1)$ value functions.

2. $k = k_{th}$: $k = k_{th} \implies j \geq 0$. Thus, we need to store $(C + 1 - j)$ value functions, for each value of j $(0 \leq j \leq C)$.

3. $W \geq k > k_{th}$: $k > k_{th} \implies (i+j) = C$. Therefore, we need to store value functions of $(C + 1)$ states for each value of k.

Therefore, the total number of value functions which need to be stored is $(C + 1)k_{th} + \frac{(C+1)(C+2)}{2} + (C + 1)(W - k_{th})$, which is equal to $\frac{(C+1)(C+2)}{2} + (C + 1)W$. Note that, this is a considerable reduction in storage complexity in comparison to the PDS learning scheme having a storage complexity of $O(C^2W)$. For example, when $W = C$, the storage complexity reduces from $O(C^3)$ to $O(C^2)$. Furthermore, feasible actions corresponding to each state need not be stored separately since the threshold vector completely characterizes the policy. The per-iteration computational complexity of this scheme (see Equation (5.4)) is $O(1)$. This scheme also involves updating a single component of the threshold vector (Equation (5.6)) with a computational complexity of $O(1)$.

Table 5.1: Computational and storage complexities of different algorithms.

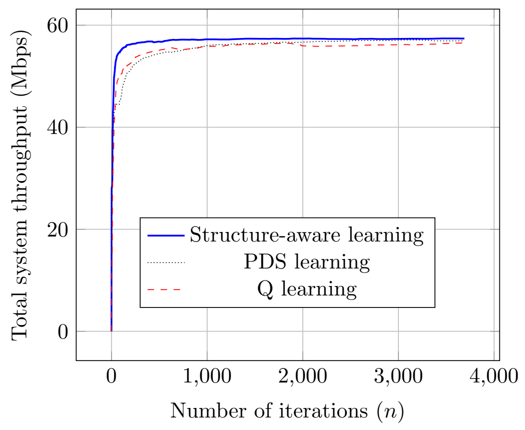
Algorithm	**Storage complexity**	**Computational complexity**
Q-learning	$O(\|\mathcal{S}\| \times \|\mathcal{A}\|)$	$O(\|\mathcal{A}\|)$
PDS learning	$O(\|\mathcal{S}\|) = O(C^2W)$	$O(\|\mathcal{A}\|)$
Structure-aware learning	$O(C^2 + CW)$	$O(1)$

5.4 Simulation Results

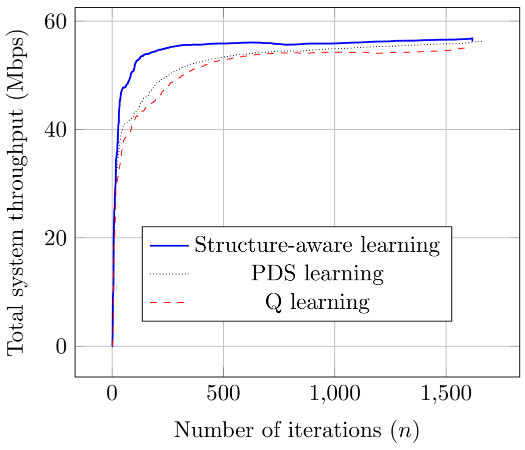
In this section, the structure-aware learning algorithm is simulated in ns-3 to characterize the convergence behavior. Convergence rate of the proposed algorithm is compared with those of PDS learning and Q-learning algorithms, as proposed in Chapter 4. Simulation results establish that the proposed PDS learning algorithm provides improved convergence than Q-learning. Furthermore, it is observed that the knowledge of the structural properties further reduces the convergence time. Therefore, the convergence rate of the proposed structure-aware learning algorithm is the fastest among the considered algorithms.

5.4.1 Simulation Model and Evaluation Methodology

The simulation setup considered in this chapter is identical to the setup considered in Section 3.5 of Chapter 3. The simulation setup comprises a 3GPP LTE BS and an IEEE 802.11g [13] WiFi AP. Data users are distributed uniformly within 30 m radius of the WiFi AP which is approximately 50 m away from the LTE BS. LTE and WiFi network parameters used in our simulation have been summarized in Table 3.3 and 3.4 of Chapter



(a) ($\mu_v = \mu_d = 1/20$).



(b) ($\mu_v = \mu_d = 1/30$).

Figure 5.1: Plot of total system throughput vs. number of iterations (n) for different algorithms.

3. For the update of the PDS value functions, the threshold vector and the LM, we consider $g(n) = \frac{1}{(\lfloor \frac{n}{1000} \rfloor + 2)^{0.6}}$ and $h(n) = \frac{10}{n}$.

5.4.2 Convergence Analysis

Fig.(5.1a) and (5.1b) illustrate how the Q-learning, PDS learning and structure-aware learning algorithms converge with increasing number of iterations (n). We keep $\lambda_v = \lambda_d = 1$.

It is evident that both PDS learning and structure-aware learning algorithms outperform Q-learning algorithm in terms of convergence speed. Contrary to PDS learning, even after a considerable amount of iterations, Q-learning explores different actions with finite probabilities. This reduces the convergence speed of Q-learning algorithm. The knowledge of structural properties reduces the feasible policy space. Therefore, the structure-aware learning algorithm offers the fastest convergence to the optimal policy. Contrary to PDS learning and Q-learning algorithms, we no longer need to learn the optimal actions in a subset of states, where the optimal policy is determined using structural properties. As observed in Fig.(5.1a) and (5.1b), the number of iterations before convergence reduces from 2000 (for PDS learning algorithm) to 300 (for structure-aware learning algorithm) and 1000 (for PDS learning) to 300 (for structure-aware learning algorithm), respectively.

Fig. 5.2 depicts the convergence of the LM as n increases, in the case of structure-

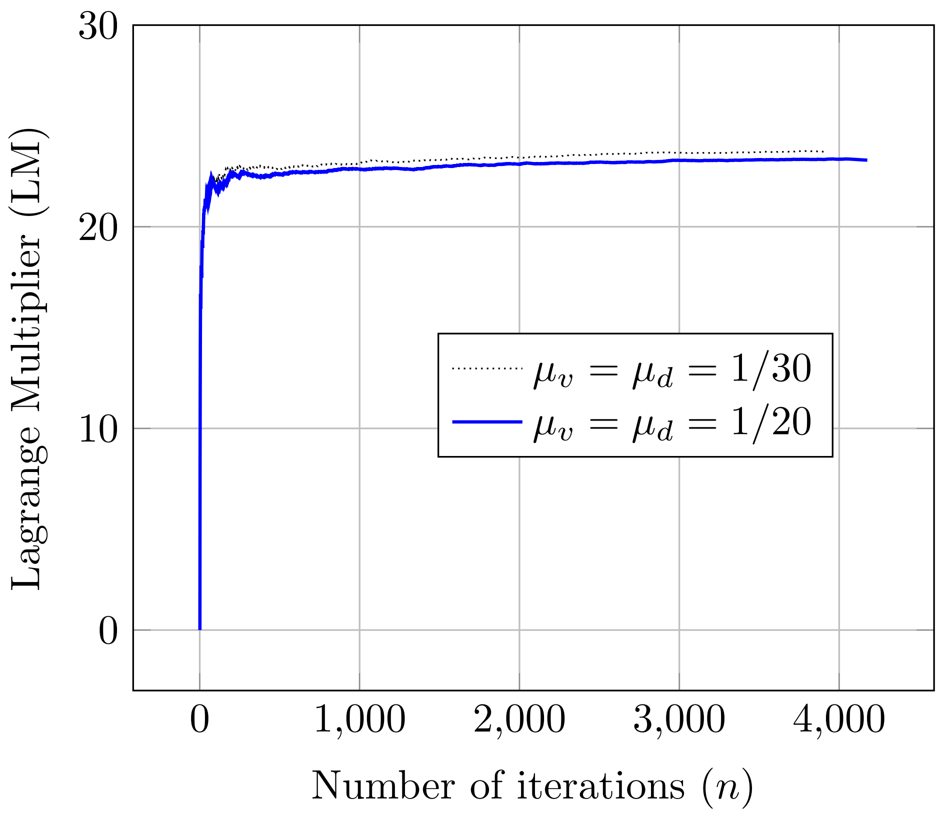


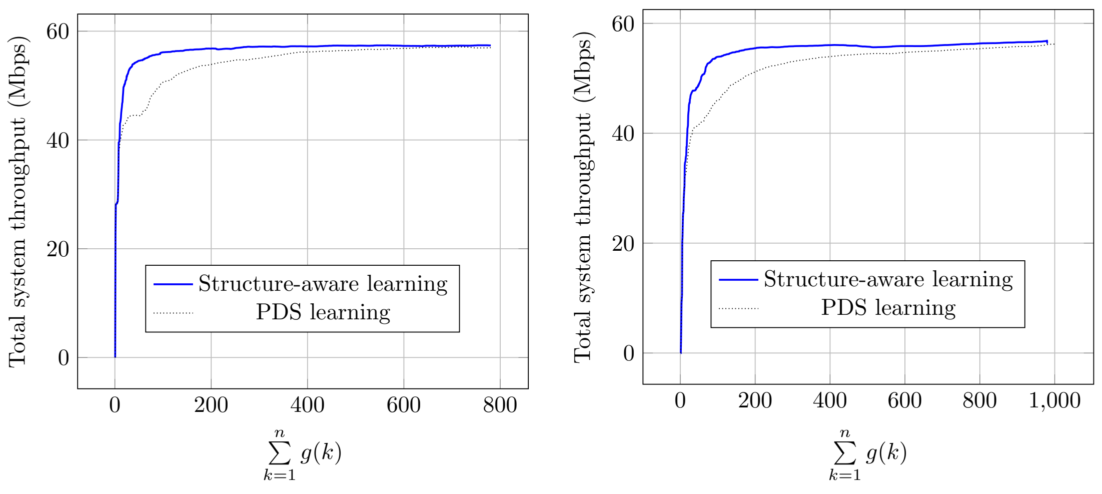
Figure 5.2: Plot of LM vs. number of iterations (n) for structure-aware learning algorithm.

aware learning. It is evident that as the number of iteration increases, the LM converges.

5.4.3 Stopping Criteria

While simulating an online learning algorithm, in practical cases, we may not want to wait till the actual convergence. Rather, we can observe the total system throughput over a moving window of the sums of step sizes till the present iteration ($\sum\limits_{k=1}^{n} g(k)$) and calculate the ratio of maximum and minimum values of the total system throughput over this window. We set the window size to be 100. If the ratio is more than 0.99, then we conclude that stopping criteria is reached, i.e., the obtained policy is in a close neighborhood of the optimal policy with high probability.

In Fig.(5.3a) and (5.3b), the total system throughput is plotted against $\sum\limits_{k=1}^{n} g(k)$. Note that, $\sum\limits_{k=1}^{n} g(k)$ is chosen instead of n, to decouple the effect of diminishing step size, while analyzing the convergence behavior of the schemes. In other words, this parameter is selected to establish that the convergences of the algorithms are indeed due to



(a) $(\lambda_d = \mu_v = 1,\ \mu_v = \mu_d = 1/20)$. (b) $(\lambda_d = \mu_v = 1,\ \mu_v = \mu_d = 1/30)$.

Figure 5.3: Plot of total system throughput vs. sum of step sizes till nth iteration ($\sum\limits_{k=1}^{n} g(k)$) for different algorithms.

the convergence to the optimal policy and not due to very small values of step size $g(n)$ as n becomes large. We observe in Fig. (5.3a) and (5.3b) that the PDS learning algorithm reaches the stopping criteria when $\sum\limits_{k=1}^{n} g(k)$ becomes 700, which corresponds to approximately 1000 iterations. On the other hand, the structure-aware learning algorithm reaches the stopping criteria when $\sum\limits_{k=1}^{n} g(k)$ becomes 300 which translates into almost 450 iterations.

5.4.4 Consideration of Realistic Scenarios

While the above simulation results provide significant insights into the convergence behavior of the proposed algorithm over traditional Q-learning and PDS learning algorithms, in this section, we evaluate the performance of the proposed algorithm in realistic scenarios. We compare the total system throughput and the voice user blocking probability performance of the proposed algorithm with those of PDS learning algorithm, on-the-spot offloading [57] and LTE-preferred schemes.

Although in the system model (see Section 5.1) we consider single resource block allocation to LTE data users, in simulations we relax this assumption and consider proportional fair scheduling for the LTE BS which dynamically assigns resources to the users based on user bandwidth demands. Users randomly generate individual bandwidth demands. However, we assume that the maximum data rate achievable for a single data user is 5 Mbps, and the bottleneck is in the access network. Furthermore, in the previous subsections, there is no consideration of channel fading effects in LTE and WiFi. To address that, whenever we choose an action involving offloading of a user from one RAT to another (A_4 and A_5), the user with the worst channel is selected for offloading. For example, whenever A_4 is chosen and we offload a data user from LTE to WiFi, we always choose the data user with the lowest SNR. Since, in general, a user with bad channel provides bad throughput to the system, the user with the worst channel is chosen for offloading. We consider Extended Pedestrian A model [79] for fading in LTE and Rayleigh fading for WiFi.

In on-the-spot offloading [57], data users always choose WiFi unless WiFi coverage is not present. Therefore, in our system model, on-the-spot offloading always associates data users with WiFi until capacity is reached in WiFi. Voice users are associated with LTE, and when LTE reaches its capacity, voice users are blocked. In LTE-preferred scheme, voice and data users are associated with LTE until LTE reaches its capacity. When LTE reaches its capacity and a voice user arrives, the voice user is blocked if there is no data user in LTE. Otherwise, one existing data user is offloaded to WiFi if capacity is available in WiFi. Upon the departure of an existing voice or data user from LTE, an existing data user in WiFi, if any, is offloaded to LTE. While offloading, we always choose the user with the worst channel.

5.4.4.1 Voice User Arrival Rate Variation

Fig. (5.4a) depicts the blocking percentage of voice users for on-the-spot offloading, LTE-preferred, PDS learning and structure-aware learning algorithms for varying λ_v. Since on-the-spot offloading blocks voice users when LTE reaches its capacity, blocking probability of voice users increases with λ_v. Since PDS learning and structure-aware learning algorithms learn in which states blocking is to be chosen as the optimal action, voice user blocking probabilities corresponding to these algorithms converge to the same value. Since the both the algorithms may block voice users even when the LTE system does not reach its capacity, the blocking probability values are marginally higher than that of on-the-spot offloading. Voice users may be blocked to save LTE resources for future data

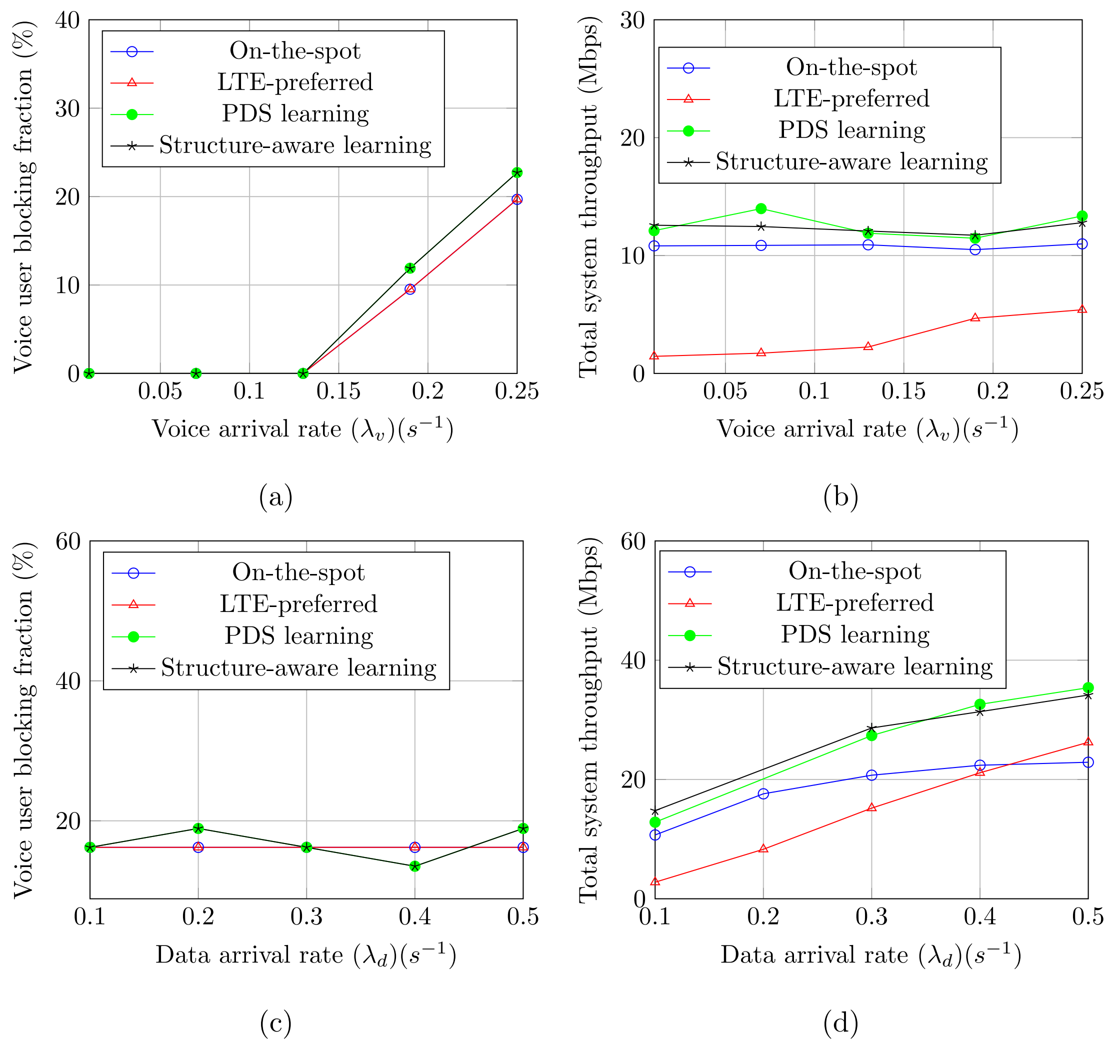


Figure 5.4: Plot of voice user blocking fraction and total system throughput for different algorithms. (a) Voice user blocking percentage vs. λ_v, (b)Total system throughput vs. λ_v ($\lambda_d = 1/20, \mu_v = 1/60$ and $\mu_d = 1/10$). (c) Voice user blocking percentage vs. λ_d, (d)Total system throughput vs. λ_d ($\lambda_v = 1/6, \mu_v = 1/60$ and $\mu_d = 1/10$).

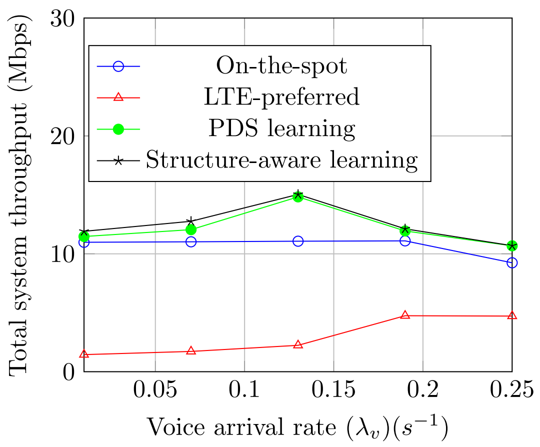
user arrivals which have a higher throughput contribution to the system. LTE-preferred scheme blocks a voice user when LTE system is full and there is no data user in LTE. Therefore, on-the-spot offloading and LTE-preferred schemes provide similar blocking probability performance.

Fig. (5.4b) illustrates the total system throughput performance of different algorithms under varying λ_v. With an increase in λ_v, the average number of voice users in the system increases while the number of WiFi data users remains the same. Therefore, in the case of on-the-spot offloading, the total system throughput increases with λ_v. However, since the throughput of voice users is small compared to data users, the rate of increment is very small. PDS learning and structure-aware learning algorithms learn the optimal policy which does significant load balancing via actions A_4 and A_5. Also, while offloading, they take the channel state of the users into account. Thus, these algorithms outperform on-the-spot offloading in terms of the total system throughput with performance improvements varying from 10.72% (for $\lambda_v = 0.13$) to 28.72% (for $\lambda_v = 0.07$). With increase in λ_v, LTE-preferred scheme starts offloading data users to WiFi to accommodate incoming voice users. Under low WiFi load, total throughput of the system increases. As WiFi load increases, the rate of increment decreases. However, both PDS learning and structure-aware learning algorithms perform better than LTE-preferred scheme.

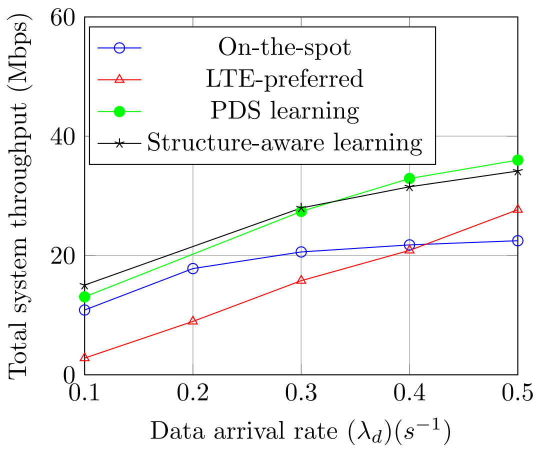
5.4.4.2 Data User Arrival Rate Variation

As observed in Fig.(5.4c), since in on-the-spot offloading, data and voice users are served using WiFi and LTE, respectively, changes in λ_d do not impact the blocking probability of voice users. Performances of both PDS learning and structure-aware learning algorithms are similar to that of on-the-spot offloading. Due to the presence of a constraint on the voice user blocking probability, most of the voice users are blocked when the LTE system reaches capacity. Therefore, PDS learning and structure-aware learning algorithms do most of the blocking of voice users in the same decision epochs as that of on-the-spot offloading. Since LTE-preferred scheme blocks voice users only when LTE does not have any available capacity and there is no data user in LTE, the blocking probabilities of LTE-preferred scheme and on-the-spot offloading are same.

Since on-the-spot offloading associates data user with WiFi, with increase in λ_d, the load in WiFi increases. As a result, as λ_d (See Fig.(5.4d)) increases, the effect of contention and channel fading reduces the rate of increment of total system throughput. Both PDS learning and structure-aware learning algorithms perform better than on-the-spot offloading by virtue of optimal RAT selection and load balancing actions which reduce the effect of contention in WiFi. Also, while offloading, PDS learning and structure-aware learning algorithms take channel state of users into account. Therefore, PDS learning and structure-aware learning algorithms outperform on-the-spot offloading with performance improvements varying from 20% (for $\lambda_d = 0.1$) to 54.6% (for $\lambda_d = 0.5$). As λ_d increases,



(a)



(b)

Figure 5.5: Plot of total system throughput for different algorithms under user mobility. (a) Total system throughput vs. λ_v ($\lambda_d = 1/20, \mu_v = 1/60$ and $\mu_d = 1/10$), (b) Total system throughput vs. λ_d ($\lambda_v = 1/6, \mu_v = 1/60$ and $\mu_d = 1/10$).

LTE-preferred scheme starts offloading more data users to WiFi. Therefore, the total system throughput increases. Under high λ_d, the effect of contention is lesser than that of on-the-spot offloading, resulting in a better performance than on-the-spot offloading. However, both PDS learning and structure-aware learning algorithms perform better than LTE-preferred scheme.

5.4.5 Consideration of User Mobility

In this section, we evaluate how PDS learning and structure-aware learning algorithms perform in comparison to on-the-spot offloading and LTE-preferred scheme in the face of user mobility. In addition to ns-3 simulation settings described in the last section, we also consider random waypoint model [80] for the mobility of voice and data users.

As evident from Fig. (5.5a) and Fig. (5.5b), although total system throughputs provided by different algorithms change due to mobility, comparative performances of PDS learning and structure-aware learning algorithms with respect to on-the-spot offloading and LTE-preferred scheme remain the same.

Since mobility does not have any impact on the blocking probability of voice users, the blocking probability performances of the considered algorithms are exactly the same as that described in Fig. (5.4a) and (5.4c).

5.5 Conclusions

In this chapter, an online learning algorithm, which exploits the threshold structure of the optimal policy has been proposed. The knowledge of the threshold structure provides improvements in computational and storage complexities and convergence time. The proposed algorithm provides a novel framework that can be applied for designing online learning algorithms for any general problem and is of independent interest. We have proved that the structure-aware learning algorithm converges to the globally optimal threshold vector. Simulation results have been presented to exhibit how the knowledge of structural properties provides improvement in convergence time over traditional online association algorithms. Simulation results establish that the proposed scheme outperforms on-the-spot offloading and LTE-preferred scheme under realistic network scenarios.

The RAT selection algorithms proposed in Chapters 3, 4 and 5 aim to maximize the total system throughput subject to a constraint on the blocking probability of voice users. However, there is no consideration of channel states of users in the system model. Channel states of users may play a vital role in RAT selection decisions. Also, the proposed schemes in Chapters 3, 4 and 5 may induce excessive offloading of data users, leading to a huge amount of backhaul control signaling. In the next chapter, we propose RAT selection schemes which address these issues.

Chapter 6

Channel and Backhaul-aware RAT Selection

In this chapter, we propose RAT selection algorithms based on extensions of the system model and the problem formulation adopted in Chapters 3, 4 and 5. In the previous three chapters, while optimizing the total system throughput subject to a constraint on the blocking probability of voice users, we do not take into account the channel states of users. In this chapter, we consider the channel states of users in our system model to investigate the role of channel conditions in RAT selection decisions.

We consider an LTE-WiFi HetNet where users of different priorities are present. We assume that high priority users (such as VoIP, live streaming) are always served using LTE since WiFi may not provide the required QoS in terms of delay and packet loss. Low priority users are best effort class of users which may be served using LTE or WiFi. When a high priority user is served using LTE, a guaranteed rate is provided to the high priority user. Therefore, high priority users are provided a fixed number of resource blocks in LTE depending on their channel conditions. However, when a low priority user is served using LTE, no rate guarantee is provided. We assume that we allocate the required number of resource blocks in LTE to high priority users for providing the guaranteed bit rate. After that, remaining resources blocks, if any, are equally distributed among the low priority users. However, the rate obtained for individual low priority user may be different depending on the channel condition of the user. A high priority user may be blocked if it is not possible to provide the required QoS using LTE. The channel condition of the high priority user may also have an impact on the association decision. For example, a high priority user with bad channel condition may be blocked, whereas a high priority user with good channel condition may be admitted in LTE. When a high priority user is admitted in LTE, an existing low priority user can be offloaded to WiFi. Moreover, departure of a user from LTE (WiFi) may trigger the offloading of a low priority user to LTE (WiFi).

As discussed in Chapter 3, maximizing the total system throughput may result in excessive blocking of high priority users since their contribution towards the system throughput is usually less than that of low priority users. Therefore, it is necessary to consider a constraint on the blocking probability of high priority users. In general, low priority users are best effort in nature, and no admission control is adopted for them. Therefore, no QoS constraint in terms of blocking probability is required for low priority users.

Maximizing the total system throughput subject to the blocking probability constraint, as considered in Chapters 3, 4 and 5, may lead to the following optimal policy. Whenever a high priority user is admitted in LTE, a low priority user may be offloaded to WiFi. However, if another high priority user departs from LTE, it may be optimal to offload one existing WiFi user to LTE. As a result, it may happen that within a short time interval, one user moves from LTE to WiFi and moves back to LTE again, thereby leading to ping-pong kind of behavior. Similar instances can occur in case of departures followed by arrivals also. This may generate additional control signaling in the backhaul. To address this, along with the high priority user blocking probability constraint, we also take into account the offloading probability (i.e., fraction of time traffic is offloaded) of low priority users as a constraint. Therefore, we target to maximize the total system throughput subject to constraints on the blocking probability of high priority users and the offloading probability of low priority users. This problem is formulated as a CMDP problem. We reduce the dimensionality of the action space by proving the suboptimality of different actions in different states of the system. However, the conventional DP methods to solve the CMDP problem are computationally expensive in the face of even moderately large state and action spaces. Furthermore, the computation of the optimal policy requires the knowledge of the transition probabilities of the underlying model. The transition probabilities are governed by the statistics of the system dynamics, viz., the arrival rates of high and low priority users, which are difficult to obtain in reality. To address these issues, in this chapter, we propose two RAT selection algorithms which have low computational complexities. Furthermore, the proposed algorithms do not require the knowledge of the statistics of the system dynamics. These features make these algorithms suitable for practical online implementation.

The rest of the chapter is organized as follows. Section 6.1 describes the system model. In Section 6.2, the problem formulation within the framework of CMDP is described. In Section 6.3, we derive the suboptimal actions and eliminate them from the action space. We describe the proposed algorithms in Section 6.4 and demonstrate a comparison of computational and storage complexities in Section 6.5. Section 6.6 concludes the chapter.

6.1 System Model

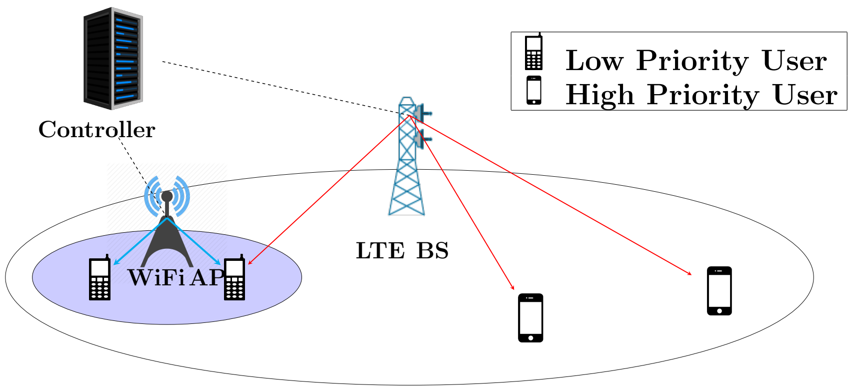


Figure 6.1: LTE-WiFi network with users of multiple priorities.

The system model considered in this chapter is similar to those of Chapters 3, 4 and 5. However, we have not considered the channel states and the priority of users in the system model in these chapters. Therefore, for the sake of completeness, we describe the system model considered in this chapter in details.

The system model consists of an LTE BS and a WiFi AP inside the coverage area of the LTE BS, as shown in Fig. 6.1. The LTE BS and the WiFi AP are connected to a centralized controller using high capacity lossless links. We assume that high and low priority users are present at any geographical point inside the coverage area of the LTE BS. Low priority users which are present outside the dual coverage area of the LTE BS and the WiFi AP, always get associated with the LTE BS. Without loss of generality, we consider only those low priority users which are present in the common coverage area of the LTE BS and the WiFi AP. Low priority users can be associated with either the LTE BS or the WiFi AP. We assume that high and low priority users are allocated resources in LTE from a common resource pool. We assume that in LTE, high and low priority users can be of either "good" or "bad" channel state. We assume that based on the location of users, the coverage area of an LTE BS is divided into two regions, viz., cell center and cell edge regions. Since cell edge users are present in the vicinity of the cell boundary, usually they receive weaker signal strength than that of the cell center users. Therefore, it is assumed that users present in the cell center region have good channels, whereas cell edge users have bad channels. Cell center/ cell edge region can be chosen based on the average Channel Quality Indicator (CQI) experienced by the users in LTE. If the average CQI of a user exceeds a certain threshold, then the user is called a cell center user. Otherwise, it is called a cell edge user. Although this formulation considers binary channel state, it can be extended easily to multiple channel state scenarios. Since RAT selection decisions are made for a sufficiently long period of time, we assume that users are distributed in cell edge/cell center region depending on their average radio conditions. We assume that instantaneous fading effects are averaged out over the timescale in which decisions are taken. We assume that the users are stationary, and the channel states do not change with time once the user is admitted. The channel states of incoming users are assumed to be known at the controller, however, the channel states are random with finite probabilities. Since the coverage area of the WiFi AP is small, we assume that the channel states of users in WiFi are always good.

We assume that high and low priority user arrivals are Poisson processes with means λ_H and λ_L, respectively. The service times for high and low priority users are exponentially distributed with means $\frac{1}{\mu_H}$ and $\frac{1}{\mu_L}$, respectively. Assumptions on service times are in accordance with [70].

Remark 13. *We have considered a single LTE BS and a single WiFi AP for brevity of notation. Nevertheless, the considered system model can be extended to multiple BSs and*

APs easily with small modifications.

6.1.1 State Space

The system can be modeled as a controlled continuous time stochastic process $\{X(t)\}_{t\geq 0}$. Any state s in the state space $\mathcal{S}$ is represented as $s = (i_G, i_B, j_G, j_B, k_G, k_B)$, where i_G, i_B denote the number of high priority users associated with the LTE BS with good and bad channels in LTE, j_G, j_B denote the number of low priority users associated with the LTE BS with good and bad channels in LTE, and k_G, k_B denote the number of low priority users associated with the WiFi AP, however, with respect to LTE they have good and bad channels, respectively. Note that we do not explicitly mention the channel states of users in WiFi since the channel states of users in WiFi are always good. The arrivals and departures of high and low priority users with good and bad channel states in LTE are taken as decision epochs. It is easy to see that the system changes state only at these decision epochs. Moreover, due to the Markovian nature of the system, it is sufficient to observe the system state at these decision epochs and not at other points in time.

Whenever there is an arrival or a departure of user, we refer it as an event. The system changes state whenever an event occurs. Let the set of all events be denoted by $\mathcal{E}$. The following events can occur.

- E_1: Arrival of high priority user with good channel,
- E_2: Arrival of low priority user with good channel,
- E_3: Arrival of high priority user with bad channel,
- E_4: Arrival of low priority user with bad channel,
- E_5: Departure of high priority user with good channel from LTE,
- E_6: Departure of high priority user with bad channel from LTE,
- E_7: Departure of low priority user with good channel from LTE,
- E_8: Departure of low priority user with bad channel from LTE,
- E_9: Departure of a low priority user from WiFi with good channel in LTE,

- E_{10}: Departure of a low priority user from WiFi with bad channel in LTE.

Note that, the channel states of users in WiFi do not come into the picture in the event space because we have assumed that the channel states of users in WiFi are always good. At every decision epoch, the controller chooses a decision based on the current system state and the event. Based on the decision chosen, the system makes a transition to a state with a finite probability.

Let the LTE system be composed of C_L resource blocks. We assume that $s = (i_G, i_B, j_G, j_B, k_G, k_B) \in \mathcal{S}$ if $(i_G + 2i_B) \leq C_L$, $(j_G + 2j_B) \leq N$ and $(k_G + k_B) < W$, where N is a sufficiently large positive integer ($N \gg C_L$). The first two conditions are based on the assumption that a user with bad channel requires twice as many resource blocks as required by a user with good channel. The first condition also signifies that the admitted high priority user is provided the required number of resource blocks, whenever resources are available. The quantity W signifies the maximum number of users that can be supported in WiFi with a specified minimum per-user throughput guarantee. Note that the per-user throughput of WiFi decreases monotonically with the number of WiFi users [72]. Since high priority users require guaranteed bit rate ($R_{L,H}$, say), a fixed number of resource blocks are allocated to high priority users based on the channel condition of the user. However, since low priority users are best-effort in nature, the remaining resources in LTE are allocated uniformly among low priority users. Therefore, the rates obtained by low priority users (which is a function of the channel state) depend on the number of high priority users in the system. We assume that the bit rate obtained by a low priority user with bad channel is $\frac{1}{d}(d > 1)$ times that of a low priority user with good channel, where d is a constant.

6.1.2 Action Space

Let the action space (set of all possible association decisions in case of arrivals and departures) be denoted by $\mathcal{A}$. The set of actions in $\mathcal{A}$ is listed below:

- A_1: Block arriving users/ do nothing at departure,
- A_2: Accept high/low priority users in LTE,
- A_3: Accept low priority users in WiFi,
- A_4: Accept a high priority user in LTE and offload a low priority user with bad channel in LTE to WiFi,
- A_5: Offload a low priority user with bad (good) channel from LTE (WiFi) to WiFi (LTE) upon the departure of a user from WiFi (LTE),
- A_6: Accept a high priority user in LTE and offload a low priority user with good channel in LTE to WiFi,
- A_7: Offload a low priority user with good (bad) channel from LTE (WiFi) to WiFi (LTE) upon the departure of a user from WiFi (LTE).

In case of high priority user arrivals, the feasible action set is $\{A_1, A_2, A_4, A_6\}$. In case of low priority user arrivals, the feasible action set is $\{A_2, A_3\}$. In case of departures, the feasible action set comprises A_1, A_5 and A_7, respectively. However, some of the actions may be infeasible in different states. Note that blocking is a feasible action for high priority users only when the system is non-empty. On the contrary, low priority users are blocked only when $(j_G + 2j_B)$ becomes N.

6.1.3 Transition Probabilities

From each state $s \in \mathcal{S}$ and under each feasible action $a \in \mathcal{S}$, the system moves to a state $s' \in \mathcal{S}$ with a positive probability $p_{ss'}(a)$. Let the sum of arrival and service rates of users in state $s = (i_G, i_B, j_G, j_B, k_G, k_B)$ be denoted by $v(i_G, i_B, j_G, j_B, k_G, k_B)$. Therefore,

$$v(i_G, i_B, j_G, j_B, k_G, k_B) = \lambda_H + \lambda_L + (i_G + i_B)\mu_H + (j_G + j_B + k_G + k_B)\mu_L.$$

Let $\hat{s} = (i'_G, i'_B, j'_G, j'_B, k'_G, k'_B)$ and $e_{\{i:1\leq i\leq 6\}}$ be a set of 6 dimensional vectors with all elements being zero except i^{th} element being '1'. Then,

$$p_{ss'}(a) = \begin{cases} \frac{\lambda_H p_g}{v(i'_G,i'_B,j'_G,j'_B,k'_G,k'_B)}, & s' = \hat{s}, \\ \frac{\lambda_H (1-p_g)}{v(i'_G,i'_B,j'_G,j'_B,k'_G,k'_B)}, & s' = \hat{s}, \\ \frac{\lambda_L p_g}{v(i'_G,i'_B,j'_G,j'_B,k'_G,k'_B)}, & s' = \hat{s}, \\ \frac{\lambda_L (1-p_g)}{v(i'_G,i'_B,j'_G,j'_B,k'_G,k'_B)}, & s' = \hat{s}, \\ \frac{i'_G \mu_H}{v(i'_G,i'_B,j'_G,j'_B,k'_G,k'_B)}, & s' = \hat{s} - e_1, \\ \frac{i'_B \mu_H}{v(i'_G,i'_B,j'_G,j'_B,k'_G,k'_B)}, & s' = \hat{s} - e_2, \\ \frac{j'_G \mu_L}{v(i'_G,i'_B,j'_G,j'_B,k'_G,k'_B)}, & s' = \hat{s} - e_3, \\ \frac{j'_B \mu_L}{v(i'_G,i'_B,j'_G,j'_B,k'_G,k'_B)}, & s' = \hat{s} - e_4, \\ \frac{k'_G \mu_L}{v(i'_G,i'_B,j'_G,j'_B,k'_G,k'_B)}, & s' = \hat{s} - e_5, \\ \frac{k'_B \mu_L}{v(i'_G,i'_B,j'_G,j'_B,k'_G,k'_B)}, & s' = \hat{s} - e_6, \end{cases}$$

where p_g denotes the probability that the channel state of the arriving user in LTE is good. Values of $i'_G, i'_B, j'_G, j'_B, k'_G, k'_B$ as a function of different actions a (conditioned on events E_l) are described in Table 6.1.

6.1.4 Rewards and Costs

Depending on the system state and the action chosen, a finite amount of reward is obtained. In WiFi, the total throughput depends on the total load of WiFi comprising low priority users with good and bad channels in LTE. As defined in Section 3.1 of Chapter 3, $R_{W,D}(k)$ denotes the per-user throughput of k users in WiFi under full buffer traffic model [72]. $R_{W,D}(k)$ is a function of success and collision probabilities which arise due to the contention-based medium access of WiFi users and slot times for idle, busy (due to collision) and successful transmissions.

Let the reward rate in state s and under action a be denoted by $r(s, a)$. The reward rate is defined as the sum of throughput of all users in LTE plus the sum of throughput

Table 6.1: Transition probability table.

$\boldsymbol{a\|E_l}$	$\boldsymbol{(i'_G, i'_B, j'_G, j'_B, k'_G, k'_B)}$
$A_1\|\mathcal{E} \cap (E_2 \cup E_4)^{\complement}$	$(i_G, i_B, j_G, j_B, k_G, k_B)$
$A_2\|E_1$	$(i_G+1, i_B, j_G, j_B, k_G, k_B)$
$A_2\|E_2$	$(i_G, i_B, j_G+1, j_B, k_G, k_B)$
$A_2\|E_3$	$(i_G, i_B+1, j_G, j_B, k_G, k_B)$
$A_2\|E_4$	$(i_G, i_B, j_G, j_B+1, k_G, k_B)$
$A_3\|E_2$	$(i_G, i_B, j_G, j_B, k_G+1, k_B)$
$A_3\|E_4$	$(i_G, i_B, j_G, j_B, k_G, k_B+1)$
$A_4\|E_1$	$(i_G+1, i_B, j_G, j_B-1, k_G, k_B+1)$
$A_4\|E_3$	$(i_G, i_B+1, j_G, j_B-1, k_G, k_B+1)$
$A_5\|(E_5 \cup \ldots \cup E_8)$	$(i_G, i_B, j_G+1, j_B, k_G-1, k_B)$
$A_5\|(E_9 \cup E_{10})$	$(i_G, i_B, j_G, j_B-1, k_G, k_B+1)$
$A_6\|E_1$	$(i_G+1, i_B, j_G-1, j_B, k_G+1, k_B)$
$A_6\|E_3$	$(i_G, i_B+1, j_G-1, j_B, k_G+1, k_B)$
$A_7\|(E_5 \cup \ldots \cup E_8)$	$(i_G, i_B, j_G, j_B+1, k_G, k_B-1)$
$A_7\|(E_9 \cup E_{10})$	$(i_G, i_B, j_G-1, j_B, k_G+1, k_B)$

of users in WiFi under an action. Let us define

$$R(i_G, i_B, j_G, j_B, k_G, k_B) = (i_G + i_B)R_{L,H} + \frac{(C_L - i_G - 2i_B)}{(j_G + j_B)} R_{L,L}(j_G + \frac{j_B}{d}) \\ + (k_G + k_B)R_{W,D}(k_G + k_B), \tag{6.1}$$

where $R_{L,L}$ is the data rate corresponding to a single resource block in LTE for low priority data users with good channel condition. The exhaustive description of reward rates in state s under different event-action pairs is provided in Table 6.2.

We consider two types of cost functions, one due to blocking and another due to offloading. Let the cost rates for blocking and offloading in state s under action a be denoted by $c_b(s,a)$ and $c_o(s,a)$, respectively. Whenever the controller blocks one high priority user, $c_b(s,a)$ is unity, else it is zero. Therefore,

$$c_b(s,a) = \begin{cases} 1, & \text{if high priority users are blocked,} \\ 0, & \text{otherwise.} \end{cases}$$

Whenever the controller offloads one low priority user from one RAT to another, $c_o(s,a)$ is unity, else it is zero. Therefore,

$$c_o(s,a) = \begin{cases} 1, & \text{if } a = (A_4 || \ldots || A_7), \\ 0, & \text{otherwise.} \end{cases}$$

6.2 Problem Formulation & Solution Techniques

As discussed before, since high priority users provide smaller contributions to the system throughput than those by low priority users, there exists an associated trade-off between the total system throughput and the blocking probability of high priority users. Furthermore, maximizing the total system throughput may result in excessive offloading of low priority users and hence, an increase in the associated control signaling in the backhaul. Therefore, we consider a constraint on the offloading probability of low priority users as well. We aim to determine a policy for the association of high and low priority users which maximizes the total system throughput subject to constraints on the blocking probability of high priority users and the offloading probability of low priority users. The considered problem can be formulated as a CMDP problem, where maximizing the total system throughput is subject to constraints on the blocking probability of high priority users and the offloading probability of low priority users. Since arrivals and departures of high and low priority users can occur at any arbitrary time, the problem is continuous time in nature. As discussed in Section 3.2 of Chapter 3, a stationary randomized optimal policy, i.e., a mixture of pure policies with corresponding probabilities, exists [73].

6.2.1 Problem Formulation

Let the set of memoryless policies be denoted by $\mathcal{M}$. We assume that the Markov chains induced by memoryless policies are unichain which guarantees that the Markov chains have unique stationary distributions. Let the average reward, the cost due to blocking of high priority users and the cost due to offloading of low priority users over infinite horizon under policy $M \in \mathcal{M}$ be denoted by V^M, $C^{B,M}$ and $C^{O,M}$, respectively. Let the total reward, the cost due to blocking and the cost due to offloading till time t be denoted by

$R(t)$, $C_B(t)$ and $C_O(t)$, respectively. The CMDP problem can be described as follows.

$$\begin{aligned} &\text{Maximize:} \quad V^M = \lim_{t\to\infty} \frac{1}{t}\mathbb{E}_M[R(t)], \\ &\text{subject to:} \quad C^{B,M} = \lim_{t\to\infty} \frac{1}{t}\mathbb{E}_M[C_B(t)] \leq B_{\max} \text{ and} \\ &C^{O,M} = \lim_{t\to\infty} \frac{1}{t}\mathbb{E}_M[C_O(t)] \leq O_{\max}, \end{aligned} \tag{6.2}$$

where $\mathbb{E}_M$ is the expectation operator under policy M, and $B_{\max}, O_{\max}$ denote constraints on the blocking probability of high priority users and the offloading probability of low priority users, respectively. Since the optimal policy is stationary, the limits in Equation (6.2) exist.

6.2.2 Conversion to Discrete-Time MDP and Lagrangian Approach

The approach adopted in this subsection is analogous to the one adopted in Chapters 3, 4 and 5. However, due to the presence of an additional constraint in this chapter, we describe the approach to capture the notational specificities associated with the additional constraint.

Optimal policy can be obtained using RVIA [69]. However, before that, we need to adopt the Lagrangian approach [73]. For fixed values of LMs β_b and β_o, the equivalent unconstrained reward function is given by

$$r(s,a;\beta_b;\beta_o) = r(s,a) - \beta_b c_b(s,a) - \beta_o c_o(s,a).$$

Using DP, the optimality equation for the considered CMDP $\forall s, s' \in \mathcal{S}$ is

$$V(s) = \max_a [r(s,a;\beta_b;\beta_o) + \sum_{s'} p_{ss'}(a)V(s') - \rho\bar{t}(s,a)],$$

where $V(s), \rho, \bar{t}(s,a)$ denote the value function of state $s \in \mathcal{S}$, the optimal average reward of the system and the mean transition time from state s under action a, respectively. Since the sojourn times are known to be exponential, this becomes a special case of continuous time controlled Markov chain for which the following equation holds,

$$0 = \max_a [r(s,a;\beta_b;\beta_o) - \rho + \sum_{s'} q(s'|s,a)V(s')], \tag{6.3}$$

where $q(s'|s,a)$ are controlled transition rates which satisfy $q(s'|s,a) \geq 0$, for $s' \neq s$ and $\sum_{s'} q(s'|s,a) = 0$. Scaling the transition rates by a positive scalar quantity is equivalent to time scaling. This scales the average reward for every policy including the optimal policy without changing the optimal policy. Therefore, without loss of generality, we assume that $-q(s|s,a) \in (0,1), \forall a$. This implies that $q(s'|s,a) \in [0,1]$ for $s' \neq s$. We add $V(s)$ to both sides of Equation (6.3) to obtain the following equation for an equivalent discrete-time MDP ($\{X_n\}$ say) with controlled transition probabilities $p_{ss'}(a)$.

$$V(s) = \max_a [r(s,a;\beta_b;\beta_o) - \rho + \sum_{s'} p_{ss'}(a)V(s')], \tag{6.4}$$

where $p_{ss'}(a) = q(s'|s,a)$ for $s' \neq s$ and $p_{ss'}(a) = 1 + q(s'|s,a)$ for $s' = s$. For the rest of the chapter, instead of the original continuous-time MDP, we focus on the equivalent discrete-time MDP in Equation (6.4).

For fixed values of β_b and β_o, RVIA can be used to solve the unconstrained maximization problem (as described in Equation (6.4)) using the following equation,

$$V_{n+1}(s) = \max_a [r(s,a;\beta_b;\beta_o) + \sum_{s'} p_{ss'}(a)V_n(s') - V_n(s^*)], \tag{6.5}$$

where $V_n(s)$ is the value function estimate of state s after n iterations, and s^* is a fixed state. We aim to obtain the optimal values for β_b and β_o, viz., ${\beta_b}^*$ and ${\beta_o}^*$, which maximize the average reward subject to cost constraints. The following equations describe gradient descent routines to update the values of β_b and β_o in k^{th} iteration.

$$\beta_{b,k+1} = \beta_{b,k} + \frac{1}{k}(B^{\pi_{\beta_{b,k}}} - B_{\max}),$$

$$\beta_{o,k+1} = \beta_{o,k} + \frac{1}{k}(O^{\pi_{\beta_{o,k}}} - O_{\max}),$$

where $\beta_{b,k}, \beta_{o,k}$ are the values of β_b and β_o in k^{th} iteration, and $B^{\pi_{\beta_{b,k}}}, O^{\pi_{\beta_{o,k}}}$ denote the high priority user blocking probability and the low priority user offloading probability in k^{th} iteration, respectively.

Note that the optimal policy for the considered CMDP is a randomized policy with randomizations in at most two states [75] where in each state, one of the two actions is chosen randomly.

6.3 Action Elimination

The DP equations (Equations (6.4) and (6.5)) are exploited to prove the suboptimality of certain actions in different states. Using this, the number of feasible actions in different states can be reduced. This fact is utilized in analyzing the computational complexities of RAT selection algorithms described in the next section. The suboptimality of different actions is established with the help of some lemmas.

6.3.1 Suboptimal Actions for Departures

The subsequent lemmas describe the suboptimality of certain actions in a subset of states. Specifically, whenever a high/low priority user departs from LTE, A_5 is better than A_7. Therefore, in this case, A_7 is a suboptimal action. Similarly, in the case of a low priority user departure from WiFi, A_5 is better than A_7. Therefore, in this case also, A_7 is a suboptimal action.

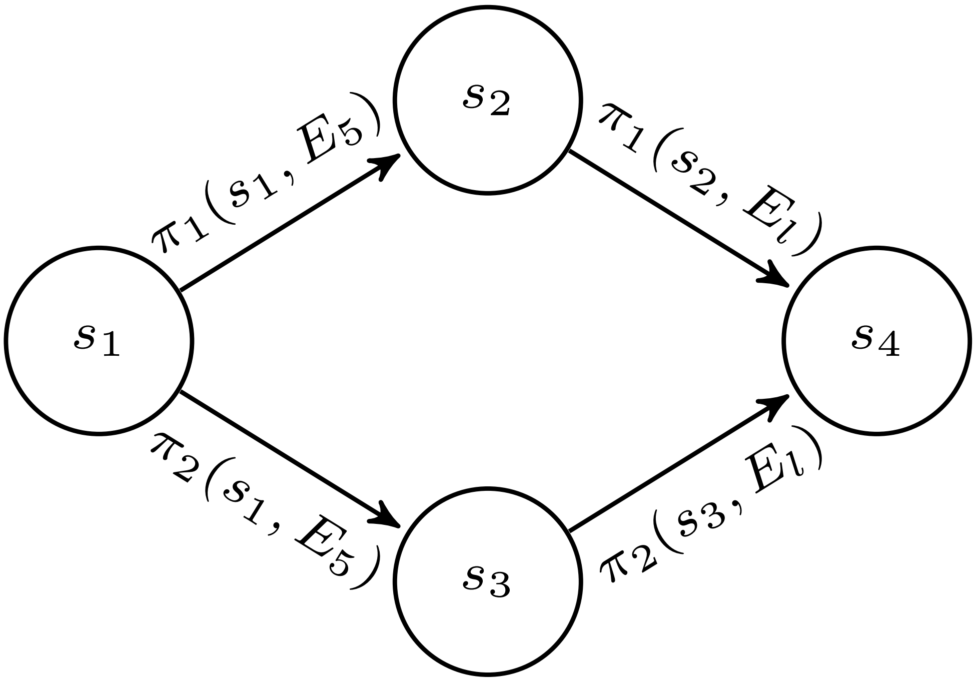


Figure 6.2: Sample paths under different policies.

Lemma 10. A_5 *is better than* A_7 *in case of high/low priority user departure from LTE (events* E_5, E_6, E_7, E_8*).*

Proof. We prove the lemma using sample path arguments. We consider the case of event E_5. Proofs for the other events follow in a similar manner. We assume that the system starts at time $t = 0$. Suppose that the system is in state $s_1 = (i_G, i_B, j_G, j_B, k_G, k_B)$, when event E_5 occurs at time t_1. Consider a policy which chooses A_7 in state $(i_G, i_B, j_G, j_B, k_G, k_B)$ and denote it by π_1. Consider another policy (which may be a non-stationary policy) π_2 which selects A_5 in state $(i_G, i_B, j_G, j_B, k_G, k_B)$. As illustrated in Fig. 6.2, let us assume that under policies π_1 and π_2, the system moves from state s_1 to state $s_2 = (i_G, i_B, j_G, j_B + 1, k_G, k_B - 1)$ and $s_3 = (i_G, i_B, j_G + 1, j_B, k_G - 1, k_B)$, respectively. Since we have considered a Markovian system, the inter-arrival times and service times are same for the considered sample paths. We assume that based on the next event E_l and following the policy π_1, the system moves from state s_2 to state s_4. Suppose the policy π_2 is such that in response to event E_l, it chooses the same action as that of π_1. Additionally, it offloads one good user from LTE to WiFi and one bad user from WiFi to LTE. Therefore, under the policy π_2, the system moves from state s_3 to s_4. We construct the policy π_2 in such a way that here onwards, it chooses the same action as that of policy π_1. Therefore, from state s_4 onwards, both the sample paths follow the same trajectory. The difference of value functions of state s_1 under policies π_1 and π_2 is given by

$$V_{\pi_1}(s_1) - V_{\pi_2}(s_1) = \frac{(C_L - i_G - 2i_B)}{(j_G + j_B + 1)} R_{L,L}(\frac{1}{d} - 1) < 0.$$

Therefore, policy π_2 is strictly better than π_1. Since the Markov chains under various policies are recurrent in nature, state s_1 is visited infinitely often. Upon each visit, action A_7 induced by policy π_1 provides less reward than that of action A_5 corresponding to policy π_2. Therefore, A_5 is better than A_7. This completes the proof of the lemma. □

Lemma 11. *A_5 is better than A_7 in case of low priority user departure from WiFi (events E_9, E_{10}).*

Proof. Similar to Lemma 10, we prove this lemma for event E_9. Proof for event E_{10} follows in a similar way. Suppose the system is in state $s_1 = (i_G, i_B, j_G, j_B, k_G, k_B)$ when event E_9 occurs at time t_1. Consider policies π_1 and π_2 which choose A_5 and A_7 in state s_1, respectively. We assume that under policies π_1 and π_2, the system moves from state s_1 to state $s_2 = (i_G, i_B, j_G, j_B - 1, k_G, k_B + 1)$ and $s_3 = (i_G, i_B, j_G - 1, j_B, k_G + 1, k_B)$, respectively. We assume that based on the next event E_l and following the policy π_1, the system moves from state s_2 to state s_4. Suppose the policy π_2 is such that in response to event E_l, it chooses the same action as that of π_1, offloads one bad user from LTE to WiFi and one good user from WiFi to LTE. Therefore, under the policy π_2, the system moves from state s_3 to s_4. We construct the policy π_2 in such a way that here onwards, it chooses the same action as that of policy π_1. Therefore, from state s_4 onwards, both the sample paths follow the same trajectory. Therefore, the difference of value functions of state s_1 under policies π_1 and π_2 is given by

$$V_{\pi_1}(s_1) - V_{\pi_2}(s_1) = \frac{(C_L - i_G - 2i_B)}{(j_G + j_B + 1)} R_{L,L}(1 - \frac{1}{d}) > 0.$$

Therefore, policy π_1 is strictly better than π_2. Due to the recurrent nature of the Markov chain, similar to Lemma 10, A_5 is better than A_7. □

The physical significance of Lemmas 10 and 11 is that whenever there is a departure of user, if we choose to offload a low priority user to the RAT from which the departure has taken place, it is always better to choose the user with good channel condition for offloading to LTE and the user with bad channel condition for offloading to WiFi. Intuitively, since a bad user degrades the throughput of all other low priority users in LTE, it is better to offload a bad user to WiFi. Since we have assumed that in WiFi, every low priority user experiences good channel conditions, offloading a user with bad channel condition in LTE to WiFi improves the total system throughput. Similar argument holds for the offloading of a user with good channel condition to LTE.

6.3.2 Suboptimal Actions for Arrivals

In this section, we characterize the suboptimal actions in the case of high priority data user arrivals. As described in the subsequent lemma, whenever there is a high priority user arrival, then irrespective of the channel condition, action A_4 is better than A_6. In other words, whenever a high priority user is accepted in LTE, and we decide to offload an existing low priority user to WiFi, it is always better to choose a user with bad channel condition rather than choosing one with good channel.

Lemma 12. *A_4 is better than A_6 in case of high priority user arrivals (events E_1, E_3).*

Proof. Proof is similar to Lemma 11. □

6.4 Proposed RAT Selection Algorithms

In Section 6.2, maximization of the total system throughput subject to constraints on the high priority user blocking probability and the low priority user offloading probability is formulated as a CMDP problem which can be solved using DP techniques. However, DP based methods suffer from the curse of dimensionality. For example, in traditional policy iteration [69], the computational complexity is $O(|\mathcal{A}|^{|\mathcal{S}|})$ which is exponential in the cardinality of the state space. Although elimination of suboptimal actions, as discussed in Section 6.3, reduces the size of the action space, the complexity still remains exponential in $|\mathcal{S}|$. Furthermore, computation of the optimal policy requires the knowledge of state transition probabilities which are governed by the statistics of the arrival processes of high and low priority users. In practice, the statistics of the arrival processes may be unknown. Although learning based approaches (as described in Chapter 4) which do not require the knowledge of the statistics of the arrival processes may be adopted, usually their convergence rates are very slow. To address these issues, we propose low-complexity algorithms which are practically implementable. Moreover, they do not require the knowledge of the statistics of the arrival processes.

6.4.1 Myopic with Constraint Satisfaction Algorithm

In this subsection, we propose an algorithm which is myopic in the sense that it only optimizes based on the current reward and does not look into the future utility. However, the proposed algorithm, called Myopic with Constraint Satisfaction Algorithm (MCSA), satisfies the associated constraints on the blocking probability of high priority users and the offloading probability of low priority users. The complete description of MCSA is provided in Algorithm 6.

As discussed in Algorithm 6, we first determine the event in the current decision epoch. Then, we determine the best action (denoted by a^*) based on the current reward (Line 4). If the current event is low priority user arrival (events E_2 and E_4), then irrespective of the channel condition of the incoming user, we always choose the action a^*. Since the feasible actions (A_2 and A_3) corresponding to a low priority user arrival affect neither the blocking probability of high priority users nor the offloading probability of low

Algorithm 6 Myopic with Constraint Satisfaction Association Algorithm.

```
Input: $R_{L,H}, R_{L,L}, R_W(.), B_{\max}, O_{\max}$.
 1: Initialize number of high priority user arrivals $A_H \leftarrow 0$, number of departures $D \leftarrow 0$, blocking
    probability of high priority users $B_H \leftarrow 0$ and offloading probability of low priority users $O_L \leftarrow 0$,
    $F_B \leftarrow 0$, $F_O \leftarrow 0$.
 2: while TRUE do
 3:     Determine the event $E$ in the current decision epoch.
 4:     Set $a^* \leftarrow \arg\max_{a \in \mathcal{A}} r(s, a)$.
 5:     if $(E = E_2 || E_4)$ then
 6:         Choose action $a = a^*$.
 7:     else if $(E = E_1 || E_3)$ then
 8:         $A_H \leftarrow A_H + 1$.
 9:         if $B_H > (B_{\max} - \epsilon_B)$ then
10:             procedure HP–CONSTRAINT–VIOLATION
11:                 If $O_L < (O_{\max} - \epsilon_O)$ choose $a = a^*$.
12:                 Else choose $a = A_2$.
13:                 $F_0 \leftarrow I_{\{a = A_4 || A_6\}}$.
14:             end procedure
15:         else
16:             Choose action $a = A_1$.
17:         end if
18:         procedure UPDATE–BP–OP
19:             $F_B \leftarrow I_{\{a = A_1\}}$.
20:             $B_H \leftarrow \frac{B_H A_H + F_B}{(A_H + 1)}$.
21:             $O_L \leftarrow \frac{O_L (A_H + D) + F_O}{(A_H + D + 1)}$.
22:         end procedure
23:     else
24:         procedure DEPARTURE–POLICY
25:             $D \leftarrow D + 1$.
26:             if $O_L < (O_{\max} - \epsilon_O)$ then
27:                 Choose action $a = a^*$.
28:             else
29:                 Choose action $a = A_1$.
30:             end if
31:             $F_0 \leftarrow I_{\{a = A_5 || A_7\}}$.
32:             $O_L \leftarrow \frac{O_L (A_H + D) + F_O}{(A_H + D + 1)}$.
33:         end procedure
34:     end if
35: end while
```

priority users, we always act in a myopic manner in the case of low priority user arrivals. However, if the current event is high priority user arrival (events E_1 and E_3), then we initially increment the counter corresponding to the number of high priority user arrivals (denoted by A_H). If the current value of blocking probability (denoted by B_H) is less than the specified constraint $B_{\max}$, then we block the arriving high priority user (Line 16). Note that, we keep a small margin ϵ_B on the blocking probability constraint $B_{\max}$ to ensure that in the long run the system operates below $B_{\max}$. However, if B_H is more than $B_{\max} - \epsilon_B$ and the current value of the offloading probability of low priority users (denote by O_L) is less than the specified constraint $O_{\max}$, then we always choose the action a^* (Line 11). If the current value of O_L violates the constraint, then A_2 is selected (Line 12) since choosing A_4 or A_6 may further increase the value of O_L. Similar to the case of $B_{\max}$, we keep a small margin ϵ_O on $O_{\max}$. Based on whether an action involving blocking (A_1) or offloading (A_4 and A_6) is chosen or not (denoted by F_B and F_O, respectively), we update the current values of B_H and O_L (Line 20 and 21). Similar procedures are followed in case of departures, where depending on the value of O_L, actions are selected (Line 25-29). Based on whether A_5 or A_7 is chosen, the value of O_L is updated (Line 32). Note that unlike DP methods, MCSA does not require the knowledge of the transition probabilities of the underlying model, and hence, the knowledge of λ_H, λ_L and p_g.

6.4.2 State-aware Myopic with Constraint Satisfaction Algorithm

In this subsection, we describe the shortcomings of the proposed MCSA and propose a State-aware Myopic with Constraint Satisfaction Algorithm (SMCSA) which addresses these shortcomings.

Whenever the current value of the blocking probability is lower than the provided constraint, the proposed MCSA blocks an incoming high priority user. Hence, when the arrival rate of high priority users is small, it may lead to unnecessary blocking of high priority users. On the other hand, the optimal policy may result in a lower value of blocking probability of high priority users than that of MCSA, depending on $B_{\max}$. In this case, the optimal policy corresponding to the unconstrained problem may result in a high priority user blocking probability which is significantly lower than $B_{\max}$. Since there is a trade-off between the high priority user blocking probability and the total system throughput, MCSA may result in higher total system throughput than that of the optimal policy. However, the blocking probability corresponding to MCSA may be higher than that of the optimal policy. Intuitively, MCSA blindly aims to satisfy the constraints of the considered problem without a consideration of the system state. Thus, MCSA always results in a high priority user blocking probability values which are close to the given constraint, irrespective of λ_H and λ_L. Due to similar reasons, MCSA results in a high value of offloading probability of low priority users which is always close to $O_{\max}$, irrespective of λ_H and λ_L.

To address this, we propose SMCSA which is described in Algorithm 7. The procedures for low priority user arrivals are same as those of Algorithm 6. In the case of high priority user arrival, when the constraints on the blocking probability and the offloading probability are not satisfied, the procedure is exactly same as that of Algorithm 6 (Line 13). However, when the constraints on the blocking probability and the offloading probability are met, we modify the RAT selection strategy in the following way. We divide the entire state space into regions based on the number of high and low priority users in the system. Let us divide the entire state space into regions denoted by $R_1, R_2, \ldots R_P$. For a given region $R_n (1 \leq n \leq P)$, let the probability of blocking and offloading be denoted by $q(n)$ and $p(n)$ $(0 \leq q(n) \leq 1, 0 \leq p(n) \leq 1)$, respectively, where $q(n)$ and $p(n)$ are increasing functions of n, and $q(P) = p(P) = 1$. Whenever an event happens, we determine the current state of the system and evaluate the region in which it falls. If it falls in R_n, we block (choose A_1) the user with probability $q(n)$ and accept (choose A_2) with probability $(1 - q(n))$ (Line 17). Similarly, if it falls in R_n and the optimal action involves offloading, we offload with probability $p(n)$ and choose the other action with probability $(1 - p(n))$ (Line 11-12). Similar procedures are followed for the departures. If the constraint on O_L is met and the optimal action involves offloading, we offload with probability $p(n)$ and choose the other action with probability $(1 - p(n))$ (Line 25-26). The procedures for the update of B_H and O_L are same as that of Algorithm 6.

The key advantage of the proposed SMCSA is that when the value of λ_H is low, we block the incoming high priority users with low probability. As λ_H increases and the system gradually fills up with high priority users, the probability of blocking increases.

Algorithm 7 State-aware Myopic with Constraint Satisfaction Association Algorithm.

```
Input: $R_{L,H}, R_{L,L}, R_W(.), B_{\max}, O_{\max}$.
 1: Initialize $A_H \leftarrow 0$, $D \leftarrow 0$, $B_H \leftarrow 0$ and $O_L \leftarrow 0$, $F_B \leftarrow 0$, $F_O \leftarrow 0$.
 2: while TRUE do
 3:     Determine the event $E$ in the current decision epoch and the region $R_n$ in which the current state
    $s$ falls.
 4:     Set $a^* \leftarrow \arg\max_{a \in \mathcal{A}} r(s,a)$.
 5:     if $(E = E_2 || E_4)$ then
 6:         Choose $a = a^*$.
 7:     else if $(E = E_1 || E_3)$ then
 8:         $A_H \leftarrow A_H + 1$.
 9:         if $B_H > (B_{\max} - \epsilon_B)$ then
10:             procedure HP–CONSTRAINT–VIOLATION–SA
11:                 If $O_L < (O_{\max} - \epsilon_O)$ and $a^* = (A_4 || A_6)$
12:                 Choose $a = a^*(A_2)$ w.p. $p(n)(1 - p(n))$.
13:                 Else choose $a = A_2$.
14:                 $F_0 \leftarrow I_{\{a = A_4 || A_6\}}$.
15:             end procedure
16:         else
17:             Choose $a = A_1(A_2)$ w.p. $q(n)(1 - q(n))$.
18:         end if
19:         procedure UPDATE–BP–OP
20:             See Algorithm 6.
21:         end procedure
22:     else
23:         procedure DEPARTURE–POLICY–SA
24:             $D \leftarrow D + 1$.
25:             if $O_L < (O_{\max} - \epsilon_O)$ and $a^* = (A_5 || A_7)$ then
26:                 Choose $a^*(A_1)$ w.p. $p(n)(1 - p(n))$.
27:             else
28:                 Choose action $a = A_1$.
29:             end if
30:             $F_0 \leftarrow I_{\{a = A_5 || A_7\}}$.
31:             $O_L \leftarrow \frac{O_L(A_H + D) + F_O}{(A_H + D + 1)}$.
32:         end procedure
33:     end if
34: end while
```

Hence, effectively, the system observes less blocking probability than that of MCSA, when λ_H is low. As λ_H increases, the blocking probability of high priority users increases since $q(n)$ is an increasing function of n. The performance of the resulting policy in the case of SMCSA is closer to the optimal policy than that in the case of MCSA. This is because unlike MCSA, the blocking is state-dependent. The blocking probability of high priority users gradually increases with λ_H, similar to the optimal policy. Therefore, the problem of high blocking probability (which is close to $B_{\max}$) for high priority users for all values of λ_H, as seen in MCSA, does not arise in the case of SMCSA. Similar observation holds in the case of the offloading probability of low priority users also. As λ_L grows, the offloading probability of low priority users gradually rises. Similar to MCSA, SMCSA does not require the knowledge of λ_H, λ_L and p_g and hence, is practically implementable.

6.5 Comparison of Complexities

In this subsection, we analyze the computational and storage complexities associated with the proposed algorithms and the optimal policy. We provide the details of the comparison of complexities of MCSA and SMCSA in Table 6.3.

The optimal policy needs to store the optimal action corresponding to every state, resulting in a storage complexity of $O(|\mathcal{S}|)$. Moreover, the computation of the optimal policy using traditional policy iteration involves the worst case complexity of $O(|\mathcal{A}|^{|\mathcal{S}|})$ since the total number of feasible policies is $|\mathcal{A}|^{|\mathcal{S}|}$. Therefore, it is very cumbersome to compute the optimal policy using traditional DP methods.

In the case of MCSA, whenever an event occurs, we need to compute the best action a^*. Therefore, the per-iteration computational complexity of MCSA is $O(|\mathcal{A}|)$. As discussed in Section 6.3, action elimination reduces the effective cardinality of the action space. Although this does not reduce the theoretical computational complexity of MCSA, in practice, the computation time may reduce. MCSA requires to store the running values of A_H, D, B_H and O_L. However, it does not need to store any information regarding the state space. Therefore, the resulting storage complexity is $O(1)$.

The per-iteration worst case computational complexity of SMCSA is also $O(|\mathcal{A}|)$ because when the current values of the constraints are below the specified value, we need to compute the best action. However, the complexity involved with the probabilistic state-aware blocking and offloading is $O(1)$ because no comparison among the actions is present. Apart from A_H, D, B_H and O_L, SMCSA needs to store the information regarding the regions R_n, where $1 \leq n \leq P$ and the corresponding probabilities $q(n)$ and $p(n)$. Therefore, the storage complexity of SMCSA is $O(P)$.

Remark 14. *A well-studied approach for MDP problems is the investigation of structural properties, see e.g., [81–83], which often leads to threshold-type optimal policy. In Chapter 3, which does not consider the offloading probability as a constraint, we have proved the optimality of threshold policies. Although the computational complexity of the resulting algorithm is lower than that of traditional policy iteration, the computational complexity is still exponential in one of the parameter of the state space. The problem addressed in this chapter does not result in a threshold based optimal policy. However, the proposed algorithms provide significantly lower computational complexity compared to what would have been achieved corresponding to a threshold structure.*

Remark 15. *The loss of threshold sturcture of the optimal policy after incorporating the additional constraint on the offloading probability is due to the consideration of channel states of users. Due to the consideration of channel states of users, the monotonicity of the reward function as a function number of users no longer holds (See Equation (6.1)).*

6.6 Conclusions

In this chapter, we have considered a RAT selection problem where we aim to maximize the total system throughput subject to constraints on the high priority user blocking probability and the low priority user offloading probability. This problem has been formulated as a CMDP problem. We have reduced the dimensionality of the action space by proving the suboptimality of different actions in various states. We have proposed two low-complexity online heuristic algorithms for RAT selection. These algorithms do not require the knowledge of the underlying transition probabilities of the model. Contrary to the first algorithm where the blocking probability and the offloading probability do not depend on the statistics of the arrival processes, in the second algorithm, blocking and offloading is performed based on the system state. Implementation of the proposed algorithms in an ns-3 based software defined networking-compliant evaluation platform is presented in the next chapter.

Table 6.2: Reward rate table.

$(a\|E_l)$	$r(s,a)$
$(A_1\|E_1)$	$R(i_G, i_B, j_G, j_B, k_G, k_B)$
$(A_1\|E_3)$	$R(i_G, i_B, j_G, j_B, k_G, k_B)$
$(A_1\|E_5)$	$R(i_G, i_B, j_G, j_B, k_G, k_B)$
$(A_1\|E_6)$	$R(i_G, i_B, j_G, j_B, k_G, k_B)$
$(A_1\|E_7)$	$R(i_G, i_B, j_G, j_B, k_G, k_B)$
$(A_1\|E_8)$	$R(i_G, i_B, j_G, j_B, k_G, k_B)$
$(A_1\|E_9)$	$R(i_G, i_B, j_G, j_B, k_G, k_B)$
$(A_1\|E_{10})$	$R(i_G, i_B, j_G, j_B, k_G, k_B)$
$(A_2\|E_1)$	$R(i_G+1, i_B, j_G, j_B, k_G, k_B)$
$(A_2\|E_2)$	$R(i_G, i_B, j_G+1, j_B, k_G, k_B)$
$(A_2\|E_3)$	$R(i_G, i_B+1, j_G, j_B, k_G, k_B)$
$(A_2\|E_4)$	$R(i_G, i_B, j_G, j_B+1, k_G, k_B)$
$(A_3\|E_2)$	$R(i_G, i_B, j_G, j_B, k_G+1, k_B)$
$(A_3\|E_4)$	$R(i_G, i_B, j_G, j_B, k_G, k_B+1)$
$(A_4\|E_1)$	$R(i_G+1, i_B, j_G, j_B-1, k_G, k_B+1)$
$(A_4\|E_3)$	$R(i_G, i_B+1, j_G, j_B-1, k_G, k_B+1)$
$(A_5\|E_5)$	$R(i_G, i_B, j_G+1, j_B, k_G-1, k_B)$
$(A_5\|E_6)$	$R(i_G, i_B, j_G+1, j_B, k_G-1, k_B)$
$(A_5\|E_7)$	$R(i_G, i_B, j_G+1, j_B, k_G-1, k_B)$
$(A_5\|E_8)$	$R(i_G, i_B, j_G+1, j_B, k_G-1, k_B)$
$(A_5\|E_9)$	$R(i_G, i_B, j_G, j_B-1, k_G, k_B+1)$
$(A_5\|E_{10})$	$R(i_G, i_B, j_G, j_B-1, k_G, k_B+1)$
$(A_6\|E_1)$	$R(i_G+1, i_B, j_G-1, j_B, k_G+1, k_B)$
$(A_6\|E_2)$	$R(i_G, i_B+1, j_G-1, j_B, k_G+1, k_B)$
$(A_7\|E_5)$	$R(i_G, i_B, j_G, j_B+1, k_G, k_B-1)$
$(A_7\|E_6)$	$R(i_G, i_B, j_G, j_B+1, k_G, k_B-1)$
$(A_7\|E_7)$	$R(i_G, i_B, j_G, j_B+1, k_G, k_B-1)$
$(A_7\|E_8)$	$R(i_G, i_B, j_G, j_B+1, k_G, k_B-1)$
$(A_7\|E_9)$	$R(i_G, i_B, j_G-1, j_B, k_G+1, k_B)$
$(A_7\|E_{10})$	$R(i_G, i_B, j_G-1, j_B, k_G+1, k_B)$

Table 6.3: Computational and storage complexities of different algorithms.

Algorithm	**Storage complexity**	**Computational complexity**
MCSA	$O(1)$	$O(\|\mathcal{A}\|)$
SMCSA	$O(P)$	$O(\|\mathcal{A}\|)$

Chapter 7

Software Defined Networking based Implementation of RAT Selection Algorithms

In this chapter, we implement the RAT selection algorithms proposed in Chapter 6 in an SDN based evaluation platform [1] developed using ns-3. Development of the evaluation platform requires significant changes in the existing modules of ns-3. For the sake of completeness, we describe the details of the evaluation platform in this chapter. However, the main contribution in this chapter is to implement the proposed RAT selection algorithms (in Chapter 6) in the SDN based evaluation platform and investigate the practicality of the proposed solutions.

With the 5G cellular mobile system [6, 7] standardization in progress, it is expected that the next generation networks will be a mixture of a large number of RATs. This gives rise to the need for a unified framework which can control and manage multiple RATs together. In existing networks, every RAT is controlled and managed by RAT-specific elements. For example, LTE is controlled by control elements such as Mobility Management Entity (MME) and Evolved NodeB (eNodeB), and WLAN is controlled by WLAN controllers. Although the upcoming 5G network [6, 7] supports multiple RATs, radio access related decisions are individually taken by RAT-specific elements. Due to this

[1]Ashish Sharma, Rohan Kharade and Abhishek Dandekar contributed towards the development of the SDN based evaluation platform in ns-3.

distributed nature of control and management of various RATs, a unified global view of the entire network comprising different RATs is not present in the existing network. This leads to a suboptimal utilization of resources in the network. However, a unified framework for control and management related decisions can facilitate the optimal resource utilization in a network. Therefore, it is necessary that the common functionalities supported by diverse RATs such as admission control, flow control, mobility management are controlled and managed in a unified manner. Emergence of SDN [41–43] paradigm may enable us to achieve unified control and management of various RATs.

The basic idea behind SDN is the split of control and data plane elements and functionalities in a network. While the control plane consists of control and management protocols and elements, the data plane consists of protocols and elements for data transfer. Using SDN, the control plane functionalities of different RATs can be decoupled from the network elements of various RATs and aggregated in the control plane. The interface which separates the control and data planes enables the configuration of data plane elements using policy-based rules provided by control plane elements. Since the control plane has a global view of the entire network, this approach allows us to design these policy-based rules in an optimal manner. Therefore, optimal utilization of network resources can be achieved contrary to distributed control in today's network.

In this chapter, we implement the RAT selection algorithms presented in Chapter 6 in an SDN based LTE-WiFi network. To this end, we propose an SDN based network architecture which unifies the control and management functionalities of LTE and WiFi networks using an SDN controller. The LTE BS and the WiFi AP forward the radio resource management messages to the SDN controller which takes control and management decisions. Note that although we consider LTE BS and WiFi AP in the proposed framework, the framework can be easily adopted for any RAT including 5G. The SDN controller takes RAT selection and offloading decisions based on the implemented algorithm. For implementation purposes, we develop an SDN based evaluation platform in ns-3. Experimental results demonstrate that the algorithms proposed in Chapter 6 provide near-optimal performances. Also, performances of the proposed algorithms in Chapter 6 are compared with other traditional RAT selection algorithms in the literature. We also verify the robustness of our algorithms in the presence of user mobility and compare their performances with other algorithms.

The rest of the chapter is organized as follows. Section 7.1 describes the proposed system architecture. In Section 7.2, we present the simulation results obtained by implementing the RAT selection algorithms within the SDN based evaluation platform. Section 7.3 concludes the chapter.

7.1 Proposed SDN based System Architecture

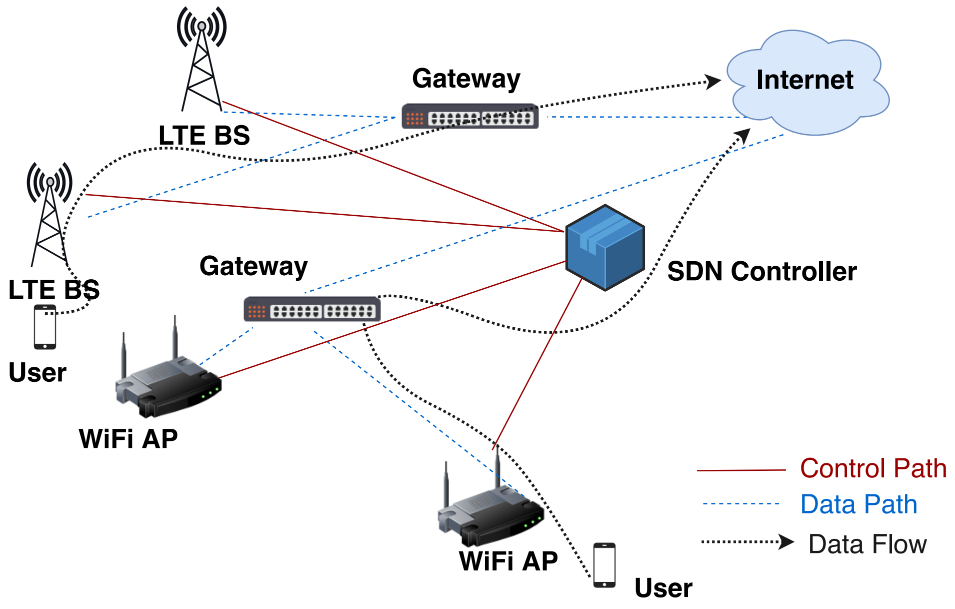


Figure 7.1: SDN based LTE-WiFi architecture.

In this section, we propose an overlay architecture that allows us to handle multiple RATs together using an SDN controller. As demonstrated in Fig. 7.1, the SDN controller handles all control and management related functionalities of different RATs. To achieve that, the Radio Resource Management (RRM) unit of the LTE BS is moved to the SDN controller. The LTE RRM consists of radio bearer control, radio admission control, connection mobility control, dynamic resource allocation and inter-cell interference coordination. However, as considered in Chapter 6, we focus on aspects related to radio bearer, admission and mobility control only. In effect, decision making related functionalities are implemented in the SDN controller which has a global view of the entire network. RRM related control messages sent by the users in LTE are forwarded by the LTE BS to the SDN controller. For example, the Radio Resource Control (RRC) connection request message reaches the SDN controller via the LTE BS. After that, the RRC connection response message sent by the SDN controller is forwarded to the user via the LTE BS. Similarly, the association request message is forwarded by the WiFi AP to the SDN controller. However, control message exchanges which are not related to RRM, remain as it is there in existing LTE network. This is due to the fact that the remaining functionalities of RRC after the removal of RRM, remain in the LTE BS. In spite of the fact that we have a single controller, scalability issues do not arise since only a small fraction of control signals (which are RRM related) is handled by the SDN controller. Note that the information regarding the channel conditions of users is needed for taking RAT selection and offloading related decisions. For this purpose, channel condition information of users are forwarded to the SDN controller at the time of association of users.

7.2 Simulation Results

In this section, we implement the RAT selection algorithms proposed in Chapter 6 in an evaluation platform which is built according to the proposed SDN based LTE-WiFi architecture in the last section. The evaluation platform is constructed by the modification of existing components of ns-3 in such a manner that it is in agreement with the proposed architecture. We observe performances of the proposed algorithms in terms of the blocking probability of high priority users, the offloading probability of low priority users and the total system throughput and compare with those of the optimal policy. We also compare the performances of the proposed algorithms with the association scheme adopted in existing network where SDN is not present. In the absence of SDN, the association scheme results in on-the-spot offloading [57], where low priority users are always associated with WiFi and high priority users are associated with LTE. However, when capacity is reached in LTE, high priority users are blocked. We also evaluate the performances of the proposed algorithms in the face of user mobility and establish that indeed the proposed algorithms perform well in comparison to other algorithms under different realistic network scenarios.

7.2.1 Simulation Setup and Methodology

We setup an evaluation platform (implemented in ns-3) which provides a framework to simulate the SDN controller present in the proposed LTE-WiFi architecture in Section 7.1.

7.2.1.1 Description of Evaluation Platform

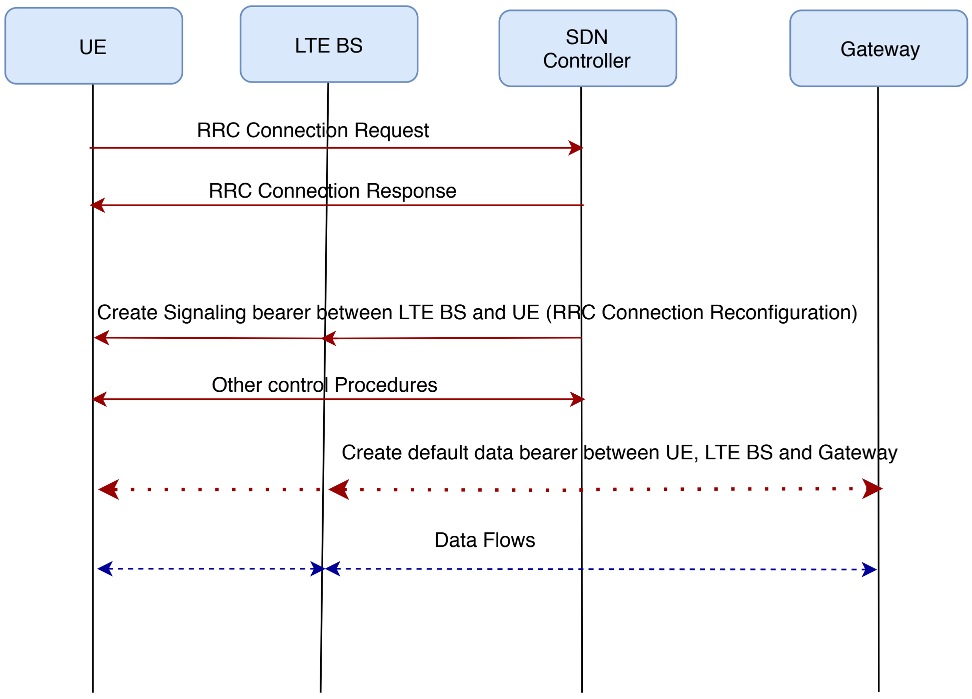

UE
LTE BS
SDN Controller
Gateway
RRC Connection Request
RRC Connection Response
Create Signaling bearer between LTE BS and UE (RRC Connection Reconfiguration)
Other control Procedures
Create default data bearer between UE, LTE BS and Gateway
Data Flows

Figure 7.2: Association procedure in LTE.

In this section, we describe the details of the SDN based evaluation platform which is implemented in ns-3. We create an SDN controller node which has two interfaces, one towards the LTE BS and the other towards the WiFi AP, over IP connections. The SDN controller consists of an LTE controller and a WiFi controller as logical entities. The RRM functionalities present in the LTE BS are moved to the SDN controller. RRM related control signals in LTE and control signals in WiFi are forwarded to the SDN controller. Deep Packet Inspection (DPI) functionality present in Packet Data Convergence Protocol (PDCP) layer of LTE is utilized to filter the control packets and route them to the SDN controller. However, the data plane traffic is routed directly from the LTE BS to the gateway. To enable communication between the LTE BS and the SDN controller, an

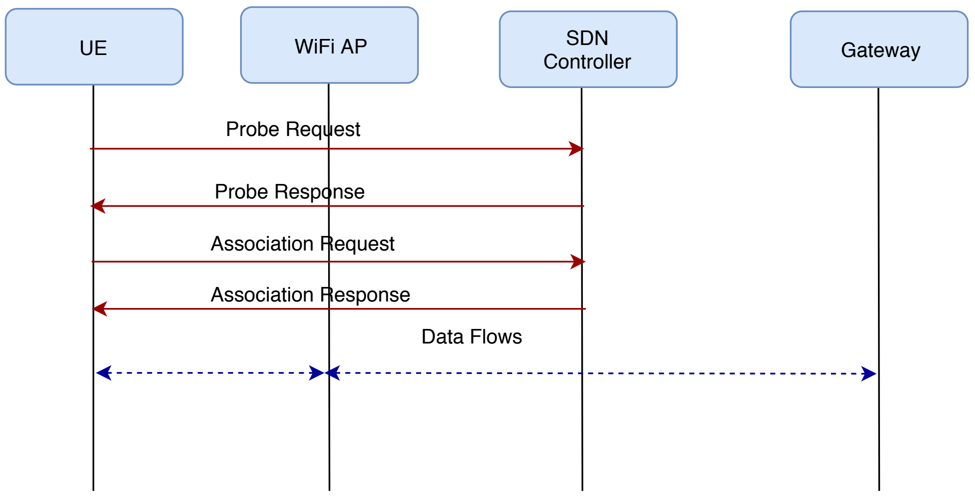


Figure 7.3: Association procedure in WiFi.

application is developed. This application sends the traditional control messages of LTE encapsulated in other control messages with suitable headers. For example, the RRC connection request message is forwarded by the LTE BS to the SDN controller, and the RRC connection response message sent by the SDN controller is forwarded by the LTE BS to the user. Similarly, association request message received by the MAC layer of WiFi AP is forwarded to the SDN controller. The association response message sent by the SDN controller is forwarded by the WiFi AP to the user. We describe this procedures through the call flows in Fig. 7.2 and 7.3, respectively.

For the communication between the LTE BS/ WiFi AP and the SDN controller, we define a control plane protocol which encapsulates the LTE and WiFi control messages (such as RRC connection request and association request) by identifying them using headers. The protocol works over TCP. The control packets are received in the SDN controller, and appropriate response messages are sent to the LTE BS/WiFi AP. We create a point-to-point link between the SDN controller and the WiFi AP. The SDN controller listens to a specific port number for incoming control packets forwarded by the WiFi AP. Then, routing paths are configured between the users and the controller for 1-hop communication. Similarly, a point-to-point link is created between the SDN controller and the LTE BS. The routing path between the user and the controller is configured then. In this case, the controller listens to a different port for receiving control packets forwarded by the LTE BS. Whenever the controller receives a control message, then a callback function is triggered. The callback function is responsible for identifying whether the association message is generated by low priority or high priority users based on the size of the control packet which is different for users of different priorities. After that, appropriate functions which take the RAT selection decisions are triggered. Whenever an event (an arrival or a departure of a user) occurs, a control packet is sent from the user to the BS/AP which forwards the packet to the SDN controller. The SDN controller then triggers the function which chooses an action in response to the event, following the algorithm implemented in the controller. Based on the chosen action, the association of the user is made. If the chosen action involves offloading of a low priority user, viz., actions A_4, A_5, A_6, A_7 (see Section 6.1 in Chapter 6), then the existing association is destroyed, and the user is associated to the other RAT.

7.2.2 Description of Simulation Scenario

The network model consists of a 3GPP LTE BS and an IEEE 802.11g WiFi AP inside the coverage area of the LTE BS. Users are assumed to be stationary. We consider that the radius of the coverage area of the WiFi AP is approximately 30 m. The WiFi AP is approximately 50 m away from the LTE BS. Data users are uniformly distributed inside the coverage area of the WiFi AP. We assume that the WiFi AP is deployed by the cellular operators, and hence, the interworking is trusted in nature. LTE and WiFi network parameters are summarized in Tables 7.1 and 7.2. Note that LTE and WiFi network parameters are chosen based on 3GPP [76]- [77] models and saturation throughput [72] IEEE 802.11g WiFi [13] model, respectively. In simulations, we assume that the maximum data rate which a low priority user can obtain is 10 Mbps due to the bottleneck in the access network. We set $B_{\max} = O_{\max} = 0.05, \epsilon_B = \epsilon_O = 0.01$ (see Chapter 6). As described in Chapter 6, in the case of SMCSA, we divide the entire state space into two regions, viz., R_1 and R_2. We keep $q(1) = p(1) = 0$ and $q(2) = p(2) = 1$.

Table 7.1: LTE network model.

Parameter	**Value**
Maximum high priority user capacity	4 users
Bit rate of a single high priority user	20 kbps
Voice packet payload	50 bits
Data packet payload	600 bits
Tx power for BS and UE	46 dBm and 23 dBm
Noise figure for BS and UE	5 dB and 9 dB
Antenna height for BS and UE	32 m and 1.5 m
Antenna parameter for BS and UE	Isotropic Antenna
Path loss (R in kms)	$128.1 + 37.6 \log(R)$
Multi-path fading	Extended Pedestrian A model [79]

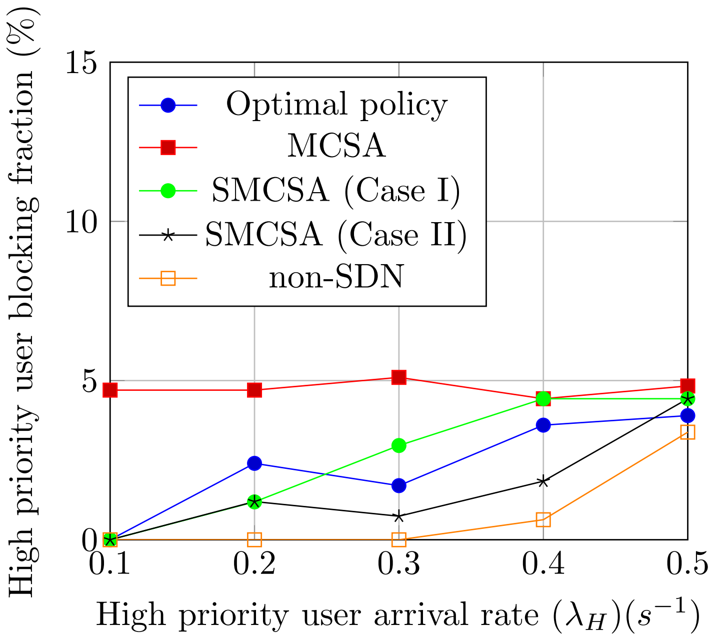


Figure 7.4: High priority user blocking percentage vs. λ_H $(\lambda_L = 1, \mu_H = 1$ and $\mu_L = 1)$.

7.2.3 High Priority User Arrival Rate Variation

Fig. 7.4 describes the blocking probability of high priority users of the proposed algorithms, optimal policy and on-the-spot offloading (existing non-SDN scenario) as a function of λ_H. As λ_H increases, the blocking probability of the optimal policy increases. This

Table 7.2: WiFi network model.

Parameter	**Value**
Channel bit rate	54 Mbps
UDP header	224 bits
Packet payload	1500 bytes
Slot duration	20 μs
SIFS	10μs
DIFS	50μs
Minimum acceptable per-user throughput	4.5 Mbps
Tx power for AP	23dBm
Noise figure for AP	4 dB
Antenna height for AP	2.5 m
Antenna parameter	Isotropic antenna
Path loss (R in kms)	$140.3 + 36.7 \log(R)$
Fading	Rayleigh fading

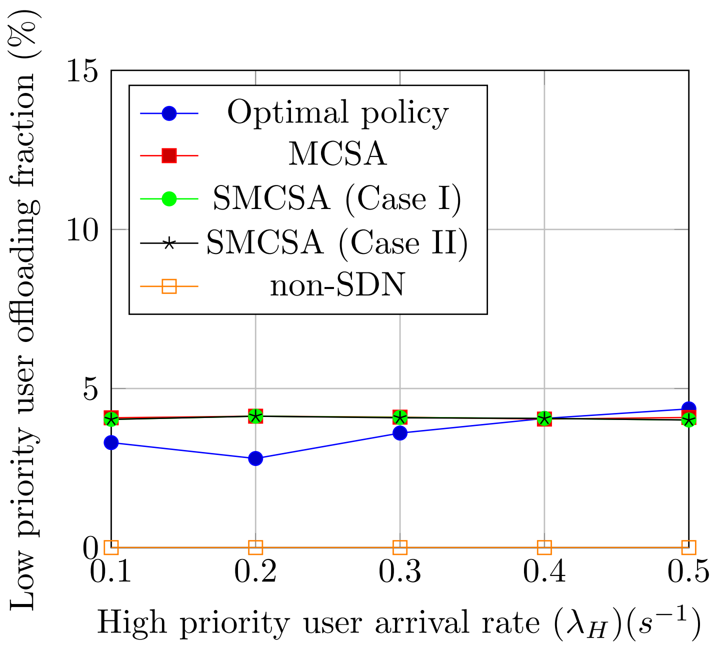


Figure 7.5: Low priority offloading percentage vs. λ_H ($\lambda_L = 1, \mu_H = 1$ and $\mu_L = 1$).

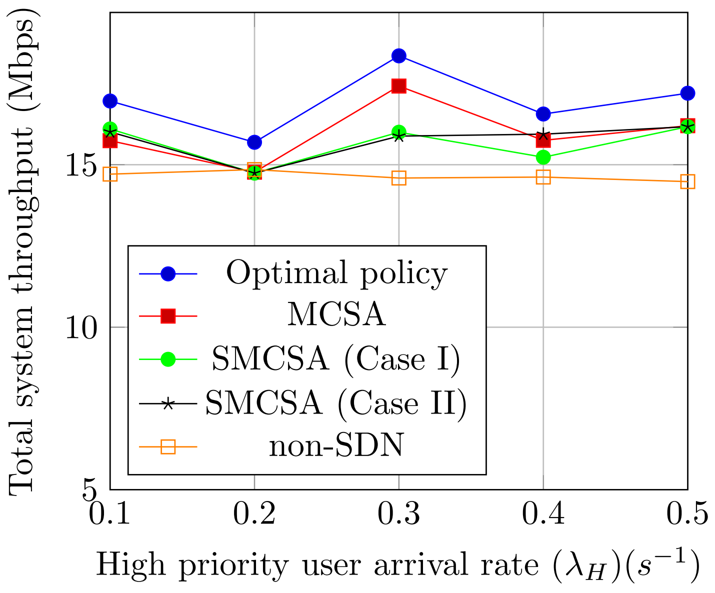


Figure 7.6: Total system throughput vs. λ_H $(\lambda_L = 1, \mu_H = 1$ and $\mu_L = 1)$.

happens because as λ_H increases, more number of high priority users having low contributions to the system throughput arrive in the system and are blocked in order to maximize the total system throughput. Since MCSA blocks the high priority users based on $B_{\max}$ without considering λ_H, the blocking probability is same for all λ_H. SMCSA is designed in such a way that it blocks high priority users only when the system reaches region R_2. We consider two cases, viz., $R_1 : (i_G + 2i_B) \leq C_L - 2$, and $R_1 : (i_G + 2i_B) \leq C_L - 1$, respectively. The high priority user blocking probability of SMCSA gradually increases with λ_H, similar to the optimal policy. This happens because when the value of λ_H is low, we block the incoming high priority users with low probability. As λ_H increases and the system gradually fills up with high priority users, the probability of blocking increases as $q(n)$ is an increasing function of n. Since the size of region R_1 is smaller in the first case, the blocking probability in the second case is lower than that of the first case. In the absence of SDN, high priority user blocking probability gradually rises with λ_H. Since in this case, blocking happens only when the system reaches the capacity, the blocking probability values are lower than those of other algorithms.

In Fig. 7.5, we plot the low priority user offloading fractions for the considered algorithms. In the absence of SDN, offloading from one RAT to another is not possible. The low priority user offloading fractions provided by MCSA and SMCSA are similar for all values of λ_H since the value of λ_L is fixed. Changes in λ_H do not have much impact on the offloading probability of low priority users. However, the low priority user offloading fraction of the optimal policy gradually rises with λ_H. This happens because with increasing λ_H, actions involving offloading (A_4, A_5, A_6, A_7) are selected more frequently.

The total system throughput provided by MCSA is very close to the total system throughput of the optimal policy, as observed in Fig. 7.6. The total throughput of SMCSA in both the cases are slightly lower than that of MCSA. This happens because MCSA blocks more fraction of high priority users than SMCSA. Since there is a trade-off between the total system throughput and the high priority user blocking probability, the total system throughput is higher in the case of MCSA. All these algorithms outperform on-the-spot algorithm. Since low priority users are always associated with WiFi in the non-SDN scenario, no load balancing mechanism is present. Moreover, the total throughput in WiFi degrades due to the contention among low priority users. Hence, the throughput performance of on-the-spot offloading is the worst among all the algorithms.

7.2.4 Low Priority User Arrival Rate Variation

Fig. 7.7 describes the high priority user blocking probability performance of the proposed algorithms, optimal policy and on-the-spot algorithm as a function of λ_L. As observed from the figure, similar to Fig. 7.4, MCSA provides a blocking probability which is close to $B_{\max}$ for all λ_L. SMCSA is designed in such a way that it blocks high priority users and offloads low priority users only when the system reaches region R_2. We consider two cases, viz., $R_1 : (i_G + 2i_B) \leq C_L - 2, (k_G + k_B) \leq 4, (j_G + j_B) \leq 2$ and $R_1 : (i_G + 2i_B) \leq C_L - 2, (k_G + k_B) \leq 4, (j_G + j_B) < 2$, respectively. The performance of SMCSA for the first case is close to that of the optimal policy. In the second case, the blocking probabilities are slightly higher than those of the first case. This happens because region R_1 is smaller in the second case.

In Fig. 7.8, we plot the offloading probability of low priority users as a function of λ_L. The offloading probability of the optimal policy grows with λ_L. MCSA provides an offloading probability which is close to the given constraint for all values of λ_L. In case of SMCSA, offloading probability grows with λ_L, similar to the optimal policy. The reason behind this is that the offloading probability $p(n)$ is an increasing function of n. The

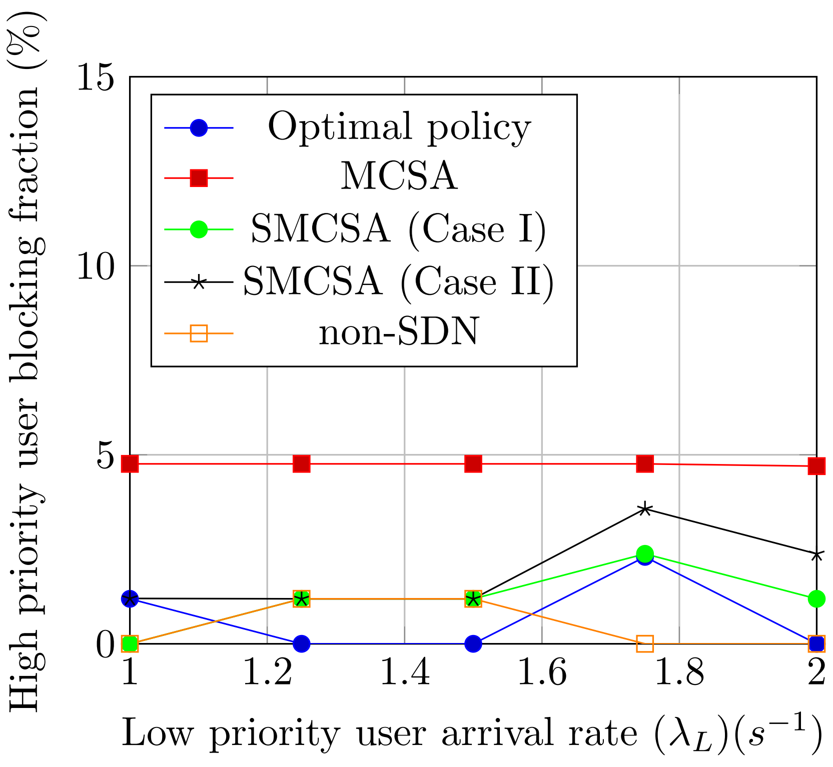


Figure 7.7: High priority user blocking percentage vs. λ_L ($\lambda_H = 0.2, \mu_H = 1$ and $\mu_L = 1$).

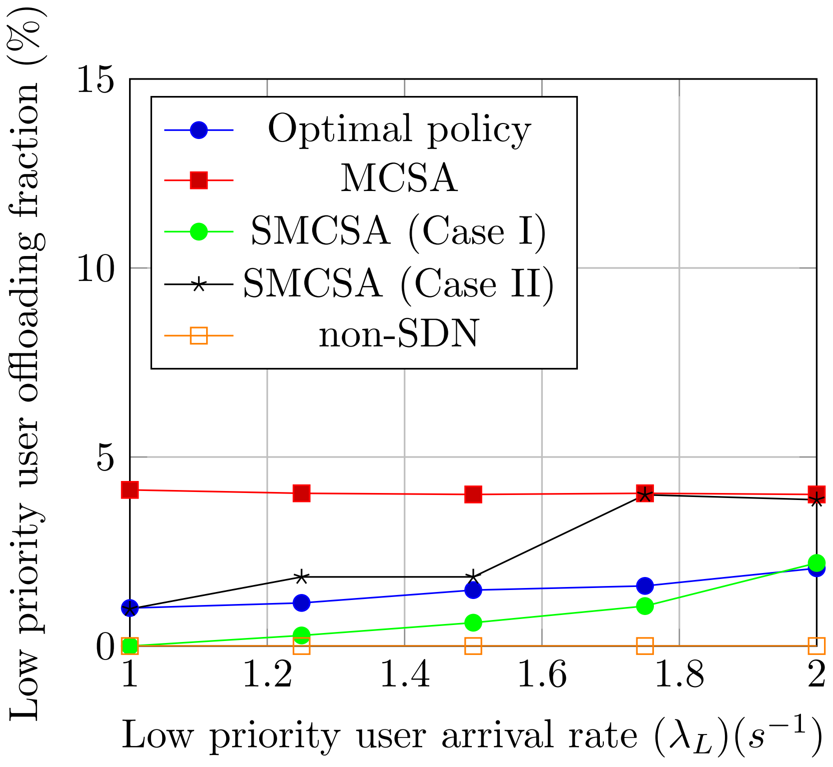


Figure 7.8: Low priority offloading percentage vs. λ_L ($\lambda_H = 0.2, \mu_H = 1$ and $\mu_L = 1$).

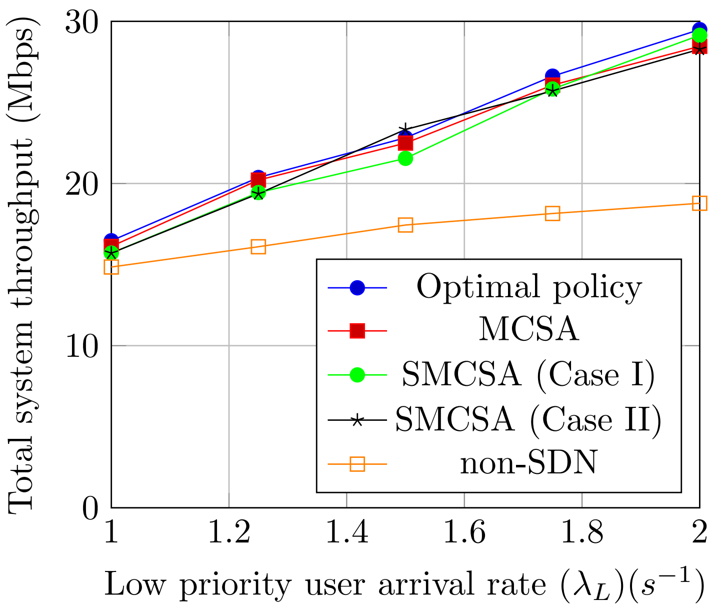


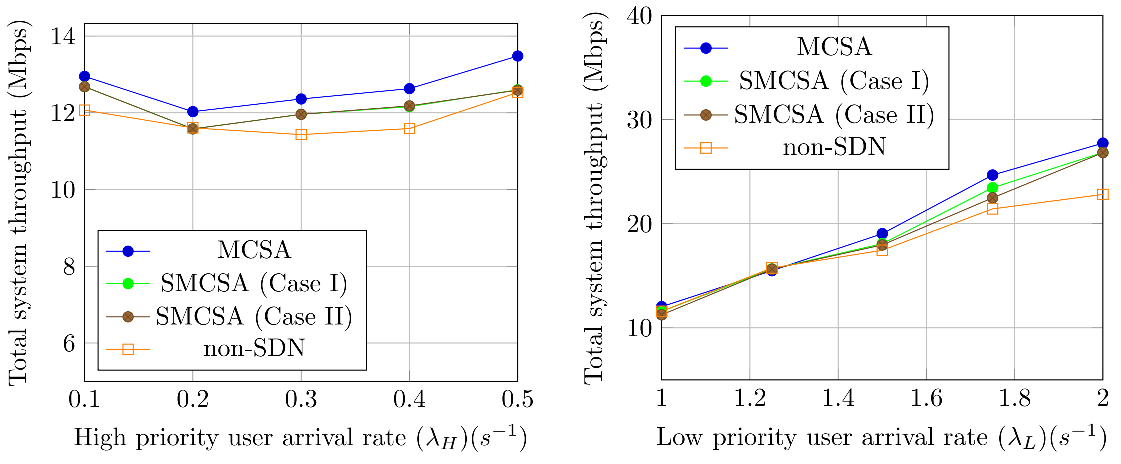
Figure 7.9: Total system throughput vs. λ_L ($\lambda_H = 0.2, \mu_H = 1$ and $\mu_L = 1$).

offloading probability in in the second case is slightly better than in the first case since region R_1 is smaller in the second case. Since offloading is not possible when SDN is not present, in the non-SDN case, the offloading probability of low priority users is always zero.

In Fig. 7.9, we observe that the performances of both MCSA and SMCSA are close to optimal, outperforming on-the-spot offloading algorithm. Though the proposed algorithms take into account only the instantaneous rewards while optimizing, these algorithms facilitate load balancing between LTE and WiFi RATs. The total system throughput of on-the-spot offloading does not increase much with λ_L due to contention among users in WiFi. Fig. 7.9 demonstrates that indeed our proposed algorithms provide near-optimal performances.

7.2.5 Consideration of multiple channel states in WiFi

Throughout this chapter, we have assumed that channel states of users in WiFi are always good. In this section, we evaluate the performance of our proposed algorithms considering multiple channel states of users in WiFi. We consider two types of low priority users. As assumed previously, users present within the coverage area of the WiFi AP are taken to be users with good channel states in WiFi. Users present outside the coverage area of



(a) Total system throughput vs. λ_H ($\lambda_L = 1, \mu_H = 1$ and $\mu_L = 1$).

(b) Total system throughput vs. λ_L ($\lambda_H = 0.2, \mu_H = 1$ and $\mu_L = 1$).

Figure 7.10: Total system throughput for different algorithms under varying λ_H and λ_L.

the WiFi AP are assumed to be users with bad channel states in WiFi. Such users are always associated with LTE, irrespective of their channel states in LTE. We also assume that separate resources are reserved in LTE for users having bad channel states in WiFi.

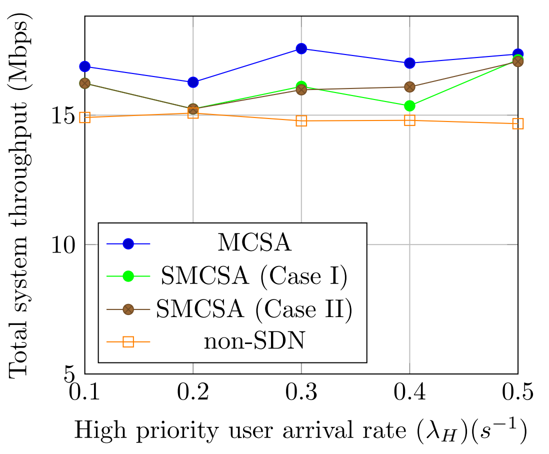
In Fig. 7.10a, we plot the total system throughputs for the considered algorithms as a function of λ_H. As observed from the figure, both the proposed algorithms perform better than on-the-spot offloading. The throughput performance of MCSA is slightly better than that of SMCSA. However, the performances of SMCSA and on-the-spot offloading are very close to each other. This is due to the fact that for users with bad channel in WiFi, both on-the-spot offloading and the proposed algorithms work in a similar fashion since all these algorithms associate them with the LTE BS. Therefore, performance benefits corresponding to the proposed algorithms are achieved only due to users which do not have bad channels with respect to the WiFi AP. Since separate resources are reserved in LTE for low priority users outside the coverage area of the WiFi AP, the blocking probabilities and the offloading probabilities are identical to those of Figs. 7.4 and 7.5.

Similarly, in Fig. 7.10b, we illustrate the comparative performances of MCSA, SMCSA and on-the-spot offloading in terms of the total system throughput as a function of λ_L. Clearly, both MCSA and SMCSA outperform on-the-spot offloading. As λ_L increases, the performance gap between the proposed algorithms and on-the-spot offloading, increases. The blocking probabilities and the offloading probabilities are identical to those of Figs. 7.7 and 7.8.

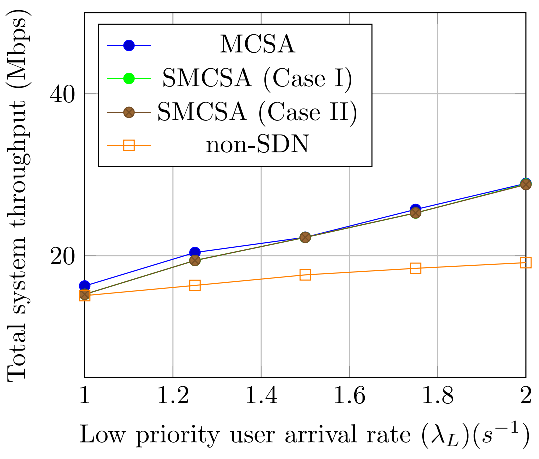
7.2.6 Consideration of Mobility

In this section, we evaluate the performances of the algorithms in the presence of user mobility. In the simulation setting, we consider random waypoint model [80] for user mobility. In this model, each user waits for a random interval and then chooses a random waypoint and a random speed. The user moves towards the waypoint with the chosen speed. After reaching the destination, the user waits and then chooses another random waypoint and random speed. The process continues thereafter. We set the user speed uniformly and randomly in the range $[0, 40]$ km/h. We deterministically set the random variable which signifies the wait time to be always zero. The initial positions of low and high priority users are allocated using the random rectangle position allocator where the x-axis and y-axis coordinates are chosen uniformly randomly between two given values. We choose these values in such a way that the low and high priority users remain within the coverage area of the LTE BS and the WiFi AP, respectively. Similar to the previous scenario, we use the extended pedestrian A model for multipath fading in LTE and Rayleigh fading in WiFi. However, the channel gains are not fixed anymore since the distances of the users to the LTE BS/ WiFi AP vary due to the mobility of users. The path loss models adopted in these scenarios are same as those of listed in Table 7.1 and Table 7.2, respectively. We classify the channel states of high priority users in LTE as good or bad depending on their locations (cell center/cell edge) at the time of their arrivals (See Section 6.1). However, the channel state of a user may change from good to bad or vice versa depending on the location of the user.

In general, due to high mobility associated with the users, the users in the system may be offloaded frequently from one RAT to another. This may increase the offloading probability associated with the overall system. Since the algorithms are designed in such a way that the offloading probability satisfies the associated constraint, a user with high mobility may significantly increase the offloading probability of the system. As a result, it may happen that the low mobility users get very less number of offloading opportunities. Furthermore, a user with high mobility is expected to drain a lot of battery due to excessive offloading from one RAT to another. To take into account these factors, we



(a) Total system throughput vs. λ_H ($\lambda_L = 1, \mu_H = 1$ and $\mu_L = 1$).



(b) Total system throughput vs. λ_L ($\lambda_H = 0.2, \mu_H = 1$ and $\mu_L = 1$).

Figure 7.11: Plot of total system throughput for different algorithms under varying λ_H and λ_L.

modify the algorithms in the following way. Apart from the constraint on the overall offloading probability of low priority users, we consider the offloading profile of individual low priority users while offloading. To be precise, whenever an action involving offloading of low priority users (A_4, A_5, A_6, A_7) is chosen, we choose a user which has not been offloaded till now. If no such user is present, then we choose the user which has been offloaded the earliest before.

In 7.11a, we observe that both MCSA and SMCSA outperform on-the-spot offloading in terms of the total system throughput for all values of λ_H. Similar observation is made in Fig. 7.11b for different values of λ_L. We do not demonstrate the blocking probability and the offloading probability performances in the presence of mobility since they are exactly same as those of Fig. 7.4, 7.7, 7.5 and 7.8.

7.3 Conclusions

In this chapter, we have implemented the proposed RAT selection algorithms in Chapter 6 in an ns-3 based evaluation platform. The evaluation platform has been built according to an SDN based LTE-WiFi network architecture proposed by us. The propose architecture unifies the control and management functionalities of LTE and WiFi networks using an

SDN controller. Experimental results have demonstrated that the proposed RAT selection algorithms outperform the traditional algorithm and provide near-optimal performances. We have also verified that the proposed algorithms provide better performances than that of the traditional algorithm, even in the face of user mobility.

Chapter 8

Summary of Results and Future Directions

8.1 Summary of Results

In this thesis, we have addressed the optimal RAT selection problem in a heterogeneous network. To cope up with an unprecedented growth in data traffic, mobile data traffic offloading from LTE to WiFi has been proposed in 3GPP Release 12 specifications. As discussed in Chapter 2, there are two kinds of approaches for RAT selection and offloading in the literature, viz., user-initiated and network-initiated approaches. In user-initiated approaches, users take individual RAT selection decisions to maximize their own utilities. In this thesis, we have adopted network-initiated approaches for RAT selection so that the overall system performance can be optimized. Although our proposed RAT selection and offloading solutions take into account LTE and WiFi as representative RATs, the framework proposed by us is generic enough so that it can be modified easily to consider other RATs as well.

Whenever a user arrives in the LTE-WiFi system, it is necessary to choose the RAT selection strategy governed by the optimal association policy. Furthermore, offloading of active users from one RAT to another may be triggered by the optimal policy during the association and the departures of users. As discussed in Chapter 2, few network-initiated solutions in the literature have addressed the issue of RAT selection and offloading, separately. However, in this thesis, for the first time, we have addressed the issue of joint

RAT selection and offloading for an LTE-WiFi HetNet in a dynamic environment where users arrive and depart from the system. Furthermore, no other work in the literature has considered the trade-off between the total system throughput and the blocking probability of voice users for an LTE-WiFi HetNet within an optimization framework. The objective of this thesis is to determine the optimal RAT selection algorithms such that the total system throughput can be maximized while satisfying various constraints on desired factors such as the blocking probability of voice users and the offloading probability of low priority users.

These optimization problems of determining the optimal policy can be formulated within the framework of Markov Decision Process (MDP). MDP problems can in general be solved using DP based schemes like Value Iteration Algorithm (VIA), policy iteration [69]. Often these solution techniques suffer from the curse of dimensionality where the computational complexities of these schemes become exponential in the size of the state space. Hence, they encounter practical implementation issues. Moreover, these techniques require the knowledge of the underlying transition probabilities between the states of the associated Markov chain which are governed by the statistics of the system dynamics such as the arrival rates of users. In reality, these may be difficult to gather. To address this issue, we usually make simplifying assumptions regarding the model. However, in practice, these assumptions may be inaccurate. Even if the initial guess about the statistics of the system dynamics are accurate, they tend to become worse with time since the system may change with time. We have addressed these issues in this thesis.

To summarize, we have devised optimal (or sub-optimal) algorithms which bring about reductions in computational and storage complexities compared to traditional schemes of determining the optimal policy. To achieve that, we have utilized the optimality of threshold policies which has been established in this thesis. Moreover, we have focused on developing model-unaware Reinforcement Learning (RL) schemes which can be implemented online without the knowledge of the statistics of the arrival processes of the users. We have also exploited the knowledge of the optimality of threshold policies in the RL framework to provide significant reductions in computational and storage complexities in comparison to traditional RL schemes. We have proposed low-complexity algorithms which aim at maximizing the total system throughput subject to constraints on the blocking probability of high priority users and the offloading probability of low priority users. The proposed algorithms are implemented in a practical ns-3 based SDN-compliant evaluation platform. We now describe the major contributions of the thesis in detail.

In Chapter 3, we have considered the problem of optimal RAT selection in an LTE-WiFi network where we aim to maximize the total system throughput. Additionally, we have considered another formulation where maximizing the total system throughput is subject to a constraint on the voice user blocking probability. These problems have been formulated within the frameworks of MDP and Constrained MDP (CMDP), respectively. Traditional VIA and policy iteration algorithms to determine the optimal policy are computationally prohibitive. This has inspired us to look for structural properties that reduce the number of feasible policies. We have established the existence of a threshold structure of the optimal policy. A policy search over the entire policy space may become computationally inefficient. Therefore, we have proposed algorithms which search the optimal policy only from the set of threshold policies in order to determine the optimal RAT selection policy. We have demonstrated that the proposed algorithms provide significant improvements in computational and storage complexities over the traditional policy iteration algorithm. ns-3 simulation results have established that the algorithms proposed by us perform better than other state-of-the-art schemes in terms of the total system throughput and the blocking probability of voice users.

However, the proposed algorithms in Chapter 3 require the knowledge of the statistics of the arrival processes of voice and data users for the computation of the optimal policy. In practice, these statistics may be difficult to obtain. To address this, an online algorithm for optimal RAT selection has been proposed in Chapter 4 based on a two timescale Q-learning approach. The proposed algorithm can be implemented without any explicit knowledge of the statistics of the arrival processes of voice and data users. The value functions are updated in the faster timescale and the Lagrange Multiplier (LM) in the slower one. The proposed scheme converges to the optimal policy. The proposed approach utilizes an online implementable version of Relative Value Iteration Algorithm (RVIA) within the frameworks of RL [45] and Stochastic Approximation (SA) [78]. Simulation results have demonstrated the convergence behavior of the proposed scheme. Although the proposed Q-learning algorithm asymptotically converges to the optimality, it needs to learn iteratively the value functions for all state-action pairs. Hence, the Q-learning algorithm possesses a high storage complexity. Moreover, due to the associated exploration mechanism, the convergence rate is slow.

To address this issue, in Chapter 4, we have proposed a two timescale PDS learning algorithm which speeds up the learning process by removing the requirement of action exploration mechanism prevailing in the Q-learning algorithm. This approach is based on the reformulation of the RVIA by introducing the notion of the Post-Decision State (PDS). The proposed algorithm has a nice structure, and therefore, it can be implemented online within the SA framework. Instead of the state-action pair values in the case of Q-learning, we need to store only the value functions of the states. Therefore, the storage complexity of the PDS learning algorithm is lower than that of the Q-learning algorithm. We have proved that the PDS based learning algorithm indeed converges to the optimality. ns-3 simulation results have been presented to illustrate the convergence behaviors of the Q-learning and the PDS learning algorithms.

Q-learning and PDS learning approaches in Chapter 4 have been popular and observed to perform well in practice. The convergence speed and the storage complexity of these approaches can be further improved if the underlying threshold structure of the optimal policy can be exploited. To this end, in Chapter 5, we have proposed a novel two timescale structure-aware online learning algorithm which reduces the feasible policy space, thereby offering lesser storage and computational complexity and faster convergence. The main idea behind this algorithm is to compute the gradient of the system metric, i.e., the average reward of the system, with respect to the threshold vector and improve the policy by updating the threshold vector in the direction of the gradient. Since the dynamics of the LM and the threshold vector are not dependent on each other directly, we update the value of the LM and the threshold vector in the same slower timescale. We have proved the convergence of the proposed algorithm to the globally optimal threshold policy. Contrary to the existing learning algorithms where the computational complexity scales linearly with the cardinality of the action space, the proposed algorithm provides a computational complexity of $O(1)$. Simulation results have demonstrated how the knowledge of structural properties affects the convergence speed, when compared to traditional learning schemes. Furthermore, we have observed through simulations that the proposed algorithm outperforms other RAT selection algorithms in the literature in the presence of various realistic network conditions such as channel fading, user mobility and dynamic resource allocation.

The arguments for the proofs of convergences of the proposed learning algorithms to the optimality are developed on the basis of the standard Ordinary Differential Equation (ODE) approach of analyzing SA algorithms [84] as a noisy discretization of a limiting ODE. We have utilized the idea of the two timescale approach [78] in proving the convergence of the coupled iterations of the value functions, the LM and the threshold vector iterates to the optimal policy. While the proofs of convergences of the algorithms are asymptotic in nature, we have demonstrated through simulations that in practice, convergences to close neighborhoods of the optimal policy are achieved in reasonable numbers of iterations.

In Chapters 3, 4 and 5, RAT selection solutions lack the consideration of few practical network aspects, viz., the channel states of users and the control traffic in the backhaul generated due to mobile data offloading. In Chapter 6, we have considered an LTE-WiFi HetNet where users of different priorities are present. We have addressed the problem of optimal RAT selection to maximize the total system throughput subject to constraints on the blocking probability of high priority users and the offloading probability of low priority users. The offloading probability of low priority users is taken into account to accommodate the issue of extra control signaling in the backhaul due to offloading of users from one RAT to another. We have also considered the channel states of users in the system model. We have formulated the optimal RAT selection problem as a CMDP. Furthermore, we have reduced the dimensionality of the action space by eliminating the provably sub-optimal actions using sample path arguments. To address the curse of modeling and the curse of dimensionality associated with the computation of the optimal policy, we have proposed two online heuristic algorithms, namely Myopic with Constraint Satisfaction Algorithm (MCSA) and State-aware Myopic with Constraint Satisfaction Algorithm (SMCSA), for RAT selection. Although greedy in nature, these algorithms are tuned to satisfy the associated constraints. In MCSA, the blocking probability of high priority users and the offloading probability of low priority users do not depend on the statistics of the arrival processes. To address this issue, in SMCSA, blocking and offloading are performed based on the system state. Hence, the blocking probability and the offloading probability gradually rise with increments in arrival rates, similar to the optimal policy. The proposed algorithms have low computational and storage complexities. Moreover, they do not require the knowledge of the statistics of system dynamics and hence, are practically implementable.

In Chapter 7, RAT selection algorithms proposed in Chapter 6 have been implemented in an SDN based evaluation platform developed in ns-3. We have proposed an SDN based network architecture which unify the radio resource management related functionalities of LTE and WiFi networks and modified the existing elements of ns-3 in accordance with the proposed architecture. Experimental observations have indicated that the performances of MCSA and SMCSA are close to optimal. Furthermore, MCSA and SMCSA outperform other traditional RAT selection algorithms under realistic network conditions. Modifications to MCSA and SMCSA have been suggested in Chapter 7 so that user mobility can be handled. We have also provided performance comparisons with other RAT selection algorithms in the literature in the presence of user mobility.

The main theme of this thesis has been the development of computationally efficient and storage-efficient RAT selection algorithms that aim at counteracting the curse of dimensionality and the curse of modeling. While we have addressed the optimal RAT selection problem and investigated the role of LTE-WiFi offloading in RAT selection, several extensions of the considered algorithms and scenarios are possible. In the next section, we provide a discussion on the possible extensions for the future work.

8.2 Future Directions

For the online learning algorithms proposed in Chapters 4 and 5, we have established the proofs of convergence in an asymptotic sense. However, in future, theoretical convergence rates of these algorithms can be analyzed. This will provide us an insight into the numbers of iterations required after which the algorithms converge to close vicinity of the optimal policy. This aspect directly translates into the practical utilities of these algorithms. Furthermore, investigation of the effects of arrival rates and other system parameters on the convergence rate can be performed. Based on that, we can propose adaptive schemes which take up different learning strategies based on the system parameters. In the proposed learning schemes in this thesis, we have considered decreasing step-size schedules. Often, in practice, constant step size parameter is adopted due to its ease of implementation. One future direction may be the investigation of convergence aspects of the proposed algorithms with the consideration of constant step-size schedule.

In this thesis, we have focussed on maximizing the total system throughut subject to certain constraints. However, the proposed framework can be extended to maximize some other utility functions such as fairness (which is a concave function of long-term throughput) subject to the chosen constraints. Extension of our proposed framework to this case may not be straightforward.

In Chapter 6, we have proposed RAT selection algorithms which are implemented in an ns-3 based SDN-compliant evaluation platform in Chapter 7. In future, different aspects of the SDN system can be incorporated in the system model to investigate how they affect the RAT selection decisions. The aspects may include latency and packet loss between the LTE BS/ WiFi AP and the SDN controller. However, optimization problems involving these parameter require indpendent investigation since the system model and solution technique may be entirely different. For example, the optimization problem involving delay and power is a challenging problem [85], even without the consideration of data user offloading.

In this thesis, we have considered the total system throughput, the blocking probability and the offloading probability as the QoSs of interest. However, various other QoS attributes such as delay, packet loss rate and energy efficiency can be considered in future.

As we have discussed in Chapter 1, with the emergence of future 5G system, next generation networks are expected to consist of a number of RATs with varying characteristics. The RAT selection frameworks and the problem formulations considered by us are in general applicable to any RAT including 5G. In future, optimal RAT selection strategy for a 5G-4G HetNet using the proposed framework can be investigated.

We believe that the algorithms proposed in this thesis hold great promises towards practical implementation purposes since they address the issues related to the curse of dimensionality and the curse of modeling. However, different extensions of the proposed algorithms in various other prevalent scenarios in next generation wireless networks can be investigated in future.

Appendices

Appendix A

Selected Proofs in Thesis

A.1 Proof of Lemma 5

Let D_i be the difference operator which is defined as $D_iV(i,j,k) = V(i+1,j,k) - V(i,j,k)$. Similarly, we define the second difference operator as $D_{ii}(.) = D_i(D_i(.))$.

To prove this lemma, we consider two cases, (1) $k \geq k_{th}$ and (2) $k < k_{th}$. We prove the lemma for the first case. Proof of the second case follows in a similar manner. From Lemma 4, we know that for $k \geq k_{th}$, A_2 is better than A_4. Thus, for $k \geq k_{th}$, the choice is between A_1 and A_2. To prove this lemma, we first prove that the value function $V(i,j,k)$ is concave in i. In Lemma 1 and 2, we have already derived the structure of the optimal policy for data user arrival and departure of voice and data users. Now, for $k \geq k_{th}$, with the aid of this, the optimality equation is as follows.

$$\begin{aligned}
V(i,j,k) &= \lambda_v\delta \max\{f(i,j,k) - \beta + V(i,j,k), f(i+1,j,k) + V(i+1,j,k)\}\\
&+ \lambda_d\delta\big(f(i,j+1,k) + V(i,j+1,k)\big) + i\mu_v\delta\big(f(i-1,j+1,k-1) + V(i-1,j+1,k-1)\big)\\
&+ j\mu_d\delta\big(f(i,j,k-1) + V(i,j,k-1)\big) + k\mu_d\delta\big(f(i,j,k-1) + V(i,j,k-1)\big)\\
&+ \big(1 - v(i,j,k)\big)V(i,j,k). \qquad \text{(A.1)}
\end{aligned}$$

Let the components in Equation (A.1) be denoted by $V^1(i,j,k), V^2(i,j,k)$, $V^3(i,j,k)$, $V^4(i,j,k)$,$V^5(i,j,k)$ and $V^6(i,j,k)$, respectively. We prove the concavity of $V(i,j,k)$ component-wise. Start the VIA with $V_0(i,j,k) = 0$. Hence, $V_0(i,j,k)$ is concave in i. Let us assume that $V_{1,n}(i,j,k) = \max\{f(i,j,k) - \beta + V_{n-1}(i,j,k), f(i+1,j,k) + V_{n-1}(i+1,j,k)\}$. Equivalently, $V_{1,n}(i,j,k) = \max\{-\beta + V_{n-1}(i,j,k), R_{L,V} + V_{n-1}(i+1,j,k)\}$. Let us define the function $V_{1,n}(i,j,k,a)$ as follows.

$$V_{1,n}(i,j,k,a) = \begin{cases} -\beta + V_{n-1}(i,j,k), & a = A_1, \\ R_{L,V} + V_{n-1}(i+1,j,k), & a = A_2. \end{cases}$$

By definition,

$$V_{1,n}(i,j,k) = \max_{a \in \{A_1,A_2\}} V_{1,n}(i,j,k,a).$$

Thus, we have,

$$V^1(i,j,k) = \lim_{n\to\infty} V_{1,n}(i,j,k).$$

Let us define $D_iV(i,j,k,a) = V(i+1,j,k,a) - V(i,j,k,a)$.

$$D_iV_{1,n}(i,j,k,a) = \begin{cases} D_iV_{n-1}(i,j,k), & a = A_1, \\ D_iV_{n-1}(i+1,j,k), & a = A_2. \end{cases}$$

$$D_{ii}V_{1,n}(i,j,k,a) = \begin{cases} D_{ii}V_{n-1}(i,j,k), & a = A_1, \\ D_{ii}V_{n-1}(i+1,j,k), & a = A_2. \end{cases}$$

Since $V_{n-1}(i,j,k)$ is concave in i, $V_{1,n}(i,j,k,a)$ is concave in i.

Now, we need to prove that $V_{1,n}(i,j,k)$ is concave in i. In other words, we need to prove that $V_{1,n}(i+2,j,k)+V_{1,n}(i,j,k) \leq 2V_{1,n}(i+1,j,k)$. Let us assume that $a_1 \in \{A_1,A_2\}$ and $a_2 \in \{A_1,A_2\}$ are the maximizing actions in states $(i+2,j,k)$ and (i,j,k), respectively. Therefore,

$$\begin{aligned} 2V_{1,n}(i+1,j,k) &\geq V_{1,n}(i+1,j,k,a_1) + V_{1,n}(i+1,j,k,a_2) \\ &= V_{1,n}(i+2,j,k,a_1) + V_{1,n}(i,j,k,a_2) - D_iV_{1,n}(i+1,j,k,a_1) + D_iV_{1,n}(i,j,k,a_2). \end{aligned}$$

Let us take $X = D_iV_{1,n}(i,j,k,a_2) - D_iV_{1,n}(i+1,j,k,a_1)$. To prove that $V_{1,n}(i,j,k)$ is concave in i, we need to prove that $X \geq 0$. There are four cases as described below.

Case 1 : $a_1 = a_2 = A_1$,

$$X = D_iV_{n-1}(i,j,k) - D_iV_{n-1}(i+1,j,k) = -D_{ii}V_{n-1}(i,j,k) \geq 0.$$

Case 2 : $a_1 = A_1, a_2 = A_2$,

$$X = D_iV_{n-1}(i+1,j,k) - D_iV_{n-1}(i+1,j,k) = 0.$$

Case 3 : $a_2 = a_2 = A_2$,

$$X = D_i V_{n-1}(i+1,j,k) - D_i V_{n-1}(i+2,j,k) = -D_{ii} V_{n-1}(i+1,j,k) \geq 0.$$

Case 4 : $a_1 = A_2, a_2 = A_1$,

$$X = D_i V_{n-1}(i,j,k) - D_i V_{n-1}(i+2,j,k) = -D_{ii} V_{n-1}(i,j,k) - D_{ii} V_{n-1}(i+1,j,k) \geq 0.$$

Thus, it is proved that $V_{1,n}(i,j,k)$ is concave in i. Since this holds for every n and every β, $V^1(i,j,k)$ is concave in i.

Similarly, in the case of the second component, let $V_{2,n}(i,j,k) = f(i+1,j,k) + V_{n-1}(i,j+1,k)$. Thus, $D_{ii} V_{2,n}(i,j,k) = D_{ii} V_{n-1}(i,j+1,k)$. Therefore, $V_{2,n}(i,j,k)$ is concave in i. Similarly, other components also can be proved to be concave in i. Therefore, $V(i,j,k)$ is concave in i.

Let us define $x(i,j,k) = -\beta - R_{L,V}$. In order to prove this lemma, we know that if state (i,j,k) is blocking, then $V(i+1,j,k) - V(i,j,k) \leq x(i,j,k)$. Due to concavity of $V(i,j,k)$, $V(i+2,j,k) - V(i+1,j,k) \leq V(i+1,j,k) - V(i,j,k)$. Now, $x(i,j,k) = x(i+1,j,k)$. As a consequence, $V(i+2,j,k) - V(i+1,j,k) \leq x(i+1,j,k)$. Thus, it is proved that if the state (i,j,k) is blocking, then the state $(i+1,j,k)$ is also blocking.

To prove that if the state (i,j,k) is blocking, then the state $(i,j+1,k)$ is also blocking, we first need to prove that the value function is submodular in (i,j). In other words, we need to prove that $V_n(i+1,j,k) + V_n(i,j+1,k) \geq V_n(i,j,k) + V_n(i+1,j+1,k)$. Similar to the previous proof, we prove the above statement component-wise. Let us assume that a_1 and a_2 are the maximizing actions in states (i,j,k) and $(i+1,j+1,k)$, respectively. Start the VIA with $V_0(i,j,k) = 0$. Therefore, $V_0(i,j,k)$ is submodular in (i,j). In other words, $D_{ij} V_0(i,j,k) \leq 0$. We have,

$$\begin{aligned} &V_{1,n}(i+1,j,k) + V_{1,n}(i,j+1,k) \geq V_{1,n}(i+1,j,k,a_1) + V_{1,n}(i,j+1,k,a_2) \\ &= V_{1,n}(i,j,k,a_1) + V_{1,n}(i+1,j+1,k,a_2) + D_i V_{1,n}(i,j,k,a_1) - D_i V_{1,n}(i,j+1,k,a_2). \end{aligned}$$

Now, we consider four possible cases.

Case 1 : $a_1 = a_2 = A_1$,

$$\begin{aligned} &D_i V_{1,n}(i,j,k,a_1) - D_i V_{1,n}(i,j+1,k,a_2) = D_i V_{n-1}(i,j,k) - D_i V_{n-1}(i,j+1,k) \\ &= -D_{ij} V_{n-1}(i,j,k) \geq 0. \end{aligned}$$

Case 2 : $a_1 = a_2 = A_2$,

$$D_iV_{1,n}(i,j,k,a_1) - D_iV_{1,n}(i,j+1,k,a_2) = D_iV_{n-1}(i+1,j,k) - D_iV_{n-1}(i+1,j+1,k)$$
$$= -D_{ij}V_{n-1}(i+1,j,k) \geq 0.$$

Case 3 : $a_1 = A_1, a_2 = A_2$,

$$D_iV_{1,n}(i,j,k,a_1) - D_iV_{1,n}(i,j+1,k,a_2) = D_iV_{n-1}(i,j,k) - D_iV_{n-1}(i+1,j+1,k)$$
$$= D_iV_{n-1}(i,j,k) - D_iV_{n-1}(i,j+1,k) + D_iV_{n-1}(i,j+1,k) - D_iV_{n-1}(i+1,j+1,k)$$
$$= -D_{ij}V_{n-1}(i,j,k) - D_{ii}V_{n-1}(i,j+1,k) \geq 0.$$

Case 4 : $a_1 = A_2, a_2 = A_1$,

$$V_{1,n}(i+1,j,k) + V_{1,n}(i,j+1,k) \geq V_{1,n}(i+1,j,k,a_2) + V_{1,n}(i,j+1,k,a_1)$$
$$= -\beta + V_{n-1}(i+1,j,k) + R_{L,V} + V_{n-1}(i+1,j+1,k)$$
$$= V_{1,n}(i,j,k,2) + V_{1,n}(i+1,j+1,k,1)$$
$$= V_{1,n}(i,j,k) + V_{1,n}(i+1,j+1,k).$$

Thus, it is proved that $V_{1,n}(i,j,k)$ is submodular in (i,j).

Similarly, in the case of the second component, we have, $V_{2,n}(i,j,k) = f(i,j+1,k) + V_n(i,j+1,k)$. Therefore, we have, $D_{ij}V_{2,n}(i,j,k) = D_{ij}V_n(i,j+1,k) \leq 0$. Similarly, other components also can be proved to be submodular in (i,j). Therefore, the value function is submodular in (i,j).

Now, if the state (i,j,k) is blocking then we have, $V(i+1,j,k) - V(i,j,k) \leq x(i,j,k)$. Again, we have, $x(i,j,k) = x(i,j+1,k)$. Due to submodularity, we have $V(i+1,j+1,k) - V(i,j+1,k) \leq V(i+1,j,k) - V(i,j,k) \leq x(i,j,k) = x(i,j+1,k)$. Thus, in the case of voice arrival, if A_1 is optimal in the state (i,j,k), then in the state $(i,j+1,k)$ also, A_1 is optimal.

Proof of (ii) follows directly from the proof of part (i).

A.2 Proof of Lemma 6

Let E_i be another difference operator defined as $E_iV(i,j,k) = V(i+1,j-1,k+1) - V(i,j,k)$. We define the second difference operator as $E_{ii}(.) = E_i(E_i(.))$. Similarly, we define $F_iV(i,j,k) = V(i+1,j-1,k) - V(i,j,k)$.

To prove this lemma, we consider two cases, (1) $k \geq k_{th}$ and (2) $k < k_{th}$. We demonstrate the proof of the lemma for the first case. Proof for the second case follows in similar manner. To prove this lemma, we first need to prove that for $(i+j) = C$, the difference of value functions $V(i+1, j-1, k+1) - V(i, j, k)$ is decreasing in i. For $(i+j) = C$ and $k \geq k_{th}$, the optimality equation can be described as

$$
\begin{aligned}
V(i,j,k) &= \lambda_v \delta \max\{f(i,j,k) - \beta + V(i,j,k), f(i+1,j-1,k+1) + V(i+1,j-1,k+1)\} \\
&+ \lambda_d \delta\big(f(i,j,k+1) + V(i,j,k+1)\big) + i\mu_v \delta\big(f(i-1,j+1,k-1) + V(i-1,j+1,k-1)\big) \\
&+ j\mu_d \delta\big(f(i,j,k-1) + V(i,j,k-1)\big) + k\mu_d \delta\big(f(i,j,k-1) + V(i,j,k-1)\big) \\
&+ \big(1 - v(i,j,k)\big)V(i,j,k).
\end{aligned}
$$

Let us assume that $V_{1,n}(i,j,k,a) = \max\{f(i,j,k) - \beta + V_{n-1}(i,j,k), f(i+1,j-1,k+1) + V_{n-1}(i+1,j-1,k+1)\}$.
Equivalently, $V_{1,n}(i,j,k,a) = \max\{-\beta + V_{n-1}(i,j,k), R_{L,V} - R_{L,D} + \tilde{R}_{W,D}(k) + V_{n-1}(i+1,j-1,k+1)\}$.

In other words,

$$
V_{1,n}(i,j,k,a) = \begin{cases} -\beta + V_{n-1}(i,j,k), & a = A_1, \\ R_{L,V} - R_{L,D} + \tilde{R}_{W,D}(k) + V_{n-1}(i+1,j-1,k+1), & a = A_4. \end{cases}
$$

We prove the above claim component-wise. Start the VIA with $V_0(i,j,k) = 0$. Therefore, $V_0(i+1,j-1,k+1) - V_0(i,j,k)$ is decreasing in i. We also have, $E_i F_i V_0(i,j,k) \leq 0$. Now, Let us define $E_i V(i,j,k,a) = V(i+1,j-1,k+1,a) - V(i,j,k,a)$ and $F_i V(i,j,k,a) = V(i+1,j-1,k,a) - V(i,j,k,a)$.

$$
E_i V_{1,n}(i,j,k,a) = \begin{cases} E_i V_{n-1}(i,j,k), & a = A_1, \\ E_i V_{n-1}(i+1,j-1,k+1), & a = A_4. \end{cases}
$$

Therefore, $E_i F_i V_{1,n}(i,j,k,a) \leq 0$.

$$
E_{ii} V_{1,n}(i,j,k,a) = \begin{cases} E_{ii} V_{n-1}(i,j,k), & a = A_1, \\ E_{ii} V_{n-1}(i+1,j-1,k+1), & a = A_4. \end{cases}
$$

Therefore, $V_{1,n}(i+1,j-1,k+1,a) - V_{1,n}(i,j,k,a)$ is decreasing in i.
Now, we need to prove that $V_{1,n}(i+1,j-1,k+1) - V_{1,n}(i,j,k)$ is decreasing in i. In

other words, we need to prove that $V_{1,n}(i+2,j-2,k+1)+V_{1,n}(i,j,k) \leq V_{1,n}(i+1,j-1,k+1)+V_{1,n}(i+1,j-1,k)$. Let us assume that $a_1 \in \{A_1, A_4\}$ and $a_2 \in \{A_1, A_4\}$ are the maximizing actions in states $(i+2,j-2,k+1)$ and (i,j,k), respectively. Therefore,

$$\begin{aligned}
&V_{1,n}(i+1,j-1,k+1)+V_{1,n}(i+1,j-1,k)\\
&\geq V_{1,n}(i+1,j-1,k+1,a_2)+V_{1,n}(i+1,j-1,k,a_1)\\
&= V_{1,n}(i,j,k,a_2)+V_{1,n}(i+2,j-2,k+1,a_1)\\
&- E_iV_{1,n}(i+1,j-1,k,a_1)+E_iV_{1,n}(i,j,k,a_2).
\end{aligned}$$

Let us take $Y = E_iV_{1,n}(i,j,k,a_2) - E_iV_{1,n}(i+1,j-1,k,a_1)$. To prove that $V_{1,n}(i+1,j-1,k+1) - V_{1,n}(i,j,k)$ is decreasing in i, we need to prove that $Y \geq 0$. There are four cases as described below.

Case 1 : $a_1 = a_2 = A_1$,

$$Y = E_iV_{n-1}(i,j,k) - E_iV_{n-1}(i+1,j-1,k) = -E_iF_iV_{n-1}(i,j,k) \geq 0.$$

Case 2 : $a_1 = A_4, a_2 = A_1$,

$$\begin{aligned}
Y &= E_iV_{n-1}(i,j,k) - E_iV_{n-1}(i+2,j-2,k+1)\\
&= -E_{ii}V_{n-1}(i,j,k) - E_iF_i(i+1,j-1,k+1) \geq 0.
\end{aligned}$$

Case 3 : $a_2 = a_2 = A_4$,

$$\begin{aligned}
Y &= E_iV_{n-1}(i+1,j-1,k+1) - E_iV_{n-1}(i+2,j-2,k+1)\\
&= -E_iF_iV_{n-1}(i+1,j-1,k+1) \geq 0.
\end{aligned}$$

Case 4 : $a_1 = A_1, a_2 = A_4$,

$$\begin{aligned}
&V_{1,n}(i+1,j-1,k+1)+V_{1,n}(i+1,j-1,k)\\
&\geq V_{1,n}(i+1,j-1,k+1,a_1)+V_{1,n}(i+1,j-1,k,a_2)\\
&= -\beta + V_{n-1}(i+1,j-1,k+1) + R_{L,V} - R_{L,D} + \tilde{R}_{W,D}(k) + V_{n-1}(i+2,j-2,k+1)\\
&= V_{1,n}(i+2,j-2,k+1,1)+V_{1,n}(i,j,k,4)\\
&= V_{1,n}(i+2,j-2,k+1)+V_{1,n}(i,j,k).
\end{aligned}$$

Thus, it is proved that $V_{1,n}(i+1,j-1,k+1) - V_{1,n}(i,j,k)$ is decreasing in i. Since this holds for every n and every β, $V^1(i+1,j-1,k+1) - V^1(i,j,k)$ is decreasing in i. Let $V_{2,n}(i,j,k) = f(i,j,k+1) + V_{n-1}(i,j,k+1)$. Thus, $E_{ii}V_{2,n}(i,j,k) = E_{ii}V_{n-1}(i,j,k+1)$.

Therefore, $V_{2,n}(i{+}1, j{-}1, k{+}1) - V_{2,n}(i, j, k)$ is decreasing in i. Similarly, other components can be proved to be decreasing in i. Therefore, $V(i{+}1, j{-}1, k{+}1) - V(i, j, k)$ is decreasing in i.

Similar to Lemma 5, using this property it can be proved that if the optimal action for voice user arrival in the state (i, j, k) is blocking, then the optimal action in the state $(i+1, j-1, k)$ is also blocking.

Proof of (ii) follows directly from the proof of part (i).

A.3 Proof of Theorem 1

The idea of the proof is similar to that of the online algorithm in [85]. The arguments are developed on the basis of the standard Ordinary Differential Equation (ODE) approach [78, Chapter 2] of analyzing stochastic approximation algorithms, which can be considered as a noisy discretization of a limiting ODE. Learning parameters are considered as discrete time steps and iterates, when linearly interpolated, are compared with the trajectory of the ODE. Standard assumptions on step-sizes, viz., Equations (4.2) and (4.3), ensure that the discretization error and error due to noise is asymptotically negligible. Therefore, the iterates track the behavior of the associated ODEs asymptotically and hence, converge to the globally asymptotically stable equilibrium a.s. However, for the sake of completeness, we restate the details of the proof for the proposed algorithm.

We rewrite the update of fast and slow timescale as

$$\begin{aligned} \hat{V}_{n+1}(\hat{s}) &= \hat{V}_n(\hat{s}) + g(\gamma(\hat{s}, n))\{\max_a[r(s', a; \beta) + \hat{V}_n(\hat{s'})] - \hat{V}_n(\hat{s^*}) - \hat{V}_n(\hat{s})\}, \\ \hat{V}_{n+1}(\hat{s''}) &= \hat{V}_n(\hat{s''}) \quad \forall \hat{s''} \neq \hat{s}. \end{aligned} \tag{A.2}$$

and

$$\beta_{n+1} = \Lambda[\beta_n + h(n)(B_n - B_{\max})]. \tag{A.3}$$

Due to the condition $\dfrac{h(n)}{g(n)} \to 0$, we have a two-timescale leader-follower behavior. The PDS value functions are updated in the faster timescale and the LM in the slower one. Let $H_1 : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ be a map defined by (for $\hat{s} \in \mathcal{S}$),

$$H_1(\hat{s}) = \sum_{s'} p(\hat{s}, s') \max_a[r(s', a; \beta) + \hat{V}_n(\hat{s'})] - \hat{V}_n(\hat{s^*}). \tag{A.4}$$

Note that the knowledge of $p(\hat{s}, s')$ is not required for this algorithm. We consider them in Equation (A.4) for the sake of analysis. Following the two time-scale analysis in [78, Section 6.1], we analyze Equation (A.2) first keeping the LM constant. This translates into analyzing the following limiting ODE which tracks Equation (A.2).

$$\dot{\hat{V}}(t) = H_1(\hat{V}(t)) - \hat{V}(t). \tag{A.5}$$

Following [86], as $t \to \infty$, $\hat{V}(t)$ converges to the unique fixed point of $H_1(.)$, i.e., $\hat{V}$ such that

$$H_1(\hat{V}) = \hat{V}.$$

Note that, this scheme is similar to [86, 87].

Lemma 13. *The PDS value function iterates and the LM iterates are bounded a.s.*

Proof. Consider a mapping $H_0 : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ as follows.

$$H_0(\hat{s}) = \sum_{s'} p(\hat{s}, s')[\max_a \hat{V}_n(\hat{s'}) - \hat{V}_n(\hat{s^*})]. \tag{A.6}$$

Note that Equation (A.6) corresponds to Equation (A.4) with zero immediate reward. Now, we have, $\lim_{b\to\infty} \frac{H_1(b\hat{V})}{b} = H_0(\hat{V})$. Also, the globally asymptotically stable equilibrium of the limiting ODE $\dot{\hat{V}}(t) = H_0(\hat{V}(t)) - \hat{V}(t)$, which is a scaled limit of the original ODE (A.5), is the origin (Using arguments of [85, Lemma 1]). The boundedness of $\hat{V}$ follows from [84].

The physical interpretation of this approach, as stated in [78], is as follows. If the iterates of the PDSs become unbounded along a subsequence, then a suitably scaled version of the original ODE approximately follows the limiting ODE. Since the origin is the globally asymptotically stable equilibrium of the scaled ODE, the scaled ODE must return towards the origin. Therefore, the original PDS iterates also begin to move towards a bounded set, ensuring stability of the iterates.

The iterates of β are bounded since they are constrained to remain in $[0, L]$, by definition. □

Lemma 14. *We have $\hat{V}_n \to \hat{V}^{\beta_n}$ a.s., where $\hat{V}^{\beta_n}$ is the value function of the PDS for $\beta = \beta_n$.*

Proof. It can be seen that the LM is varied on a much slower timescale than $\hat{V}$. Therefore, the $\hat{V}$ iterations consider LM to be almost constant. Therefore, the β iterations can be written as $\beta_{n+1} = \beta_n + \gamma(n)$, where $\gamma(n) = O(h(n)) = o(g(n))$. Thus, the limiting ODEs associated with the iterates are $\dot{\hat{V}}(t) = H_1(\hat{V}(t)) - \hat{V}(t)$ and $\dot{\beta}(t) = 0$. Since $\dot{\beta}(t) = 0$, for analyzing the $\hat{V}(.)$ iterates, it is sufficient to consider the ODE $\dot{\hat{V}}(t) = H_1(\hat{V}(t)) - \hat{V}(t)$, for a fixed value of the LM β. The rest of the proof follows from [88]. □

The lemma presented next proves that the LM iterates converge to the optimal LM β^* and hence, $(\hat{V}_n, \beta_n)$ converges to $(\hat{V}, \beta^*)$.

Lemma 15. *LM iterates converge to β^*.*

Proof. The proof outline is similar to that of [85]. Note that in Equations (4.10) and (4.11), the iterations for the PDSs determine the maximum of a Lagrangian for the policy keeping the LM almost constant. Therefore, the limiting ODE for the LM iterations is same as a gradient descent for the Lagrangian which minimizes over the PDS value functions. The result follows from "envelope theorem" [88], which allows the interchange of min and gradient operator.

For each fixed policy, the reward is linear in β with negative slope. Thus $\hat{V}(.)$, which is by standard argument, the upper envelope, is piecewise linear with finitely many linear pieces and convex in β for each component. Let the stationary randomized policy M has a unique stationary distribution η^M. Let $E_M[.]$ denote the expectation under a stationary randomized policy M. Let $\mathcal{L}(M, \beta) = E_M[r(X_n, Z_n) - \beta c(X_n, Z_n)]$ and $I(\beta) = \max_M \mathcal{L}(M, \beta)$, where the controlled Markov chain $\{X_n\}$ on a finite state space $\mathcal{S}$ is controlled by a control process $\{Z_n\}$ taking values in a finite action space. Therefore, I is piecewise linear and convex. Define $z(\beta) = \sum_{s,a} \eta^{M^\beta}(s)[r(s,a) - \beta c(s,a)]$, where M^β is an optimal stationary randomized policy when multiplier β is used. The limiting ODE is

$$\dot{\beta}(t) = z(\beta(t)).$$

According to the results in [78, Section 10.2], stochastic gradient descent for a convex function tracks this ODE and hence converges to the optimal β^*. Thus, the desired β^* is the global minimum, as there is no local minimum which is not a global minimum. □

A.4 Proof of Lemma 7

Let us denote $D_N\hat{V}(i,j,k) = \hat{V}_N(i+1,j,k) - \hat{V}_N(i,j,k)$. Therefore, we need to prove that $D_{N+1}\hat{V}(i,j,k) \leq D_N\hat{V}(i,j,k)$, $\forall N$. We prove this using induction arguments on VIA. When $N=0$, $\hat{V}_N(i,j,k)=0$. Thus, $D_0\hat{V}(i,j,k)=0$. We have,

$$\begin{aligned}
&\hat{V}_{N+1}(i,j,k) = \lambda_v \max\{f(i,j,k) - \beta + \hat{V}_N(i,j,k), f(i+1,j,k) + \hat{V}_N(i+1,j,k),\\
&f(i+1,j-1,k+1) + \hat{V}_N(i+1,j-1,k+1)\} + \lambda_d \max\{f(i,j+1,k) + \hat{V}_N(i,j+1,k),\\
&f(i,j,k+1) + \hat{V}_N(i,j,k+1)\} + i\mu_v \max\{f(i-1,j,k) + \hat{V}_N(i-1,j,k), f(i-1,j+1,k-1)+\\
&\hat{V}_N(i-1,j+1,k-1)\} + j\mu_d \max\{f(i,j-1,k) + \hat{V}_N(i,j-1,k), f(i,j,k-1) + \hat{V}_N(i,j,k-1)\}\\
&+ k\mu_d \max\{f(i,j,k-1) + \hat{V}_N(i,j,k-1), f(i,j-1,k) + \hat{V}_N(i,j-1,k)\}\\
&+ (1 - v(i,j,k))\hat{V}_N(i,j,k),
\end{aligned}$$

where $f(i,j,k) = iR_{L,V} + jR_{L,D} + kR_{W,D}(k)$.

For $k \geq k_{th}$, we prove the claim component wise. Proof for $k < k_{th}$ follows in a similar way. Let us denote the p^{th} component of $\hat{V}_N(i,j,k)$ by $\hat{V}_N^p(i,j,k)$, for $p = 1,2,\ldots,6$. Therefore, we have,

$$\begin{aligned}
&\hat{V}_1^1(i,j,k) = \max\{f(i,j,k) - \beta + \hat{V}_0(i,j,k), f(i+1,j,k)+\\
&\hat{V}_0(i+1,j,k), f(i+1,j-1,k+1) + \hat{V}_0(i+1,j-1,k+1)\}.
\end{aligned}$$

We know, for $k \geq k_{th}$, A_4 is suboptimal (Property 2). Therefore, subtracting $f(i,j,k)$ from the rewards corresponding to both actions,

$$\hat{V}_1^1(i,j,k) = \max\{-\beta + \hat{V}_0(i,j,k), R_{L,V} + \hat{V}_0(i+1,j,k)\}.$$

Therefore, $D_1\hat{V}^1(i,j,k) = 0$. Again, we have,

$$\hat{V}_1^2(i,j,k) = \max\{f(i,j+1,k) + \hat{V}_0(i,j+1,k), f(i,j,k+1) + \hat{V}_0(i,j,k+1)\}.$$

Or, equivalently,

$$\hat{V}_1^2(i,j,k) = \max\{R_{L,D} + \hat{V}_0(i,j+1,k), \tilde{R}_{W,D}(k) + \hat{V}_0(i,j,k+1)\}.$$

We know, for $k \geq k_{th}$, A_2 is optimal. Thus,

$$\hat{V}_1^2(i,j,k) = R_{L,D} + \hat{V}_0(i,j+1,k).$$

Therefore, $D_1\hat{V}^2(i,j,k) = 0$. Similarly, other components can be proved to be equal to zero. Therefore, $D_1\hat{V}(i,j,k) \leq D_0\hat{V}(i,j,k)$.

Now, assume that the claim holds for arbitrary N, i.e., $D_{N+1}\hat{V}(i,j,k) \leq D_N\hat{V}(i,j,k)$. Now, we need to prove that $D_{N+2}\hat{V}(i,j,k) \leq D_{N+1}\hat{V}(i,j,k)$. Similar to the case of $N=0$, we have, $D_{N+2}\hat{V}^p(i,j,k) - D_{N+1}\hat{V}^p(i,j,k) = D_{N+1}\hat{V}(i,j,k) - D_N\hat{V}(i,j,k) \leq 0$, $p \neq 1$. Therefore, it remains to prove that $D_{N+2}\hat{V}^1(i,j,k) - D_{N+1}\hat{V}^1(i,j,k) \leq 0$. Let $a_0, a_1 \in \{A_1, A_2\}$ be the maximizing actions in PDSs (i,j,k) and $(i+1,j,k)$, respectively, at $(N+2)^{\text{th}}$ iteration. Let $b_0, b_1 \in \{A_1, A_2\}$ be the maximizing actions in PDSs (i,j,k) and $(i+1,j,k)$, respectively, at $(N+1)^{\text{th}}$ iteration. Now, it is not possible to have $a_1 = A_2$ and $b_0 = A_1$. This is because if $b_0 = A_1$, we have, $D_N\hat{V}(i,j,k) \leq -\beta - R_{L,V}$. Using concavity with respect to i (shown in Chapter 3), we must have $D_N\hat{V}(i+1,j,k) \leq -\beta - R_{L,V}$. However, if $a_1 = A_2$, we have $D_{N+1}\hat{V}(i+1,j,k) \geq -\beta - R_{L,V}$, which contradicts the inductive assumption. Therefore, we consider three cases as follows. In every case, given values of a_1 and b_0, if the inequality holds for any chosen values of a_0 and b_1, then the inequality must hold for maximizing actions a_0 and b_1.

1) If $a_1 = b_0 = A_1$, then we can choose $a_0 = b_1 = A_1$, resulting in

$$D_{N+2}\hat{V}^1(i,j,k) - D_{N+1}\hat{V}^1(i,j,k) = D_{N+1}\hat{V}(i,j,k) - D_N\hat{V}(i,j,k) \leq 0.$$

2) If $a_1 = b_0 = A_2$, then we can choose $a_0 = b_1 = A_2$, and the inequality satisfies due to same reasoning as above.

3) If $a_1 = A_1$ and $b_0 = A_2$, then we can choose $a_0 = A_2$ and $b_1 = A_1$. In this case,

$$\begin{aligned} &D_{N+2}\hat{V}^1(i,j,k) - D_{N+1}\hat{V}^1(i,j,k) = -\beta + \hat{V}_{N+1}(i+1,j,k) \\ &- R_{L,V} - \hat{V}_{N+1}(i+1,j,k) + \beta - \hat{V}_N(i+1,j,k) + R_{L,V} + \\ &\hat{V}_N(i+1,j,k) = 0. \end{aligned}$$

Thus, we have, $D_{N+2}\hat{V}(i,j,k) \leq D_{N+1}\hat{V}(i,j,k)$.

A.5 Proof of Lemma 8

Let us denote $E_N\hat{V}(i,j,k) = \hat{V}_N(i+1,j-1,k+1) - \hat{V}_N(i,j,k)$. Therefore, we need to prove that $E_{N+1}\hat{V}(i,j,k) \leq E_N\hat{V}(i,j,k)$. When $N=0$, $\hat{V}_N(i,j,k) = 0$. Thus, $E_0\hat{V}(i,j,k) = 0$.

We have,

$$\hat{V}_1^1(i,j,k) = \max\{-\beta + \hat{V}_0(i,j,k), R_{L,V} - R_{L,D} + \tilde{R}_{W,D}(k) + \hat{V}_0(i+1,j-1,k+1)\}.$$

Now, we have the following cases.

1) If $-\beta \geq R_{L,V} - R_{L,D} + \tilde{R}_{W,D}(k)$, then using Assumption 2, we have $-\beta \geq R_{L,V} - R_{L,D} + \tilde{R}_{W,D}(k+1)$. Therefore, $E_1\hat{V}^1(i,j,k) = 0$.

2) If $-\beta \leq R_{L,V} - R_{L,D} + \tilde{R}_{W,D}(k+1)$, then we have $E_1\hat{V}^1(i,j,k) = \tilde{R}_{W,D}(k+1) - \tilde{R}_{W,D}(k) \leq 0$ (using Assumption 2).

3) If $\tilde{R}_{W,D}(k) > -\beta - R_{L,V} + R_{L,D} > \tilde{R}_{W,D}(k+1)$, we have $E_1\hat{V}^1(i,j,k) = -\beta - (R_{L,V} - R_{L,D} + \tilde{R}_{W,D}(k)) < 0$.

Similar to Lemma 7, other components can be proved to be equal to zero. Therefore, $E_1\hat{V}(i,j,k) \leq E_0\hat{V}(i,j,k)$.

Now, assume that the claim holds for arbitrary N, i.e., $E_{N+1}\hat{V}(i,j,k) \leq E_N\hat{V}(i,j,k)$. Now, we need to prove that $E_{N+2}\hat{V}(i,j,k) \leq E_{N+1}\hat{V}(i,j,k)$. Similar to the case of $N = 0$, we have, $E_{N+2}\hat{V}^p(i,j,k) - E_{N+1}\hat{V}^p(i,j,k) = E_{N+1}\hat{V}(i,j,k) - E_N\hat{V}(i,j,k) \leq 0$, $p \neq 1$. Therefore, it remains to prove that $E_{N+2}\hat{V}^1(i,j,k) - E_{N+1}\hat{V}^1(i,j,k) \leq 0$. Let $a_0, a_1 \in \{A_1, A_4\}$ be the maximizing actions in PDSs (i,j,k) and $(i+1, j-1, k+1)$, respectively, at $(N+2)^{\text{th}}$ iteration. Let $b_0, b_1 \in \{A_1, A_4\}$ be the maximizing actions in PDSs (i,j,k) and $(i+1, j-1, k+1)$, respectively, at $(N+1)^{\text{th}}$ iteration. Therefore, we consider four cases as follows.

1) If $a_1 = b_0 = A_1$, we choose $a_0 = b_1 = A_1$, resulting in

$$E_{N+2}\hat{V}^1(i,j,k) - E_{N+1}\hat{V}^1(i,j,k) = E_{N+1}\hat{V}(i,j,k) - E_N\hat{V}(i,j,k) \leq 0.$$

2) If $a_1 = b_0 = A_4$, then we can choose $a_0 = b_1 = A_4$, and the inequality satisfies due to same reasoning as above.

3) If $a_1 = A_1$ and $b_0 = A_4$, then we can choose $a_0 = A_4$ and $b_1 = A_1$. Similar to Case 3 in Lemma 7, $E_{N+2}\hat{V}^1(i,j,k) - E_{N+1}\hat{V}^1(i,j,k) = 0$.

4) If $a_1 = A_4$ and $b_0 = A_1$, then we can choose $a_0 = A_1$ and $b_1 = A_4$.

$$E_{N+2}\hat{V}^1(i,j,k) - E_{N+1}\hat{V}^1(i,j,k) = [E_{N+1}\hat{V}(i+1,j-1,k+1) + E_{N+1}\hat{V}(i,j,k)] - [E_N\hat{V}(i+1,j-1,k+1) + E_N\hat{V}(i,j,k)] \leq 0.$$

Thus, we have $E_{N+2}\hat{V}(i,j,k) \leq E_{N+1}\hat{V}(i,j,k)$.

A.6 Proof of Lemma 9

We prove this lemma componentwise, i.e., we prove that the average reward is unimodal with respect to each $\theta(T)$, hence proving unimodality with respect to θ. Consider value Iteration Algorithm (VIA). Let the value function of PDS (i,j,k) at N^{th} iteration be denoted by $\hat{V}_N(i,j,k)$. We consider both the cases, where after a certain threshold on i, the optimal action changes from (1) A_2 to A_1 and (2) A_4 to A_1, respectively.

Proof of (1): We know that if the optimal action in PDS (i,j,k) is A_1, then we have, $\hat{V}(i+1,j,k) - \hat{V}(i,j,k) \leq -\beta - R_{L,V}$. Since VIA converges to the threshold policy with threshold θ^*, there exists an integer N_1 such that for all $N \geq N_1$, $\hat{V}_N(i+1,j,k) - \hat{V}_N(i,j,k) \leq -\beta - R_{L,V}$ for $i \geq \theta^*(T)$ and $\hat{V}_N(i+1,j,k) - \hat{V}_N(i,j,k) \geq -\beta - R_{L,V}$ for $i < \theta^*(T)$. Let us assume U_N for $N \geq 1$ to be
$U_N = \min\{i \in \mathbb{N}_0 : \hat{V}_N(i+1,j,k) - \hat{V}_N(i,j,k) \leq -\beta - R_{L,V}\}$. If U_N is empty, then we assume $U_N = M(T)$. Hence, U_N can be referred to as the optimal threshold at N^{th} iteration of VIA. Now, since $\hat{V}_N(i+1,j,k) - \hat{V}_N(i,j,k) \leq -\beta - R_{L,V}$ implies $\hat{V}_{N+1}(i+1,j,k) - \hat{V}_{N+1}(i,j,k) \leq -\beta - R_{L,V}$ (using Lemma 7), U_N monotonically decreases with N. Also, $\lim_{N\to\infty} U_N = \theta^*(T)$.

Given a threshold $\theta_1(T)$ ($\theta^*(T) < \theta_1(T) \leq M(T)$), we consider a re-defined problem where blocking is not allowed at any state with $i < \theta_1(T)$, i.e., the action in these states is to always accept in LTE. In this case also, $\hat{V}_N(i+1,j,k) - \hat{V}_N(i,j,k) \leq -\beta - R_{L,V}$ implies $\hat{V}_{N+1}(i+1,j,k) - \hat{V}_{N+1}(i,j,k) \leq -\beta - R_{L,V}$. Let in this case, $N_{\theta_1(T)} = \min\{N : U_N \leq \theta_1(T)\}$. That is, $N_{\theta_1(T)}$ is the first iteration of VIA at which the threshold drops to $\theta_1(T)$. Therefore, $N_{\theta_1(T)}$ must be finite, because for $N < N_{\theta_1(T)}$, the value function iterates take exactly the same values as that of the original problem. The fact that blocking is not allowed for $i < \theta_1(T)$ makes no difference (for $N < N_{\theta_1(T)}$) since blocking is never chosen as the optimal action in these states upto $N^{\text{th}}_{\theta_1(T)}$ iteration in the original problem. Since the original problem converges to the threshold $\theta^*(T)$, at $N^{\text{th}}_{\theta_1(T)}$ iteration we have, $\hat{V}_{N+1}(\theta_1(T)+1,j,k) - \hat{V}_{N+1}(\theta_1(T),j,k) \leq -\beta - R_{L,V}$, and the same inequality holds for this redefined problem. Since $\hat{V}_N(i+1,j,k) - \hat{V}_N(i,j,k) \leq -\beta - R_{L,V}$ implies $\hat{V}_{N+1}(i+1,j,k) - \hat{V}_{N+1}(i,j,k) \leq -\beta - R_{L,V}$ (Lemma 7), the same inequality holds for all $N \geq N_{\theta_1(T)}$. Hence, the VIA converges to the policy with threshold $\theta_1(T)$ in the redefined problem, which implies that $\theta_1(T)$-threshold policy is superior to $\theta_1(T)+1$-threshold policy. Since this argument holds for any threshold $\theta_1(T)$, we can claim that the average reward is monotonically decreasing with $\theta_1(T)$, for $\theta_1(T) > \theta^*(T)$.

Therefore, if we have $\rho(\theta(T)) \geq \rho(\theta(T)+1)$, we must have $\theta(T) \geq \theta^*(T)$. Thus, we have, $\rho(\theta(T)+1) \geq \rho(\theta(T)+2)$. This completes the proof of unimodality for (1).

Proof of (2): In this case, if the optimal action in PDS (i,j,k) is blocking, we have $\hat{V}(i+1,j-1,k+1) - \hat{V}(i,j,k) \leq -\beta - R_{L,V} + R_{L,D} - \tilde{R}_{W,D}(k)$. Rest of the proof is similar to (1) (Using Lemma 8).

A.7 Proof of Theorem 2

Proof approach is similar to the approach adopted in the proof of Theorem 1. We describe the arguments which are specific to this proof. We rewrite the updates of the iterates as

$$\begin{aligned} V_{n+1}(s,\theta) &= V_n(s,\theta) + g(\gamma(s,n))\{[r(s,a;\beta) + V_n(s',\theta) - V_n(s^*,\theta)) - V_n(s,\theta)]\}, \\ V_{n+1}(s'',\theta) &= V_n(s'',\theta) \quad \forall s'' \neq s, \end{aligned} \tag{A.7}$$

$$\begin{aligned} \theta_{n+1}(T) &= \Delta_T[\theta_n(T) + h(n)\nabla f(s,\theta_n(T))(-1)^{\alpha_n} V_n(s',\theta_n)], \\ \theta_{n+1}(T') &= \theta_n(T') \quad \forall T' \neq T, \end{aligned} \tag{A.8}$$

$$\beta_{n+1} = \Lambda[\beta_n + h(n)(B_n - B_{\max})]. \tag{A.9}$$

Similar to Theorem 1, following the two timescale analysis [78], we analyze Equation (A.7) keeping the threshold vector θ and LM β constant. Therefore, it can be argued that as $t \to \infty$, $\hat{V}$ converges to the asymptotically stable equilibrium of the associated ODE.

Lemma 16. *The value functions, LM and the threshold vector iterates are bounded a.s.*

Proof. The boundedness of value functions and LM follows in an approach similar to Lemma 13. Also, the iterates of the threshold vector are bounded (See Equation (5.7)). □

Lemma 17. *We have $V_n \to V^{\beta_n,\theta_n}$ a.s., where V^{β_n,θ_n} is the value function of states for $\beta = \beta_n, \theta = \theta_n$.*

Proof. Since the iterations of the threshold vector θ can be expressed as $\theta_{n+1} = \theta_n + \delta(n)$, where $\delta(n) = o(g(n))$, the proof approach is similar to that of Lemma 14. □

Convergence of LM follows immediately from Lemma 15. The only thing left to prove is the convergence of threshold vector iterations θ_n to the optimal threshold vector θ^* and hence, the convergence of (V_n, β_n, θ_n) to (V, β^*, θ^*). However, before that, we need to prove the following lemma which establishes the unimodality of the average reward attained under a threshold policy with threshold θ (which is $\rho(\theta)$), with respect to θ.

Lemma 18. *The threshold vector iterates $\theta_n \to \theta^*$.*

Proof. The limiting ODE for the threshold vector iterations (Equation (A.8)) is same as a gradient ascent of the form

$$\dot{\theta}(t) = \nabla \rho(\theta(t)).$$

Since the average reward is known to be unimodal in θ (Lemma 9), there does not exist any local maxima which is not a global maxima. Therefore, $\theta_n \to \theta^*$. $\square$

Contrary to Theorem 1, convergence of the threshold vector iterates does not require individual clocks for every state, as long as all components are updated comparably often, i.e., the relative frequencies of their update remain bounded away from zero. This is true in general for stochastic gradient schemes, see [78, Chapter 7].

This completes the proof of Theorem 2.

Appendix B

A Structure-aware Online Learning Algorithm for Markov Decision Processes

The framework of Markov Decision Process (MDP) [69] is used in modeling and optimization of stochastic systems with temporal decision making. An MDP is a controlled stochastic process on a state space with an associated control process of ‘actions’, where the transition from one state to the next depends only on the current state-action pair and not on the past history of the system (known as the controlled Markov property). Each state transition is associated with a reward. The MDP problem aims to maximize the average reward and provides an optimal policy as a solution. A policy is a mapping from a state to an action describing which action is to be chosen in a state. An optimal policy maximizes the average reward.

A common approach for solving MDP problems is Dynamic Programming (DP) [69]. In this appendix, we consider an MDP problem and prove that the optimal policy has a threshold structure using DP methods. In other words, we prove that up to a certain threshold in the state space, a specific action is preferred and thereafter another action is preferred.

Classical iterative methods for DP are computationally inefficient in the face of large state and action spaces. This is known as the curse of dimensionality. Moreover, the computation of the optimal policy using DP methods requires the knowledge of the state transition probability matrix which is often governed by the statistics of unknown system dynamics. For example, in a telecommunication system, transition probabilities between different states are determined by the statistics of the arrival rates of users. This is known as the curse of modeling. In practice, it may be difficult to gather the knowledge regarding the statistics of the system dynamics beforehand. When we do not have any prior knowledge of the statistics of the system dynamics, a popular approach is Reinforcement Learning (RL) techniques which learn the optimal policy iteratively by trial and error [45]. Examples of RL techniques include TD(λ) [45], Q-learning [89], actor-critic [88], policy gradient [90] and Post-Decision State (PDS) learning [85, 91]. Consider, e.g., Q-learning and PDS learning. Q-learning [89] is one of most popular learning algorithms. Q-learning iteratively computes the Q-function associated with every state-action pair using a combination of exploration and exploitation. Since Q-learning needs to learn the optimal policy for all state-action pairs, the storage complexity of the scheme is of the order of the cardinality of the state space times the cardinality of the action space. In many cases of practical interest, the state and action spaces are large which renders Q-learning impractical. Furthermore, due to the presence of exploration, the convergence rate of Q-learning is generally slow. The idea of PDS [85, 91] learning obtained by reformulating the Relative Value Iteration Algorithm (RVIA) [69] equation is adopted in literature for various problems. The main advantage of PDS learning is that it circumvents the action exploration, thereby improving the convergence rate. Also, there is no need of storing the Q functions of state-action pairs. Instead, it requires only storing the value functions associated with the states. Therefore the storage complexity of the PDS learning scheme is lower than that of Q-learning.

A common drawback of the learning schemes described above is that they do not exploit any known properties related to the structure of the optimal policy, but search over the entire policy space. In application areas such as communications and operations research, examples abound where properties such as convexity/ concavity, submodularity/ supermodularity, monotonicity of the value function can be leveraged to prove additional structure for optimal policy such as threshold structure or index rule. Also, one may achieve reduction of the effective state space by rendering certain states transient or unreachable under the optimal policy, see, e.g., [92, 93]. This can in principle be exploited to reduce the search space and the computational budget of the learning scheme.

In this appendix, we propose a Structure-Aware Learning (SAL) algorithm which exploits the threshold nature of the optimal policy and searches the optimal policy only from the set of threshold policies. In other words, it only learns the threshold in the state space where the optimal action changes. Based on the gradient of the average reward of the system, the threshold is updated on a slower timescale than that of the value function iterates. As a result, the convergence time of the proposed algorithm reduces along with reductions in computational and storage complexities in comparison to traditional schemes such as Q-learning and PDS learning. We prove that the proposed scheme indeed converges to the optimal policy. In general, the proposed technique is applicable to a large variety of optimization problems where the optimal policy is threshold in nature, e.g., [81, 92, 94–96]. Simulation results are presented where the proposed technique is employed on a well-known problem from queuing theory [95] to demonstrate that the proposed algorithm indeed offers faster convergence than traditional learning algorithms like Q-learning and PDS learning.

There are a few works in the literature [97–99] which exploit the structural properties [93] in the learning framework. In [98], an online learning algorithm which approximates the value functions using piecewise linear functions is proposed. However, there is a trade-off between the complexity and the approximation accuracy in this scheme. In [97], authors propose a variant of Q-learning where the value function iterates are projected in such a manner that they preserve the monotonicity in system state. Similar model is adopted in [99]. Although there is an improvement in convergence rate over the conventional Q-learning, not much gain in computational complexity is achieved. Unlike us, none of these works consider the threshold as a parameter in the learning framework. Therefore they are computationally less efficient than our solution.

The rest of the appendix is organized as follows. The system model and problem formulation are described in Section B.1. In Section B.2, the optimality of threshold policy is established. In Section B.3, the structure-aware learning algorithm is proposed. We also provide a proof of convergence of the proposed algorithm to the optimality in this section. We provide a comparison of computational and storage complexities of different RL schemes in Section B.4. Simulation results are provided in Section B.5. Sections B.6

discusses possible future extensions of the problem. We provide the conclusions in Section B.7.

B.1 System Model & Problem Formulation

We consider a controlled time-homogeneous Discrete Time Markov Chain (DTMC) and denote it by $\{X_n\}_{n\geq 0}$, which takes values from the finite state space $\mathcal{S}$. Without loss of generality, we assume that $\mathcal{S} = \{0, 1, 2, \ldots, N\}$, where N is a fixed positive integer. For the sake of simplicity, we assume that each state $i \in \mathcal{S}$ is associated with an action space $\mathcal{A}$. Let the action space $\mathcal{A}$ consists of two actions, viz., A_1 and A_2. Let the transition probability of going from state $i \in \mathcal{S}$ to state $j \in \mathcal{S}$ under action $a \in \mathcal{A}$ be denoted as $p_{ij}(a)$. Therefore, we have, $p_{ij}(a) \in [0, 1]$ $\forall i, j, a$ and $\sum\limits_{j} p_{ij}(a) = 1$. Let the action process be denoted by $Z_n, n \geq 0$. Therefore, the evolution of X_n can be described by

$$P(X_{n+1} = j | X_m, Z_m, m \leq n, X_n = i) = p_{ij}(Z_n), n \geq 0.$$

Let us assume that whenever A_1 is chosen in state $i \in \mathcal{S}$, no reward is obtained, and the system remains in the same state with probability p and goes to state $(i-1)^+$ with probability $(1-p)$, where $(i)^+ = \max\{i, 0\}$. We further assume that whenever the system is in state $i \in \mathcal{S}$ and A_2 is chosen, a non-negative fixed reward r is obtained and the system moves to state $(i+1)$ with probability p and moves to state $(i-1)^+$ with probability $(1-p)$. Note that the A_2 is not feasible in state N.

We have used this model for sake of specificity and because it does arise in practice. Analogous schemes can be developed for other models that naturally lead to a threshold structure.

We aim to obtain a policy which maximizes the average expected reward of the system. Let $\mathcal{Q}$ be the set of memoryless policies where the decision rule at time t depends only on the state of the system at time t and not on the past history. Under the assumption of unichain nature of the underlying Markov chain which guarantees the existence of unique stationary distribution, let the average reward of the system over infinite horizon under policy $Q \in \mathcal{Q}$ be independent of the initial condition and be denoted by σ_Q. We intend to maximize

$$\sigma_Q = \lim_{H\to\infty} \frac{1}{H} \sum_{h=1}^{H} \mathbb{E}_Q[r(X_h, Z_h)], \tag{B.1}$$

where $r(X_h, Z_h)$ denotes the reward function in state X_h under action Z_h, and $\mathbb{E}_Q$ denotes the expectation operator under policy Q. The limit in Equation (B.1) may be taken to exist because the optimal policy is known to be stationary. The DP equation depicted below provides the necessary condition for optimality $\forall i \in \mathcal{S}$.

$$V(i) = \max_{a \in \mathcal{A}} \left[r(i,a) + \sum_{j \in \mathcal{S}} p_{ij}(a) V(j) - \sigma \right], \tag{B.2}$$

where $V(i)$ and σ denote the value function of state $i \in \mathcal{S}$ and the optimal average reward, respectively. The arg max above yields the optimal policy, i.e., optimal action as a function of current state. RVIA can be used to solve this problem using the iterative scheme described below.

$$V_{n+1}(i) = \max_{a \in \mathcal{A}} \left[r(i,a) + \sum_{j \in \mathcal{S}} p_{ij}(a) V_n(j) - V_n(i^*) \right], \tag{B.3}$$

where $V_n(.)$ is the value function estimate in n^{th} iteration of RVIA, and $i^* \in \mathcal{S}$ is a fixed state.

B.2 Structure of Optimal Policy

In this section, we investigate the structure of the optimal policy. We prove the structural properties using the 'non-increasing difference' property of the value function in the lemma described next.

Lemma 19. *$V(i+1) - V(i)$ is non-increasing in i.*

Proof. We rewrite the optimality equation for the value function as

$$V(i) = p \max\{V(i), r + V(i+1)\} + (1-p)V((i-1)^+).$$

Let the value function of state i in n^{th} iteration of Value Iteration Algorithm (VIA) be denoted by $v_n(i)$. Start with $v_0(i) = 0$. Hence, $v_0(i+1) - v_0(i)$ is non-increasing in i. We have,

$$v_{n+1}(i) = p \max\{v_n(i), r + v_n(i+1)\} + (1-p)v_n((i-1)^+). \tag{B.4}$$

Using Equation (B.4), $v_1(i+1) - v_1(i)$ is non-increasing in i. Now, we assume that $v_n(i+1) - v_n(i)$ is non-increasing in i. We need to prove that $v_{n+1}(i+1) - v_{n+1}(i)$ is non-increasing in i. Let us define $v'_{n+1}(i,a)$ as follows.

$$v'_{n+1}(i,a) = \begin{cases} v_n(i), & a = A_1, \\ r + v_n(i+1), & a = A_2. \end{cases}$$

Also define $v'_n(i) = \max_{a \in \mathcal{A}} v'_n(i,a)$. Let us define $Dv_n(i) = v_n(i+1) - v_n(i)$. Therefore,

$$Dv'_{n+1}(i,a) = \begin{cases} Dv_n(i), & a = A_1, \\ Dv_n(i+1), & a = A_2. \end{cases}$$

$$D^2v'_{n+1}(i,a) = \begin{cases} D^2v_n(i), & a = A_1, \\ D^2v_n(i+1), & a = A_2. \end{cases}$$

Since $v_n(i+1)-v_n(i)$ is non-increasing in i, $v'_{n+1}(i+1,a)-v'_{n+1}(i,a)$ is non-increasing in i, $\forall a \in \mathcal{A}$. Let $a_1 \in \mathcal{A}$ and $a_2 \in \mathcal{A}$ be the maximizing actions in states $(i+2)$ and i, respectively.

$$\begin{aligned} &2v'_{n+1}(i+1) \geq v'_{n+1}(i+1,a_1) + v'_{n+1}(i+1,a_2) \\ &= v'_{n+1}(i+2,a_1) + v'_{n+1}(i,a_2) + Dv'_{n+1}(i,a_2) - Dv'_{n+1}(i+1,a_1). \end{aligned}$$

Let $B = Dv'_{n+1}(i,a_2) - Dv'_{n+1}(i+1,a_1)$. For proving that $v'_{n+1}(i+1) - v'_{n+1}(i)$ is non-increasing in i, we need to prove $B \geq 0$. Let us consider four cases as follows.

- $a_1 = a_2 = A_1$

$$B = Dv_n(i) - Dv_n(i+1) = -D^2v_n(i) \geq 0.$$

- $a_1 = A_1, a_2 = A_2$

$$B = Dv_n(i+1) - Dv_n(i+1) = 0.$$

- $a_1 = a_2 = A_2$

$$B = Dv_n(i+1) - Dv_n(i+2) = -D^2v_n(i+1) \geq 0.$$

- $a_1 = A_2, a_2 = A_1$

$$B = Dv_n(i) - Dv_n(i+2) = -D^2v_n(i) - D^2v_n(i+1) \geq 0.$$

Since $v'_{n+1}(i+1)-v'_{n+1}(i)$ and $v_n(i+1)-v_n(i)$ are non-increasing in i, $v_{n+1}(i+1)-v_{n+1}(i)$ is non-increasing in i (Using Equation (B.4)). Since $V(.) = \lim_{n\to\infty} v_n(.)$, $V(i+1)-V(i)$ is non-increasing in i. $\square$

The following theorem describes that the optimal policy is of threshold type where A_2 is optimal only upto a certain threshold.

Theorem 3. *The optimal policy has a threshold structure where the optimal action changes from A_2 to A_1 after a certain threshold in $i \in \mathcal{S}$.*

Proof. If A_1 is optimal in state i, then $r+V(i+1) \leq V(i)$. Using Lemma 19, $V(i+1)-V(i)$ is non-increasing in i. Therefore, it follows that there exists a threshold such that A_2 is optimal only below the threshold, A_1 thereafter. $\square$

B.3 Structure-aware Online RL Algorithm

In this section, we propose a learning algorithm by exploiting the threshold properties of the optimal policy. Unlike the traditional RL algorithms which optimize over the entire policy space, our algorithm searches the optimal policy only from the set of threshold policies. As a result, the proposed algorithm converges faster than traditional RL algorithms like Q-learning, PDS learning. Also, the computational complexity and the storage complexity of learning is reduced as argued later.

B.3.1 Gradient Based RL Framework

Since we know that the optimal policy is threshold in nature where the optimal action changes from A_2 to A_1 after a certain threshold, if we know the value of the threshold, we can specify the optimal policy completely. However, the value of the threshold depends on the transition probabilities (i.e., p) between different states. Therefore, in the absence of knowledge regarding p, instead of learning the optimal policy from the set of all policies, we only learn the the optimal value of the threshold. We target to optimize over the threshold using an update rule so that the value of threshold converges to the optimal threshold.

We consider the set of threshold policies and describe them in terms of the value of parameter threshold (T, say). The approach we adopt in this appendix is to compute the gradient of the average expected reward of the system with respect to the threshold T and improve the threshold policy in the direction of the gradient by updating the the value of T. Before proceeding, we need to explicitly indicate the dependence of the associated MDP on T by redefining the notations in the context of threshold policies.

Let the steady state stationary probability of state i, the value function of state i and the average reward of the Markov chain in terms of threshold parameter T be denoted by $\pi(i,T)$, $V(i,T)$ and $\sigma(T)$, respectively. Let the transition probability from state i to state j under threshold T be denoted as $P_{ij}(T)$. Therefore,

$$P_{ij}(T) = P(X_{n+1} = j | X_n = i, T).$$

We later embed the discrete parameter T into a continuous valued one. With this in mind, we make the following assumptions regarding $P_{ij}(T)$.

Assumption 4. *$P_{ij}(T)$ is a twice differentiable function of T with bounded first and second derivatives. Moreover, $P_{ij}(T)$ is bounded.*

The proposition described below provides a closed-form expression for the gradient of the average reward $\sigma(T)$.

Proposition 2. *Under Assumption 4,*

$$\nabla\sigma(T) = \sum_{i\in\mathcal{S}} \pi(i,T) \sum_{j\in\mathcal{S}} \nabla P_{ij}(T) V(j,T).$$

Proof. Detailed proof can be found in [74]. □

The system model considered by us is a special case of the model considered in [74], with the exception that unlike in [74], the reward function in our case does not have any dependence on T.

B.3.2 Online RL Algorithm

Optimal policy can be obtained using RVIA if the transition probabilities between different states are known beforehand. In the absence of knowledge regarding transition probabilities, we can use the theory of Stochastic Approximation (SA) [78] to remove the expectation operation in Equation (B.3) and converge to the optimal policy by averaging over time. Let $g(n)$ be a positive step-size sequence having the following properties.

$$\sum_{n=1}^{\infty} g(n) = \infty; \sum_{n=1}^{\infty} (g(n))^2 < \infty. \tag{B.5}$$

Let $h(n)$ be another step-size sequence with similar properties as in Equation (B.5) along with the following additional property.

$$\lim_{n\to\infty} \frac{h(n)}{g(n)} \to 0. \tag{B.6}$$

In order to learn the optimal policy, we adopt the following strategy. We update the value function of one state at a time and keep others unchanged. Let S_n be the state whose value function is updated at n^{th} iteration. Let $\eta(i,n)$ denote the number of times the value function of the state i is updated till n^{th} iteration. Symbolically,

$$\eta(i,n) = \sum_{m=0}^{n} I\{i = S_m\}.$$

The scheme for the update of value function can be described as follows.

$$\begin{aligned} V_{n+1}(i,T) &= (1 - g(\eta(i,n)))V_n(i,T) + g(\eta(i,n))[r(i,a) + V_n(j,T) - V_n(i^*,T)], \\ V_{n+1}(i',T) &= V_n(i',T), \forall i' \neq i, \end{aligned} \tag{B.7}$$

where $V_n(i,T)$ denotes the value function of state i at the n^{th} iteration on the faster timescale when the current value of threshold is T. The scheme (B.7) solves a dynamic programming equation for a fixed value of threshold T, referred to as primal RVIA. To obtain the optimal threshold value, T has to be iterated in a separate timescale $h(n)$. Intuitively, in order to learn the value of the optimal threshold, we can determine the value of $\nabla\sigma(T)$ based on the current value of threshold T_n at the n^{th} iteration and then update the value of threshold in the direction of the gradient. This is similar to a stochastic gradient scheme which can be expressed as

$$T_{n+1} = T_n + h(n)\nabla\sigma(T_n). \tag{B.8}$$

The assumptions described in Equations (B.5) and (B.6) guarantee that value function and threshold parameter are updated in two separate timescales without interfering in each other's convergence behavior. The value functions are updated in a faster timescale than that of the threshold. From the faster timescale, the value of threshold appears to be fixed. From the slower timescale, the value functions seem to be equilibrated according to the current threshold value. This behavior is commonly known as "leader-follower" scheme.

Given a threshold T, we assume that the transition from state i is determined by the rule $P^1(j|i)$, if $i < T$ and by the rule $P^0(j|i)$, otherwise. For example, consider that the system is in state i and $i < T$. Then the next state to which the system moves is governed by the rule $P^1(j|i)$ for action A_2. Therefore, the system moves to the state $(i+1)$. However, if $i \geq T$, then the state transition is given by the rule $P^0(j|i)$ for action A_1. Therefore, the system remains in state i. This scheme is applied to Equation (B.7) for a fixed value of threshold T.

To update the threshold, we need to interpolate the value of threshold which takes discrete values, to continuous domain so that the online rule can be applied. Since the threshold policy can be described as a step function which takes discrete non-negative values as input and follows $P^1(j|i)$ upto a threshold and $P^0(j|i)$ thereafter, the derivative does not exist at all points (See Assumption 4). Therefore, we propose an approximation to the threshold policy using a randomized policy. The randomized policy is a mixture of two policies depicted by $P^0(j|i)$ and $P^1(j|i)$ with corresponding probabilities $f(i,T)$ and $(1 - f(i,T))$. To be precise,

$$P_{ij}(T) \approx P^0(j|i)f(i,T) + P^1(j|i)(1 - f(i,T)). \tag{B.9}$$

Note that the function $f(.,.)$ which decides how much importance is to be given to respective policies, is a function of state i and the current value of threshold T. For a convenient approximation, $f(i,T)$ should be an increasing function of i. The idea is to provide comparable importances to both $P^0(j|i)$ and $P^1(j|i)$ near the threshold and reduce the importance of $P^0(.|.)$ $(P^1(.|.))$ away from the threshold in the left (right) direction. We choose the following function owing to its nice properties such as continuous differentiability and the existence of non-zero derivative everywhere.

$$f(i,T) = \frac{e^{(i-T-0.5)}}{1 + e^{(i-T-0.5)}}. \tag{B.10}$$

This does not satisfy Assumption 1 at $T = i, i-1$, but that does not affect our subsequent analysis if we take right or left derivatives at these points.

Remark 16. *Another choice of $f(i,T)$ could be the following.*

$$f(i,T) = 0.I\{i \leq T\} + 1.I\{i \geq T+1\} + (i-T).I\{T < i < T+1\}.$$

Since this function exactly replicates the step function nature of the optimal policy in the interval $[0,T]$ and $[T+1,N]$ and uses approximation only in the interval $(T,T+1)$, the approximation error in this case is less than that of Equation (B.10). However, the derivative of the function is nonzero only in the interval $(T,T+1)$. Therefore, if the initial guess of the threshold is outside this range, then the proposed learning scheme may not converge to the optimal threshold as the gradient becomes zero.

Note also that having a continuous valued threshold does not create any issue. The state space remains discrete, and only an ordinal comparison separates the two action choices.

While devising an update rule for the threshold, we evaluate $\nabla P_{ij}(T)$ as a representative of $\nabla \sigma(T)$ and use that in Equation (B.8). From Equation (B.9), we get,

$$\nabla P_{ij}(T) = (P^0(j|i) - P^1(j|i))\nabla f(i,T). \tag{B.11}$$

Since multiplication by a constant factor does not impact the online update of the proposed scheme, we incorporate an extra multiplicative factor of $\frac{1}{2}$ to the right hand side of Equation (B.11). This operation can be described in the following manner. In every iteration, we choose transition according to $P^0(.|.)$ and $P^1(.|.)$ with equal probabilities. $\nabla f(i,T)$ is a state-dependent term which denotes how much importance is to be given to the value function of the state. Therefore, the update of T in the slower timescale $h(n)$ is as follows.

$$T_{n+1} = \Lambda[T_n + h(n)\nabla f(i,T_n)(-1)^{\gamma} V_n(k,T_n)],$$

where γ is a random variable which takes values 0 and 1 with equal probabilities. If $\gamma = 0$, then the transition is determined by the rule $P^0(.|.)$, else by $P^1(.|.)$. Therefore, $k \sim \tilde{P}_{ik}$ where $\tilde{P}_{ik} = \gamma P^0(k|i) + (1-\gamma)P^1(k|i)$. The averaging effect of SA scheme enables us to obtain the effective drift in Equation (B.11). The projection operator is defined as $\Lambda : x \mapsto 0 \vee (x \wedge N) \in [0,N]$. Λ is introduced to guarantee that the iterates remain bounded in $[0,N]$.

Therefore, the online RL scheme where the value functions are updated in the faster timescale and the threshold parameter in the slower one, can be summarized as

$$\begin{aligned}&V_{n+1}(i,T) = (1-g(\eta(i,n)))V_n(i,T) + g(\eta(i,n))[r(i,a)+V_n(j,T)-V_n(i^*,T)];\\&V_{n+1}(i',T) = V_n(i',T), \forall i' \neq i,\end{aligned} \tag{B.12}$$

$$T_{n+1} = \Lambda[T_n + h(n)\nabla f(i,T_n)(-1)^{\gamma}V_n(k,T_n)]. \tag{B.13}$$

The transitions in (B.12) from i to j correspond to a single run of a simulated chain as is common in RL. For each current state i, the k in (B.13) is generated separately as per $\tilde{P}_{ik}$.

Theorem 4. *The schemes (B.12) and (B.13) converge to optimality almost surely (a.s.).*

Proof. Proof methodologies adopted in this appendix are similar to that of [85]. The idea of adoption of Ordinary Differential Equation (ODE) approach for analyzing SA algorithms by considering them as a noisy discretization of a limiting ODE [78], is considered. Step size parameters are considered as discrete time steps, and if the discrete values of the iterates are linearly interpolated, they closely follow the trajectory of the ODE. Assumptions on step sizes, viz., (B.5) and (B.6) are made to guarantee that the discretization error and error due to noise are negligible asymptotically. As a result, in the asymptotic sense, the iterates closely follow the trajectory of the ODEs and converge a.s. to the globally asymptotically stable equilibrium.

Update rules for value functions and threshold in the faster and slower timescale, respectively, are as follows.

$$\begin{aligned}&V_{n+1}(i,T) = (1-g(\eta(i,n)))V_n(i,T) + g(\eta(i,n))[r(i,a)+V_n(j,T)-V_n(i^*,T)];\\&V_{n+1}(i',T) = V_n(i',T), \forall i' \neq i,\end{aligned} \tag{B.14}$$

$$T_{n+1} = \Lambda[T_n + h(n)\nabla f(i,T_n)(-1)^{\gamma}V_n(k,T_n)]. \tag{B.15}$$

Following the two timescale analysis adopted in [78], we consider Equation (B.14) first keeping threshold T fixed. Let $M_1 : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ be a map given by

$$M_1(s) = \sum_j P_{ij}(T)[r(i,a)+V_n(j,T)] - V_n(i^*,T). \tag{B.16}$$

The knowledge of $P_{ij}(T)$ is required only for the sake of analysis. However, the proposed algorithm can operate without the knowledge of $P_{ij}(T)$. Since T is kept constant, this gives rise to the following limiting ODE which tracks Equation (B.14).

$$\dot{V}(t) = M_1(V(t)) - V(t). \tag{B.17}$$

As $t \to \infty$, $V(t)$ converges to the fixed point of $M_1(.)$ (i.e., $M_1(V) = V$) [86], which is the asymptotically stable equilibrium of the ODE. Similar approaches are adopted in [86,87].

The lemma presented next establishes the boundedness of value functions and threshold iterates.

Lemma 20. *The value function and the threshold iterates are bounded a.s.*

Proof. Let $M_0 : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ be a map given by

$$M_0(s) = \sum_j P_{ij}(T) V_n(j, T) - V_n(i^*, T). \tag{B.18}$$

Note that Equation (B.16) reduces to Equation (B.18) if the immediate reward is zero. Now, $\lim\limits_{b \to \infty} \frac{M_1(bV)}{b} = M_0(V)$. Consider the limiting ODE

$$\dot{V}(t) = M_0(V(t)) - V(t). \tag{B.19}$$

Observe that the globally asymptotically stable equilibrium of the ODE (B.19) is the origin. Also, notice that the ODE (B.19) is a scaled limit of the ODE (B.17). Boundedness of $V(.)$ follows [84].

Boundedness of iterates of T follows from (B.13). □

The physical interpretation behind the proof is as follows. If the iterates of the value functions become unbounded along a subsequence, then a scaled version of the original ODE follows the ODE approximately. Since we have shown that the scaled ODE must globally asymptotically converge to the origin, the scaled ODE must return to the origin. Therefore, the value function iterates must also move towards a bounded set. This ensures the stability of the value function iterates.

Lemma 21. $V_n - V^{T_n} \to 0$ *a.s., where* V^{T_n} *is the value function of the states for* $T = T_n$.

Proof. We know that the threshold is varied on a slower timescale than that of V. Therefore, the value function iterates treat the threshold value as a constant. Therefore, T iterations can be viewed as $T_{n+1} = T_n + \alpha(n)$, where $\alpha(n) = O(h(n)) = o(g(n))$. Thus, the limiting ODEs associated with value function and threshold iterates are $\dot{V}(t) = M_1(V(t)) - V(t)$ and $\dot{T}(t) = 0$, respectively. Since $\dot{T}(t) = 0$, it is sufficient to consider the ODE $\dot{V}(t) = M_1(V(t)) - V(t)$, for a fixed value of T. The rest of the proof is similar to [88]. □

The subsequent lemmas prove that the average reward under a threshold T ($\sigma(T)$) is unimodal in T, and hence the threshold iterations T_n converge to the optimal threshold T^*. Therefore, (V_n, T_n) converges to (V, T^*).

Lemma 22. *$v_n(i+1) - v_n(i)$ is non-increasing in n.*

Proof. We need to prove that $Dv_n(i)$ is non-increasing in n. We prove using induction. When $n = 0$, $v_0(i) = 0$ and $Dv_0(i) = 0$. We have,

$$v_{n+1}(i) = p \max\{v_n(i), r + v_n(i+1)\} + (1-p)v_n((i-1)^+).$$

$$v'_{n+1}(i) = \max\{v_n(i), r + v_n(i+1)\}.$$

Let $v''_{n+1}(i) = v_n((i-1)^+)$. Then $Dv'_1(i) = 0$ and $Dv_1(i) \leq Dv_0(i)$.

Now, assume that the claim holds for any n, i.e., $Dv_{n+1}(i) \leq Dv_n(i)$. We need to prove that $Dv_{n+2}(i) \leq Dv_{n+1}(i)$. It is easy to see that $Dv''_{n+2}(i) \leq Dv''_{n+1}(i)$. Therefore, to complete the proof, we need to prove that $Dv'_{n+2}(i) \leq Dv'_{n+1}(i)$. Let $a_0, a_1 \in \{A_1, A_2\}$ be the maximizing actions in states i and $i+1$, respectively, at $(n+2)^{\text{th}}$ iteration. Let $b_0, b_1 \in \{A_1, A_2\}$ be the maximizing actions in states i and $i+1$, respectively, at $(n+1)^{\text{th}}$ iteration. Now, it is impossible to have $a_1 = A_2$ and $b_0 = A_1$. If $b_0 = A_1$, we have, $Dv_n(i) \leq -r$. From Lemma 19, we must have $Dv_n(i+1) \leq -r$. If $a_1 = A_2$, we have $Dv_{n+1}(i+1) \geq -r$. This contradicts the inductive assumption. Therefore, we consider three cases as follows. For a given value of a_1 and b_0, if the inequality holds for any values of a_0 and b_1, then the inequality will hold for maximizing actions as well.

1) $a_1 = b_0 = A_1$, then choose $a_0 = b_1 = A_1$. We have,

$$Dv'_{n+2}(i) - Dv'_{n+1}(i) = Dv'_{n+1}(i) - Dv'_n(i) \leq 0.$$

2) If $a_1 = b_0 = A_2$, then we choose $a_0 = b_1 = A_2$, and the inequality satisfies similar to the previous case.
3) If $a_1 = A_1$ and $b_0 = A_2$, then we choose $a_0 = A_2$ and $b_1 = A_1$.

$$\begin{aligned} Dv'_{n+2}(i) - Dv'_{n+1}(i) = v_{n+1}(i+1) - r - v_{n+1}(i+1) - v_n(i+1) \\ + r + v_n(i+1) = 0. \end{aligned}$$

Thus, we have, $Dv_{n+2}(i) \leq Dv_{n+1}(i)$. □

Lemma 23. *$\sigma(T)$ is unimodal in T.*

Proof. We know that if the optimal action in state i is A_1, then $V(i+1) - V(i) \leq -r$. Since VIA converges to the policy with threshold T^*, $\exists N_0 > 0$ such that $\forall n \geq N_0$, $v_n(i+1) - v_n(i) \leq -r \ \forall i \geq T^*$ and $v_n(i+1) - v_n(i) \geq -r \ \forall i \leq T^*$. Let $t_n, n \geq 1$ be the optimal threshold at nth iteration of VIA. Symbolically, $t_n = \min\{i \in \mathbb{N}_0 : v_n(i+1) - v_n(i) \leq -r\}$. If for no values of i, the inequality holds, then t_n is taken as N. Using Lemma 22, t_n must monotonically decrease with n and $\lim\limits_{n\to\infty} t_n = T^*$.

Consider a modified problem where A_1 is not permitted in any state $i < \hat{T}$, for a given threshold $\hat{T}(T^* < \hat{T} \leq N)$. Lemma 22 holds for this modified problem too. Let $n_{\hat{T}}$ be the first VIA iteration where the threshold drops to $\hat{T}$. The value function iterates for the modified and the original problem are same for $n < n_{\hat{T}}$ because A_1 is never chosen as the optimal action for $i < \hat{T}$ in the original problem in these iterations. Therefore, $n_{\hat{T}}$ must be finite and the following inequality holds for both original and the modified problem after $n_{\hat{T}}$ iterations.

$$v_n(\hat{T}+1) - v_n(\hat{T}) \leq -r. \tag{B.20}$$

Using Lemma 22, Equation (B.20) holds $\forall n \geq n_{\hat{T}}$. Therefore, in the considered modified problem, t_n converges to $\hat{T}$. This implies that the threshold policy with threshold $\hat{T}$ is better than that of $\hat{T}+1$. Since $\hat{T}$ can be chosen arbitrarily, average reward is monotonically decreasing with $\hat{T}$, $\forall \hat{T} > T^*$.

Now, if we have $\sigma(T) \geq \sigma(T+1)$, we must have $T \geq T^*$. Therefore, $\sigma(T+1) \geq \sigma(T+2)$. Thus, $\sigma(T)$ is unimodal in T. □

Lemma 24. *The threshold iterates $T_n \to T^*$.*

Proof. The limiting ODE for Equation (B.15) is the gradient ascent

$$\dot{T} = \nabla \sigma(T),$$

with inward pointing gradient at $0, N$. Using Lemma 23, there does not exist any local maximum other than the global maximum T^*. Therefore $T_n \to T^*$. □

Remark 17. *In general, in an MDP problem with a threshold structure, unimodality of average reward may not hold. In such cases, the threshold iterates may converge to a local maximum only.*

□

We describe the resulting two-timescale SAL algorithm in Algorithm 8 . As described

Algorithm 8 Two-timescale SAL algorithm

```
1: Initialize number of iterations $n \leftarrow 1$, value function $V(i) \leftarrow 0, \forall i \in \mathcal{S}$ and the threshold $T \leftarrow 0$.
2: while TRUE do
3:     Choose action $a$ governed by the current value of $T$.
4:     Update the value function of state $i$ using Equation (B.12).
5:     Update threshold $T$ using Equation (B.13).
6:     Update $i \leftarrow j$ and $n \leftarrow n + 1$.
7: end while
```

in Algorithm 8, the number of iterations, value functions and the threshold are initialized at the beginning. On every decision epoch, we choose the action which is specified by the current value of threshold. Based on the reward obtained, the value function of states and the value of threshold are updated in faster and slower timescale, respectively. The rules for the updates are provided in Equation (B.12) and (B.13), respectively.

Remark 18. *Even if the optimal policy in an MDP problem does not have a threshold structure, the methodologies presented in this appendix which is guaranteed to converge to the optimal (at least locally) threshold policy, can be used. In general threshold policies are easy to implement and have low storage complexity. Besides, often a well chosen threshold policy provides a good performance.*

B.4 Computational and Storage Complexity

In this section, we provide a comparative study of computational and storage complexities associated with traditional learning algorithms such as Q-learning, PDS learning and the SAL algorithm. The comparison is summarized in Table B.1.

Table B.1: Computational and storage complexities of various RL algorithms.

Algorithm	**Storage complexity**	**Computational complexity**
Q-learning [45, 89]	$O(\lvert\mathcal{S}\rvert \times \lvert\mathcal{A}\rvert)$	$O(\lvert\mathcal{A}\rvert)$
PDS learning [85, 91]	$O(\lvert\mathcal{S}\rvert)$	$O(\lvert\mathcal{A}\rvert)$
SAL	$O(\lvert\mathcal{S}\rvert)$	$O(1)$

As described in Table B.1, Q-learning algorithm needs to store the value function associated with every state-action pair. Thus, the storage complexity associated with Q-learning is $O(|\mathcal{S}| \times |\mathcal{A}|)$. PDS learning algorithm needs to store the value functions associated with only the PDSs along with feasible actions in every state, thereby requiring $O(|\mathcal{S}|)$ storage. The SAL algorithm proposed by us needs to store the value functions of all the states and the value of threshold. We no longer need to store feasible actions corresponding to every state since the value of threshold completely specifies the policy. Therefore, the storage complexity of SAL algorithm is $O(|\mathcal{S}|)$. However, for all practical purposes, once the algorithm converges, it is sufficient to store only the value of threshold instead of optimal actions associated with every state, as required by Q-learning and PDS learning.

Q-learning algorithm updates the value function associated with a state-action pair in every iteration by evaluating $|\mathcal{A}|$ functions and choosing the best one. Therefore, the per-iteration complexity associated with Q-learning is $O(|\mathcal{A}|)$. In the case of PDS learning, each iteration involves the evaluation of $|\mathcal{A}|$ functions, thereby having a per-iteration complexity of $O(|\mathcal{A}|)$. As evident form Equations (B.12) and (B.13), single iteration of the proposed algorithm involves updating the value function of a state and the value of threshold. Therefore, the computational complexity of our proposed algorithm is $O(1)$. This is a considerable reduction in computational complexity in comparison to Q-learning and PDS learning.

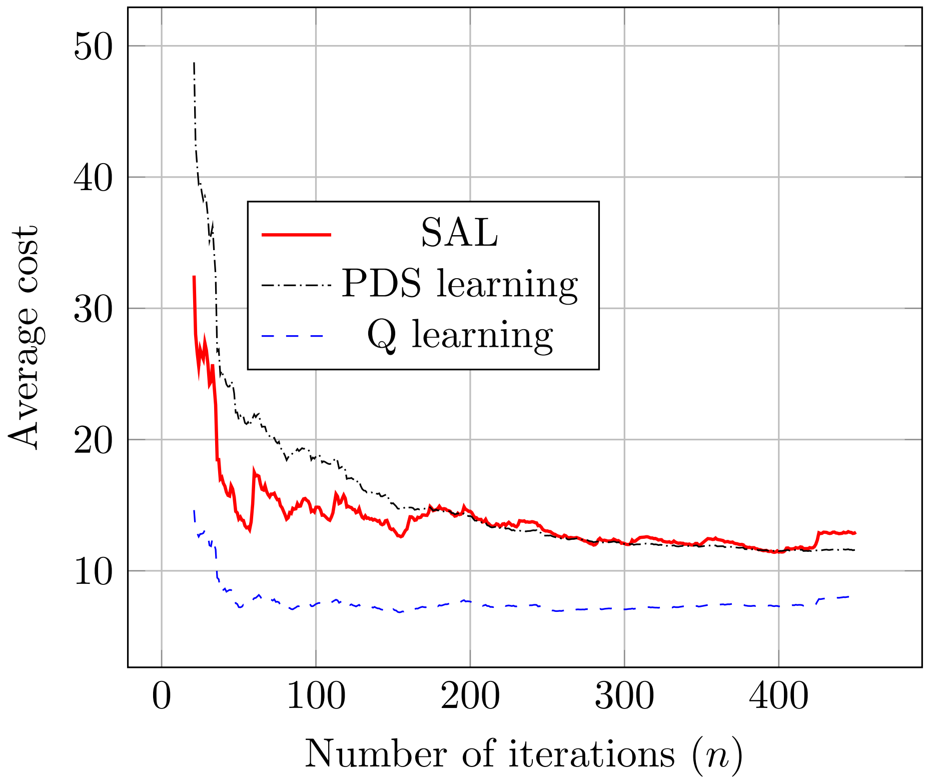
B.5 Simulation Results

In this section, we demonstrate the advantages offered by the proposed algorithm in terms of the convergence speed with respect to other traditional algorithms such as Q-learning [45] and PDS learning [85]. We adopt a simple queuing model from [95] and exhibit that the SAL algorithm converges faster than other RL algorithms. In general, the proposed learning technique is applicable to models involving threshold structure of the optimal policy, such as [81, 92, 94, 96].

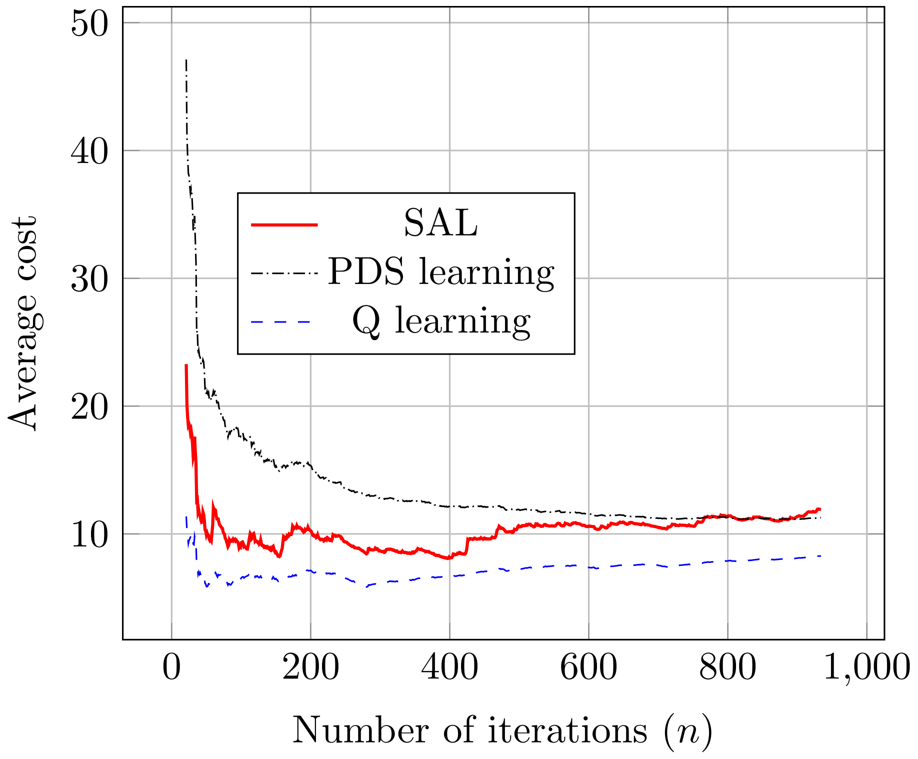
The authors in [95] consider a single queue where the service time is exponentially distributed (with parameter $\frac{1}{\mu}$, say), and the arrival process is Poisson. The system incurs a constant cost upon blocking a user. Additionally, there is a holding cost which is a convex function of the number of customers in the system. The authors prove that it is optimal to admit a user only below a threshold on the number of customers. We conduct ns-3 simulations of SAL algorithm to exploit the threshold structure of optimal policy in [95] and compare the convergence performance with those of Q-learning and PDS learning algorithms.

B.5.1 Convergence Analysis

As illustrated in Fig. B.1a and B.1b, SAL algorithm converges faster than both Q-learning and PDS learning. Due to the absence of exploration mechanism, PDS learning has better convergence behavior than Q-learning. However, SAL algorithm outperforms both Q-learning and PDS learning due to the fact that it operates on a smaller policy space (set of threshold policies only) than other algorithms. On the other hand, for both Q-learning and PDS learning, the policy at any given iteration may be non-threshold in nature. This increases the convergence time to optimality. As observed in Fig. B.1a, while PDS learning requires around 200 iterations for convergence, SAL algorithm requires only 100 iterations. These iterations take 219 s and 108 s, respectively. Similarly, in Fig. B.1b, the number of iterations for convergence reduces from 600 in PDS learning to 450 in SAL algorithm. The approximate times taken for convergence in PDS learning and SAL



(a) $\mu = 1.2s^{-1}$.



(b) $\mu = 1.5s^{-1}$.

Figure B.1: Plot of average cost vs. number of iterations (n) for different algorithms.

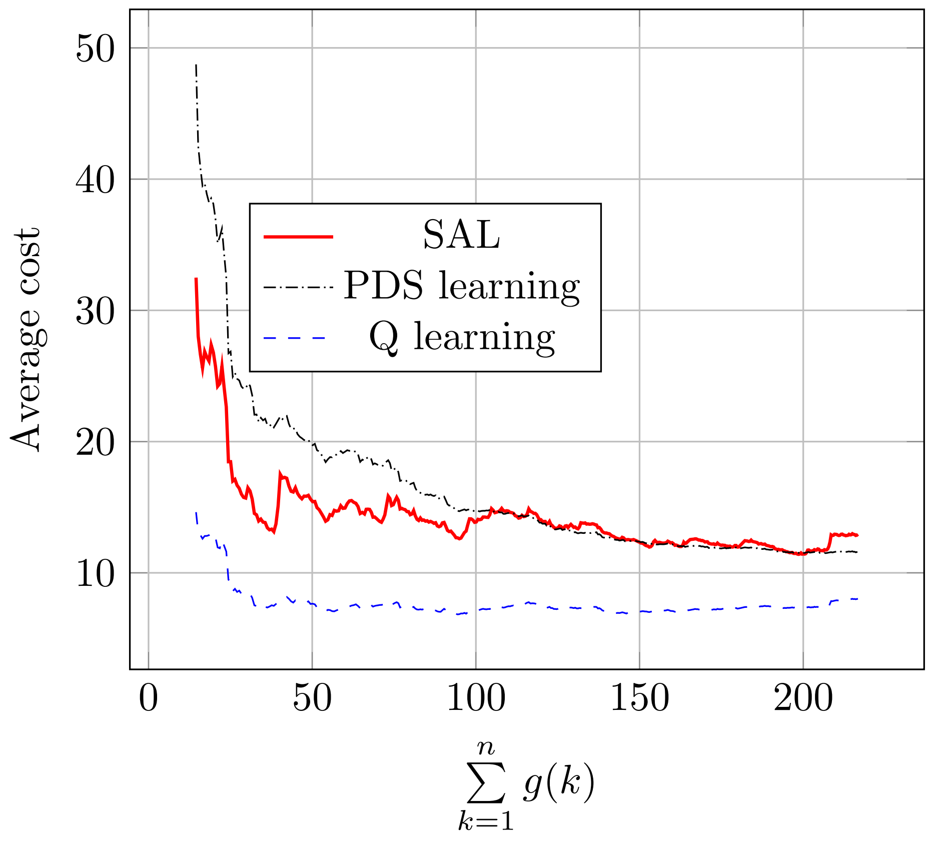
algorithms are 638 s and 480 s, respectively. In both the cases, even after a sufficient number of iterations, Q-learning does not converge to the optimal value. However, the gap to optimality reduces with the number of iterations. Note that we omit initial 20 burn-in period iterates for better representation of the convergence behaviors.

However, for practical purposes, even if we do not converge to the optimal policy, if the average cost of the system does not change much over a window of iterations, we can say that the stopping criterion is reached. In other words, the current policy is close to the optimal policy with a high probability. Instead of a window of iterations, we consider the sum of step sizes till the present iteration as the parameter of choice to eliminate the effect of declining step size in convergence. We choose the window size equal to 50 and observe in Fig. B.2a that convergences for Q-learning, PDS learning and SAL algorithm are achieved approximately in $500, 175$ and 50 iterations, respectively. Similarly, we observe in Fig. B.2b that number of iterations required for practical convergence reduces from 750 and 400 (which correspond to 790 and 444 s) in Q-learning and PDS learning to 200 (which corresponds to 219 s) in SAL algorithm.

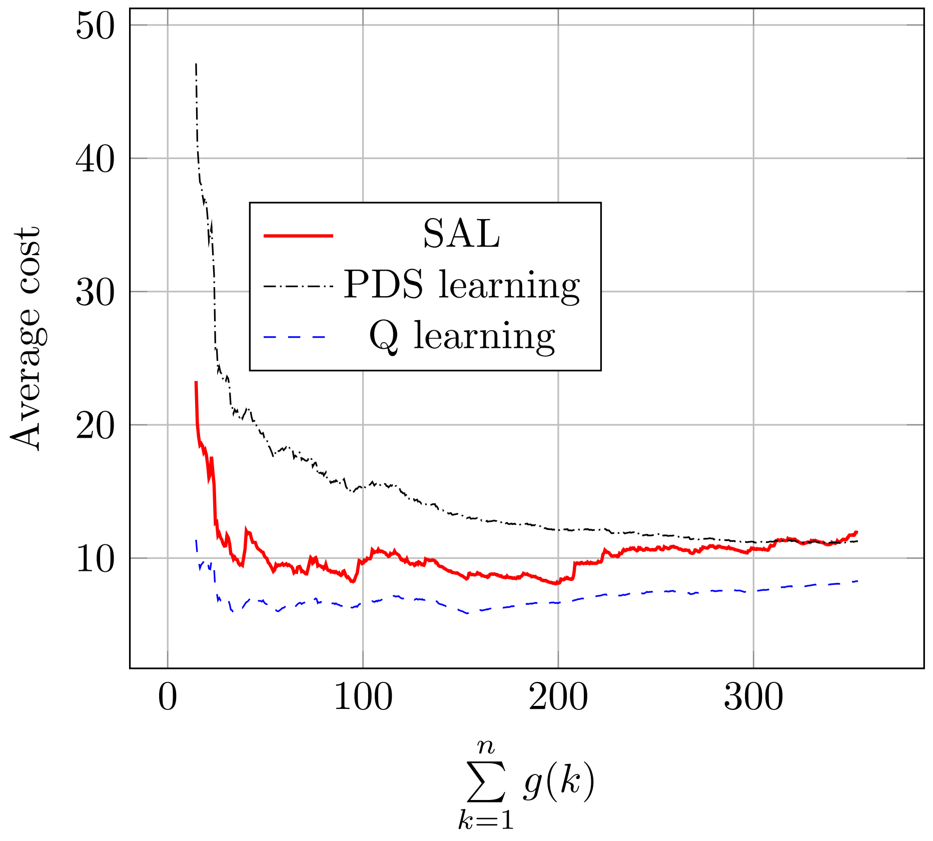
B.6 Possible Extensions

In this section, we describe the possible extensions of the techniques proposed in this appendix. Although the techniques employed in this appendix are primarily focused towards solving MDP problems, the techniques can be employed for learning problems involving Constrained MDP (CMDP) problem also. Due to the presence of constraints, usually a two-timescale learning approach is adopted [78], where the value functions are updated in one timescale and the associated Lagrange Multiplier (LM) in another. The consideration of structure-aware learning may introduce another timescale where the value of threshold is updated. However, since the iterates for the LM and the threshold are not dependent on each other, they can be updated in the same timescale.

The proposed learning technique can also be extended to MDP/ CMDP problems parameterized by a set of threshold parameters rather than only one. In the slower timescale, one threshold parameter can be updated in a single iteration based on the visited state and rest can be kept fixed. Since the update of threshold parameters follows



(a) $\mu = 1.2 s^{-1}$.



(b) $\mu = 1.5 s^{-1}$.

Figure B.2: Plot of average cost vs. sum of step sizes till n^{th} iteration for different algorithms.

a stochastic gradient scheme, contrary to value function iterates, the threshold parameter iterates do not need individual local clocks for convergence. However, for the scheme to work, the relative frequencies of the update of individual threshold parameters have to be bounded away from zero [78]. Yet another future direction is to develop RL schemes for restless bandits wherein threshold policies often lead to simple index-based policies, see [100] for a step towards this.

B.7 Conclusions

In this appendix, we have considered an MDP problem and proved the optimality of threshold policies. We have proposed an RL algorithm which exploits the threshold structure of the optimal policy while learning. Contrary to traditional RL algorithms, the proposed algorithm searches the optimal policy only from the set of threshold policies and hence provides faster convergence. We have proved that the proposed scheme indeed converges to the globally optimal threshold policy. Analysis has been presented to exhibit the effectiveness of the proposed technique in reducing the computational and storage complexities. Simulation results demonstrate the improvement in convergence behavior of the proposed algorithm in comparison to those of Q-learning and PDS learning.

Appendix C

Overview of Markov Decision Process, Reinforcement Learning and Stochastic Approximation

In this appendix, we describe the basics of Markov Decision Process (MDP), Reinforcement Learning (RL) and Stochastic Approximation (SA). We describe the results related to the average reward MDP since we have considered average reward MDP and constrained MDP (CMDP) throughout the thesis. The standard results on MDP follow [69, 73]. Furthermore, we describe the RL fundamentals following [45, 89, 101]. Finally, we introduce the SA technique and discuss the approaches for proving the convergence of SA algorithms. The description is based on the analysis provided in [78, 101, 102].

C.1 Markov Decision Process

The framework of MDP is used for controlling dynamic stochastic systems. The system *state* is a collection of parameters which can describe the system. The state of the system can change based on the chosen *action*. Each action is associated with a *reward* (or *cost*). Based on the current state of the system and the chosen action, the *state transition probabilities* to different states can be determined. The transitions to different states are probabilistic because the effect of an action on the system state is coupled with a *noise*. An MDP follows the Markovian property which states that given the current system state and the action, the past and future states are independent.

C.1.1 Optimality Criteria and Dynamic Programming

In a discrete time MDP, the state of the system is observed at discrete points in time. If the length of the observation window is infinite, then the problem is called an infinite horizon MDP problem. Consider a slotted system where decisions are to be chosen at every slot n. Let the system is described by a stochastic process $\{X_n\}_{n\geq 0}$ which takes values from the finite state space $\mathcal{S}$. Let the action process be denoted by $Z_n, n \geq 0$ which takes entries from the action space $\mathcal{A}$. Let $p(X_n, Z_n, X_n')$ represent the transition probability from state X_n to X_n' under the action Z_n. Let $r(X_n, Z_n)$ denote the immediate reward obtained by taking action Z_n in state X_n. Note that we have assumed that the transition probabilities and the reward functions do not change with time.

A decision rule $\pi_n : \mathcal{S} \to \mathcal{A}$ is defined as a mapping from the state space to the action space at n^{th} decision epoch (slot). A policy is a sequence of decision rules $\{\pi_1, \pi_2, \ldots\}$ taken at consecutive decision epochs. A *stationary policy* is a policy where the decision rules are independent of time and depends only on the current state of the system. An MDP is said to be *unichain* if the Markov chain induced by a policy contains a single ergodic class. For an average reward MDP problem, the objective is to maximize the average expected reward over infinite horizon. Let $\mathcal{Q}$ be the set of memoryless policies where the decision rule at time t depends only on the state of the system at time t and not on the past history. Under the assumption of unichain nature of the underlying Markov chain which guarantees the existence of unique stationary distribution, let the average reward of the system over infinite horizon under policy $Q \in \mathcal{Q}$ be independent of the initial condition and be denoted by σ_Q. We intend to maximize

$$\sigma_Q = \lim\inf_{H\to\infty} \frac{1}{H} \sum_{h=1}^{H} \mathbb{E}_Q[r(X_h, Z_h)],$$

where H is the length of the horizon, and $\mathbb{E}_Q$ denotes the expectation operator under policy Q.

The following Dynamic Programming (DP) equation is known as the Bellman equation which describes the necessary condition for optimality.

$$V(s) = \max_{a \in \mathcal{A}} \left[r(s,a) + \sum_{s' \in \mathcal{S}} p(s,a,s')V(s') - \sigma \right],$$

where $V(s)$ and σ denote the value function of state $s \in \mathcal{S}$ and the optimal average reward, respectively. Relative Value Iteration Algorithm (RVIA) is a well-known algorithm which can be used to solve this problem using the iterative scheme as follows.

$$V_{n+1}(s) = \max_{a \in \mathcal{A}} \left[r(s,a) + \sum_{s' \in \mathcal{S}} p(s,a,s')V_n(s') - V_n(s^*) \right], \tag{C.1}$$

where $V_n(.)$ is the value function estimate in n^{th} iteration of RVIA, and $s^* \in \mathcal{S}$ is a fixed state. It is well known that for a unichain MDP, $V_n(.) \to V(.)$ and $V_n(s^*) \to \sigma$. Furthermore, the optimal policy is a *pure* policy where the optimal action in every state is deterministic.

C.1.2 Constrained Markov Decision Process

In the case of CMDP, we aim to maximize an objective function subject to constraints on other average costs. Let us assume that there are K constraints. Let $\bar{c} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^K$ denote the cost function associated with the constraints. The objective for the CMDP problem can be summarized as follows:

maximize

$$\sigma_Q = \lim \inf_{H \to \infty} \frac{1}{H} \sum_{h=1}^{H} \mathbb{E}_Q[r(X_h, Z_h)],$$

subject to

$$\lim \sup_{H \to \infty} \frac{1}{H} \sum_{h=1}^{H} \mathbb{E}_Q[\bar{c}(X_h, Z_h)] \leq \bar{\omega},$$

where $\bar{\omega}$ is a K dimensional vector.

Let $\bar{\beta} \in \mathbb{R}_+^K$ be a vector with K dimensions. The approach to solve the CMDP problem is to convert the constrained problem into an unconstrained one using the Lagrangian approach. We define $r(s,a;\bar{\beta})$ as

$$r(s,a,\bar{\beta}) = r(s,a) - \bar{\beta}^T \bar{c}(s,a).$$

The idea is to solve the unconstrained problem with the modified reward function $r(s,a;\bar{\beta})$ for a fixed value of $\bar{\beta}$ which is known as the Lagrange Multiplier (LM).

A *randomized* stationary policy is a policy where there exist more than or equal to one state where the optimal action is randomized between more than one actions. It is well known that for a CMDP problem with K constraints, there exists a stationary optimal policy with at most K randomizations.

C.2 Reinforcement Learning

In RL, a *learning agent* takes certain actions, observes the rewards obtained and by trial-and-error learns the optimal policy in order to maximize a certain objective function. Note that the RVIA introduced in the last section relies on the knowledge of the state transition probabilities in order to compute the optimal policy. These transition probabilities depend on the knowledge of the unknown system dynamics which may be difficult to gather in reality. This is known as the *curse of modeling.* However, RL algorithms can operate without such knowledge. In addition, DP algorithms face the *curse of dimensionality* where the DP algorithms become computationally prohibitive for a large state space. However, RL algorithms are computationally efficient and can operate without the knowledge of the system dynamics. We show the efficacy of RL algorithms using an example. We study the well known Q-learning algorithm due to its simplicity and popularity.

C.2.1 Q-learning

In Q-learning, let the *Q-value*, the expected long-term average reward associated with a state-action pair (s, a) be $Q(s, a)$. The aim of Q-learning is to learn the optimal policy, i.e., the optimal Q-value for each state-action pair. The RVIA based iterative equation for Q-learning can be expressed as:

$$\begin{aligned} Q_{n+1}(s,a) &= (1-\rho)Q_n(s,a) + \rho[r(s,a) + \max_{a'\in\mathcal{A}} Q_n(s',a') - Q_n(s^*,a^*)]; \\ Q_{n+1}(\tilde{s},\tilde{a}) &= Q_n(\tilde{s},\tilde{a}) \quad \forall(\tilde{s},\tilde{a}) \neq (s,a), \end{aligned} \tag{C.2}$$

where $Q_n(s, a)$ is the Q-value associated with state s and action a at the n^{th} iteration, $\rho(0 < \rho < 1)$ is the learning rate, and (s^*, a^*) is a fixed state-action pair. Based on the chosen action a, the system moves from state s to state s'. The idea is to update the value of one state-action pair at a time.

Note that Equation (C.2) does not need the knowledge of the transition probabilities $p(s, a, s')$. It updates the Q-value associated with a state-action pair by observing the actual transition using simulations. Since we learn the Q-values one state at a time, the Q-learning algorithm is computationally efficient. Note that if $\rho = 0$, then the agent does not learn. If $\rho = 1$, then the most recent Q-value is considered. If the learning rates are slowly reduced to zero and each state-action pair is visited a reasonable number of times, then Q-learning iterates converge to the optimal Q-values. Therefore, depending on the cardinality of the state space and the learning rate schedule, the convergence time to the optimality may vary.

C.3 Stochastic Approximation

RVIA can be employed to solve the Bellman equation iteratively (see Equation (C.1)). However, in some cases, the observed values of different variables are noisy. SA algorithms are iterative algorithms which can function well even in the presence of noise. In general, SA algorithms are used to solve various optimization problems.

C.3.1 Iterative SA algorithms

In general, SA algorithms can be used to solve a system of equations. Consider an example where we want to solve a system of equation

$$Y = g(Y), \tag{C.3}$$

where Y is a d-dimensional variable and $g(.) : \mathbb{R}^d \to \mathbb{R}^d$ is a function of Y.

The following scheme can be used to solve Equation (C.3):

$$Y_{n+1} = Y_n + ag(Y_n),$$

where a is a small step size. We consider the scenario where the exact measurement of $g(.)$ is difficult to obtain. However, we have an a noisy version of $g(.)$ in hand. Let us denote the noisy version as $g(Y_n) + M_{n+1}$, where M_{n+1} is a random noise. In the presence of noise, the iterative scheme can be written as:

$$Y_{n+1} = Y_n + a(n)(g(Y_n) + M_{n+1}),$$

where $a(n)$ is a step size schedule. Specifically, $a(n)$ needs to satisfy the following equation:

$$\sum_{n=1}^{\infty} a(n) = \infty; \sum_{n=1}^{\infty} (a(n))^2 < \infty. \tag{C.4}$$

While the first condition guarantees that the entire timescale is covered, the second condition ensures that the noise is asymptotically negligible.

Ordinary Differential Equation (ODE) method of analyzing the asymptotic convergence of SA algorithms is popular in the literature. Under the assumption that

$$\mathbb{E}[M_{n+1} | M_m, Y_m, m \leq n] = 0, \forall n,$$

we compute the cumulative noise as

$$C_n = \sum_{i=0}^{n-1} a(i) M_{i+1}, n \geq 1.$$

C_n satisfies the following property

$$\mathbb{E}[C_{n+1} | M_m, Y_m, m \leq n] = C_n, \forall n.$$

Therefore, C_n can be viewed as a martingale process. We make the assumptions that $\sup_n \mathbb{E}[||M_n||] \leq \infty$ and $\sup_n ||Y_n|| \leq \infty$ (iterates are stable). Under the first assumption and using the martingale convergence theorem, we obtain the almost sure (a.s.) convergence of C_n. Now, consider the well-posed ODE

$$\dot{y}(t) = g(y(t)). \tag{C.5}$$

We assume that the globally asymptotically stable attractor set of this ODE is L. Let L_ϵ denote an ϵ-neighborhood of L for some $\epsilon > 0$. Let $p(.) : [0, \infty] \rightarrow \mathbb{R}^d$ be a bounded function. $p(.)$ is a (T, δ) perturbation of Equation (C.5) for $T, \delta > 0$ if we can determine $\lim_{n \to \infty} T_n \rightarrow \infty$ under the condition that $T_{n+1} - T_n \geq T, \forall n$ and solutions of Equation (C.5) $y^n(t)$, where $t \in [T_n, T_{n+1}]$ such that the following condition holds.

$$\sup_{t \in [T_n, T_{n+1}]} ||y^n(t) - p(n)|| < \delta.$$

The theorem described next can be used to establish the convergence of SA algorithms.

Theorem 5. *Given $\epsilon > 0$ and $T > 0$, there exists $\hat{\delta} > 0$ such that for every $\delta < \hat{\delta}$, if we have a (T, δ) perturbation of Equation (C.5), then the perturbation converges to L_ϵ.*

The interpolated iterates can be viewed as (T, δ) perturbations of Equation (C.5), and the convergence of the iterates to the optimality follows immediately.

C.3.2 Two timescale SA algorithms

In some cases, one needs to solve two systems of equations in order to estimate two variables which depend on each other. Consider the following example

$$X = g_1(X, Y); Y = g_2(X, Y),$$

where the measurements of $g_1(.,.)$ and $g_2(.,.)$ are noisy. In such a scenario, the SA scheme can be stated as:

$$X_{n+1} = X_n + a(n)(g_1(X_n, Y_n) + \hat{M}_{n+1}),$$
$$Y_{n+1} = Y_n + b(n)(g_2(X_n, Y_n) + \tilde{M}_{n+1}),$$

where $\{\hat{M}\}$ and $\{\tilde{M}\}$ are Martingale noise sequences. The system of iterates can be viewed as two subroutines where the outer subroutine can be updated after the inner subroutine nearly converges. However, in principle, the same effect can be obtained by performing simultaneous update of both the variables on two different timescales. The variable corresponding to the faster timescale forms the inner subroutine, whereas the variable on the slower timescale constitutes the outer subroutine. This is ensured by imposing the following condition on the step size parameters along with the conditions introduced in Equation (C.4).

$$\lim_{n\to\infty} \frac{a(n)}{b(n)} \to 0.$$

The inner subroutine views the slow component as quasi-static, whereas the outer subroutine views the fast component as almost equilibrated. Convergence of the joint scheme to the corresponding optimalities follows from the standard ODE approach.

Bibliography

[1] Cisco, "Global mobile data traffic forecast update, 2015–2020 white paper," *white paper*, 2016. [Online]. Available: `https://www.cisco.com/c/dam/m/en_in/innovation/enterprise/assets/mobile-white-paper-c11-520862.pdf`.

[2] P. Cerwall, P. Jonsson, R. Möller, S. Bävertoft, S. Carson, and I. Godor, "Ericsson mobility report," *On the Pulse of the Networked Society*, 2015. [Online]. Available: `https://www.ericsson.com/assets/local/mobility-report/documents/2015/ericsson-mobility-report-june-2015.pdf`.

[3] Z. Wang, *Internet QoS: architectures and mechanisms for quality of service.* Morgan Kaufmann, 2001.

[4] "Third generation partnership project." [Online]. Available: `http://www.3gpp.org/specs/specs.htm."`.

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

[6] 3GPP TS 23.501, "System Architecture for the 5G System," 2017. [Online]. Available: `https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3144`.

[7] 3GPP TS 38.401, "Next Generation Radio Access Network (NG-RAN) Architecture Description," 2017. [Online]. Available: `https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3219`.

[8] 3GPP TR 37.834 v0.3.0, "Study on WLAN/3GPP Radio Interworking," 2013. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2623.

[9] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge university press, 2005.

[10] B. Sklar, "Rayleigh fading channels in mobile digital communication systems part I: Characterization," *IEEE Communications magazine*, vol. 35, no. 7, pp. 90–100, 1997.

[11] M. Gast, *802.11 wireless networks: the definitive guide*. O'Reilly Media, Inc., 2005.

[12] J. H. Schiller, *Mobile communications*. Pearson education, 2003.

[13] IEEE 802.11-2012, Part 11, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," 2012.

[14] www.3gpp.org, "Overview of 3gpp release 6." [Online]. Available: http://www.3gpp.org/ftp/Information/WORK_PLAN/Description_Releases/.

[15] 3GPP TS 23.402 V10.7.0, "Architecture enhancements for non-3GPP accesses," 2012. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=850.

[16] B. O'hara and A. Petrick, *IEEE 802.11 handbook: a designer's companion*. IEEE Standards Association, 2005.

[17] Rackus, "Hotspot 2.0: Making the wi-fi roaming experience as secure and easy to use as with cellular," 2013. [Online]. Available: https://www.ruckussecurity.com/HotSpot-2-0.asp.

[18] A. Whittier, P. Kulkarni, F. Cao, and S. Armour, "Mobile data offloading addressing the service quality vs. resource utilisation dilemma," in *IEEE PIMRC*, pp. 1–6, 2016.

[19] F. Moety, M. Ibrahim, S. Lahoud, and K. Khawam, "Distributed heuristic algorithms for RAT selection in wireless heterogeneous networks," in *IEEE WCNC*, pp. 2220–2224, 2012.

[20] E. Aryafar, A. Keshavarz-Haddad, M. Wang, and M. Chiang, "RAT selection games in HetNets," in *IEEE INFOCOM*, pp. 998–1006, 2013.

[21] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong, "Mobile data offloading: How much can WiFi deliver?," *IEEE/ACM Transactions on Networking*, vol. 21, no. 2, pp. 536–550, 2013.

[22] D. Suh, H. Ko, and S. Pack, "Efficiency analysis of WiFi offloading techniques," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 5, pp. 3813–3817, 2016.

[23] N. Cheng, N. Lu, N. Zhang, X. Zhang, X. S. Shen, and J. W. Mark, "Opportunistic WiFi offloading in vehicular environment: A game-theory approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 7, pp. 1944–1955, 2016.

[24] M. Gerasimenko, N. Himayat, S. P. Yeh, S. Talwar, S. Andreev, and Y. Koucheryavy, "Characterizing performance of load-aware network selection in multi-radio (WiFi/LTE) heterogeneous networks," in *IEEE Globecom Workshops*, pp. 397–402, 2013.

[25] K. Khawam, S. Lahoud, M. Ibrahim, M. Yassin, S. Martin, M. El Helou, and F. Moety, "Radio access technology selection in heterogeneous networks," *Physical Communication*, vol. 18, pp. 125–139, 2016.

[26] S. Barmpounakis, A. Kaloxylos, P. Spapis, and N. Alonistioti, "Context-aware, user-driven, network-controlled RAT selection for 5G networks," *Computer Networks*, vol. 113, pp. 124–147, 2017.

[27] B. H. Jung, N. O. Song, and D. K. Sung, "A network-assisted user-centric WiFi-offloading model for maximizing per-user throughput in a heterogeneous network," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 4, pp. 1940–1945, 2014.

[28] W. Song, Y. Cheng, and W. Zhuang, "Improving voice and data services in cellular/WLAN integrated networks by admission control," *IEEE Transactions on Wireless Communications*, vol. 6, no. 11, pp. 4025–4037, 2007.

[29] X. Gelabert, J. Pérez-Romero, O. Sallent, and R. Agustí, "A Markovian approach to radio access technology selection in heterogeneous multiaccess/multiservice wireless

networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 10, pp. 1257–1270, 2008.

[30] W. Song, H. Jiang, W. Zhuang, and A. Saleh, "Call admission control for integrated voice/data services in cellular/WLAN interworking," in *IEEE ICC*, pp. 5480–5485, 2006.

[31] W. Song and W. Zhuang, "QoS provisioning via admission control in cellular/wireless LAN interworking," in *IEEE BroadNets*, pp. 543–550, 2005.

[32] G. S. Kasbekar, P. Nuggehalli, and J. Kuri, "Online client-AP association in WLANs," in *IEEE WiOpt*, pp. 1–8, 2006.

[33] D. Kumar, E. Altman, and J. M. Kelif, "Globally optimal user-network association in an 802.11 WLAN & 3G UMTS hybrid cell," in *Managing Traffic Performance in Converged Networks*, pp. 1173–1187, Springer, 2007.

[34] E. Khloussy, X. Gelabert, and Y. Jiang, "A revenue-maximizing scheme for radio access technology selection in heterogeneous wireless networks with user profile differentiation," in *Meeting of the European Network of Universities and Companies in Information and Communication Engineering*, pp. 66–77, Springer, 2013.

[35] M. El Helou, M. Ibrahim, S. Lahoud, K. Khawam, D. Mezher, and B. Cousin, "A network-assisted approach for RAT selection in heterogeneous cellular networks," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 6, pp. 1055–1067, 2015.

[36] A. Y. Panah, S. P. Yeh, N. Himayat, and S. Talwar, "Utility-based radio link assignment in multi-radio heterogeneous networks," in *IEEE Globecom Workshops*, pp. 618–623, 2012.

[37] H. Tabrizi, G. Farhadi, and J. Cioffi, "A learning-based network selection method in heterogeneous wireless systems," in *IEEE Globecom*, pp. 1–5, 2011.

[38] D. Pacheco-Paramo, V. Pla, V. Casares-Giner, and J. Martinez-Bauset, "Optimal radio access technology selection on heterogeneous networks," *Physical Communication*, vol. 5, no. 3, pp. 253–271, 2012.

[39] A. Kumar and V. Kumar, “Optimal association of stations and APs in an IEEE 802.11 WLAN,” in *IEEE NCC*, pp. 1–5, 2005.

[40] G. Dandachi, S. E. Elayoubi, T. Chahed, and N. Chendeb, “Network-centric versus user-centric multihoming strategies in LTE/WiFi networks,” *IEEE Transactions on Vehicular Technology*, vol. 66, no. 5, pp. 4188–4199, 2017.

[41] E. Haleplidis, K. Pentikousis, S. Denazis, J. H. Salim, D. Meyer, and O. Koufopavlou, “RFC 7426 - Software-Defined Networking (SDN): Layers and architecture terminology,” 2015. [Online]. Available: `https://tools.ietf.org/html/rfc7426`.

[42] A. M. Nayak, A. Roy, P. Jha, and A. Karandikar, “Control and management of multiple RATs in wireless networks: An SDN approach,” *arXiv preprint arXiv:1801.03819*, 2018.

[43] A. M. Nayak, P. Jha, and A. Karandikar, “A centralized SDN architecture for the 5G cellular network,” in *IEEE 5GWF*, pp. 147–152, 2018.

[44] “Network simulator-3.” [Online]. Available: `http://code.nsnam.org/ns-3-dev/`”.

[45] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, vol. 1. MIT press Cambridge, 1998.

[46] M. Singh and P. Chaporkar, “An efficient and decentralised user association scheme for multiple technology networks,” in *IEEE WiOpt*, pp. 460–467, 2013.

[47] Q. Ye, B. Rong, Y. Chen, M. Al Shalash, C. Caramanis, and J. G. Andrews, “User association for load balancing in heterogeneous cellular networks,” *IEEE Transactions on Wireless Communications*, vol. 12, no. 6, pp. 2706–2716, 2013.

[48] K. Son, S. Chong, and G. De Veciana, “Dynamic association for load balancing and interference avoidance in multi-cell networks,” *IEEE Transactions on Wireless Communications*, vol. 8, no. 7, pp. 3566–3576, 2009.

[49] C. Liu, M. Li, S. V. Hanly, and P. Whiting, “Joint downlink user association and interference management in two-tier HetNets with dynamic resource partitioning,” *IEEE Transactions on Vehicular Technology*, vol. 66, no. 2, pp. 1365–1378, 2017.

[50] R. Li, Z. Zhao, J. Zheng, C. Mei, Y. Cai, and H. Zhang, "The learning and prediction of application-level traffic data in cellular networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 3899–3912, 2017.

[51] K. Kumar, A. Gupta, R. Shah, A. Karandikar, and P. Chaporkar, "On analyzing indian cellular traffic characteristics for energy efficient network operation," in *IEEE NCC*, pp. 1–6, 2015.

[52] U. Paul, A. P. Subramanian, M. M. Buddhikot, and S. R. Das, "Understanding traffic dynamics in cellular data networks," in *IEEE INFOCOM*, pp. 882–890, 2011.

[53] K. Adachi, M. Li, P. H. Tan, Y. Zhou, and S. Sun, "Q-learning based intelligent traffic steering in heterogeneous network," in *IEEE VTC Spring*, pp. 1–5, 2016.

[54] S. Anbalagan, D. Kumar, D. Ghosal, G. Raja, and V. Muthuvalliammai, "SDN-assisted learning approach for data offloading in 5G hetnets," *Mobile Networks and Applications*, pp. 1–12, 2017.

[55] F. Rebecchi, M. D. De Amorim, V. Conan, A. Passarella, R. Bruno, and M. Conti, "Data offloading techniques in cellular networks: a survey," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 580–603, 2015.

[56] Y. He, M. Chen, B. Ge, and M. Guizani, "On WiFi offloading in heterogeneous networks: Various incentives and trade-off strategies," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2345–2385, 2016.

[57] F. Mehmeti and T. Spyropoulos, "Performance analysis of on-the-spot mobile data offloading," in *IEEE Globecom*, pp. 1577–1583, 2013.

[58] S. Ranjan, N. Akhtar, M. Mehta, and A. Karandikar, "User-based integrated offloading approach for 3GPP LTE-WLAN network," in *IEEE NCC*, pp. 1–6, 2014.

[59] N. Wang and J. Wu, "Opportunistic WiFi offloading in a vehicular environment: Waiting or downloading now?," in *Computer Communications, IEEE INFOCOM*, pp. 1–9, 2016.

[60] M. H. Cheung, R. Southwell, and J. Huang, "Congestion-aware network selection and data offloading," in *IEEE CISS*, pp. 1–6, 2014.

[61] D. H. Hagos, "The performance of network-controlled mobile data offloading from LTE to WiFi networks," *Telecommunication Systems*, vol. 61, no. 4, pp. 675–694, 2016.

[62] C. K. Ho, D. Yuan, and S. Sun, "Data offloading in load coupled networks: A utility maximization framework," *IEEE Transactions on Wireless Communications*, vol. 13, no. 4, pp. 1921–1931, 2014.

[63] F. Malandrino, C. Casetti, and C. F. Chiasserini, "LTE offloading: When 3GPP policies are just enough," in *IEEE WONS*, pp. 1–8, 2014.

[64] X. Duan, X. Wang, and A. M. Akhtar, "Partial mobile data offloading with load balancing in heterogeneous cellular networks using software-defined networking," in *IEEE PIMRC*, pp. 1348–1353, 2014.

[65] K. Chen, J. Liu, J. Martin, K. C. Wang, and H. Hu, "Improving integrated LTE-WiFi network performance with SDN based flow scheduling," in *IEEE ICCCN*, pp. 1–9, 2018.

[66] S. Borst, A. Ö. Kaya, D. Calin, and H. Viswanathan, "Dynamic path selection in 5G multi-RAT wireless networks," in *IEEE INFOCOM*, pp. 1–9, 2017.

[67] A. Raschellà, F. Bouhafs, G. Deepak, and M. Mackay, "QoS aware radio access technology selection framework in heterogeneous networks using SDN," *Journal of Communications and Networks*, vol. 19, no. 6, pp. 577–586, 2017.

[68] D. Harutyunyan, S. Herle, D. Maradin, G. Agapiu, and R. Riggio, "Traffic-aware user association in heterogeneous LTE/WiFi radio access networks," in *IEEE/IFIP NOMS*, pp. 1–8, 2018.

[69] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming.* John Wiley & Sons, 2014.

[70] T. Bonald and J. W. Roberts, "Internet and the Erlang formula," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 1, pp. 23–30, 2012.

[71] A. Kumar, "Discrete event stochastic processes," *Lecture Notes for Engineering Curriculum*, 2012.

[72] G. Bianchi, “Performance analysis of the IEEE 802.11 distributed coordination function,” *IEEE Journal on selected areas in communications*, vol. 18, no. 3, pp. 535–547, 2000.

[73] E. Altman, *Constrained Markov decision processes.* CRC Press, 1999.

[74] P. Marbach and J. N. Tsitsiklis, “Simulation-based optimization of Markov reward processes,” *IEEE Transactions on Automatic Control*, vol. 46, no. 2, pp. 191–209, 2001.

[75] F. J. Beutler and K. W. Ross, “Optimal policies for controlled Markov chains with a constraint,” *Journal of mathematical analysis and applications*, vol. 112, no. 1, pp. 236–252, 1985.

[76] 3GPP TR 36.814 v9.0.0, “Further Advancements for E-UTRA Physical Layer Aspects,” 2010. [Online]. Available: `https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2493`.

[77] 3GPP TR 36.839 v11.1.0, “Mobility Enhancements in Heterogeneous Networks,” 2012. [Online]. Available: `https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2540`.

[78] V. S. Borkar, *Stochastic approximation: a dynamical systems viewpoint.* Cambridge University Press, 2008.

[79] 3GPP TS 36.104 V10.2.0, “Base Station (BS) Radio Transmission and Reception,” 2011. [Online]. Available: `https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2412`.

[80] D. B. Johnson and D. A. Maltz, “Dynamic source routing in Ad Hoc wireless networks,” in *Mobile computing*, pp. 153–181, Springer, 1996.

[81] G. A. Brouns and J. Van Der Wal, “Optimal threshold policies in a two-class preemptive priority queue with admission and termination control,” *Queueing Systems*, vol. 54, no. 1, pp. 21–33, 2006.

[82] E. Altman, T. Jiménez, and G. Koole, "On optimal call admission control in resource-sharing system," *IEEE Transactions on Communications*, vol. 49, no. 9, pp. 1659–1668, 2001.

[83] A. Turhan, M. Alanyali, and D. Starobinski, "Optimal admission control in two-class preemptive loss systems," *Operations Research Letters*, vol. 40, no. 6, pp. 510–515, 2012.

[84] V. S. Borkar and S. P. Meyn, "The ODE method for convergence of stochastic approximation and reinforcement learning," *SIAM Journal on Control and Optimization*, vol. 38, no. 2, pp. 447–469, 2000.

[85] N. Salodkar, A. Bhorkar, A. Karandikar, and V. S. Borkar, "An on-line learning algorithm for energy efficient delay constrained scheduling over a fading channel," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, pp. 732–742, 2008.

[86] V. R. Konda and V. S. Borkar, "Actor-critic-type learning algorithms for Markov decision processes," *SIAM Journal on control and Optimization*, vol. 38, no. 1, pp. 94–123, 1999.

[87] J. Abounadi, D. Bertsekas, and V. S. Borkar, "Learning algorithms for Markov decision processes with average cost," *SIAM Journal on Control and Optimization*, vol. 40, no. 3, pp. 681–698, 2001.

[88] V. S. Borkar, "An actor-critic algorithm for constrained Markov decision processes," *Systems & control letters*, vol. 54, no. 3, pp. 207–213, 2005.

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

[90] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in neural information processing systems*, pp. 1057–1063, 2000.

[91] W. B. Powell, *Approximate Dynamic Programming: Solving the curses of dimensionality*, vol. 703. John Wiley & Sons, 2007.

[92] M. Agarwal, V. S. Borkar, and A. Karandikar, “Structural properties of optimal transmission policies over a randomly varying channel,” *IEEE Transactions on Automatic Control*, vol. 53, no. 6, pp. 1476–1491, 2008.

[93] J. E. Smith and K. F. McCardle, “Structural properties of stochastic dynamic programs,” *Operations Research*, vol. 50, no. 5, pp. 796–809, 2002.

[94] A. Sinha and P. Chaporkar, “Optimal power allocation for a renewable energy source,” in *IEEE NCC*, pp. 1–5, 2012.

[95] G. Koole, “Structural results for the control of queueing systems using event-based dynamic programming,” *Queueing systems*, vol. 30, no. 3-4, pp. 323–339, 1998.

[96] M. H. Ngo and V. Krishnamurthy, “Optimality of threshold policies for transmission scheduling in correlated fading channels,” *IEEE Transactions on Communications*, vol. 57, no. 8, 2009.

[97] S. Kunnumkal and H. Topaloglu, “Exploiting the structural properties of the underlying Markov decision problem in the Q-learning algorithm,” *INFORMS Journal on Computing*, vol. 20, no. 2, pp. 288–301, 2008.

[98] F. Fu and M. van der Schaar, “Structure-aware stochastic control for transmission scheduling,” *IEEE Transactions on Vehicular Technology*, vol. 61, no. 9, pp. 3931–3945, 2012.

[99] M. H. Ngo and V. Krishnamurthy, “Monotonicity of constrained optimal transmission policies in correlated fading channels with ARQ,” *IEEE Transactions on Signal Processing*, vol. 58, no. 1, pp. 438–451, 2010.

[100] V. S. Borkar and K. Chadha, “A reinforcement learning algorithm for restless bandits,” in *IEEE ICC*, pp. 89–94, 2018.

[101] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-dynamic programming*, vol. 5. Athena Scientific Belmont, MA, 1996.

[102] J. C. Spall, *Introduction to stochastic search and optimization: estimation, simulation, and control*, vol. 65. John Wiley & Sons, 2005.

List of Publications

Part of Ph.D. Thesis

Journal Publications

J1 A. Roy, P. Chaporkar, and A. Karandikar, “Optimal Radio Access Technology Selection Algorithm for LTE-WiFi Network”, *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 6446-6460, 2018.

J2 A. Roy, V. Borkar, P. Chaporkar, and A. Karandikar, “Low Complexity Online Radio Access Technology Selection Algorithm in LTE-WiFi HetNet”, Available in early access in *IEEE Transactions on Mobile Computing*, 2019.

International Conference Publications

C1 A. Roy and A. Karandikar, “Optimal Radio Access Technology Selection Policy for LTE-WiFi Network”, in *IEEE WiOpt*, pp. 291-298, 2015.

C2 A. Roy, P. Chaporkar, and A. Karandikar, “An On-Line RAT Selection Algorithm in an LTE-WiFi Network”, in *IEEE WCNC*, pp. 1-6, 2017.

C3 A. Roy, V. Borkar, A. Karandikar, and P. Chaporkar, “A Structure-aware Online Learning Algorithm for Markov Decision Processes”, in *ACM VALUETOOLS*, pp. 71-78, 2019.

C4 A. Roy, P. Chaporkar, A. Karandikar, and P. Jha,“Optimal Radio Access Technology Selection in an SDN based LTE-WiFi Network”, in *RAWNET Workshop (WiOpt)*, pp. 1-8, 2019.

Preprints/Submitted/Under Preparation

J3 A. Roy, P. Chaporkar, A. Karandikar, and P. Jha, "Optimal Radio Access Technology Selection Algorithms in an SDN based LTE-WiFi Network", In Preparation (Journal).

Other Publications

International Conference Publications

C5 P. K. Taksande, A. Roy, and A. Karandikar, "Optimal Traffic Splitting Policy in LTE-based Heterogeneous Network", in *IEEE WCNC*, pp. 1-6, 2018.

Preprints/Submitted/Under Preparation

C6 A. Nayak M., A. Roy, P. Jha, and A. Karandikar, "Control and Management of Heterogeneous RATs in 5G Wireless Networks: An SDN Approach", Under review.

J4 P. S. Deogun, A. Roy, and A. Karandikar, "An Energy-Aware WLAN Discovery Scheme for LTE HetNet", ArXiv e-prints, [Online] Available: `https://arxiv.org/pdf/1710.11455.pdf`.

J5 P. K. Taksande, A. Roy, and A. Karandikar, "Optimal Traffic Splitting Algorithm in LTE- based Heterogeneous Network", In Preparation (Journal).

Patents

P1 A. Karandikar, P. K. Jha, A. Nayak M., N. Shah, A. Roy, O. Kanhere, P. Pola, A. Dandekar, and R. Kharade, "Methods and Systems for Controlling a SDN based Multi-RAT Communication Network", Patent filed, 2017 (India) (201721008015) and and Patent Granted, 2018 (USA) (15/582145).