ONLINE LEARNING ALGORITHMS FOR NETWORK OPTIMIZATION WITH UNKNOWN VARIABLES

by

Yi Gai

A Dissertation Presented to the FACULTY OF THE USC GRADUATE SCHOOL UNIVERSITY OF SOUTHERN CALIFORNIA In Partial Fulfillment of the Requirements for the Degree DOCTOR OF PHILOSOPHY (ELECTRICAL ENGINEERING)

December 2012

Copyright 2012 Yi Gai
Dedication

To my beloved parents, Hua Zhao and Wenquan Gai.
Acknowledgments

This work would not have been possible if it were not for the help of many people. First, I would like to thank my advisor, Prof. Bhaskar Krishnamachari. I could not have been more fortunate to have such a fine advisor as Bhaskar. He opened my mind to a world of opportunities and taught me about research, learning, teaching, career planning, and so much else. He has always been very kind, patient, and generous. I enjoyed the many insightful discussions with him. Those moments of deep happiness from conquering research difficulties with him will always be precious to me.

I thank my dissertation committee for their guidance and support, including Prof. Rahul Jain and Prof. Shaddin Dughmi.

Special thanks go to my research collaborators. Matthew P. Johnson is a terrific friend and researcher who gave me plenty of very useful suggestions. I also thank Prof. Mingyan Liu, Prof. Qing Zhao, Prof. Amotz Bar-Noy, Wenhan Dai, George Rabanca and Naumaan Nayyar. This dissertation benefited enormously from all of them.

At the University of Southern California, I received the support of various faculty, students, and colleagues. Prof. Michael Neely’s classes were the most fun and educational classes that I have ever taken. I thank Prof. David Kempe for his valuable feedback.
on my papers. I was also helped by Prof. Urbashi Mitra, Prof. Antonio Ortega, Ying Chen, Scott Moeller, Maheswaran Sathiamoorthy, Yi Wang, Vlad Balan, Mi Zhang, Hua Liu, Pai-Han Huang, Joon Ahn, Sangwon Lee, Suvil Singh, Amitabha Ghosh, Majed Al-resaini, Marjan Baghaie, Sundeep Pattem, Avinash Sridharan, Yanting Wu, Qiumin Xu, Shangxing Wang and Peter Ungsunan.

I thank all my advisors from the earlier stages of my academic career, each of whom has influenced me, including Prof. Lin Zhang, who gave me lots of support at Tsinghua University.

Finally, I’d like to thank my parents and friends for their emotional support.
# Table of Contents

Dedication ii  

Acknowledgments iii  

List of Figures viii  

List of Tables x  

Abstract xii  

**Chapter 1: Introduction** 1  

1.1 Contributions 6  

1.1.1 Learning with Linear Rewards 6  

1.1.1.1 I.I.D Rewards 6  

1.1.1.2 Rested Markovian Rewards 9  

1.1.1.3 Restless Markovian Rewards 11  

1.1.2 Learning for Stochastic Water-Filling with Linear and Nonlinear Rewards 12  

1.1.3 Learning in Decentralized Settings 14  

1.2 Organization 15  

**Chapter 2: Background: Classic Multi-Armed Bandits** 17  

2.1 Problem Formulation 17  

2.2 Online Learning Algorithms for Classic MABs 19  

2.2.1 Lai-Robbins Policy 19  

2.2.2 UCB1 Policy 20  

**Chapter 3: Related Work** 23  

3.1 Independent Arms with Temporally I.I.D. Rewards 23  

3.2 Dependent Arms with Temporally I.I.D. Rewards 26  

3.3 Independent Arms with Rested Markovian Rewards 28  

3.4 Independent Arms with Restless Markovian Rewards 29  

3.5 Decentralized Policy for Multi-Armed Bandits 30
<table>
<thead>
<tr>
<th>Chapter 4: Learning with I.I.D. Linear Rewards</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Overview</td>
<td>34</td>
</tr>
<tr>
<td>4.2 Problem Formulation</td>
<td>35</td>
</tr>
<tr>
<td>4.3 Policy Design</td>
<td>37</td>
</tr>
<tr>
<td>4.3.1 A Naive Approach</td>
<td>37</td>
</tr>
<tr>
<td>4.3.2 A New Policy</td>
<td>38</td>
</tr>
<tr>
<td>4.4 Analysis of Regret</td>
<td>41</td>
</tr>
<tr>
<td>4.5 Applications</td>
<td>48</td>
</tr>
<tr>
<td>4.5.1 Maximum Weighted Matching</td>
<td>48</td>
</tr>
<tr>
<td>4.5.2 Shortest Path</td>
<td>52</td>
</tr>
<tr>
<td>4.5.3 Minimum Spanning Tree</td>
<td>55</td>
</tr>
<tr>
<td>4.6 Examples and Simulation Results</td>
<td>56</td>
</tr>
<tr>
<td>4.7 K Simultaneous Actions</td>
<td>58</td>
</tr>
<tr>
<td>4.8 LLR with Approximation Algorithm</td>
<td>61</td>
</tr>
<tr>
<td>4.9 Summary</td>
<td>64</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 5: Learning with Rested Markovian Rewards</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1 Overview</td>
<td>66</td>
</tr>
<tr>
<td>5.2 Problem Formulation</td>
<td>67</td>
</tr>
<tr>
<td>5.3 Matching Learning for Markovian Rewards</td>
<td>70</td>
</tr>
<tr>
<td>5.4 Analysis of Regret</td>
<td>72</td>
</tr>
<tr>
<td>5.5 Examples and Simulation Results</td>
<td>87</td>
</tr>
<tr>
<td>5.6 Summary</td>
<td>91</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 6: Learning with Restless Markovian Rewards</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Overview</td>
<td>92</td>
</tr>
<tr>
<td>6.2 Problem Formulation</td>
<td>92</td>
</tr>
<tr>
<td>6.3 Policy Design</td>
<td>95</td>
</tr>
<tr>
<td>6.4 Analysis of Regret</td>
<td>99</td>
</tr>
<tr>
<td>6.5 Applications and Simulation Results</td>
<td>118</td>
</tr>
<tr>
<td>6.5.1 Stochastic Shortest Path</td>
<td>118</td>
</tr>
<tr>
<td>6.5.2 Stochastic Bipartite Matching for Channel Allocation</td>
<td>123</td>
</tr>
<tr>
<td>6.6 Summary</td>
<td>126</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 7: Learning for Stochastic Water-Filling</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1 Overview</td>
<td>127</td>
</tr>
<tr>
<td>7.2 Problem Formulation</td>
<td>128</td>
</tr>
<tr>
<td>7.3 Online Learning for Maximizing the Sum-Rate</td>
<td>132</td>
</tr>
<tr>
<td>7.3.1 Policy Design</td>
<td>133</td>
</tr>
<tr>
<td>7.3.2 Analysis of Regret</td>
<td>135</td>
</tr>
<tr>
<td>7.4 Online Learning for Sum-Pseudo-Rate</td>
<td>142</td>
</tr>
</tbody>
</table>
# List of Figures

1.1 Overview of this dissertation. ........................................ 5

4.1 An illustrative scenario. ............................................. 50

4.2 Simulation results of a system with 7 orthogonal channels and 4 users. ... 56

4.3 Simulation results of a system with 9 orthogonal channels and 5 users. ... 57

5.1 Simulation results of example 1 with $\Delta_{\text{min}} = 0.1706$. ............... 89

5.2 Simulation results of example 2 with $\Delta_{\text{min}} = 0.0091$. ............... 90

6.1 An illustration of CLRMR. ........................................... 98

6.2 Two example graphs for stochastic shortest path routing. ..................... 119

6.3 Normalized regret $\frac{\mathcal{R}(n)}{\ln n}$ vs. $n$ time slots. ..................... 121

6.4 Normalized regret $\frac{\mathcal{R}(n)}{\ln n}$ vs. $n$ time slots. ..................... 125

7.1 Normalized regret $\frac{\mathcal{R}(n)}{\log n}$ vs. $n$ time slots. ..................... 150

7.2 Numerical results of $E[\tilde{T}_i(n)]/\log n$ and theoretical bound. ........... 152

7.3 Normalized regret $\frac{\mathcal{R}(n)}{\log n}$ vs. $n$ time slots. ..................... 153

8.1 Illustration of rotating the prioritization vector. ............................. 170

8.2 Normalized regret $\frac{\mathcal{R}(n)}{\ln n}$ vs. $n$ time slots. ..................... 178
8.3 Number of times that channel $i$ has been chosen by user $m$ up to time $n = 10^6$.

8.4 Comparison of DLF and TDFS.
List of Tables

4.1 Notation. .................................................. 40
4.2 Comparison of regret bounds. .................................. 55
4.3 Regret when $t = 2 \times 10^6$. .................................. 58
5.1 Notation. .................................................. 74
5.2 Transition probabilities. ...................................... 88
5.3 Rewards on each state. ....................................... 88
5.4 Expected rewards. ........................................... 88
5.5 Number of times that resource $j$ has been matched with user $i$ up to time $n = 10^7$. .................................. 90
5.6 Rewards on each state. ....................................... 90
5.7 Expected rewards. ........................................... 91
5.8 Number of times that resource $j$ has been matched with user $i$ up to time $n = 10^7$. .................................. 91
6.1 Notation for Algorithm 7. ................................. 96
6.2 Notation for regret analysis. ............................... 101
6.3 Transition probabilities. ..................................... 120
6.4 Transition probabilities. ..................................... 120
6.5 Transition probabilities $p_{01}, p_{10}$ for each user-channel pair.  

6.6 Transition probabilities $p_{01}, p_{10}$ for each user-channel pair.
Abstract

The formulations and theories of multi-armed bandit (MAB) problems provide fundamental tools for optimal sequential decision making and learning in uncertain environments. They have been widely applied to resource allocation, scheduling, and routing in communication networks, particularly in recent years, as the field is seeing an increasing focus on adaptive online learning algorithms to enhance system performance in stochastic, dynamic, and distributed environments. This dissertation addresses several key problems in this domain.

Our first focus is about MAB with linear rewards. As they are fundamentally about combinatorial optimization in unknown environments, one would indeed expect to find even broader use of multi-armed bandits. However, a barrier to their wider application in practice has been the limitation of the basic formulation and corresponding policies, which generally treat each arm as an independent entity. They are inadequate to deal with many combinatorial problems of practical interest in which there are large numbers of arms. In such settings, it is important to consider and exploit any structure in terms of dependencies between the arms. In this dissertation, we show that when the dependencies take a linear form, they can be handled tractably with algorithms that have provably good...
performance in terms of regret as well as storage and computation. We develop a new class of learning algorithms for different problem settings including i.i.d. rewards, rested Markovian rewards, and restless Markovian rewards, to improve the cost of learning, compared to prior work, for large-scale stochastic network optimization problems.

We then consider the problem of optimal power allocation over parallel channels with stochastically time-varying gain-to-noise ratios for maximizing information rate (stochastic water-filling) with both linear and non-linear multi-armed bandit formulations and propose new efficient online learning algorithms for these.

Finally, we focus on learning in decentralized settings. The desired objective is to develop decentralized online learning algorithms running at each user to make a selection among multiple choices, where there is no information exchange, such that the sum-throughput of all distributed users is maximized. We make two contributions in this problem. First, we consider the setting where the users have a prioritized ranking, such that it is desired for the $K$-th ranked user to learn to access the arm offering the $K$-th highest mean reward. For this problem, we present the first distributed algorithm that yields regret that is uniformly logarithmic over time without requiring any prior assumption about the mean rewards. Second, we consider the case when a fair access policy is required, i.e., it is desired for all users to experience the same mean reward. For this problem, we present a distributed algorithm that yields order-optimal regret scaling with respect to the number of users and arms, better than previously proposed algorithms in the literature.
Chapter 1

Introduction

Multi-armed bandit (MAB) problems provide a fundamental approach to learning under stochastic rewards, and find rich applications in a wide range of networking contexts, from Internet advertising [69] to medium access in cognitive radio networks [59, 75]. In the simplest, classic non-Bayesian version of the problem, studied by Lai and Robbins [52], there are $K$ independent arms, each generating stochastic rewards that are i.i.d. over time. The player is unaware of the parameters for each arm, and must use some policy to play the arms in such a way as to maximize the cumulative expected reward over the long term. The policy’s performance is measured in terms of its regret, defined as the gap between the the expected reward that could be obtained by an omniscient user that knows the parameters for the stochastic rewards generated by each arm and the expected cumulative reward of that policy. It is of interest to characterize the growth of regret with respect to time as well as with respect to the number of arms/players. Intuitively, if the regret grows sublinearly over time, the time-averaged regret tends to zero.
There is inherently a tradeoff between exploration and exploitation in the learning process in a multi-armed bandit problem: on the one hand all arms need to be sampled periodically by the policy used, to ensure that the “true” best arm is found; on the other hand, the policy should play the arm that is considered to be the best often enough to accumulate rewards at a good pace. To quote Peter Whittle [78]: bandit problems “embody in essential form a conflict evident in all human action. This is the conflict between taking those actions which yield immediate reward and those (such as acquiring information or skill, or preparing the ground) whose benefit will come only later.”

The classical treatment of multi-armed bandits assumes that arms are independent. Further, existing algorithms for the classic MAB, such as the Lai-Robbins policy [52] and Auer et al.’s UCB1 [13], yield regret that is linear in the number of arms. In many problems of practical interest, particularly those arising in the context of communication networks such as stochastic shortest path and minimum spanning tree computation and scheduling based on maximum-weight matching on bipartite graphs (with unknown stochastic weights), there are dependencies between a large number of arms that can be described by a smaller set of unknown parameters (the number of arms could be exponential in the number of these parameters, for example, the number of possible paths vs. the number of edges).

As they are fundamentally about combinatorial optimization in unknown environments, one would indeed expect to find even broader use of multi-armed bandits. However, we argue that a barrier to their wider application in practice has been the limitation
of the basic formulation and corresponding policies, which generally treat each arm as an independent entity. They are inadequate to deal with many combinatorial problems of practical interest in which there are large (exponential) numbers of arms.

Our first focus in this dissertation is to consider and exploit any structure in terms of dependencies between the arms to learn more efficiently for stochastic network optimization problems under different settings.

When the dependencies take a linear form, we show in Chapter 4 that for the i.i.d. rewards, they can be handled tractably with policies that have provably good performance in terms of regret as well as storage and computation.

While the majority of the literature on MAB has focused on the i.i.d. reward model, extensions exist to the rested Markovian reward model where the reward state of each arm evolves as an unknown Markov process over successive plays and remains frozen when the arm is not played [10, 74], and restless Markovian reward model where the reward state of each arm evolves dynamically following unknown stochastic processes no matter it is played or not [25, 56, 57, 75]. However, these prior works deal with the dependencies between arms inefficiently.

For rested and restless Markovian rewards when the dependencies take a linear form, we present in Chapter 5 and Chapter 6 online learning algorithms that are designed for the setting where the edge weights are modeled by finite-state Markov chains, with unknown transition matrices.
While Chapter 4 to Chapter 6 provide efficient algorithms for MAB with linear rewards, we are also interested in going beyond the linear reward formulation to solve the more general formulations of MAB with non-linear rewards.

One classic optimization problem in communication systems with a non-linear objective function is that of rate-maximizing constrained power allocation over parallel channels, which is solved by the well-known water-filling algorithm in the deterministic setting [40]. We show in Chapter 7 two distinct but related MAB formulations of stochastic water-filling in which the gain to noise ratio for each channel evolves over time as an i.i.d. random process. For the first problem, we propose a cognitive water-filling algorithm that exploits the linear structure of this problem. For the second problem, we present the first MAB policy with provable regret performance to exploit non-linear dependencies between arms.

Another focus of this dissertation is on designing and analyzing decentralized algorithms for MAB. The classical MAB formulation considers only a single player, which, in a network setting, can only handle centralized configurations where all players act collectively as a single entity by exchanging observations and making decisions jointly. In many applications, however, information exchange among players and joint decision making can be costly or even infeasible due to the competitions among players. Thus, we have a decentralized problem in which multiple distributed players learn from their local observations and make decisions independently. Decentralized players’ actions affect each other since other players’ observations and actions are unknown. Conflicts occur
online Learning Algorithms (with Applications in Networks)

Learning for Combinatorial Network Optimization

- MAB with Linear Rewards
  - i.i.d. (Chapter 3)
  - Rested MAB (Chapter 4)

- MAB with non-Linear Rewards (Chapter 6)
  - Markovian
  - Restless MAB (Chapter 5)

Decentralized Learning

- Prioritized Access (Chapter 7)
- Fair Access (Chapter 7)

Figure 1.1: Overview of this dissertation.

when multiple players choose the same arm at the same time and conflicting players can only share the reward offered by the arm, not necessarily with conservation. All these make learning more difficult for decentralized MAB problems. We present in Chapter 8 decentralized learning algorithms in the context of opportunistic spectrum access.

Figure 1.1 shows the structure of research mentioned in this dissertation on online learning algorithms with the multi-armed bandit formulations. We have also obtained some additional related results on learning algorithms for non-bayesian restless MAB (RMAB), that are not included in this dissertation. In [26], we propose an algorithm to learn the optimal policy for the non-Bayesian RMAB with identical transition matrices, by employing a suitable meta-policy which treats each policy from a finite set as an arm in a different non-Bayesian multi-armed bandit problem for which a single-arm selection policy is optimal. Our results on the non-Bayesian RMAB with non-identical transition matrices can be found in [66].
While this dissertation has focused on the stochastic reward model, we should note that another variant of the MAB problem, adversarial reward model, has been widely studied [2, 14]. For adversarial reward model (or adversarial bandits), no statistical assumptions are made about the generation of rewards. An adversary, rather than a well-behaved stochastic process, has complete control over the payoffs.

1.1 Contributions

Given the ready applicability of MAB to a wide range of communication systems, as evidenced by many papers, it is clear that progress in expanding the boundaries of knowledge on algorithms and performance beyond classical MAB will have significant impact on the design of efficient communication network protocols in unknown stochastic environments. We now detail our contributions.

1.1.1 Learning with Linear Rewards

1.1.1.1 I.I.D Rewards

In Chapter 4, we consider the following multi-armed bandit problem. There are \( N \) random variables with unknown mean that are each instantiated in an i.i.d. fashion over time. At each time a particular set of multiple random variables can be selected, subject to a general arbitrary constraint on weights associated with the selected variables.
All of the selected individual random variables are observed at that time, and a linearly weighted combination of these selected variables is yielded as the reward.

Our general formulation of multi-armed bandits with linear rewards is applicable to a very broad class of combinatorial network optimization problems with linear objectives. These include maximum weight matching in bipartite graphs (which is useful for user-channel allocations in cognitive radio networks), as well as shortest path, and minimum spanning tree computation. In these examples, there are random variables associated with each edge on a given graph, and the constraints on the set of elements allowed to be selected at each time correspond to sets of edges that form relevant graph structures (such as matchings, paths, or spanning trees).

Because our formulation allows for arbitrary constraints on the multiple elements that are selected at each time, prior work on multi-armed bandits that only allow for a fixed-number of multiple plays along with individual observations at each time [3, 11] cannot be directly used for this more general problem. On the other hand, by treating each feasible weighted combination of elements as a distinct arm, it is possible to handle these constraints using prior approaches for multi-armed bandits with single play (such as the well-known UCB1 index policy of Auer et al. [13]). However this approach turns out to be naive, and yields poor performance scaling in terms of regret, storage, and computation. This is because this approach maintains and computes quantities for each possible combination separately and does not exploit potential dependencies between them. In Chapter 4, we instead propose smarter policies to handle the arbitrary constraints, that
explicitly take into account the linear nature of the dependencies and base all storage and computations on the unknown variables directly. As we shall show, this saves not only on storage and computation, but also substantially reduces the regret compared to the naive approach.

Specifically, we first present a novel policy called Learning with Linear Rewards (LLR) that requires only $O(N)$ storage, and yields a regret that grows essentially\(^1\) as $O(N^4 \log n)$, where $n$ is the time index. We also discuss how this policy can be modified in a straightforward manner while maintaining the same performance guarantees when the problem is one of cost minimization rather than reward maximization. A key step in these policies we propose is the solving of a deterministic combinatorial optimization with a linear objective. While this is NP-hard in general (as it includes 0-1 integer linear programming), there are still many special-case combinatorial problems of practical interest which can be solved in polynomial time. For such problems, the policy we propose would thus inherit the property of polynomial computation at each step. Further, we present suitably relaxed results on the regret that would be obtained for computationally harder problems when an approximation algorithm with a known guarantee is used.

We also present in Chapter 4 a more general $K$-action formulation, in which the policy is allowed to pick $K \geq 1$ different combinations of variables each time. We show how the basic LLR policy can be readily extended to handle this and present the regret analysis for this case as well.

---

\(^1\)This is a simplification of our key result in section 4.4 which gives a tighter expression for the bound on regret that applies uniformly over time, not just asymptotically.
The examples of combinatorial network optimization that we present are far from exhausting the possible applications of the formulation and the policies we present in this work — there are many other linear-objective network optimization problems [6, 46]. Our framework, for the first time, allows these problems to be solved in stochastic settings with unknown random coefficients, with provably efficient performance. Besides communication networks, we expect that our work will also find practical application in other fields where linear combinatorial optimization problems arise naturally, such as algorithmic economics, data mining, finance, operations research and industrial engineering.

1.1.1.2 Rested Markovian Rewards

In Chapter 5, we formulate a novel combinatorial generalization of the multi-armed bandit problem that allows for rested Markovian rewards and propose an efficient policy for it. In particular, there is a given bipartite graph of $M$ users and $N \geq M$ resources. For each user-resource pair $(i, j)$, there is an associated state that evolves as an aperiodic irreducible finite-state Markov chain with unknown parameters, with transitions occurring each time the particular user $i$ is allocated resource $j$. The user $i$ receives a reward that depends on the corresponding state each time it is allocated the resource $j$. A key difference from the classic multi-armed bandit is that each user can potentially see a different reward process for the same resource. If we therefore view each possible matching of users to resources as an arm, then we have a exponential number of arms with dependent
rewards. Thus, this new formulation is significantly more challenging than the traditional multi-armed bandit problems.

Because our formulation allows for user-resource matching, it could be potentially applied to a diverse range of networking settings such as switching in routers (where inputs need to be matched to outputs) or frequency scheduling in wireless networks (where nodes need to be allocated to channels) or for server assignment problems (for allocating computational resources for various processes), etc., with the objective of learning as quickly as possible so as to maximize the usage of the best options. For instance, our formulation is general enough to be applied to the channel allocation problem in cognitive radio networks considered in [35] if the rewards for each user-channel pair come from a discrete set and are i.i.d. over time (which is a special case of Markovian rewards).

Our main contribution in Chapter 5 is the design of a novel policy for this problem that we refer to Matching Learning for Markovian Rewards (MLMR). Since we treat each possible matching of users to resources as an arm, the number of arms in our formulation grows exponentially. However, MLMR uses only polynomial storage, and requires only polynomial computation at each step. We analyze the regret for this policy with respect to the best possible static matching, and show that it is uniformly logarithmic over time under some restrictions on the underlying Markov process. We also show that when these restrictions are removed, the regret can still be made arbitrarily close to logarithmic with respect to time. In either case, the regret is polynomial in the number of users and resources.
1.1.1.3 Restless Markovian Rewards

We present in Chapter 6 an online learning algorithm that is designed for the setting where the edge weights are modeled by finite-state Markov chains, with unknown transition matrices. We specifically model this problem as a combinatorial multi-armed bandit problem with restless Markovian rewards.

We consider a single-action regret definition, whereby the genie is assumed to know the transition matrices for all edges, but is constrained to stick with one action (corresponding to a particular network structure) at all times\(^2\). We prove that our algorithm, which we refer to as CLRMR (Combinatorial Learning with Restless Markov Rewards) achieves a regret that is polynomial in the number of Markov chains (i.e., number of edges), and logarithmic with time. This implies that our learning algorithm, which does not know the transition matrices, asymptotically achieves the maximum time averaged reward possible with any single-action policy, even if that policy is given advanced knowledge of the transition matrices. By contrast, the conventional approach of estimating the mean of each edge weight and then finding the desired network structure via deterministic optimization would incur greater overhead and provide only linearly increasing regret over time, which is not asymptotically optimal.

While recent work has shown how to address multi-armed bandits with restless Markovian rewards in the classic non-combinatorial setting [75], this dissertation is the first to show how to efficiently implement online learning for stochastic combinatorial network

---

\(^2\)Although a stronger notion of regret can be defined, allowing the genie to vary the action at each time, the problem of minimizing such a stronger regret is much harder and remains open even for simpler settings than the one we consider here.
optimization when edge weights are dynamically evolving as restless Markovian processes. We perform simulations to evaluate our new algorithm over two combinatorial network optimization problems: stochastic shortest path routing and bipartite matching for channel allocation, and show that its regret performance is substantially better than that of the algorithm presented in [75], which can handle restless Markovian rewards but does not exploit the dependence between the arms, resulting in a regret that grows exponentially in the number of unknown variables.

1.1.2 Learning for Stochastic Water-Filling with Linear and Nonlinear Rewards

A fundamental resource allocation problem that arises in many settings in wireless communication systems is to allocate a constrained amount of power across many parallel channels in order to maximize the sum-rate. Assuming that the power-rate function for each channel is proportional to $\log(1 + SNR)$ as per the Shannon’s capacity theorem for AWGN channels, it is well known that the optimal power allocation can be determined by a water-filling strategy [24]. The classic water-filling solution is a deterministic algorithm, and requires perfect knowledge of all channel gain to noise ratios. Water-filling is used widely in practical wireless networks, for example, for power allocation to subcarriers in multi-user OFDM systems such as WiMax.

In practice, however, channel gain-to-noise ratios are stochastic quantities. Traditionally this is handled by estimating their mean and applying the deterministic optimization.
We consider here an alternative approach based on online learning, specifically stochastic multi-armed bandits. We formalize stochastic water-filling as follows: time is discretized into slots; each channel’s gain-to-noise ratio is modeled as an i.i.d. random variable with an unknown distribution. In our general formulation, the power-to-rate function for each channel is allowed to be any sub-additive function \(^3\).

We consider in Chapter 7 two distinct but related MAB formulations of stochastic water-filling.

First, we seek a power allocation that maximizes the expected sum-rate (i.e., an optimization of the form \(\mathbb{E}\left[\sum_i \log(1 + SNR_i)\right]\)). Even if the channel gain-to-noise ratios are random variables with known distributions, this turns out to be a hard combinatorial stochastic optimization problem. Our focus is thus on a more challenging case.

We present a novel combinatorial policy for this first problem that we call CWF1, that yields regret growing polynomially in \(N\) and logarithmically over time. Despite the exponential growing set of arms, the CWF1 observes and maintains information for \(P \cdot N\) variables, one corresponding to each power-level and channel, and exploits linear dependencies between the arms based on these variables.

Second, we consider identifying the power allocation that maximizes \(\sum_i \log(1 + \mathbb{E}[SNR_i])\). This is motivated by the fact that typically, the way the randomness in the channel gain to noise ratios is dealt with is that the mean channel gain to noise ratios are

\(^3\)A function \(f\) is subadditive if \(f(x + y) \leq f(x) + f(y)\); for any concave function \(g\), if \(g(0) \geq 0\) (such as \(\log(1 + x)\)), \(g\) is subadditive.
estimated first based on averaging a finite set of training observations and then the estimated gains are used in a deterministic water-filling procedure. Essentially this approach tries to identify the power allocation that maximizes a pseudo-sum-rate, which is determined based on the power-rate equation applied to the mean channel gain-to-noise ratios. For this second formulation, we present a different stochastic water-filling algorithm that we call CWF2, which learns the optimal power allocation to maximize this function in an online fashion. This algorithm observes and maintains information for $N$ variables, one corresponding to each channel, and exploits non-linear dependencies between the arms based on these variables. To our knowledge, CWF2 is the first MAB algorithm to exploit non-linear dependencies between the arms. We show that the number of times CWF2 plays a non-optimal combination of powers is uniformly bounded by a function that is logarithmic in time. Under some restrictive conditions, CWF2 may also solve the first problem more efficiently.

### 1.1.3 Learning in Decentralized Settings

There are two problem formulations of interest when considering distributed MAB: a) the *prioritized access problem*, where it is desired to prioritize a ranked set of users so that the $K$-th ranked user learns to access the arm with the $K$-th highest reward, and b) the *fair access problem*, where the goal is to ensure that each user receives the same reward in expectation.
In Chapter 8, we make significant new contributions to both problem formulations.

For the prioritized access problem, we present a distributed learning policy DLP that results in a regret that is uniformly logarithmic in time and, unlike the prior work in [9], does not require any prior knowledge about the arm reward means. For the fair access problem, we present another distributed learning policy DLF, which yields regret that is also uniformly logarithmic in time and that scales as $O(M(N - M))$ with respect to the number of users $M$ and the number of arms $N$. As it has been shown in [59] that the lower-bound of regret for distributed policies also scales as $\Omega(M(N - M))$, this is not only a better scaling than the previous state of the art, it is, in fact, order-optimal.

A key subroutine of both decentralized learning policies running at each user involves selecting an arm with the desired rank order of the mean reward. For this, we present a new policy that we refer to as SL($K$), which is a non-trivial generalization of UCB1 in [13]. SL($K$) provides a general solution for selecting an arm with the $K$-th largest expected rewards for classic MAB problems with $N$ arms.

### 1.2 Organization

The rest of the dissertation is organized as follows. We present some background material and a brief survey of relevant studies in the literature in Chapter 2 and Chapter 3. In Chapter 4, we formulate the combinatorial multi-armed bandit (MAB) problem with linear rewards and individual observations, and develop new efficient policies for this problem, that are shown to achieve regret that grows logarithmically with time, and
polynomially in the number of unknown variables. In Chapter 5, we consider the problem with rested Markovian rewards. Then, we investigate the restless setting in Chapter 6. In Chapter 7, we consider the Stochastic Water-Filling problem, which involves the problem formulations with both linear and nonlinear rewards. We investigate the decentralized online learning for multi-armed bandits in Chapter 8. Finally, we present concluding comments and indicate some open directions for future work in Chapter 9.
Chapter 2

Background: Classic Multi-Armed Bandits

The first multi-armed bandit (MAB) problem is posed by Thompson in 1933 for the application of clinical trials [76]. Since then, MAB has developed into an important branch in stochastic optimization and machine learning. It has recently gained increasing attention from the communications and networking research community due to its ability of formulating and tackling the optimization of learning and activation in a dynamic environment, often under unknown models.

2.1 Problem Formulation

In the simplest, classic non-Bayesian version of the problem, studied by Lai and Robbins [52], there are $K$ independent arms, each generating stochastic rewards that are i.i.d. over time, defined by random variables $X_{i,n}$ for $1 \leq i \leq K$. Time is slotted and indexed by $n \geq 1$. The stochastic rewards are generated independently across arms, i.e., $X_{i,n}$ and
$X_{j,n_j}$ are independent (and usually not identically distributed) for each $1 \leq i < j \leq K$ and each $n_i, n_j \geq 1$.

A policy, or allocation strategy, $\phi = \{\phi(n)\}_{n=1}^{\infty}$ is an algorithm that chooses the next arm to play based on the sequence of past plays and obtained rewards.

The player is unaware of the parameters for each arm (or the distributions of the random variables), and must use some algorithm, or policy, or allocation strategy, to play the arms (based on the sequence of past plays and obtained rewards) in such a way as to maximize the cumulative expected reward over the long term.

The policy’s performance is measured in terms of its regret, defined as the gap between the expected reward that could be obtained by an omniscient player that knows the parameters for the stochastic rewards generated by each arm and the expected cumulative reward of that policy. Let $T_i(n)$ be the number of times arm $i$ has been played by $\phi$ during the first $n$ plays. Then the regret of $\phi$ after $n$ plays is defined by

$$\theta^* n - \theta \sum_{i=1}^{K} \mathbb{E}[T_j(n)]$$

where $\theta_i$ is the expected reward got on arm $i$ and $\theta^* = \max_{1 \leq j \leq K} \theta_j$.

It is of interest to characterize the growth of regret with respect to time as well as with respect to the number of arms/players. Intuitively, if the regret grows sublinearly over time, the time-averaged regret tends to zero.
2.2 Online Learning Algorithms for Classic MABs

2.2.1 Lai-Robbins Policy

Multi-armed bandit problems provide a fundamental approach to learning under stochastic rewards. Lai and Robbins [52] are among the earliest ones to study the classic multi-armed bandit problems with the i.i.d formulation, for specific families of reward distributions which indexed by a single real parameter.

Denoted by $\tau_{i,t}$ the number of times that arm $i$ has been played up to (but excluding) time $t$. Let $S_{i,1}, \ldots, S_{i,\tau_{i,t}}$ be the past observations got from arm $i$. Fix $\delta \in (0, 1/N)$.

Algorithm 1 has been proposed by them to achieve

$$E[T_i(n)] \leq \left(\frac{1}{I(\theta_i, \theta^*)} + o(1)\right) \ln n \quad (2.2)$$

where $o(1) \to 0$ as $n \to \infty$. $I(\theta_i, \theta^*) = \mathbb{E}_{\theta_i}[\log(f(y, \theta_i)/f(y, \theta^*))]$ is the Kullback-Leibler divergence between two distributions parameterized by $\theta_i$ and $\theta^*$.

Lai and Robbins also showed the lower bound of regret for classic multi-armed bandit problems indexed by a single real parameter. They have shown that for any suboptimal arm $i$,

$$E[T_i(n)] \geq \frac{\ln n}{I(\theta_i, \theta^*)}. \quad (2.3)$$

So, the regret of algorithm 1 is the best possible (order optimal).
Algorithm 1 Lai-Robbins Policy [52] for Classic MAB

1: // INITIALIZATION
2: for $p = 1$ to $N$ do
3: $n = p$;
4: Play each arm once;
5: end for
6: // MAIN LOOP
7: while 1 do
8: $n = n + 1$;
9: Among all the arms that have been played at least $(t - 1)\delta$ times, denoted by $l_t$ the leader, which is the arm with the largest point estimate

$$l_t = \arg \max_{i: \tau_{i,t} \geq (t-1)\delta} h_{\tau_{i,t}} (S_{i,1}, \ldots, S_{i,\tau_{i,t}}).$$

(2.4)

where $h_{\tau_{i,t}} (S_{i,1}, \ldots, S_{i,\tau_{i,t}})$ is the point estimate of arm $i$.
10: Let $r_t = t \mod N$ be the round robin candidate at time $t$. The player plays $l_t$ if $h_{\tau_{l,t}} (S_{l,1}, \ldots, S_{l,\tau_{l,t}}) > g_{t,\tau_{l,t}} (S_{r,1}, \ldots, S_{r,\tau_{r,t}})$, and the round-robin candidate $r_t$ otherwise. $g_{t,\tau_{l,t}} (S_{r,1}, \ldots, S_{r,\tau_{r,t}})$ is upper confidence bounds.
11: end while

For Gaussian, Bernoulli, and Poisson reward models, $h_k$ and $g_{t,k}$ satisfy 2.5 and 2.6 as follows:

$$h_k = \frac{1}{k} \sum_{j=1}^{k} S_{n,j},$$

(2.5)

$$g_{t,k} = \inf \{\lambda : \lambda \geq h_k \land I(h_k, \lambda) \geq \frac{\log(t-1)}{k} \}.$$  

(2.6)

2.2.2 UCB1 Policy

Our work in this dissertation is significantly influenced by the paper by Auer et al. [13], which considers arms with non-negative rewards that are i.i.d. over time with an arbitrary un-parameterized distribution that has the only restriction that it have a finite support.
Further, they provide a simple policy (referred to as UCB1, as shown in Algorithm 2), which achieves logarithmic regret uniformly over time as shown in Theorem 1, rather than only asymptotically.

**Algorithm 2 UCB1 Policy [13] for Classic MAB**

1: // INITIALIZATION
2: for $p = 1$ to $N$ do
3: \hspace{1em} $n = p$;
4: \hspace{1em} Play each arm once;
5: end for
6: // MAIN LOOP
7: while 1 do
8: \hspace{1em} $n = n + 1$;
9: \hspace{1em} Play an arm $i$ which solves the maximization problem
10: \hspace{2em} $i = \arg \max x_i + \sqrt{\frac{2 \ln n}{n_i}}$ \hspace{1em} (2.7)

where $\overline{x}_i$ is the average reward got on arm $i$; $n_i$ is the number of times arm $i$ has been played up to the current time slot.
10: end while

(2.7) in Algorithm 2 shows how the algorithm handles the tradeoff between exploration and exploitation in MABs. If an arm $i$ is not played often enough, $n_i$ is small and hence $\sqrt{\frac{2 \ln n}{n_i}}$ is relatively big and will dominate in (2.7). So the arms are not played often enough are more likely to be picked (exploration). On the other hand, if all the arms are played often enough, $\overline{x}_i$ dominates in (2.7), so the arm with highest observed mean is more likely to be played (exploitation).

**Theorem 1.** The expected regret under UCB1 policy is at most

$$\left[ 8 \sum_{k: \theta_k < \theta^*} \left( \frac{\ln n}{\Delta_k} \right) \right] + \left(1 + \frac{\phi^2}{3}\right)(\sum_{k: \theta_k < \theta^*} \Delta_k) \hspace{1em} (2.8)$$
where $\Delta_k = \theta^* - \theta_k$, $\theta_k = \sum_{i \in A_k} a_i \theta_i$.

Proof. See [13, Theorem 1].

The key idea of the proof presented by Auer et al. [13] is to find the upper bound expected number of times that each non-optimal arm is played and then sum over all arms to get the upper bound of regret. Auer et al. show that the probability that each non-optimal arm is played is equivalent to sum of probabilities of three events: (i) the observed mean of the optimal arm is below a certain distance of its expectation; (ii) the observed mean of some arm $i$ is above a certain distance of its; (iii) the expectation of the optimal arm is smaller than the expectation of arm $i$ plus a value which is a function of the number of times arm $i$ has been played. Then they bound the probabilities of the first two events based on the Chernoff-Hoeffding bound [70], as stated in Lemma 1, for the concentration of sample mean. They also show that the probability of the third event is zero when the number of times that arm is played is large enough.

**Lemma 1** (Chernoff-Hoeffding bound [70]). $X_1, \ldots, X_n$ are random variables with range $[0, 1]$, and $E[X_t|X_1, \ldots, X_{t-1}] = \mu$, $\forall 1 \leq t \leq n$. Denote $S_n = \sum X_i$. Then for all $a \geq 0$

\[
\begin{align*}
P\{S_n \geq n\mu + a\} &\leq e^{-2a^2/n} \\
P\{S_n \leq n\mu - a\} &\leq e^{-2a^2/n}
\end{align*}
\] (2.9)

Our work in this dissertation reshapes in innovative ways the UCB1 algorithm and its proof for more general and practical problem settings.
Chapter 3

Related Work

In this chapter, we give an overview of the previous research and literature that are relevant to our studies. We summarize below prior work, which has treated a) independent and temporally i.i.d. rewards, b) non-independent arms with temporally i.i.d, c) independent and rested Markovian state-based rewards, d) independent and restless Markovian state-based rewards, e) distributed online learning policies, or f) applications to communication networks.

3.1 Independent Arms with Temporally I.I.D. Rewards

Lai and Robbins [52] wrote one of the earliest papers on the classic non-Bayesian infinite horizon multi-armed bandit problem. Assuming $N$ independent arms, each generating rewards that are i.i.d. over time from a given family of distributions with an unknown real-valued parameter, they presented a general policy that provides expected regret that is $O(N \log n)$, i.e. linear in the number of arms and asymptotically logarithmic in $n$. 

They also show that this policy is order optimal in that no policy can do better than \(\Omega(N \log n)\). Anantharam et al. [11] extend this work to the case when \(M\) multiple plays are allowed. Agrawal et al. [4] further extend this to the case when there are multiple plays and also switching costs are taken into account.

Our study is influenced by the works by Agrawal [3] and Auer et al. [13]. The work by Agrawal [3] first presented easy to compute upper confidence bound (UCB) policies based on the sample-mean that also yield asymptotically logarithmic regret. Auer et al. [13] build on [3], and present variants of Agrawal’s policy including the so-called UCB1 policy, and prove bounds on the regret that are logarithmic uniformly over time (i.e., for any finite \(n\), not only asymptotically), so long as the arm rewards have a finite support. There are similarities in the proof techniques used in both these works [3, 13], which both use known results on large deviation upper bounds. In this thesis, we also make use of this approach, leveraging the same Chernoff-Hoeffding bound utilized in [13]. However, these works do not exploit potential dependencies between the arms\(^1\). As we show in this thesis, a direct application of the UCB1 policy therefore performs poorly for our problem formulation for the i.i.d. case.

Unlike Lai and Robbins [52], Agrawal [3], and Auer et al. [13], we consider in Chapter 4 a more general combinatorial version of the problem (with the i.i.d. formulation) that allows for the selection of a set of multiple variables simultaneously so long as they

---

\(^1\)Both the papers by Agrawal [3] and Auer et al. [13] indicate in the problem formulation that the rewards are independent across arms; however, since their proof technique bounds separately the expected time spent on each non-optimal arm, in fact the bounds the expected regret that they get through linearity of expectation applies even when the arm rewards are not independent. Nevertheless, as we indicate, the policies do not exploit any dependencies that may exist between the arm rewards.
satisfy a given arbitrary constraint. The constraint can specified explicitly in terms of sets of variables that are allowed to be picked together. There is no restriction on how these sets are constructed. They may correspond, for example, to some structural property such as all possible paths or matchings on a graph. While we credit the paper by Anantharam et al. [11] for being the first to consider multiple simultaneous plays, we note that our formulation in Chapter 4 is much more general than that work. Specifically, the work by Anantharam et al. [11] considers only a particular kind of constraint: it allows selection of all combinations of a fixed number of arms (i.e., in [11], exactly $M$ arms must be played at each time). For this reason, the algorithm presented in [11] cannot be directly used for the more general combinatorial problem in our formulation. For the same reason, the algorithm presented in [4] also cannot be used directly for our problem.

In our formulation in Chapter 4, we assume that the rewards from each individual variable in the selected combination are observed, and the total reward is a linearly weighted combination of these variables. Because we consider a fundamentally different and more general problem formulation, our proof strategy, while sharing structural similarities with [3, 13], has a non-trivial innovative component as well. In particular, in our setting, it turns out to be difficult to directly bound the number of actual plays of each weighted combination of variables, and therefore we create a carefully-defined virtual counter for each individual variable, and bound that instead.
3.2 Dependent Arms with Temporally I.I.D. Rewards

While these above key papers and many others have focused on independent arms, there have been some works treating dependencies between arms. The paper by Pandey et al. [69] divides arms into clusters of dependent arms (in our case there would be only one such cluster consisting of all the arms). Their model assumes that each arm provide only binary rewards, and in any case, they do not present any theoretical analysis on the expected regret. Ortner [68] proposes to use an additional arm color, to utilize the given similarity information of different arms to improve the upper bound of the regret. They assume that the difference of the mean rewards of any two arms with the same color is less than a predefined parameter $\delta$, which is known to the user. This is different from the linear reward model in this thesis.

Mersereau et al. [63] consider a bandit problem where the expected reward is defined as a linear function of a random variable, and the prior distribution is known. They show the upper bound of the regret is $O(\sqrt{n})$ and the lower bound of the regret is $\Omega(\sqrt{n})$. Rusmevichientong and Tsitsiklis [72] extend [63] to the setting where the reward from each arm is modeled as the sum of a linear combination of a set of unknown static random numbers and a zero-mean random variable that is i.i.d. over time and independent across arms. The upper bound of the regret is shown to be $O(N\sqrt{n})$ on the unit sphere and $O(N\sqrt{n} \log^{3/2} n)$ for a compact set, and the lower bound of regret is $\Omega(N\sqrt{n})$ for both cases. Although these papers also consider linear dependencies, a key difference, however is that in [63] and [72] it is assumed that only the total reward is observed at
each time, not the individual rewards. In this dissertation, we instead assume that all
the selected individual random variables are observed at each time (from which the total
reward can be inferred). Because of the more limited coarse-grained feedback, the prob-
lems tackled in [63] and [72] are indeed much more challenging, perhaps explaining
why they result in a higher regret bound order.

Both [12] and [27] consider linear reward models that are more general than ours, but
also under the assumption that only the total reward is observed at each time. Auer [12]
presents a randomized policy which requires storage and computation to grow linearly
in the number of arms. This algorithm is shown to achieve a regret upper bound of
\( O(\sqrt{N} \sqrt{n \log^2(n|F|)}) \). Dani et al. [27] develop another randomized policy for the case
of a compact set of arms, and show the regret is upper bounded by \( O(N \sqrt{n} \log^{3/2} n) \)
for sufficiently large \( n \) with high probability, and lower bounded by \( \Omega(N \sqrt{n}) \). They
also show that when the difference in costs (denoted as \( \Delta \)) between the optimal and next
to optimal decision among the extremal points is greater than zero, the regret is upper
bounded by \( O(\frac{N^2}{\Delta} \log^3 n) \) for sufficiently large \( n \) with high probability.

Liu and Zhao [61] have also investigated multi-armed bandit problems where the
dependencies between the arms take a linear form. Especially, their proposed policy can
handle a more general reward model by extending the results for distributions with finite
support to including any light-tailed distributions for a general compact action space.
They also consider the heavy-tailed distributions for the special case when the action
space is a polytope or finite. However their work has also focused on the case where
the feedback is coarse-grained in that only the total arm rewards are known and not for individual components.

Another paper that is related to our work is by Awerbuch and Kleinberg [15]. They consider the problem of shortest path routing in a non-stochastic, adversarial setting, in which only the total cost of the selected path is revealed at each time. For this problem, assuming the edge costs on the graph are chosen by an adaptive adversary that can view the past actions of the policy, they present a policy with regret scaling as $O(n^{\frac{2}{3}}(\log(n))^{\frac{1}{3}})$ over $n$ time steps. However, although as we discuss our formulation can also be applied to online shortest path routing, our work is different from [15] in that we consider a stochastic, non-adversarial setting and allow for observations of the individual edge costs of the selected path at each time.

### 3.3 Independent Arms with Rested Markovian Rewards

There has been relatively less work on multi-armed bandits with Markovian rewards. Anantharam *et al.* [10] wrote one of the earliest papers with such a setting. They proposed a policy to pick $M$ out of the $N$ arms each time slot and prove the lower bound and the upper bound on regret. However, the rewards in their work are assumed to be generated by rested Markov chains with transition probability matrices defined by a single parameter $\theta$ with identical state spaces. Also, the result for the upper bound is achieved only asymptotically.
For the case of single users and independent arms, a recent work by Tekin and Liu [74] has extended the results in [10] to the case with no requirement for a single parameter and identical state spaces across arms. They propose to use UCB1 from [13] for the multi-armed bandit problem with Markovian rewards and prove a logarithmic upper bound on the regret under some conditions on the Markov chain. We use elements of the proof from [74] in Chapter 5 of this thesis, which is however quite different in its combinatorial matching formulation (which allows for dependent arms).

3.4 Independent Arms with Restless Markovian Rewards

Restless arm bandits are so named because the arms evolve at each time, changing state even when they are not selected. Work on restless Markovian rewards with single users and independent arms can be found in [26, 56, 57, 66, 75]. In these papers there is no consideration of possible dependencies among arms, as in our work here.

Tekin and Liu [75] have proposed a RCA policy that achieves logarithmic single-action regret when certain knowledge about the system is known. We use elements of the policy and proof from [75] in Chapter 6 of this thesis, which is however quite different in its combinatorial matching formulation (which allows for dependent arms). Liu et al. [56, 57] adopted the same problem formulation as in [75], and proposed a policy named RUCB, achieving a logarithmic single-action regret over time when certain system knowledge is known. They also extend the RUCB policy to achieve a near-logarithmic regret asymptotically when no knowledge about the system is available.
In our recent work [25], we have also considered the same formulation, and proposed a CEE policy. When no information is available about the dynamics of the arms, CEE is the first algorithm to guarantee near-logarithmic regret uniformly over time. When some bounds corresponding to the stationary state distributions and the state-dependent rewards are known, we show that CEE can be easily modified to achieve logarithmic regret over time with less additional information compared with RCA and RUCB.

In [26], we have adopted a stronger definition of regret: the difference in expected reward compared to a model-aware genie. They develop a policy that yields regret of order arbitrarily close to logarithmic for certain classes of restless bandits with a finite-option structure, such as restless MAB with two states and identical probability transition matrices. In [66], we have developed a policy for a special case of two positively correlated restless multi-armed bandit problem and prove that it yields near-logarithmic regret with respect to any policy that achieves an expected discounted reward that is within $\epsilon$ of the optimal.

### 3.5 Decentralized Policy for Multi-Armed Bandits

While most of the prior work on MAB focused on the centralized policies, motivated by the problem of opportunistic access in cognitive radio networks, Liu and Zhao [59, 60], and Anandkumar et al. [8, 9] have both developed policies for the problem of $M$ distributed players operating $N$ independent arms. There are two problem formulations of interest when considering distributed MAB: a) the prioritized access problem, where it
is desired to prioritize a ranked set of users so that the \( K \)-th ranked user learns to access the arm with the \( K \)-th highest reward, and b) the \textit{fair access problem}, where the goal is to ensure that each user receives the same reward in expectation. For the prioritized access problem, Anandkumar \textit{et al.} [9] present a distributed policy that yields regret that is logarithmic in time, but requires prior knowledge of the arm reward means. For the fair access problem, they propose in [8, 9] a randomized distributed policy that is logarithmic with respect to time and scales as \( O(M^2N) \) with respect to the number of arms and users. Liu and Zhao [59, 60] also treat the fair access problem and present the TDFS policy which yields asymptotically logarithmic regret with respect to time and scales as \( O(M(\max\{M^2, (N - M)M\})) \) with respect to the number of arms and users.

In Chapter 8 of this thesis, we make significant new contributions to both problem formulations. For the prioritized access problem, we present a distributed learning policy DLP that results in a regret that is uniformly logarithmic in time and, unlike the prior work in [8, 9], does not require any prior knowledge about the arm reward means. For the fair access problem, we present another distributed learning policy DLF, which yields regret that is also uniformly logarithmic in time and that scales as \( O(M(N - M)) \) with respect to the number of users \( M \) and the number of arms \( N \). As it has been shown in [59, 60] that the lower-bound of regret for distributed policies also scales as \( \Omega(M(N - M)) \), this is not only a better scaling than the previous state of the art, it is, in fact, order-optimal.
Another recent work on decentralized MAB problem is by Kalathil et al. [44]. They have considered a different decentralized multi-armed bandit problem where the rewards on each arm can be distinct for each player. Decentralized policies for both i.i.d. and rested Markovian rewards are proposed, based on the use of a distributed bipartite matching algorithm.

3.6 Applications to Communication Networks

While multi-armed bandits are broadly useful for other fields such as medicine, finance, and industrial engineering, this thesis is particularly motivated and inspired by their applicability to communication networks. We briefly survey these applications below.

The application in communications where bandit formulations have found particular use in recent years is the context of dynamic spectrum access in cognitive radio networks [1, 5, 7–9, 35, 41, 43, 51, 55, 59, 60, 62, 64, 67, 75, 80]. They have been applied to other problems in wireless communications such as downlink scheduling in wireless networks [54], [65], MIMO systems [77], and channel measurements for wideband communication [42]. They have been used for opportunistic routing [19] and intrusion detection [79] in ad hoc networks, and for network selection in heterogeneous wireless multimedia networks [73]. In the context of sensor systems, they have been applied to control emissions for low probability intercept sensors [48], for scheduling communications to maximize sensor network lifetime [23], for joint coding and scheduling in sensor
networks [58], and for node discovery in mobile sensor networks [30]. Multi-armed bandit formulations have also been considered for path and wavelength selection in optical networks [45].

Given the ready applicability of MAB to a wide range of communication systems, as evidenced by these many papers, it is clear that progress in expanding the boundaries of knowledge on algorithms and performance beyond classical MAB will have significant impact on the design of efficient communication network protocols in unknown stochastic environments.
Chapter 4

Learning with I.I.D. Linear Rewards

4.1 Overview

In this chapter¹, we formulate the following combinatorial multi-armed bandit (MAB) problem: there are $N$ random variables with unknown mean that are each instantiated in an i.i.d. fashion over time. At each time multiple random variables can be selected, subject to an arbitrary constraint on weights associated with the selected variables. All of the selected individual random variables are observed at that time, and a linearly weighted combination of these selected variables is yielded as the reward. The goal is to find a policy that minimizes regret. This formulation is broadly applicable and useful for stochastic online versions of many interesting tasks in networks that can be formulated as tractable combinatorial optimization problems with linear objective functions, such as maximum weighted matching, shortest path, and minimum spanning tree computations.

¹This chapter is based on [35] and [36].
Prior work on multi-armed bandits with multiple plays cannot be applied to this formulation because of the general nature of the constraint. On the other hand, the mapping of all feasible combinations to arms allows for the use of prior work on MAB with single-play, but results in regret, storage, and computation growing exponentially in the number of unknown variables. We present new efficient policies for this problem, that are shown to achieve regret that grows logarithmically with time, and polynomially in the number of unknown variables. Furthermore, these policies only require storage that grows linearly in the number of unknown parameters. For problems where the underlying deterministic problem is tractable, these policies further require only polynomial computation. For computationally intractable problems, we also present results on a different notion of regret that is suitable when a polynomial-time approximation algorithm is used.

4.2 Problem Formulation

We consider a discrete time system with $N$ unknown random processes $X_i(n), 1 \leq i \leq N$, where time is indexed by $n$. We assume that $X_i(n)$ evolves as an i.i.d. random process over time, with the only restriction that its distribution have a finite support. Without loss of generality, we normalize $X_i(n) \in [0, 1]$. We do not require that $X_i(n)$ be independent across $i$. This random process is assumed to have a mean $\theta_i = E[X_i]$ that is unknown to the users. We denote the set of all these means as $\Theta = \{\theta_i\}$.

At each decision period $n$ (also referred to interchangeably as time slot), an $N$-dimensional action vector $a(n)$ is selected under a policy $\phi(n)$ from a finite set $\mathcal{F}$. We
assume $a_i(n) \geq 0$ for all $1 \leq i \leq N$. When a particular $a(n)$ is selected, only for those $i$ with $a_i(n) \neq 0$, the value of $X_i(n)$ is observed. We denote $A_{a(n)} = \{i : a_i(n) \neq 0, 1 \leq i \leq N\}$, the index set of all $a_i(n) \neq 0$ for an action $a$. The reward is defined as:

$$R_{a(n)}(n) = \sum_{i=1}^{N} a_i(n)X_i(n). \quad (4.1)$$

When a particular action $a(n)$ is selected, the random variables corresponding to non-zero components of $a(n)$ are revealed$^2$, i.e., the value of $X_i(n)$ is observed for all $i$ such that $a(n) \neq 0$.

We evaluate policies with respect to regret, which is defined as the difference between the expected reward that could be obtained by a genie that can pick an optimal action at each time, and that obtained by the given policy. Note that minimizing the regret is equivalent to maximizing the rewards. Regret can be expressed as:

$$\mathcal{R}_n^\phi(\Theta) = n\theta^* - \mathbb{E}^\phi[\sum_{t=1}^{n} R_{\phi(t)}(t)], \quad (4.2)$$

where $\theta^* = \max_{a \in \mathcal{A}} \sum_{i=1}^{N} a_i\theta_i$, the expected reward of an optimal action. For the rest of the chapter, we use * as the index indicating that a parameter is for an optimal action. If there is more than one optimal action exist, * refers to any one of them.

$^2$As noted in the related work, this is a key assumption in our work that differentiates it from other prior work on linear dependent-arm bandits [12], [27]. This is a very reasonable assumption in many cases, for instance, in the combinatorial network optimization applications we discuss in section 4.5, it corresponds to revealing weights on the set of edges selected at each time.
Intuitively, we would like the regret $R_n^\phi(\Theta)$ to be as small as possible. If it is sub-linear with respect to time $n$, the time-averaged regret will tend to zero and the maximum possible time-averaged reward can be achieved. Note that the number of actions $|\mathcal{F}|$ can be exponential in the number of unknown random variables $N$.

### 4.3 Policy Design

#### 4.3.1 A Naive Approach

A unique feature of our problem formulation is that the action selected at each time can be chosen such that the corresponding collection of individual variables satisfies an arbitrary structural constraint. For this reason, as we indicated in our related works discussion, prior work on MAB with fixed number of multiple plays, such as [11], or on linear reward models, such as [27], cannot be applied to this problem. One straightforward, relatively naive approach to solving the combinatorial multi-armed bandits problem that we defined is to treat each arm as an action, which allows us to use the UCB1 policy given by Auer et al. [13]. Using UCB1, each action is mapped into an arm, and the action that maximizes $\hat{Y}_k + \sqrt{\frac{2\ln n}{m_k}}$ will be selected at each time slot, where $\hat{Y}_k$ is the mean observed reward on action $k$, and $m_k$ is the number of times that action $k$ has been played. This approach essentially ignores the dependencies across the different actions, storing observed information about each action independently, and making decisions based on this information alone.
Note that UCB1 requires storage that is linear in the number of actions and yields regret growing linearly with the number of actions. In a case where the number of actions grow exponentially with the number of unknown variables, both of these are highly unsatisfactory.

Intuitively, UCB1 algorithm performs poorly on this problem because it ignores the underlying dependencies. This motivates us to propose a sophisticated policy which more efficiently stores observations from correlated actions and exploits the correlations to make better decisions.

### 4.3.2 A New Policy

Our proposed policy, which we refer to as “learning with linear rewards” (LLR), is shown in Algorithm 3.

Table 4.1 summarizes some notation we use in the description and analysis of our algorithm.

The key idea behind this algorithm is to store and use observations for each random variable, rather than for each action as a whole. Since the same random variable can be observed while operating different actions, this allows exploitation of information gained from the operation of one action to make decisions about a dependent action.

We use two $1 \times N$ vectors to store the information after we play an action at each time slot. One is $\hat{\theta}_i$ in which $\hat{\theta}_i$ is the average (sample mean) of all the observed values of $X_i$ up to the current time slot (obtained through potentially different sets of
Algorithm 3 Learning with Linear Rewards (LLR)

1: // INITIALIZATION
2: If $\max_a |A_a|$ is known, let $L = \max_a |A_a|$; else, $L = N$;
3: for $p = 1$ to $N$ do
4: $n = p$;
5: Play any action $a$ such that $p \in A_a$;
6: Update $(\hat{\theta}_i)_{1 \times N}, (m_i)_{1 \times N}$ accordingly;
7: end for
8: // MAIN LOOP
9: while 1 do
10: $n = n + 1$;
11: Play an action $a$ which solves the maximization problem
   
   
   $a = \arg \max_{a \in F} \sum_{i \in A_a} a_i \left( \hat{\theta}_i + \sqrt{\frac{(L + 1) \ln n}{m_i}} \right)$; 
   

   
   (4.3)

12: Update $(\hat{\theta}_i)_{1 \times N}, (m_i)_{1 \times N}$ accordingly;
13: end while

actions over time). The other one is $(m_i)_{1 \times N}$ in which $m_i$ is the number of times that $X_i$ has been observed up to the current time slot.

At each time slot $n$, after an action $a(n)$ is played, we get the observation of $X_i(n)$ for all $i \in A_{a(n)}$. Then $(\hat{\theta}_i)_{1 \times N}$ and $(m_i)_{1 \times N}$ (both initialized to 0 at time 0) are updated as follows:

$$
\hat{\theta}_i(n) = \begin{cases} 
\frac{\hat{\theta}_i(n-1)m_i(n-1)+X_i(n)}{m_i(n-1)+1}, & \text{if } i \in A_{a(n)} \\
\hat{\theta}_i(n-1), & \text{else}
\end{cases}
$$

(4.4)

$$
m_i(n) = \begin{cases} 
m_i(n-1) + 1, & \text{if } i \in A_{a(n)} \\
m_i(n-1), & \text{else}
\end{cases}
$$

(4.5)
\( N \): number of random variables.
\( a \): vectors of coefficients, defined on set \( F \).
\( A_a \): \( \{ i : a_i \neq 0, 1 \leq i \leq N \} \).
\( * \): index indicating that a parameter is for an optimal action.
\( m_i \): number of times that \( X_i \) has been observed up to the current time slot.
\( \hat{\theta}_i \): average (sample mean) of all the observed values of \( X_i \) up to the current time slot.
Note that \( E[\hat{\theta}_i(n)] = \theta_i \).
\( \hat{\theta}_{i,m_i} \): average (sample mean) of all the observed values of \( X_i \) when it is observed \( m_i \) times.
\( \Delta_a \): \( R^* - R_a \).
\( \Delta_{\text{min}} \): \( \min_{R_a < R^*} \Delta_a \).
\( \Delta_{\text{max}} \): \( \max_{R_a < R^*} \Delta_a \).
\( T_a(n) \): number of times action \( a \) has been played in the first \( n \) time slots.
\( a_{\text{max}} \): \( \max_{a \in F} \max_i a_i \).

Table 4.1: Notation.

Note that while we indicate the time index in the above updates for notational clarity, it is not necessary to store the matrices from previous time steps while running the algorithm.

LLR policy requires storage linear in \( N \). In section 4.4, we will present the analysis of the upper bound of regret, and show that it is polynomial in \( N \) and logarithmic in time. Note that the maximization problem (4.3) needs to be solved as the part of LLR policy. It is a deterministic linear optimal problem with a feasible set \( F \) and the computation time for an arbitrary \( F \) may not be polynomial in \( N \). As we show in Section 4.5, there exist many practically useful examples with polynomial computation time.
Traditionally, the regret of a policy for a multi-armed bandit problem is upper-bounded by analyzing the expected number of times that each non-optimal action is played, and the summing this expectation over all non-optimal actions. While such an approach will work to analyze the LLR policy too, it turns out that the upper-bound for regret consequently obtained is quite loose, being linear in the number of actions, which may grow faster than polynomials. Instead, we give here a tighter analysis of the LLR policy that provides an upper bound which is instead polynomial in $N$ and logarithmic in time. Like the regret analysis in [13], this upper-bound is valid for finite $n$.

**Theorem 2.** *The expected regret under the LLR policy is at most*

$$\left[ \frac{4a^2_{\text{max}}L^2(L+1)N \ln n}{(\Delta_{\text{min}})^2} + N + \frac{\pi^2}{3}LN \right] \Delta_{\text{max}}. \quad (4.6)$$

**Proof.** Denote $C_{t,m_i}$ as $\sqrt{\frac{(L+1)\ln t}{m_i}}$. We introduce $\tilde{T}_i(n)$ as a counter after the initialization period. It is updated in the following way:

At each time slot after the initialization period, one of the two cases must happen: (1) an optimal action is played; (2) a non-optimal action is played. In the first case, $(\tilde{T}_i(n))_{1 \times N}$ won’t be updated. When an non-optimal action $a(n)$ is picked at time $n$, there must be at least one $i \in A_n$ such that $i = \arg\min_{j \in A_n} m_j$. If there is only one such action, $\tilde{T}_i(n)$ is increased by 1. If there are multiple such actions, we arbitrarily pick one, say $i'$, and increment $\tilde{T}_{i'}$ by 1.
Each time when a non-optimal action is picked, exactly one element in \( (\tilde{T}_i(n))_{1 \times N} \) is incremented by 1. This implies that the total number that we have played the non-optimal actions is equal to the summation of all counters in \( (\tilde{T}_i(n))_{1 \times N} \), i.e.,
\[
\sum_{a: R_a < R^*} T_a(n) = \sum_{i=1}^{N} \tilde{T}_i(n)
\]
and hence
\[
E[\sum_{a: R_a < R^*} T_a(n)] = E[\sum_{i=1}^{N} \tilde{T}_i(n)].
\] (4.7)

Therefore, we have:
\[
\sum_{a: R_a < R^*} E[T_a(n)] = \sum_{i=1}^{N} E[\tilde{T}_i(n)].
\] (4.8)

Also note for \( \tilde{T}_i(n) \), the following inequality holds:
\[
\tilde{T}_i(n) \leq m_i(n), \forall 1 \leq i \leq N.
\] (4.9)

Denote by \( \bar{I}_i(n) \) the indicator function which is equal to 1 if \( \tilde{T}_i(n) \) is added by one at time \( n \). Let \( l \) be an arbitrary positive integer. Then:
\[
\bar{I}_i(n) = \sum_{t=N+1}^{n} 1\{\bar{I}_i(t) = 1\}
\]
\[
\leq l + \sum_{t=N+1}^{n} 1\{\bar{I}_i(t) = 1, \bar{I}_i(t-1) \geq l\}
\] (4.10)

where \( 1(x) \) is the indicator function defined to be 1 when the predicate \( x \) is true, and 0 when it is false. When \( \bar{I}_i(t) = 1 \), a non-optimal action \( a(t) \) has been picked for which
\[
m_i = \min_j \{m_j : \forall j \in A_{a(t)}\}.
\]
We denote this action as \( a(t) \) since at each time that \( \bar{I}_i(t) = 1 \), we could get different actions. Then,
\[
\tilde{T}_i(n) \leq l + \sum_{t=N+1}^{n} \mathbb{1}\left\{ \sum_{j \in A_{a^*}} a_j^* (\tilde{\theta}_{j,m_j(t-1)} + C_{t-1,m_j(t-1)}) \right\}
\leq \sum_{j \in A_{a(t)}} a_j(t) (\tilde{\theta}_{j,m_j(t-1)} + C_{t,m_j(t-1)}), \tilde{T}_i(t-1) \geq l \right\}
\leq l + \sum_{t=N}^{n} \mathbb{1}\left\{ \sum_{j \in A_{a^*}} a_j^* (\tilde{\theta}_{j,m_j(t)} + C_{t,m_j(t)}) \right\}
\leq \sum_{j \in A_{a(t+1)}} a_j(t+1) (\tilde{\theta}_{j,m_j(t)} + C_{t,m_j(t)}), \tilde{T}_i(t) \geq l \right\}.
\] (4.11)

Note that \( l \leq \tilde{T}_i(t) \) implies,

\[
l \leq \tilde{T}_i(t) \leq m_j(t), \forall j \in A_{a(t+1)}. \tag{4.12}
\]

So,

\[
\tilde{T}_i(n) \leq l + \sum_{t=N}^{n} \mathbb{1}\left\{ \min_{0 < m_{h_1}, \ldots, m_{h_{|A_{a^*}|}} \leq t} \sum_{j = 1}^{|A_{a^*}|} a_{h_j} (\tilde{\theta}_{h_j,m_{h_j}} + C_{t,m_{h_j}}) \right\}
\leq \max_{l \leq m_{p_1}, \ldots, m_{p_{|A_{a(t+1)}|}}} \left\{ \sum_{j = 1}^{t} \sum_{l = 1}^{t} a_{p_j} (t+1)(\tilde{\theta}_{p_j,m_{p_j}} + C_{t,m_{p_j}}) \right\}
\leq l + \sum_{t=1}^{\infty} \sum_{m_{h_1} = 1}^{t} \cdots \sum_{m_{h_{|A_{a^*}|}} = 1}^{t} \sum_{m_{p_1} = l}^{m_{p_{|A_{a(t+1)}|}} = l} \sum_{m_{p_1} = l}^{m_{p_{|A_{a(t+1)}|}} = l} \mathbb{1}\left\{ \sum_{j = 1}^{|A_{a*}|} a_{h_j} (\tilde{\theta}_{h_j,m_{h_j}} + C_{t,m_{h_j}}) \leq \sum_{j = 1}^{t} a_{p_j} (t+1)(\tilde{\theta}_{p_j,m_{p_j}} + C_{t,m_{p_j}}) \right\}
\]

where \( h_j (1 \leq j \leq |A_{a^*}|) \) represents the \( j \)-th element in \( A_{a^*} \) and \( p_j (1 \leq j \leq |A_{a(t+1)}|) \) represents the \( j \)-th element in \( A_{a(t+1)} \).
\[
\sum_{j=1}^{|A_a|} a^*_h \hat{\theta}_{h_j,m_{h_j}} + C_{t,m_{h_j}} \leq \sum_{j=1}^{|A_a^{(t+1)}|} a_{p_j} (t + 1)(\hat{\theta}_{p_j,m_{p_j}} + C_{t,m_{p_j}})
\]
means that at least one of the following must be true:

\[
\sum_{j=1}^{|A_a|} a^*_h \hat{\theta}_{h_j,m_{h_j}} \leq R^* - \sum_{j=1}^{|A_a|} a^*_h C_{t,m_{h_j}}, \tag{4.13}
\]

\[
\sum_{j=1}^{|A_a^{(t+1)}|} a_{p_j} (t + 1)\hat{\theta}_{p_j,m_{p_j}} \geq R_{a^{(t+1)}} + \sum_{j=1}^{|A_a^{(t+1)}|} a_{p_j} (t + 1)C_{t,m_{p_j}}, \tag{4.14}
\]

\[
R^* < R_{a^{(t+1)}} + 2 \sum_{j=1}^{|A_a^{(t+1)}|} a_{p_j} (t + 1)C_{t,m_{p_j}}. \tag{4.15}
\]

Now we find the upper bound for \(P\left\{ \sum_{j=1}^{|A_a|} a^*_h \hat{\theta}_{h_j,m_{h_j}} \leq R^* - \sum_{j=1}^{|A_a|} a^*_h C_{t,m_{h_j}} \right\}\). We have:

\[
P\left\{ \sum_{j=1}^{|A_a|} a^*_h \hat{\theta}_{h_j,m_{h_j}} \leq R^* - \sum_{j=1}^{|A_a|} a^*_h C_{t,m_{h_j}} \right\}
\]

\[
= P\left\{ \sum_{j=1}^{|A_a|} a^*_h \hat{\theta}_{h_j,m_{h_j}} \leq \sum_{j=1}^{|A_a|} a^*_h \theta_{h_j} - \sum_{j=1}^{|A_a|} a^*_h C_{t,m_{h_j}} \right\}
\]

\[
\leq P\{ \text{At least one of the following must hold:}
\]

\[
a^*_h \hat{\theta}_{h_1,m_{h_1}} \leq a^*_h \theta_{h_1} - a^*_h C_{t,m_{h_1}};
\]

\[
a^*_h \hat{\theta}_{h_2,m_{h_2}} \leq a^*_h \theta_{h_2} - a^*_h C_{t,m_{h_2}};
\]

\[
: \]

\[
a^*_h \hat{\theta}_{h_{|A_a|},m_{|A_a|}} \leq a^*_h \theta_{h_{|A_a|}} - a^*_h C_{t,m_{|A_a|}} \}
\]
\begin{equation}
\leq \sum_{j=1}^{|A_{a^*}|} P\{a_{h_j}^* \hat{\theta}_{h_j, m_{h_j}} \leq a_{h_j}^* \theta_{h_j} - a_{h_j}^* C_{t, m_{h_j}} \} \\
= \sum_{j=1}^{|A_{a^*}|} P\{\hat{\theta}_{h_j, m_{h_j}} \leq \theta_{h_j} - C_{t, m_{h_j}} \}.
\end{equation}

∀1 \leq j \leq |A_{a^*}|, applying the Chernoff-Hoeffding bound stated in Lemma 1, we could find the upper bound of each item in the above equation as,

\begin{equation}
P\{\hat{\theta}_{h_j, m_{h_j}} \leq \theta_{h_j} - C_{t, m_{h_j}} \} \\
= P\{m_{h_j} \hat{\theta}_{h_j, m_{h_j}} \leq m_{h_j} \theta_{h_j} - m_{h_j} C_{t, m_{h_j}} \} \\
\leq e^{-2 \frac{(m_{h_j})^2 \ln t}{m_{h_j}}} \\
= e^{-2(L+1) \ln t} \\
= t^{-2(L+1)}.
\end{equation}

Thus,

\begin{equation}
P\{ \sum_{j=1}^{|A_{a^*}|} a_{h_j}^* \hat{\theta}_{h_j, m_{h_j}} \leq R^* - \sum_{j=1}^{|A_{a^*}|} a_{h_j}^* C_{t, m_{h_j}} \} \\
\leq |A_{a^*}| t^{-2(L+1)}
\end{equation}

\begin{equation}
\leq Lt^{-2(L+1)}.
\end{equation}

Similarly, we can get the upper bound of the probability for inequality (4.14):

\begin{equation}
P\{ \sum_{j=1}^{|A_{a(t+1)}|} a_{p_j} (t + 1) \hat{\theta}_{p_j, m_{p_j}} \geq R_{a(t+1)} + \sum_{j=1}^{|A_{a(t+1)}|} a_{p_j} (t + 1) C_{t, m_{p_j}} \} \leq Lt^{-2(L+1)}.
\end{equation}
Note that for \( l \geq \left\lceil \frac{4(L+1) \ln n}{\left( \frac{\Delta a(t+1)}{L a_{\text{max}}} \right)^2} \right\rceil \),

\[
R^* - R_{a(t+1)} - 2 \sum_{j=1}^{\vert A_{a(t+1)} \vert} a_{pj}(t+1)C_{t,mpj} = R^* - R_{a(t+1)} - 2 \sum_{j=1}^{\vert A_{a(t+1)} \vert} a_{pj}(t+1) \sqrt{\frac{(L+1) \ln t}{mpj}} \\
\geq R^* - R_{a(t+1)} - L a_{\text{max}} \sqrt{\frac{4(L+1) \ln n}{l}} \\
\geq R^* - R_{a(t+1)} - L a_{\text{max}} \sqrt{\frac{4(L+1) \ln n}{4(L+1) \ln n}} \left( \frac{\Delta a(t+1)}{L a_{\text{max}}} \right)^2 \\
\geq R^* - R_{a(t+1)} - \Delta a(t+1) = 0.
\]

Equation (4.19) implies that condition (4.15) is false when \( l = \left\lceil \frac{4(L+1) \ln n}{\left( \frac{\Delta a(t+1)}{L a_{\text{max}}} \right)^2} \right\rceil \). If we let \( l = \left\lceil \frac{4(L+1) \ln n}{\left( \frac{\Delta a_{\text{min}}}{L a_{\text{max}}} \right)^2} \right\rceil \), then (4.15) is false for all \( a(t+1) \).

Therefore,

\[
E[\tilde{T}_i(n)] \leq \left[ \frac{4(L+1) \ln n}{\left( \frac{\Delta a_{\text{min}}}{L a_{\text{max}}} \right)^2} \right] + \sum_{t=1}^{\infty} \left( \sum_{m_{h_1}=1}^{t} \cdots \sum_{m_{h_{\left\lvert A^* \right\rceil}}}^{t} \sum_{m_{p_1}=l}^{t} \cdots \sum_{m_{p_{\left\lvert A(t) \right\rceil}}}^{t} 2L t^{-(L+1)} \right) \\
\leq \frac{4a_{\text{max}}^2 L^2 (L+1) \ln n}{(\Delta a_{\text{min}})^2} + L \sum_{t=1}^{\infty} 2t^{-2} \\
\leq \frac{4a_{\text{max}}^2 L^2 (L+1) \ln n}{(\Delta a_{\text{min}})^2} + 1 + \frac{\pi^2}{3} L.
\]

(4.20)
So under LLR policy, we have:

\[
R_n^\phi(\Theta) = R^* n - \mathbb{E}[\sum_{t=1}^{n} R_{\phi(t)}(t)]
\]

\[
= \sum_{a: R_a < R^*} \Delta_a \mathbb{E}[T_a(n)] 
\]

\[
\leq \Delta_{\text{max}} \sum_{a: R_a < R^*} \mathbb{E}[T_a(n)] 
\]

\[
= \Delta_{\text{max}} \sum_{i=1}^{N} \mathbb{E}[\tilde{T}_i(n)] 
\]

\[
\leq \left[ \sum_{i=1}^{N} \frac{4a_{\text{max}}^2 L^2 (L + 1) \ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3} LN \right] \Delta_{\text{max}} 
\]

\[
\leq \left[ \frac{4a_{\text{max}}^2 L^2 (L + 1) N \ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3} LN \right] \Delta_{\text{max}}. 
\]

\[ (4.21) \]

**Remark 1.** Note that when the set of action vectors consists of binary vectors with a single “1”, the problem formulation reduces to an multi-armed bandit problem with \( N \) independent actions. In this special case, the LLR algorithm is equivalent to UCB1 in [13]. Thus, our results generalize that prior work.

**Remark 2.** We have presented \( \mathcal{F} \) as a finite set in our problem formation. We note that the LLR policy we have described and its analysis actually also work with a more general formulation when \( \mathcal{F} \) is an infinite set with the following additional constraints: the maximization problem in (4.3) always has at least one solution; \( \Delta_{\min} \) exists; \( a_i \) is bounded. With the above constraints, Algorithm 3 will work the same and the conclusion and all the details of the proof of Theorem 2 can remain the same.
Remark 3. In fact, Theorem 2, also holds for certain kinds of non-i.i.d. random variables $X_i, 1 \leq i \leq N$ that satisfy the condition that $\mathbb{E}[X_i(t) | X_i(1), \ldots, X_i(t - 1)] = \theta_i, \forall 1 \leq i \leq N$. This is because the Chernoff-Hoeffding bound used in the regret analysis requires only this condition to hold$^3$.

### 4.5 Applications

We now describe some applications and extensions of the LLR policy for combinatorial network optimization in graphs where the edge weights are unknown random variables.

#### 4.5.1 Maximum Weighted Matching

Maximum Weighted Matching (MWM) problems are widely used in the many optimization problems in wireless networks such as the prior work in [16, 21]. Given any graph $G = (V, E)$, there is a weight associated with each edge and the objective is to maximize the sum weights of a matching among all the matchings in a given constraint set, i.e., the general formulation for MWM problem is

$$
\max \quad R_n^{MWM} = \sum_{i=1}^{[E]} a_i W_i \\
\text{s.t.} \quad a \text{ is a matching}
$$

(4.22)

where $W_i$ is the weight associated with each edge $i$.

$^3$This does not, however, include Markov chains for which we have obtained some weaker regret results in Chapter 5 and Chapter 6.
In many practical applications, the weights are unknown random variables and we need to learn by selecting different matchings over time. This kind of problem fits the general framework of our proposed policy regarding the reward as the sum weight and a matching as an action. Our proposed LLR policy is a solution with linear storage, and the regret polynomial in the number of edges, and logarithmic in time.

Since there are various algorithms to solve the different variations in the maximum weighted matching problems, such as the Hungarian algorithm for the maximum weighted bipartite matching [50], Edmonds’s matching algorithm [31] for a general maximum matching. In these cases, the computation time is also polynomial.

Here we present a general problem of multiuser channel allocations in cognitive radio network. There are $M$ secondary users and $Q$ orthogonal channels. Each secondary user requires a single channel for operation that does not conflict with the channels assigned to the other users. Due to geographic dispersion, each secondary user can potentially see different primary user occupancy behavior on each channel. Time is divided into discrete decision rounds. The throughput obtainable from spectrum opportunities on each user-channel combination over a decision period is denoted as $S_{i,j}$ and modeled as an arbitrarily-distributed random variable with bounded support but unknown mean, i.i.d. over time. This random process is assumed to have a mean $\theta_{i,j}$ that is unknown to the users. The objective is to search for an allocation of channels for all users that maximizes the expected sum throughput.
Assuming an interference model whereby at most one secondary user can derive benefit from any channel, if the number of channels is greater than the number of users, an optimal channel allocation employs a one-to-one matching of users to channels, such that the expected sum-throughput is maximized.

Figure 4.1 illustrates a simple scenario. There are two secondary users (i.e., links) S1 and S2, that are each assumed to be in interference range of each other. S1 is proximate to primary user P1 who is operating on channel 1. S2 is proximate to primary user P2 who is operating on channel 2. The matrix shows the corresponding $\Theta$, i.e., the throughput each secondary user could derive from being on the corresponding channel. In this simple example, the optimal matching is for secondary user 1 to be allocated channel 2 and user 2 to be allocated channel 1. Note, however, that, in our formulation, the users are not a priori aware of the matrix of mean values, and therefore must follow a sequential learning policy.

Note that this problem can be formulated as a multi-armed bandits with linear regret, in which each action corresponds to a matching of the users to channels, and the reward corresponds to the sum-throughput. In this channel allocation problem, there is $M \times Q$ unknown random variables, and the number of actions are $P(Q, M)$, which can grow
exponentially in the number of unknown random variables. Following the convention, instead of denoting the variables as a vector, we refer it as a $M$ by $Q$ matrix. So the reward as each time slot by choosing a permutation $a$ is expressed as:

$$R_a = \sum_{i=1}^{M} \sum_{j=1}^{Q} a_{i,j} S_{i,j}$$

(4.23)

where $a \in \mathcal{F}$, $\mathcal{F}$ is a set with all permutations, which is defined as:

$$\mathcal{F} = \{a : a_{i,j} \in \{0, 1\}, \forall i, j \land \sum_{i=1}^{Q} a_{i,j} = 1 \land \sum_{j=1}^{Q} a_{i,j} = 1\}.$$  

(4.24)

We use two $M$ by $Q$ matrices to store the information after we play an action at each time slot. One is $(\hat{\theta}_{i,j})_{M \times Q}$ in which $\hat{\theta}_{i,j}$ is the average (sample mean) of all the observed values of channel $j$ by user $i$ up to the current time slot (obtained through potentially different sets of actions over time). The other one is $(m_{i,j})_{M \times Q}$ in which $m_{i,j}$ is the number of times that channel $j$ has been observed by user $i$ up to the current time slot.

Applying Algorithm 3, we get a linear storage policy for which $(\hat{\theta}_{i,j})_{M \times Q}$ and $(m_{i,j})_{M \times Q}$ are stored and updated at each time slot. The regret is polynomial in the number of users and channels, and logarithmic in time. Also, the computation time for the policy is also polynomial since (4.3) in Algorithm 3 now becomes the following deterministic maximum weighted bipartite matching problem:

$$\arg \max_{a \in \mathcal{F}} \sum_{(i,j) \in A_a} \left( \hat{\theta}_{i,j} + \sqrt{\frac{(L + 1) \ln n}{m_{i,j}}} \right)$$

(4.25)
on the bipartite graph of users and channels with edge weights \( \hat{\theta}_{i,j} + \sqrt{\frac{(L+1)\ln n}{m_{i,j}}} \).

It could be solved with polynomial computation time (e.g., using the Hungarian algorithm [50]). Note that \( L = \max_a |A_a| = \min\{M, Q\} \) for this problem, which is less than \( M \times Q \) so that the bound of regret is tighter. The regret is \( O(\min\{M, Q\}^3MQ\log n) \) following Theorem 2.

### 4.5.2 Shortest Path

Shortest Path (SP) problem is another example where the underlying deterministic optimization can be done with polynomial computation time. If the given directed graph is denoted as \( G = (V, E) \) with the source node \( s \) and the destination node \( d \), and the cost (e.g., the transmission delay) associated with edge \((i, j)\) is denoted as \( D_{i,j} \geq 0 \), the objective is find the path from \( s \) to \( d \) with the minimum sum cost, i.e.,

\[
\min C^{SP}_a = \sum_{(i,j) \in E} a_{i,j} D_{i,j} \tag{4.26}
\]

\[
s.t. \quad a_{i,j} \in \{0, 1\}, \forall (i, j) \in E \tag{4.27}
\]

\[
\forall i, \sum_j a_{i,j} - \sum_j a_{j,i} = \begin{cases} 
1 & : \ i = s \\
-1 & : \ i = t \\
0 & : \text{otherwise}
\end{cases} \tag{4.28}
\]

where equation (4.27) and (4.28) defines a feasible set \( \mathcal{F} \), such that \( \mathcal{F} \) is the set of all possible paths from \( s \) to \( d \). When \((D_{ij})\) are random variables with bounded support but
unknown mean, i.i.d. over time, an dynamic learning policy is needed for this multi-armed bandit formulation.

Note that corresponding to the LLR policy with the objective to maximize the rewards, a direct variation of it is to find the minimum linear cost defined on finite constraint set $\mathcal{F}$, by changing the maximization problem into a minimization problem. For clarity, this straightforward modification of LLR is shown below in Algorithm 4, which we refer to as Learning with Linear Costs (LLC).

\begin{algorithm}
\caption{Learning with Linear Cost (LLC)}
\begin{algorithmic}[1]
\STATE {// \textsc{Initialization part is same as in Algorithm 3}}
\STATE {// \textsc{Main loop}}
\WHILE {1}
\STATE $n = n + 1$;
\STATE Play an action $a$ which solves the minimization problem
\STATE \hspace{1cm} $a = \arg\min_{a \in \mathcal{F}} \sum_{i \in A_a} a_i \left( \hat{\theta}_i - \sqrt{\frac{(L + 1) \ln n}{m_i}} \right)$; \hspace{1cm} (4.29)
\STATE Update $(\hat{\theta}_i)_{1 \times N}$, $(m_i)_{1 \times N}$ accordingly;
\ENDWHILE
\end{algorithmic}
\end{algorithm}

LLC (Algorithm 4) is a policy for a general multi-armed bandit problem with linear cost defined on any constraint set. It is directly derived from the LLR policy (Algorithm 3), so Theorem 2 also holds for LLC, where the regret is defined as:

$$R_n^\phi(\Theta) = E_n^\phi \left[ \sum_{t=1}^{n} C_{\phi(t)}(t) \right] - nC^*$$ \hspace{1cm} (4.30)

where $C^*$ represents the minimum cost, which is cost of the optimal action.
Using the LLC policy, we map each path between \( s \) and \( t \) as an action. The number of unknown variables are \(|E|\), while the number of actions could grow exponentially in the worst case. Since there exist polynomial computation time algorithms such as Dijkstra’s algorithm [29] and Bellman-Ford algorithm [18, 32] for the shortest path problem, we could apply these algorithms to solve (4.29) with edge cost \( \hat{\theta}_i = \sqrt{\frac{(L+1) \ln n}{m_i}} \). LLC is thus an efficient policy to solve the multi-armed bandit formulation of the shortest path problem with linear storage, polynomial computation time. Note that \( L = \max_a |\mathcal{A}_a| = |E| \). Regret is \( O(|E|^4 \ln n) \).

Another related problem is the Shortest Path Tree (SPT), where problem formulation is similar, and the objective is to find a subgraph of the given graph with the minimum total cost between a selected root \( s \) node and all other nodes. It is expressed as [17, 47]:

\[
\min_a C_{\alpha}^{\text{SPT}} = \sum_{(i,j) \in E} a_{i,j} D_{i,j} \tag{4.31}
\]

s.t. \( a_{i,j} \in \{0, 1\}, \forall (i, j) \in E \) \tag{4.32}

\[
\sum_{(j,i) \in \mathcal{B}_S(i)} a_{j,i} - \sum_{(i,j) \in \mathcal{F}_S(i)} a_{i,j} = \begin{cases} 
-n + 1 & : \ i = s \\
1 & : \ i \in V / \{s\} 
\end{cases} \tag{4.33}
\]

where \( \mathcal{B}_S(i) = \{(u, v) \in E : v = i\} \), \( \mathcal{F}_S(i) = \{(u, v) \in E : u = i\} \). (4.32) and (4.33) defines the constraint set \( \mathcal{F} \). We can also use the polynomial computation time algorithms such as Dijkstra’s algorithm and Bellman-Ford algorithm to solve (4.29) for the LLC policy.
### 4.5.3 Minimum Spanning Tree

Minimum Spanning Tree (MST) is another combinatorial optimization with polynomial computation time algorithms, such as Prim’s algorithm [71] and Kruskal’s algorithm [49]. The objective for the MST problem can be simply presented as

$$\min_{a \in \mathcal{F}} C^a_{\text{MST}} = \sum_{(i,j) \in E} a_{i,j} D_{i,j}$$  \hspace{1cm} (4.34)$$

where $\mathcal{F}$ is the set of all spanning trees in the graph.

With the LLC policy, each spanning tree is treated as an action, and $L = |E|$. Regret bound also grows as $O(|E|^4 \log n)$.

To summarize, we show in Table 4.2 a side by side comparison for the bipartite matching, shortest paths and spanning tree problems. For the matching problem, the graph is already restricted to bipartite graphs. The problem of counting the number of

<table>
<thead>
<tr>
<th></th>
<th>Naive Policy</th>
<th>LLR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Weighted</td>
<td>$O(</td>
<td>F</td>
</tr>
<tr>
<td>Matching</td>
<td>where $</td>
<td>F</td>
</tr>
<tr>
<td>Shortest Path (Complete Graph)</td>
<td>$O(</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>where $</td>
<td>F</td>
</tr>
<tr>
<td>Minimum Spanning Tree (Complete Graph)</td>
<td>$O(</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>where $</td>
<td>F</td>
</tr>
</tbody>
</table>

Table 4.2: Comparison of regret bounds.
paths on a graph is known to be $\#-P$ complete, so there is no known simple formula for a general setting. Similarly, we are not aware of any formulas for counting the number of spanning trees on a general graph. For this reason, for the latter two problems, we present comparative analytical bounds for the special case of the complete graph, where a closed form expression for number of paths can be readily obtained, and Cayley’s formula can be used for the number of spanning trees [22].

4.6 Examples and Simulation Results

![Figure 4.2: Simulation results of a system with 7 orthogonal channels and 4 users.](image)

We present in these section the numerical simulation results with the example of multiuser channel allocations in cognitive radio network.

Fig 4.2 shows the simulation results of using LLR policy compared with the naive policy in 4.3.1. We assume that the system consists of $Q = 7$ orthogonal channels in and $M = 4$ secondary users. The throughput $\{S_{i,j}(t)\}_{t \geq 1}$ for the user-channel combination
Figure 4.3: Simulation results of a system with 9 orthogonal channels and 5 users.

is an i.i.d. Bernoulli process with mean \( \theta_{i,j} \) (where \( \theta_{i,j} \) is unknown to the players) shown as below:

\[
(\theta_{i,j}) = \begin{pmatrix}
0.3 & 0.5 & 0.9 & 0.7 & 0.8 & 0.9 & 0.6 \\
0.2 & 0.2 & 0.3 & 0.4 & 0.5 & 0.4 & 0.5 \\
0.8 & 0.6 & 0.5 & 0.4 & 0.7 & 0.2 & 0.8 \\
0.9 & 0.2 & 0.2 & 0.8 & 0.3 & 0.9 & 0.6
\end{pmatrix}
\] (4.35)

where the components in the box are in the optimal action. Note that \( P(7, 4) = 840 \) while \( 7 \times 4 = 28 \), so the storage used for the naive approach is 30 times more than the LLR policy. Fig 4.2 shows the regret (normalized with respect to the logarithm of time) over time for the naive policy and the LLR policy. We can see that under both policies the regret grows logarithmically in time. But the regret for the naive policy is a lot higher than that of the LLR policy.

Fig 4.3 is another example of the case when \( Q = 9 \) and \( M = 5 \). The throughput is also assumed to be an i.i.d. Bernoulli process, with the following mean:
For this example, $P(9, 5) = 15120$, which is much higher than $9 \times 5 = 45$ (about 336 times higher), so the storage used by the naive policy grows much faster than the LLR policy. Comparing with the regrets shown in Table 4.3 for both examples when $t = 2 \times 10^6$, we can see that the regret also grows much faster for the naive policy.

<table>
<thead>
<tr>
<th></th>
<th>Naive Policy</th>
<th>LLR</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 channels, 4 users</td>
<td>2443.6</td>
<td>163.6</td>
</tr>
<tr>
<td>9 channels, 5 users</td>
<td>24892.6</td>
<td>345.2</td>
</tr>
</tbody>
</table>

Table 4.3: Regret when $t = 2 \times 10^6$.

### 4.7 K Simultaneous Actions

The reward-maximizing LLR policy presented in Algorithm 3 and the corresponding cost-minimizing LLC policy presented in Algorithm 4 can also be extended to the setting where $K$ actions are played at each time slot. The goal is to maximize the total rewards (or minimize the total costs) obtained by these $K$ actions. For brevity, we only present
the policy for the reward-maximization problem; the extension to cost-minimization is straightforward. The modified LLR-K policy for picking the $K$ best actions are shown in Algorithm 5.

**Algorithm 5 Learning with Linear Rewards while selecting $K$ actions (LLR-K)**

1: // **INITIALIZATION PART IS SAME AS IN ALGORITHM 3**
2: // **MAIN LOOP**
3: while 1 do
4: $n = n + 1$;
5: Play actions $\{a\}_K \in \mathcal{F}$ with $K$ largest values in (4.37)
6: $\sum_{i \in A_n} a_i \left( \hat{\theta}_i + \sqrt{\frac{(L + 1) \ln n}{m_i}} \right)$; (4.37)
7: Update $(\hat{\theta}_i)_{1 \times N}, (m_i)_{1 \times N}$ for all actions accordingly;
8: end while

**Theorem 3.** The expected regret under the LLR-K policy with $K$ actions selection is at most

$$
\left[\frac{4a_{max}^2 L^2 (L + 1) N \ln n}{(\Delta_{min})^2} + N + \frac{\pi^2}{3} LK^2 L N\right] \Delta_{max}.
$$

(4.38)

**Proof.** The proof is similar to the proof of Theorem 2, but now we have a set of $K$ actions with $K$ largest expected rewards as the optimal actions. We denote this set as $\mathfrak{A}^* = \{a^{*, k}, 1 \leq k \leq K\}$ where $a^{*, k}$ is the action with $k$-th largest expected reward. As in the proof of Theorem 2, we define $\tilde{T}_i(n)$ as a counter when a non-optimal action is played in the same way. Equation (4.43), (4.9), (4.10) and (4.12) still hold.

Note that each time when $\tilde{T}_i(t) = 1$, there exists some action such that a non-optimal action is picked for which $m_i$ is the minimum in this action. We denote this action as $a(t)$. Note that $a(t)$ means there exists $m, 1 \leq m \leq K$, such that the following holds:
\[
\tilde{T}_i(n) \leq l + \sum_{t=N}^{n} 1 \{ \sum_{j \in A_{a^*,m}} a_j^*(\hat{\theta}_{j,m_j(t)} + C_{t,m_j(t)}) \} \\
\leq \sum_{j \in A_{a(t)}} a_j(t)(\hat{\theta}_{j,m_j(t)} + C_{t,m_j(t)}), \tilde{T}_i(t) \geq l}. \]

(4.39)

Since at each time \(K\) actions are played, so at time \(t\), a random variable could be observed up to \(Kt\) times. Then (4.13) should be modified as:

\[
\tilde{T}_i(n) \leq l + \sum_{t=1}^{\infty} \sum_{m_{h_1}=1}^{Kt} \cdots \sum_{m_{h_{|A^*,m|}}=1}^{Kt} \sum_{m_{p_1}=1}^{Kt} \cdots \sum_{m_{p_{|A_{a(t)}|=1}}}^{Kt} 1 \{ \sum_{j=1}^{|A^*,m|} a_{h_j}^*(\hat{\theta}_{h_j,m_{h_j}} + C_{t,m_{h_j}}) \} \leq \sum_{j=1}^{|A_{a(t)}|} a_{p_j}(t)(\hat{\theta}_{p_j,m_{p_j}} + C_{t,m_{p_j}}) \}.
\]

(4.40)

Equation (4.13) to (4.19) are similar by substituting \(a^*\) with \(a^*_{m=1}^1\). So, we have:

\[
\mathbb{E}[\tilde{T}_i(n)] \leq \left[ \frac{4(L+1)\ln n}{\Delta_{\min}} \right]^2 + \sum_{t=1}^{\infty} \left( \sum_{m_{h_1}=1}^{Kt} \cdots \sum_{m_{h_{|A^*,m|}}=1}^{Kt} \sum_{m_{p_1}=1}^{Kt} \cdots \sum_{m_{p_{|A_{a(t)}|=1}}}^{Kt} 2Lt^{-2(L+1)} \right) \\
\leq \frac{4a^2_{\max} L^2(L+1) \ln n}{(\Delta_{\min})^2} + 1 + \frac{\pi^2}{3}LK^{2L}. 
\]

Hence, we get the upper bound for the regret as:

\[
\mathcal{R}_n^{\phi}(\Theta) \leq \left[ \frac{4a^2_{\max} L^2(L+1)N \ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3}LK^{2L}N \right] \Delta_{\max}. 
\]
4.8 LLR with Approximation Algorithm

One interesting question arises in the context of NP-hard combinatorial optimization problems, where even the deterministic version of the problem cannot be solved in polynomial time with known algorithms. In such cases, if only an approximation algorithm with some known approximation guarantee is available, what can be said about the regret bound?

For such settings, let us consider that a factor-\( \beta \) approximation algorithm (i.e., which for a maximization problem yields a solution that have reward more than \( \frac{OPT}{\beta} \)) is used to solve the maximization step in (4.3) in our LLR algorithm 3. Accordingly, we define an \( \beta \)-approximate action to be an action whose expected reward is within a factor \( \beta \) of that of the optimal action, and all other actions as non-\( \beta \)-approximate. Now we define \( \beta \)-approximation regret as follows:

\[
\mathcal{R}_{\phi,n}^{\beta,\phi}(\Theta) = E[\text{total number of times non-}\beta\text{-approximate actions are played by strategy } \phi \text{ in } n \text{ time slots}]
\]

\[
= E\left[ \sum_{a:a \text{ is not a } \beta\text{-approximate action}} m_a(n) \right] \quad (4.41)
\]

where \( m_a(n) \) is the total number of time that \( a \) has been played up to time \( n \). We define \( \Delta_{\min}^{\beta} \) as the minimum distance between an \( \beta \)-approximate action and a non-\( \beta \)-approximate action. We assume \( \Delta_{\min}^{\beta} > 0 \).

We have the following theorem regarding LLR with a \( \beta \)-approximation algorithm.

61
Theorem 4. The $\beta$-approximation regret under the LLR policy with a $\beta$-approximation algorithm is at most

$$\frac{4a_{\text{max}}^2 L^2 (L+1) N \ln n}{(\Delta_{\text{min}}^\beta)^2} + N + \frac{\pi^2}{3} LN$$  \hspace{1cm} (4.42)$$

Proof. We modify the proof of Theorem 2 to show Theorem 4. We replace “optimal action” to “$\beta$-approximate action”, and “non-optimal action” to “non-$\beta$-approximate action” every where shown in the proof of Theorem 2, and we still define a virtual counter $(\tilde{T}_{i}(n))_{1 \times N}$ in a similar way. We still use * to refer to an optimal action. So (4.43) becomes,

$$\sum_{a: a \text{ is a non-$\beta$-approximate action}} \mathbb{E}[T_a(n)] = \sum_{i=1}^{N} \mathbb{E}[\tilde{T}_{i}(n)].$$ \hspace{1cm} (4.43)$$

Now we note that for LLR with a $\beta$-approximation algorithm, when $\tilde{T}_{i}(t) = 1$, a non-$\beta$-approximate action $a(t)$ has been picked for which $m_i = \min_j \{m_j : \forall j \in A_{a(t)}\}$. Define $s_{\text{max}}(t)$ is the optimal solution for (4.3) in Algorithm 3. Then, we have

$$\sum_{j \in A_{a(t)}} a_j(t) (\hat{\theta}_{j,m_j(t-1)} + C_{t-1,m_j(t-1)}) \geq \frac{1}{\beta} s_{\text{max}}(t) \geq \frac{1}{\beta} \sum_{j \in A_{a^*}} a^*_j (\hat{\theta}_{j,m_j(t-1)} + C_{t-1,m_j(t-1)})$$ \hspace{1cm} (4.44)$$

So,
\[
\tilde{T}_i(n) \leq l + \sum_{t=N+1}^{n} \mathbf{1} \left\{ \frac{1}{\beta} \sum_{j \in A^*_n} a^*_j (\hat{\theta}_{j,m_j(t-1)} + C_{t-1,m_j(t-1)}) \right\} \\
\leq \sum_{j \in A_{n(t)}} \mathbf{1} \left\{ \frac{1}{\beta} \sum_{j \in A^*_n} a^*_j (\hat{\theta}_{j,m_j(t-1)} + C_{t-1,m_j(t-1)}) \right\}
\]

(4.45)

With a similar analysis, as in (4.11) to (4.13), we have

\[
\tilde{T}_i(n) \leq l + \sum_{t=1}^{\infty} \sum_{m_{h_{A^*_n}}} \sum_{m_{p_{|A_{n(t+1)}|}}} \sum_{m_{h_{|A^*_n|}}} \sum_{m_{p_{|A_{n(t+1)}|}}} \mathbf{1} \left\{ \frac{1}{\beta} \sum_{j \in |A_{n^t}|} a^*_h (\hat{\theta}_{h_j,m_{h_j}} + C_{t,m_{h_j}}) \right\} \leq \sum_{j=1}^{A_{a*(t+1)}} \sum_{j=1}^{A_{a(t+1)}} a_p (t+1) (\hat{\theta}_{p_j,m_{p_j}} + C_{t,m_{p_j}})
\]

(4.46) and (4.47) are equivalent to (4.13) and (4.14), and we note that for (4.48),

\[
\mathbf{1} \left\{ \frac{1}{\beta} \sum_{j \in |A_{n^t}|} a^*_h (\hat{\theta}_{h_j,m_{h_j}} + C_{t,m_{h_j}}) \right\} \leq \frac{1}{\beta} R^* - \frac{1}{\beta} \sum_{j=1}^{A_{a*(t+1)}} a^*_p (t+1) (\hat{\theta}_{p_j,m_{p_j}} + C_{t,m_{p_j}})
\]

implies that at least one of the following must be true:

\[
\frac{1}{\beta} \sum_{j \in |A_{n^t}|} a^*_h (\hat{\theta}_{h_j,m_{h_j}} + C_{t,m_{h_j}}) \leq \frac{1}{\beta} R^* - \frac{1}{\beta} \sum_{j=1}^{A_{a*(t+1)}} a^*_p (t+1) (\hat{\theta}_{p_j,m_{p_j}} + C_{t,m_{p_j}})
\]

(4.46)

\[
\sum_{j=1}^{A_{a(t+1)}} a_p (t+1) (\hat{\theta}_{p_j,m_{p_j}} + C_{t,m_{p_j}}) \geq R_{a(t+1)} + \sum_{j=1}^{A_{a(t+1)}} a_p (t+1) (\hat{\theta}_{p_j,m_{p_j}} + C_{t,m_{p_j}})
\]

(4.47)

\[
\frac{1}{\beta} R^* < R_{a(t+1)} + 2 \sum_{j=1}^{A_{a(t+1)}} a_p (t+1) (\hat{\theta}_{p_j,m_{p_j}} + C_{t,m_{p_j}})
\]

(4.48)

when \( l \geq \left\lfloor \frac{\beta (L+1) \ln n}{\Delta_{\min}^2 \Delta_{\max}^2} \right\rfloor \),
\[
\frac{1}{\beta} R^* - R_{a(t+1)} - 2 \sum_{j=1}^{\lfloor A_{a(t+1)} \rfloor} a_{p_j}(t+1)C_{t,m_{p_j}} \geq \frac{1}{\beta} R^* - R_{a(t+1)} - \Delta_{\min}^\beta = 0. \tag{4.49}
\]

Therefore, (4.41) still holds, and we have the upper bound for \(\beta\)-approximation regret as,

\[
\mathcal{R}_{n,\beta,LLR}(\Theta) \leq \frac{4a_{\max}^2L^2(L+1)N \ln n}{\left(\Delta_{\min}^\beta\right)^2} + N + \frac{\pi^2}{3}LN.
\]

\(\square\)

### 4.9 Summary

In this chapter, we have considered multi-armed bandit problems in which at each time an arbitrarily constrained set of random variables are selected, the selected variables are revealed, and a total reward that is a linear function of the selected variables is yielded. For such problems, existing single-play MAB policies such as the well-known UCB1 [13] can be utilized, but have poor performance in terms of storage, computation, and regret. The LLR and LLR-K policies we have presented are smarter in that they store and make decisions at each time based on the stochastic observations of the underlying unknown-mean random variables alone; they require only linear storage and result in a regret that is bounded by a polynomial function of the number of unknown-mean random variables. If the deterministic version of the corresponding combinatorial optimization
problem can be solved in polynomial time, our policy will also require only polynomial
computation per step. We have shown a number of problems in the context of networks
where this formulation would be useful, including maximum-weight matching, shortest
path and spanning tree computations. For the case where the deterministic version is
NP-hard, one has often at hand a polynomial-time approximation algorithm. In section
4.8, we show that under a suitably relaxed definition of regret, the LLR algorithm can
also employ such an approximation to give provable performance.
Chapter 5

Learning with Rested Markovian Rewards

5.1 Overview

We consider ¹ a combinatorial generalization of the classical multi-armed bandit problem that is defined as follows. There is a given bipartite graph of $M$ users and $N \geq M$ resources. For each user-resource pair $(i, j)$, there is an associated state that evolves as an aperiodic irreducible finite-state Markov chain with unknown parameters, with transitions occurring each time the particular user $i$ is allocated resource $j$. The user $i$ receives a reward that depends on the corresponding state each time it is allocated the resource $j$. The system objective is to learn the best matching of users to resources so that the long-term sum of the rewards received by all users is maximized. This corresponds to minimizing (single-action) regret, defined here as the gap between the expected total reward that can be obtained by the best-possible static matching and the expected total reward that can be achieved by a given algorithm. We present a polynomial-storage and

¹This chapter is based in part on [37].
polynomial-complexity-per-step matching-learning algorithm for this problem. We show
that this algorithm can achieve a regret that is uniformly arbitrarily close to logarithmic
in time and polynomial in the number of users and resources.

5.2 Problem Formulation

We consider a bipartite graph with \( M \) users and \( N \geq M \) resources predefined by some
application. Time is slotted and is indexed by \( n \). At each decision period (also referred
to interchangeably as time slot), each of the \( M \) users is assigned a resource with some
policy.

For each user-resource pair \((i, j)\), there is an associated state that evolves as an aper-
iodic irreducible finite-state Markov chain with unknown parameters. When user \( i \)
is assigned resource \( j \), assuming there are no other conflicting users assigned this re-
source, \( i \) is able to receive a reward that depends on the corresponding state each time
it is allocated the resource \( j \). The state space is denoted by \( S^{i,j} = \{z_1, z_2, \ldots, z_{|S^{i,j}|}\} \).
The state of the Markov chain for each user-resource pair \((i, j)\) evolves only when re-
source \( j \) is allocated to user \( i \). We assume the Markov chains for different user-resource
pairs are mutually independent. The reward got by user \( i \) while allocated resource \( j \)
on state \( z \in S^{i,j} \) is denoted by \( \theta_{z}^{i,j} \), which is also unknown to the users. We denote by
\( P^{i,j} = \{p^{i,j}_{z_a, z_b} \} \) the transition probability matrix for the Markov chain \((i, j)\).
Denote by \( \pi_z^{i,j} \) the steady state distribution for state \( z \). The mean reward got by user \( i \)
on resource $j$ is denoted by $\mu_{i,j}$. Then we have $\mu_{i,j} = \sum_{z \in S_{i,j}} \theta_{i,j}^{z} \pi_{i,j}^{z}$. The set of all mean rewards is denoted by $\mu = \{\mu_{i,j}\}$.

We denote by $Y_{i,j}(n)$ the actual reward obtained by a user $i$ if it is assigned resource $j$ at time $n$. We assume that $Y_{i,j}(n) = \theta_{i,j}^{z(n)}$, if user $i$ is the only occupant of resource $j$ at time $n$ where $z(n)$ is the state of Markov chain associated with $(i, j)$ at time $n$. Else, if multiple users are allocated resource $j$, then we assume that, due to interference, at most one of the conflicting users $j'$ gets reward $Y_{i,j'}(n) = \theta_{i,j'}^{z'(n)}$ where $z'(n)$ is the state of Markov chain associated with $(i, j')$ at time $n$, while the other users on the resources $j \neq j'$ get zero reward, i.e., $Y_{i,j}(n) = 0$. This interference model covers scenarios in many networking settings.

A deterministic policy $\phi(n)$ at each time is defined as a map from the observation history $\{O_t\}_{t=1}^{n-1}$ to a vector of resources $o(n)$ to be selected at period $n$, where $O_t$ is the observation at time $t$; the $i$-th element in $o(n)$, $o_i(n)$, represents the resource allocation for user $i$. Then the observation history $\{O_t\}_{t=1}^{n-1}$ in turn can be expressed as $\{o_i(t), Y_{i,o_i(t)}(t)\}_{1 \leq i \leq M, 1 \leq t < n}$.

Due to the fact that allocating more than one user to a resource is always worse than assigning each a different resource in terms of sum-throughput, we will focus on collision-free policies that assign all users distinct resources, which we will refer to as a permutation or matching. There are $P(N, M)$ such permutations.

We formulate our problem as a combinatorial multi-armed bandit, in which each arm corresponds to a matching of the users to resources. We can represent the arm
corresponding to a permutation \( k (1 \leq k \leq P(N, M)) \) as the index set \( A_k = \{(i, j) : (i, j) \text{ is in permutation } k\} \). The stochastic reward for choosing arm \( k \) at time \( n \) under policy \( \phi \) is then given as

\[
Y_{\phi(n)}(n) = \sum_{(i,j) \in A_{\phi(n)}} Y_{i,j}(n) = \sum_{(i,j) \in A_{\phi(n)}} \theta_{i,j}^{z_{\phi(n)}}.
\]

Note that different from most prior work on multi-armed bandits, this combinatorial formulation results in dependence across arms that share common components.

A key metric of interest in evaluating a given policy for this problem is (single-action) regret, which is defined as the difference between the expected reward that could be obtained by the best-possible static matching, and that obtained by the given policy. It can be expressed as:

\[
\mathcal{R}_{\phi}(n) = n\mu^* - \mathbb{E}_{\phi}^\phi[\sum_{t=1}^n Y_{\phi(t)}(t)] = n\mu^* - \mathbb{E}_{\phi}^\phi[\sum_{t=1}^n \sum_{(i,j) \in A_{\phi(t)}} \theta_{i,j}^{z_{\phi(t)}}],
\]

where \( \mu^* = \max_k \sum_{(i,j) \in A_k} \mu^{i,j} \), the expected reward of the optimal arm, is the expected sum-weight of the maximum weight matching of users to resources with \( \mu^{i,j} \) as the weight.

We are interested in designing policies for this combinatorial multi-armed bandit problem with Markovian rewards that perform well with respect to regret. Intuitively, we
would like the regret $\mathcal{R}^\phi(n)$ to be as small as possible. If it is sub-linear with respect to time $n$, the time-averaged regret will tend to zero.

### 5.3 Matching Learning for Markovian Rewards

A straightforward idea for the combinatorial multi-armed bandit problem with Markovian rewards is to treat each matching as an arm, apply UCB1 policy (given by Auer et al. [13]) directly, and ignore the dependencies across the different arms. For each arm $k$, two variables are stored and updated: the time average of all the observation values of arm $k$ and the number of times that arm $k$ has been played up to the current time slot. The UCB1 policy makes decisions based on this information alone.

However, there are several problems that arise in applying UCB1 directly in the above setting. We note that UCB1 requires both the storage and computation time that are linear in the number of arms. Since the number of arms in this formulation grows as $P(N, M)$, it is highly unsatisfactory. Also, the upper-bound of regret given in [74] will not work anymore since the rewards across arms are not independent anymore and the states of an arm may involve even when this arm is not played. No existing analytical result on the upper-bound of regret can be applied directly in this setting to the best of our knowledge.

So we are motivated to propose a policy which more efficiently stores observations from correlated arms and exploits the correlations to make better decisions. Our key idea is to use two $M$ by $N$ matrices, $(\hat{\theta}_{i,j})_{M \times N}$ and $(m_{i,j})_{M \times N}$, to store the information for each user-resource pair, rather than for each arm as a whole. $\hat{\theta}_{i,j}$ is the average
Algorithm 6 Matching Learning for Markovian Rewards (MLMR)

1: // INITIALIZATION
2: for $p = 1$ to $M$ do
3:  for $q = 1$ to $N$ do
4:    $n = (M - 1)p + q$;
5:    Play any permutation $k$ such that $(p, q) \in A_k$;
6:    Update $(\hat{\theta}_{i,j})_{M \times N}, (m_{i,j})_{M \times N}$ accordingly.
7:  end for
8: end for
9: // MAIN LOOP
10: while 1 do
11:   $n = n + 1$;
12:   Solve the Maximum Weight Matching problem (e.g., using the Hungarian algorithm [50]) on the bipartite graph of users and resources with edge weights $(\hat{\theta}_{i,j} + \sqrt{\frac{L \ln n}{m_{i,j}}})_{M \times N}$ to play arm $k$ that maximizes
13:    $\sum_{(i,j) \in A_k} \left( \hat{\theta}_{i,j} + \sqrt{\frac{L \ln n}{m_{i,j}}} \right)$ (5.2)
14:    where $L$ is a positive constant.
15:   Update $(\hat{\theta}_{i,j})_{M \times N}, (m_{i,j})_{M \times N}$ accordingly.
16: end while

(sample mean) of all the observed values of resource $j$ by user $i$ up to the current time slot (obtained through potentially different sets of arms over time). $m_{i,j}$ is the number of times that resource $j$ has been assigned to user $i$ up to the current time slot.

At each time slot $n$, after an arm $k$ is played, we get the observation of $Y_{i,j}^k(n)$ for all $(i, j) \in A_k$. Then $(\hat{\theta}_{i,j})_{M \times N}$ and $(m_{i,j})_{M \times N}$ (both initialized to 0 at time 0) are updated as follows:

$$\hat{\theta}_{i,j}(n) = \begin{cases} 
\frac{\hat{\theta}_{i,j}(n-1)m_{i,j}(n-1)+Y_{i,j}^k(n)}{m_{i,j}(n-1)+1}, & \text{if } (i, j) \in A_k \\
\hat{\theta}_{i,j}(n-1), & \text{else}
\end{cases} \quad (5.3)$$
\[
m_{i,j}(n) = \begin{cases} 
  m_{i,j}(n-1) + 1, & \text{if } (i, j) \in \mathcal{A}_k \\
  m_{i,j}(n-1), & \text{else}
\end{cases}
\] (5.4)

Note that while we indicate the time index in the above updates for notational clarity, it is not necessary to store the matrices from previous time steps while running the algorithm.

Our proposed policy, which we refer to as Matching Learning for Markovian Rewards, is shown in Algorithm 6.

### 5.4 Analysis of Regret

We summarize some notation we use in the description and analysis of our MLMR policy in Table 5.1.

The regret of a policy for a multi-armed bandit problem is traditionally upper-bounded by analyzing the expected number of times that each non-optimal arm is played and then taking the summation over these expectation times the reward difference between an optimal arm and a non-optimal arm all non-optimal arms. Although we could use this approach to analyze the MLMR policy, we notice that the upper-bound for regret consequently obtained is quite loose, which is linear in the number of arms, \(P(N, M)\). Instead, we present here a novel analysis for a tighter analysis of the MLMR policy. Our analysis shows an upper bound of the regret that is polynomial in \(M\) and \(N\), and uniformly logarithmic over time.
$N$: number of resources.
$M$: number of users, $M \leq N$.
$k$: index of a parameter used for an arm, $1 \leq k \leq P(N, M)$.
$i, j$: index of a parameter used for user $i$, resource $j$.
$A_k$: \{(i, j): (i, j) is in permutation $k$\}
$K_{i,j}$: \{$A_k: (i, j) \in A_k$\}
*: index indicating that a parameter is for the optimal arm. If there are multiple optimal arms, * refers to any of them.
$m_{i,j}$: number of times that resource $j$ has been matched with user $i$ up to the current time slot.
$\hat{\theta}_{i,j}$: average (sample mean) of all observed values of resource $j$ by user $i$ up to current time slot.
$m^k_i$: $m_{i,j}$ such that $(i, j) \in A_k$ at current time slot.
$S^{i,j}$: state space of the Markov chain for user-resource pair $(i, j)$.
$P^{i,j}$: transition matrix of the Markov chain associated with user-resource pair $(i, j)$.
$\pi^z_{i,j}$: steady state distribution for state $z$ of the Markov chain associated with $(i, j)$.
$\theta^z_{i,j}$: reward obtained by user $i$ while access resource $j$ on state $z \in S^{i,j}$.
$\mu^{i,j}$: $\sum_{z \in S^{i}} \theta^z_{i,j} \pi^z_{i,j}$, the mean reward for user $i$ using resource $j$.
$\mu^k$: $\sum_{(i, j) \in A_k} \mu^{i,j}$
$\mu^*$: $\max_k \sum_{(i, j) \in A_k} \mu^{i,j}$
$\Delta_k$: $\mu^* - \mu^k$.
$\Delta_{\min}$: $\min_{k: \mu^k < \mu^*} \Delta_k$.
$\Delta_{\max}$: $\max_k \Delta_k$.
$\pi_{\min}$: $\min_{1 \leq i \leq M, 1 \leq j \leq N, z \in S^{i,j}} \pi^z_{i,j}$.
$s_{\max}$: $\max_{1 \leq i \leq M, 1 \leq j \leq N} |S_{i,j}|$.
$s_{\min}$: $\min_{1 \leq i \leq M, 1 \leq j \leq N} |S_{i,j}|$. 
\[ \theta_{\text{max}}: \max_{1 \leq i \leq M, 1 \leq j \leq N, z \in S} \theta_{i,j}^z, \]
\[ \theta_{\text{min}}: \min_{1 \leq i \leq M, 1 \leq j \leq N, z \in S} \theta_{i,j}^z. \]
\[ \epsilon_{i,j}: \text{eigenvalue gap, defined as } 1 - \lambda_2, \text{ where } \lambda_2 \text{ is the second largest eigenvalue of } P_{i,j}. \]
\[ \epsilon_{\text{max}}: \max_{1 \leq i \leq M, 1 \leq j \leq N} \epsilon_{i,j}. \]
\[ \epsilon_{\text{min}}: \min_{1 \leq i \leq M, 1 \leq j \leq N} \epsilon_{i,j}. \]
\[ T_k(n): \text{number of times arm } k \text{ has been played by MLMR in the first } n \text{ time slots.} \]
\[ \hat{\theta}_k(n): \sum_{(i,j) \in A_k} \hat{\theta}_{i,j}(n). \text{ It is the summation of all the} \]
\[ \text{average observation values in arm } k \text{ at time } n. \]
\[ \hat{\theta}_{i,m\|^k}_k: \hat{\theta}_{i,j}(n) \text{ such that }(i,j) \in A_k \text{ and } m_{i,j}(n) = m^k_i. \]
\[ \hat{\theta}_{k,m^k_1,\ldots,m^k_M}: \sum_{i=1}^{M} \hat{\theta}_{i,m^k_i}. \]

Table 5.1: Notation.

The following lemmas are needed for our main results in Theorem 5:

**Lemma 2.** (Lemma 2.1 from [10]) \( \{X_n, n = 1, 2, \ldots\} \) is an irreducible aperiodic Markov chain with state space \( S \), transition matrix \( P \), a stationary distribution \( \pi_z, \forall z \in S \), and an initial distribution \( q \). Let \( F_t \) be the \( \sigma \)-algebra generated by \( X_1, X_2, \ldots, X_t \). Let \( G \) be a \( \sigma \)-algebra independent of \( F = \bigvee_{t \geq 1} F_t \). Let \( \tau \) be a stopping time with respect to the increasing family of \( \sigma \)-algebra \( G \vee F_t, t \geq 1 \). Define \( N(z, \tau) \) such that \( N(z, \tau) = \sum_{t=1}^{\tau} I(X_t = z) \). Then,

\[ |E[N(z, \tau) - \pi_z E[\tau]]| \leq A_P, \tag{5.5} \]

for all \( q \) and all \( \tau \) such that \( E[\tau] < \infty \). \( A_P \) is a constant that depends on \( P \).
Lemma 3. (Corollary 1 from [74]) Let $\pi_{\min}$ be the minimum value among the stationary distribution, which is defined as $\pi_{\min} = \min_{z \in S} \pi_z$. Then $A_P \leq 1/\pi_{\min}$.

Lemma 4. For user-resource matching, if the state of reward associated with each user-resource pair $(i, j)$ is given by a Markov chain, denoted by $\{X_{i,j}^1, X_{i,j}^2, \ldots\}$, satisfying the properties of Lemma 2, then the regret under policy $\phi$ is bounded by:

$$R^\phi(n) \leq \sum_{k=1}^{P(N,M)} (\mu^* - \mu^k) E^\phi[T_k^\phi(n)] + A_{S,P,\Theta},$$  \hspace{1cm} (5.6)$$

where $A_{S,P,\Theta}$ is a constant that depends on all the state spaces $\{S_{i,j}\}_{1 \leq i \leq M, 1 \leq i \leq N}$, transition probability matrices $\{P_{i,j}\}_{1 \leq i \leq M, 1 \leq i \leq N}$ and the rewards set $\{\theta_{i,j}^z, z \in S_{i,j}\}_{1 \leq i \leq M, 1 \leq i \leq N}$.

Proof. $\forall 1 \leq i \leq M, 1 \leq j \leq N$, define $G_{i,j} = \bigvee_{k \neq i, l \neq j} F_{k,l}$ where $F_{k,l} = \bigvee_{t \geq 1} F_{i,j}^t$, which applies to the Markov chain $\{X_{i,j}^1, X_{i,j}^2, \ldots\}$. We note that the Markov chains of different user-resource pairs are mutually independent, so $\forall i, j, G_{i,j}$ is independent of $F_{i,j}$. $F_{i,j}$ satisfies the conditions in Lemma 2. Note that $T_{i,j}^\phi(n)$ is a stopping time with respect to $\{G_{i,j} \lor F_{n,j}^i, n > 1\}$.

Since the state of a Markov chain evolves only when it is observed, $X_{i,j}^1, X_{i,j}^2, \ldots, X_{i,j}^n_{T_{i,j}^\phi(n)}$ represents the successive states of the Markov chain up to $n$ when assigning resource $j$ to user $i$. Then the total reward obtained under policy $\phi$ up to time $n$ is given by:

$$\sum_{t=1}^{n} Y_{\phi(t)}(t) = \sum_{j=1}^{N} \sum_{i=1}^{M} \sum_{l=1}^{T_{i,j}^\phi(n)} \sum_{z \in S_{i,j}} \theta_{i,j}^z \mathbb{1}(X_{l}^{i,j} = z).$$  \hspace{1cm} (5.7)$$

75
Note that $\forall i = 1, \ldots, M, T^\phi_k(n) = T^\phi_k(n,i)$, where $T^\phi_k(n,i)$ is the number of times up to $n$ that the $i$-th component has been observed while playing arm $k$, and there exists one resource index $j$ such that $(i, j) \in A_k$. So, we have:

$$
\sum_{k=1}^{P(N,M)} \mu^k E^\phi[T^\phi_k(n)] = \sum_{k=1}^{P(N,M)} \sum_{i=1}^{M} \mu^k_i E^\phi[T^\phi_k(n)]
$$

$$
= \sum_{k=1}^{P(N,M)} \sum_{i=1}^{M} \mu^k_i \sum_{j=1}^{A_k \in \mathcal{K}(i,j)} E^\phi[T^\phi_k(i,j)]
$$

$$
= \sum_{j=1}^{N} \sum_{i=1}^{M} \mu^i_j \sum_{A_k \in \mathcal{K}(i,j)} E^\phi[T^\phi_k(i,j)]
$$

$$
= \sum_{j=1}^{N} \sum_{i=1}^{M} \mu^i_j \sum_{A_k \in \mathcal{K}(i,j)} \sum_{l=1}^{\sum_{z \in S^{i,j}}} \theta^i_j \pi^i_j \sum_{z \in S^{i,j}} E^\phi[T^\phi_k(i,j)]
$$

Hence,

$$
|R^\phi(n) - \sum_{k=1}^{P(N,M)} (\mu^* - \mu^k) E^\phi[T^\phi_k(n)]|
$$

$$
= |R^\phi(n) - (n\mu^* - \sum_{k=1}^{P(N,M)} \mu^k E^\phi[T^\phi_k(n)])|
$$

$$
= |(n\mu^* - \sum_{t=1}^{n} E^\phi[Y^\phi(t)]) - (n\mu^* - \sum_{k=1}^{P(N,M)} \mu^k E^\phi[T^\phi_k(n)])|
$$

$$
= |E^\phi \sum_{t=1}^{n} Y^\phi(t)] - \sum_{k=1}^{P(N,M)} \mu^k E^\phi[T^\phi_k(n)]|
$$

$$
= |E^\phi \sum_{j=1}^{N} \sum_{i=1}^{M} T^\phi_{i,j}(n) - \sum_{j=1}^{N} \sum_{i=1}^{M} \sum_{l=1}^{\sum_{z \in S^{i,j}}} \theta^i_j \pi^i_j \sum_{z \in S^{i,j}} E^\phi[T^\phi_{i,j}(n)]|
$$
\[
\leq \sum_{j=1}^{N} \sum_{i=1}^{M} \sum_{z \in S^{i,j}} T_{i,j}^{\phi}(n) \left| E^\phi \left[ \sum_{l=1}^{T_{i,j}^{\phi}(n)} \theta_{i,j}^{l} I(X_{i}^{j} = z) \right] - \theta_{z}^{i,j} \pi_{z}^{i,j} E\phi[T_{i,j}^{\phi}(n)] \right|
\]

\[
= \sum_{j=1}^{N} \sum_{i=1}^{M} \sum_{z \in S^{i,j}} T_{i,j}^{\phi}(n) \left| E^\phi \left[ \sum_{l=1}^{T_{i,j}^{\phi}(n)} I(X_{i}^{j} = z) \right] - \pi_{z}^{i,j} E\phi[T_{i,j}^{\phi}(n)] \right|
\]

\[
= \sum_{j=1}^{N} \sum_{i=1}^{M} \sum_{z \in S^{i,j}} \theta_{i,j}^{z} \left| E^\phi[N(z, T_{i,j}^{\phi}(n))] - \pi_{z}^{i,j} E\pi[T_{i,j}^{\phi}(n)] \right| .
\]

Based on Lemma 2, we have:

\[
|\mathcal{R}^{\phi}(n) - \sum_{k=1}^{P(N,M)} (\mu^* - \mu^k) E\phi[T_{k}^{\phi}(n)]| \leq \sum_{j=1}^{N} \sum_{i=1}^{M} \sum_{z \in S^{i,j}} \theta_{i,j}^{z} \sum_{z \in S} \theta_{z}^{i,j} C_{P_{i,j}} = A_{S,P,\Theta}. \quad (5.8)
\]

Lemma 5. (Theorem 2.1 from [39]) Let \( \{X_n, n = 1, 2, \ldots\} \) be an irreducible aperiodic Markov chain with finite state space \( S \), transition matrix \( P \), a stationary distribution \( \pi_z \), \( \forall z \in S \), and an an initial distribution \( q \). Let \( N_q = ||(q_z/\pi_z), z \in S||_2 \). The eigenvalue gap \( \epsilon \) is defined as \( \epsilon = 1 - \lambda_2 \), where \( \lambda_2 \) is the second largest eigenvalue of the matrix \( P \).

\( \forall A \subseteq S \), define \( t_A(n) \) as the total number of times that all states in the set \( A \) are visited up to time \( n \). Then \( \forall \gamma \geq 0 \),

\[
P(t_A(n) - n\pi_A \geq \gamma) \leq (1 + \frac{\gamma \epsilon}{10n} N_q e^{-\gamma^2/20n}), \quad (5.9)
\]

where \( \pi_A = \sum_{z \in A} \pi_z \).
Our main results on the regret of MLMR policy are shown in Theorem 5. We show that with using a constant \( L \) which is bigger than a value determined by the minimum eigenvalue gap of the transition matrix, maximum value of the number of states, and maximum value of the rewards, our MLMR policy is guaranteed to achieve a regret that is uniformly logarithmic in time, and polynomial in the number of users and resources.

**Theorem 5.** When using any constant \( L \geq \frac{(50+40M)\theta_{\text{max}}^2 s_{\text{max}}^2}{\epsilon_{\text{min}}} \), the expected regret under the MLMR policy specified in Algorithm 6 is at most

\[
\left[ \frac{4M^3NL\ln n}{(\Delta_{\text{min}})^2} + MN + M^2N s_{\text{max}} \left( 1 + \frac{\epsilon_{\text{max}}\sqrt{L}}{10s_{\text{min}}\theta_{\text{min}}} \right) \frac{\pi}{3} \right] \Delta_{\text{max}} + A_{S,P,\Theta},
\]

where \( \Delta_{\text{min}}, \Delta_{\text{max}}, \pi_{\text{min}}, s_{\text{max}}, s_{\text{min}}, \theta_{\text{max}}, \theta_{\text{min}}, \epsilon_{\text{max}}, \epsilon_{\text{min}} \) follow the definition in Table 5.1; \( A_{S,P,\Theta} \) follows the definition in Lemma 4.

**Proof.** Let \( C_{t,n} \) be \( \sqrt{\frac{L\ln t}{n}} \). Denote \( C_{t,n,A_k} = \sum_{(i,j) \in A_k} \sqrt{\frac{L\ln t}{m_{i,j}}} = \sum_{i=1}^{M} \sqrt{\frac{L\ln t}{n_{i}}} = \sum_{i=1}^{M} C_{t,n_i} \). It is also denoted by \( C_{t,(n_1^k,...,n_M^k)} \) sometimes for a clear explanation in this proof.

We introduce \( \tilde{T}_{i,j}(n) \) as a counter after the initialization period. It is updated in the following way:

At each time slot after the initialization period, one of the two cases must happen: (1) an optimal arm is played; (2) a non-optimal arm is played. In the first case, \( (\tilde{T}_{i,j}(n))_{M \times N} \) won’t be updated. When an non-optimal arm \( k(n) \) is picked at time \( n \), there must be at least one \( (i,j) \in A_k \) such that \( m_{i,j}(n) = \min_{(i_1,j_1) \in A_k} m_{i_1,j_1} \). If there is only one such arm,
\( \tilde{T}_{i,j}(n) \) is increased by 1. If there are multiple such arms, we arbitrarily pick one, say \((i', j')\), and increment \( \tilde{T}_{i',j'} \) by 1.

Each time when a non-optimal arm is picked, exactly one element in \( (\tilde{T}_{i,j}(n))_{M \times N} \) is incremented by 1. This implies that the total number that we have played the non-optimal arms is equal to the summation of all counters in \( (\tilde{T}_{i,j}(n))_{M \times N} \). Therefore, we have

\[
\sum_{k: \mu_k < \mu^*} \mathbb{E}[T_k(n)] = \sum_{i=1}^{M} \sum_{j=1}^{N} \mathbb{E}[\tilde{T}_{i,j}(n)].
\] (5.11)

Also note for \( \tilde{T}_{i,j}(n) \), the following inequality holds

\[
\tilde{T}_{i,j}(n) \leq m_{i,j}(n), \forall 1 \leq i \leq M, 1 \leq j \leq N.
\] (5.12)

Denote by \( \tilde{I}_{i,j}(n) \) the indicator function which is equal to 1 if \( \tilde{T}_{i,j}(n) \) is added by one at time \( n \). Let \( l \) be an arbitrary positive integer. Then

\[
\tilde{T}_{i,j}(n) = \sum_{t=MN+1}^{n} \mathbb{1}\{\tilde{I}_{i,j}(t)\} \leq l + \sum_{t=MN+1}^{n} \mathbb{1}\{\tilde{I}_{i,j}(t), \tilde{T}_{i,j}(t-1) \geq l\}
\] (5.13)

where \( \mathbb{1}(x) \) is the indicator function defined to be 1 when the predicate \( x \) is true, and 0 when it is false.

When \( \tilde{I}_{i,j}(t) = 1 \), there exists some arm such that a non-optimal arm is picked for which \( m_{i,j} \) is the minimum in this arm. We denote this arm by \( k(t) \) since at each time that \( \tilde{I}_{i,j}(t) = 1 \), we may get different arms. Then,
\[
\tilde{T}_{i,j}(n) \leq l + \sum_{t=M+1}^n 1 \{ \hat{\theta}(t-1) + C_{t-1,n^*(t-1)} \leq \hat{\theta}(t-1) + C_{t-1,n_A(t-1)}(t-1), \tilde{T}_{i,j}(t-1) \geq l \}
\]

\[
= l + \sum_{t=M+1}^n 1 \{ \hat{\theta}(t) + C_{t,n^*(t)} \leq \hat{\theta}(t) + C_{t,n_A(t)}, \tilde{T}_{i,j}(t) \geq l \}.
\]

Based on (5.12), \( l \leq \tilde{T}_{i,j}(t) \) implies:

\[
l \leq \tilde{T}_{i,j}(t) \leq m_{i,j}(t) = m_i^k(t).
\]

So,

\[
\forall 1 \leq i \leq M, m_i^k(t) \geq l.
\]

Then we can bound \( \tilde{T}_{i,j}(n) \) as,

\[
\tilde{T}_{i,j}(n) \leq l + \sum_{t=M+1}^n 1 \{ \min_{0 < m_1^*, \ldots, m_M^* \leq t} \hat{\theta} m_1^*, \ldots, m_M^* + C_{t,(m_1^*, \ldots, m_M^*)} \leq \hat{\theta}(t) + \theta_{k(t)}, m_1^k(t), \ldots, m_M^k(t) + C_{t,(m_1^k(t), \ldots, m_M^k(t))} \}
\]

\[
\leq l + \sum_{t=1}^\infty \sum_{m_1^1=1}^t \cdots \sum_{m^k_M}^t \sum_{m_1^1=1}^t \cdots \sum_{m^k_M}^t 1 \{ \hat{\theta} m_1^*, \ldots, m_M^* + C_{t,(m_1^*, \ldots, m_M^*)} \leq \hat{\theta}(t) + \theta_{k(t)}, m_1^k(t), \ldots, m_M^k(t) + C_{t,(m_1^k(t), \ldots, m_M^k(t))} \}.
\]

\[
\hat{\theta} m_1^*, \ldots, m_M^* + C_{t,(m_1^*, \ldots, m_M^*)} \leq \hat{\theta}(t) + \theta_{k(t)}, m_1^k(t), \ldots, m_M^k(t) + C_{t,(m_1^k(t), \ldots, m_M^k(t))} \] means that at least one of the following must be true:
\[
\hat{\theta}_{m^*_1,\ldots,m^*_M} \leq \mu^* - C_{t,(m^*_1,\ldots,m^*_M)},
\] (5.14)

\[
\hat{\theta}_{k(t),m^*_{k(t)}(t),\ldots,m^*_{M(t)}} \geq \mu_{k(t)} + C_{t,(m^*_{k(t)}(t),\ldots,m^*_{M(t)})},
\] (5.15)

\[
\mu^* < \mu_{k(t)} + 2C_{t,(m^*_{k(t)}(t),\ldots,m^*_{M(t)})}.
\] (5.16)

Here we first find the upper bound for \( P\{\hat{\theta}_{m^*_1,\ldots,m^*_M} \leq \mu^* - C_{t,(m^*_1,\ldots,m^*_M)}\} \):

\[
P\{\hat{\theta}_{m^*_1,\ldots,m^*_M} \leq \mu^* - C_{t,(m^*_1,\ldots,m^*_M)}\} = P\{\sum_{i=1}^{M} \hat{\theta}_{i,m^*_i} \leq \sum_{i=1}^{M} \mu^*_i - \sum_{i=1}^{M} C_{t,n^*_i}\}
\leq \sum_{i=1}^{M} P\{\hat{\theta}_{i,m^*_i} \leq \mu^*_i - C_{t,n^*_i}\}.
\] (5.17)

\( \forall 1 \leq i \leq M, \)

\[
P\{\hat{\theta}_{i,m^*_i} \leq \mu^*_i - C_{t,n^*_i}\} = P\{\sum_{z=1}^{S^*_i} \frac{\theta^*_i(z)m^*_i(z)}{m^*_i} \leq \sum_{z=1}^{S^*_i} \theta^*_i(z)\pi^*_i(z) - C_{t,n^*_i}\}
\leq P\{\sum_{z=1}^{S^*_i} (\theta^*_i(z)m^*_i(z) - m^*_i\theta^*_i(z)\pi^*_i(z)) \leq -m^*_iC_{t,n^*_i}\}
\leq P\{\text{At least one of the following must hold:}\}
\]

\[
\theta^*_i(1)m^*_i(1) - m^*_i\theta^*_i(1)\pi^*_i(1) \leq -\frac{m^*_i}{|S^*_i|}C_{t,n^*_i},
\]

\[
\vdots
\]

\[
\theta^*_i(|S^*_i|)m^*_i(|S^*_i|) - m^*_i\theta^*_i(|S^*_i|)\pi^*_i(|S^*_i|) \leq -\frac{m^*_i}{|S^*_i|}C_{t,n^*_i}
\]

\[
\leq \sum_{z=1}^{S^*_i} P\{\theta^*_i(z)m^*_i(z) - m^*_i\theta^*_i(z)\pi^*_i(z) \leq -\frac{m^*_i}{|S^*_i|}C_{t,n^*_i}\}
\]

81
\[
= \sum_{z=1}^{S^*_i} \mathbb{P}\left\{ m^*_i(z) - m^*_i \pi^*_i(z) \leq -\frac{m^*_i}{|S^*_i| \theta^*_i(z)} C_{t,n^*_i} \right\}
\]
\[= \sum_{z=1}^{S^*_i} \mathbb{P}\left\{ (m^*_i - \sum_{l \neq z} m^*_i(l)) - m^*_i(1 - \sum_{l \neq z} \pi^*_i(z)) \leq -\frac{m^*_i}{|S^*_i| \theta^*_i(z)} C_{t,n^*_i} \right\}
\]
\[= \sum_{z=1}^{S^*_i} \mathbb{P}\left\{ \sum_{l \neq z} m^*_i(l) - m^*_i \sum_{l \neq z} \pi^*_i(z) \geq \frac{m^*_i}{|S^*_i| \theta^*_i(z)} C_{t,n^*_i} \right\}. \quad (5.18)
\]

\[\forall 1 \leq z \leq |S^*_i|, \text{ applying Lemma 5, we can find the upper bound of each probability in (5.18) as,}\]
\[P\{\hat{\theta}_{i,m^*_i} \leq \mu^*_i - C_{t,n^*_i} \} \leq \sum_{z=1}^{S^*_i} \left( 1 + \frac{\epsilon_{i,j}}{10|S^*_i| \theta^*_i(z)} \right) \sqrt{\frac{L \ln t}{N_{q_{i,j}}} e^{\frac{-L \ln t}{20 S_{\max} \theta^*_i(z) \pi_{\min}}}} \leq \sum_{z=1}^{S^*_i} \left( 1 + \frac{\epsilon_{\max} \sqrt{L t}}{10 S_{\min} \theta_{\min}} \right) N_{q_{i,j}} e^{\frac{-L \ln t}{20 S_{\max} \theta^*_i(z) \pi_{\min}}} \leq S_{\max} \sqrt{t} \left( 1 + \frac{\epsilon_{\max} \sqrt{L t}}{10 S_{\min} \theta_{\min}} \right) t^{-\frac{L \epsilon_{\min}}{20 S_{\max} \theta^*_i(z) \pi_{\min}}} \quad (5.19)\]

where (5.19) holds since for any \(q_{i,j}\),
\[N_{q_{i,j}} = \left\| \frac{q_{i,j}^z}{\pi_{\min}^z}, z \in S^i \right\|_2 \leq \sum_{z=1}^{S^i} \left\| \frac{q_{i,j}^z}{\pi_{\min}^z} \right\|_2 \leq \sum_{z=1}^{S^i} \left\| q_{i,j}^z \right\|_{\pi_{\min}} = \frac{1}{\pi_{\min}}.\]

Thus,
\[P\{\hat{\theta}_1^*, ..., \hat{\theta}_M^* \leq \theta^* - C_{t,(m^*_1, ..., m^*_M)} \} \leq \frac{M s_{\max}}{\pi_{\min}} \left( 1 + \frac{\epsilon_{\max} \sqrt{L t}}{10 S_{\min} \theta_{\min}} \right) t^{-\frac{L \epsilon_{\min} - 10 s_{\max} \theta^*_i(z) \pi_{\min}^z}{20 S_{\max} \theta^*_i(z) \pi_{\min}}} . \]

82
With the similar calculation, we can also get the upper bound of the probability for (5.15):

\[
\begin{align*}
\mathbb{P}\{\tilde{\theta}_{k(t),m_1^k(t),\ldots,m_M^k(t)} \geq \mu_k + C t_{m_1^k(t),\ldots,m_M^k(t)}\} \\
\leq \sum_{i=1}^{M} \mathbb{P}\{\tilde{\theta}_{i,m_i^k} \geq \mu_i + C t_{n_i^k}\} \leq \sum_{i=1}^{M} \sum_{z=1}^{\left|S_i^k\right|} \mathbb{P}\left\{\frac{\theta_i^k(z)m_i^k(z)}{m_i^k} \geq \frac{\theta_i^k(z)\pi_i^k(z)}{\pi_i^k(z)} + C t_{n_i^k}\right\} \\
= \sum_{i=1}^{M} \sum_{z=1}^{\left|S_i^k\right|} \mathbb{P}\left\{m_i^k(z) - m_i^k\pi_i^k(z) \geq \frac{m_i^k}{\left|S_i^k\right|} C t_{n_i^k}\right\} \\
\leq \sum_{i=1}^{M} \sum_{z=1}^{\left|S_i^k\right|} \mathbb{P}\left\{m_i^k(z) - m_i^k\pi_i^k(z) \geq \frac{m_i^k}{\left|S_i^k\right|} C t_{n_i^k}\right\} \\
\leq \sum_{i=1}^{M} \frac{s_{\max}}{\pi_{\min}} \left(1 + \frac{\epsilon_{\max} \sqrt{L}}{10s_{\min}\theta_{\min}}\right) t^{-\frac{Ls_{\min} - 10s_{\max} \sigma_{\max}^2}{20s_{\max} \sigma_{\max}^2}} \\
\leq M s_{\max} \pi_{\min} \left(1 + \frac{\epsilon_{\max} \sqrt{L}}{10s_{\min}\theta_{\min}}\right) t^{-\frac{Ls_{\min} - 10s_{\max} \sigma_{\max}^2}{20s_{\max} \sigma_{\max}^2}}.
\end{align*}
\]

(5.20)

Note that for \( l \geq \left\lceil \frac{4L \ln n}{\left(\frac{\Delta_k(t)}{M}\right)^2} \right\rceil \),

\[
\begin{align*}
\mu^* - \mu_k(t) - 2C t_{m_1^k(t),\ldots,m_M^k(t)} \\
= \mu^* - \mu_k(t) - 2 \sum_{i=1}^{M} \sqrt{\frac{L \ln t}{n_i^k}} \\
\geq \mu^* - \mu_k(t) - M \sqrt{\frac{4L \ln n}{4L \ln n} \left(\frac{\Delta_k(t)}{M}\right)^2} \\
= \mu^* - \mu_k(t) - \Delta_k(t) = 0.
\end{align*}
\]

(5.21)
(5.21) implies that condition (5.16) is false when \( \ell = \left[ \frac{4L \ln n}{\left( \frac{\Delta_{ij}}{\theta_{\min}} \right)^2} \right] \). If we let \( \ell = \left[ \frac{4L \ln n}{\left( \frac{\Delta_{ij}}{\theta_{\min}} \right)^2} \right] \), then (5.16) is false for all \( k(t), 1 \leq t \leq \infty \) where,

\[
\Delta_{ij} = \min_k \{ \Delta_k : (i, j) \in \mathcal{A}_k \}.
\]  

Therefore,

\[
\mathbb{E}[\tilde{T}_{i,j}(n)] \leq \left[ \frac{4L \ln n}{\left( \frac{\Delta_{ij}}{\theta_{\min}} \right)^2} \right] + \sum_{t=1}^{\infty} \left( \sum_{m_1^* = 1}^{\pi_{\min}} \cdots \sum_{m_k^* = M}^{t} \sum_{m_1^* = 1}^{\pi_{\min}} \cdots \sum_{m_k^* = M}^{t} \right) 2M^{s_{\max}} \frac{\tau_{\min}}{\pi_{\min}} \left( 1 + \frac{\epsilon_{\max} \sqrt{L}}{10s_{\min} \theta_{\min}} \right) t^{-\frac{L_{\epsilon_{\min}} - 10s_{\max} \theta_{\max}^2}{20s_{\max} \theta_{\max}^2}} \sum_{t=1}^{\infty} 2t - \frac{L_{\epsilon_{\min}} - (40M + 10)s_{\max} \theta_{\max}^2}{20s_{\max} \theta_{\max}^2} \sum_{t=1}^{\infty} 2t^{-2}
\]  

\[
= \frac{4M^2 L \ln n}{(\Delta_{ij})^2} + M^{s_{\max}} \frac{\tau_{\min}}{\pi_{\min}} \left( 1 + \frac{\epsilon_{\max} \sqrt{L}}{10s_{\min} \theta_{\min}} \right) \frac{\pi}{3},
\]  

where (5.23) holds since \( L \geq \frac{(50 + 40M) \theta_{\max}^2 s_{\max}^2}{\epsilon_{\min}} \).

So under our MLMR policy,

\[
\mathcal{R}^\phi(n) \leq \sum_{k=1}^{P(N,M)} (\mu^* - \mu^k) \mathbb{E}[T^k(n)] + A_{\mathbf{S}, \mathbf{P}, \Theta}
\]  

\[
= \sum_{k: \theta_k < \theta^*} \Delta_k \mathbb{E}[T_k(n)] + A_{\mathbf{S}, \mathbf{P}, \Theta}
\]

84
We have,

\[
\mathcal{R}_k^*(n) \leq \Delta_{\max} \sum_{k: \theta_k < \theta^*} \mathbb{E}[T_k(n)] + A_{S,P,\Theta}
\]

\[
= \Delta_{\max} \sum_{i=1}^{M} \sum_{j=1}^{N} \mathbb{E}[\tilde{T}_{i,j}(n)] + A_{S,P,\Theta}
\]

\[
\leq \left[ \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{4M^2 L \ln n}{(\Delta_{\min})^2} + 1 + M \frac{s_{\max}}{\pi_{\min}} \left( 1 + \frac{\epsilon_{\max} \sqrt{L}}{10s_{\min} \theta_{\min}} \right) \frac{\pi}{3} \right] \Delta_{\max} + A_{S,P,\Theta}
\]

\[
\leq \left[ \frac{4M^3 N L \ln n}{(\Delta_{\min})^2} + MN + M^2 N \frac{s_{\max}}{\pi_{\min}} \left( 1 + \frac{\epsilon_{\max} \sqrt{L}}{10s_{\min} \theta_{\min}} \right) \frac{\pi}{3} \right] \Delta_{\max} + A_{S,P,\Theta}.
\]

(5.24)

Theorem 5 shows when we use a constant \( L \) which is large enough such that \( L \geq \frac{(50+40M)\theta_{\max}^2 s_{\max}^2}{\epsilon_{\min}} \), the regret of Algorithm 6 is upper-bounded uniformly over time \( n \) by a function that grows as \( O(M^3 N \ln n) \). However, when \( \theta_{\max}, s_{\max} \) or \( \epsilon_{\min} \) is unknown, the upper bound of regret can not be guaranteed to grow logarithmically in \( n \).

So when no knowledge about the system is available, we extend the MLMR policy to achieve a regret that is bounded uniformly over time \( n \) by a function that grows as \( O(M^3 N L(n) \ln n) \), by using any arbitrarily slowly diverging non-decreasing sequence \( L(n) \) such that \( L(n) \leq n \) for any \( n \) in Algorithm 6. Since \( L(n) \) can grow arbitrarily slowly, the MLMR can achieve a regret arbitrarily close to the logarithmic order. We present our analysis in Theorem 6.
Theorem 6. When using any arbitrarily slowly diverging non-decreasing sequence \( L(n) \) (i.e., \( L(n) \to \infty \) as \( n \to \infty \)) in (5.2) such that \( \forall n, L(n) \leq n \), the expected regret under the MLMR policy specified in Algorithm 6 is at most

\[
\left[ \frac{4M^3NL(n) \ln n}{(\Delta_{\min})^2} + MNB_{S,P,\Theta} + M^2N\frac{s_{\max}}{\pi_{\min}} \left( 1 + \frac{\epsilon_{\max}}{10s_{\min}\theta_{\min}} \right) \frac{\pi}{3} \right] \Delta_{\max} + A_{S,P,\Theta},
\]

(5.25)

where \( B_{S,P,\Theta} \) is a constant that depends on \( \theta_{\max} \), \( s_{\max} \) and \( \epsilon_{\min} \).

Proof. Let \( C_{t,n} \) be \( \sqrt{L(n) \ln n} \). Let \( C_{t,n_A} \) be \( \sum_{(i,j) \in A_k} \sqrt{L(t) \ln t} \). Then replacing \( L \) with \( L(t) \) in the proof of Theorem 5, (5.11) to (5.22) still stand. The upper bound of \( E[\tilde{T}_{i,j}(n)] \) in (5.23) should be modified as in (5.26).

\[
E[\tilde{T}_{i,j}(n)] \leq \left[ \frac{4L(n) \ln n}{(\Delta_{ij})^2} \right] + \sum_{t=1}^{\infty} \left( \sum_{m_1^1 = 1}^{t} \cdots \sum_{m_t^1 = M}^{t} \sum_{m_1^2 = 1}^{t} \cdots \sum_{m_t^2 = M}^{t} 2M\frac{s_{\max}}{\pi_{\min}} \left( 1 + \frac{\epsilon_{\max}}{10s_{\min}\theta_{\min}} \right) \frac{\pi}{3} \right) \Delta_{\max} + A_{S,P,\Theta},
\]

(5.26)

\( L(t) \) is a diverging non-decreasing sequence, so there exists a constant \( t_1 \), such that for all \( t \geq t_1 \), \( L(t) \geq \frac{(60+40M)\theta_{\max}^2 s_{\max}^2}{\epsilon_{\min}} \), which implies \( t - \frac{L(1)\epsilon_{\min} - (40M+10)\theta_{\max}^2 s_{\max}^2}{20\theta_{\max}^2 s_{\max}^2} \geq \frac{1}{2} \leq t^{-2} \).

Thus, we have
\[ E[\tilde{T}_{i,j}(n)] \leq \frac{4M^2 L(n) \ln n}{(\Delta_{\min}^i)^2} + M \frac{s_{\max}}{\pi_{\min}} \left( 1 + \frac{\epsilon_{\max}}{10s_{\min}\theta_{\min}} \right) \sum_{t=t_1}^{\infty} 2t^{-2} + B_{S,P,\Theta} \]

\[ = \frac{4M^2 L(n) \ln n}{(\Delta_{\min}^i)^2} + M \frac{s_{\max}}{\pi_{\min}} \left( 1 + \frac{\epsilon_{\max} \sqrt{L}}{10s_{\min}\theta_{\min}} \right) \frac{\pi}{3} + B_{S,P,\Theta} \]  

(5.27)

where \( B_{S,P,\Theta} \) is a constant as shown in (5.28), which depends on \( \theta_{\max}, s_{\max} \) and \( \epsilon_{\min} \).

\[ B_{S,P,\Theta} = 1 + M \frac{s_{\max}}{\pi_{\min}} \left( 1 + \frac{\epsilon_{\max}}{10s_{\min}\theta_{\min}} \right) \sum_{t=1}^{t_1-1} 2t \frac{L(t)\epsilon_{\min} - (40M+10)s_{\max}^2\epsilon_{\max}^2 + 1}{20s_{\max}^3\theta_{\max}^2}. \]  

(5.28)

Then for the MLMR policy with \( L(n) \)

\[ \mathfrak{R}(n) \leq \Delta_{\max} \sum_{i=1}^{M} \sum_{j=1}^{N} E[\tilde{T}_{i,j}(n)] + A_{S,P,\Theta} \]

\[ \leq \left[ \frac{4M^3 N L(n) \ln n}{(\Delta_{\min})^2} + MN B_{S,P,\Theta} \right] \sum_{t=1}^{t_1-1} 2t \frac{L(t)\epsilon_{\min} - (40M+10)s_{\max}^2\epsilon_{\max}^2 + 1}{20s_{\max}^3\theta_{\max}^2}. \]  

(5.29)

5.5 Examples and Simulation Results

We consider a system that consists of \( M = 2 \) users and \( N = 4 \) resources. The state of each resource evolves as an irreducible, aperiodic Markov chain with two states “0” and “1”. For all the tables in this section, the element in the \( i \)-th row and \( j \)-th column
represents the value for the user-resource pair \((i, j)\). The transition probabilities are shown in the Table 5.2, and the mean rewards for each state are shown in the Table 5.3.

\[
\begin{array}{cccc}
0.5 & 0.4 & 0.7 & 0.3 \\
0.2 & 0.9 & 0.9 & 0.7 \\
\end{array}
\quad
\begin{array}{cccc}
0.6 & 0.7 & 0.8 & 0.9 \\
0.9 & 0.5 & 0.4 & 0.4 \\
\end{array}
\]

\(p_{01}\)

\(p_{10}\)

Table 5.2: Transition probabilities.

\[
\begin{array}{cccc}
0.6 & 0.5 & 0.2 & 0.4 \\
0.3 & 0.7 & 0.8 & 0.3 \\
\end{array}
\quad
\begin{array}{cccc}
0.8 & 0.2 & 0.7 & 0.5 \\
0.5 & 0.3 & 0.6 & 0.6 \\
\end{array}
\]

\(\theta_0\)

\(\theta_1\)

Table 5.3: Rewards on each state.

For \(1 \leq i \leq M\), \(1 \leq j \leq N\), the stationary distribution of user-resource pair \((i, j)\) on state “0” is calculated as \(p_{01}^{i,j} / p_{01}^{i,j} + p_{10}^{i,j}\); the stationary distribution on state “1” is calculated as \(p_{10}^{i,j} / p_{01}^{i,j} + p_{10}^{i,j}\). The eigenvalue gap is \(\epsilon_{i,j} = p_{01}^{i,j} + p_{10}^{i,j}\). The expected reward \(\mu_{i,j}\) for all the pairs can be calculated as in Table 5.4.

\[
\begin{array}{cccc}
0.6909 & 0.3909 & 0.4333 & 0.425 \\
0.3363 & 0.4429 & 0.6615 & 0.4909 \\
\end{array}
\]

\(\mu\)

Table 5.4: Expected rewards.

We can see that the arm \\{(1,1), (2,3)\} is the optimal arm with greatest expected reward \(\mu^* = 0.6909 + 0.6615 = 1.3524\). \(\Delta_{\text{min}} = 0.1706\).

Figure 5.1 shows the simulation result of the regret (normalized with respect to the logarithm of time) for our MLMR policy for the above system with different choices of \(L\). We also show the theoretical upper bound for comparison. The value of \(L\) to satisfy
the condition in Theorem 5 is \( L \geq \frac{(50+40M)R^2s_{\text{max}}^2}{\epsilon_{\text{min}}} = 303 \), so we picked \( L = 303 \) in the simulation.

Note that in the proof of Theorem 5, when \( L < \frac{(50+40M)R^2s_{\text{max}}^2}{\epsilon_{\text{min}}} \), we have

\[
-\frac{L\epsilon_{\text{min}} - (40M + 10)s_{\text{max}}^2\theta_{\text{max}}^2}{20s_{\text{max}}^2\theta_{\text{max}}^2} > -2.
\]

This implies \( \sum_{t=1}^{\infty} 2t \frac{L\epsilon_{\text{min}} - (40M + 10)s_{\text{max}}^2\theta_{\text{max}}^2}{20s_{\text{max}}^2\theta_{\text{max}}^2} \) does not converge anymore and thus we can not bound \( E[\tilde{T}_{i,j}(n)] \) any more. Empirically, however, in 5.1 the case when \( L = 2 \) also seems to yield logarithmic regret over time and the performance is in fact better than \( L = 303 \), since the non-optimal arms are played less when \( L \) is smaller. However, this may possibly be due to the fact that the cases when \( \tilde{T}_{i,j}(n) \) grows faster than \( \ln(t) \) only happens with very small probability when \( L = 2 \).

Table 5.5 shows the number of times that resource \( j \) has been matched with user \( i \) up to time \( n = 10^7 \).
Table 5.5: Number of times that resource $j$ has been matched with user $i$ up to time $n = 10^7$.

<table>
<thead>
<tr>
<th>$m_{i,j}(10^7)$, $L = 2$</th>
<th>$m_{i,j}(10^7)$, $L = 303$</th>
</tr>
</thead>
<tbody>
<tr>
<td>999470 153 185 196</td>
<td>892477 30685 39410 37432</td>
</tr>
<tr>
<td>136 293 999155 420</td>
<td>26813 50341 850265 72585</td>
</tr>
</tbody>
</table>

Figure 5.2: Simulation results of example 2 with $\Delta_{\text{min}} = 0.0091$.

Figure 5.2 shows the simulation results of the regret of another example with the same transition probabilities as in the previous example and different rewards on states as in Table 5.6.

<table>
<thead>
<tr>
<th>$\theta_0$</th>
<th>$\theta_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.7 0.3 0.5 0.5</td>
<td>0.4 0.6 0.7 0.45</td>
</tr>
<tr>
<td>0.65 0.7 0.8 0.4</td>
<td>0.5 0.5 0.6 0.55</td>
</tr>
</tbody>
</table>

Table 5.6: Rewards on each state.

The expected reward $\mu_{i,j}$ for all the pairs can be calculated as in Table 5.7.

$\{(1, 1), (2, 3)\}$ is still the optimal arm. However, compared with the previous example, we can see that the expected reward of three other arms $\{(1, 3), (2, 1)\}$, $\{(1, 3), (2, 2)\}$, $\{(1, 1), (2, 2)\}$ are all very close to the expected reward of the optimal arm. For this example, $\Delta_{\text{min}} = 0.0091$, which is much smaller compared with the previous example.
Table 5.7: Expected rewards.

\[
\begin{array}{cccc}
0.5636 & 0.4091 & 0.5933 & 0.4875 \\
0.6227 & 0.5714 & 0.6615 & 0.4954 \\
\end{array}
\]

\(\mu\)

In this case, the non-optimal arms are played much more compared with the previous example. This is because we have several arms of which the expected rewards are very close to \(\mu^*\), so the policy has to spend a lot more time to explore on those non-optimal arms to make sure those are non-optimal arms. This fact can be seen clearly in Table 5.8, which presents the number of times that resource \(j\) has been matched with user \(i\) up to time \(n = 10^7\) under both cases when \(L = 2\) and \(L = 303\).

<table>
<thead>
<tr>
<th>(m_{i,j}(10^7), L = 2)</th>
<th>(m_{i,j}(10^7), L = 303)</th>
</tr>
</thead>
<tbody>
<tr>
<td>817529</td>
<td>544</td>
</tr>
<tr>
<td>175583</td>
<td>3610</td>
</tr>
<tr>
<td>346395</td>
<td>60031</td>
</tr>
<tr>
<td>301491</td>
<td>146317</td>
</tr>
</tbody>
</table>

Table 5.8: Number of times that resource \(j\) has been matched with user \(i\) up to time \(n = 10^7\).

5.6 Summary

We have presented the MLMR policy for the problem of learning combinatorial matchings of users to resources when the reward process is Markovian. We have shown that this policy requires only polynomial storage and computation per step, and yields a regret that grows uniformly logarithmically over time and only polynomially with the number of users and resources.
Chapter 6

Learning with Restless Markovian Rewards

6.1 Overview

In this chapter\(^1\), we consider how to solve the combinatorial network optimization problems when the edge weights vary as independent Markov chains with unknown dynamics. Using a stochastic restless multi-armed bandit approach, we propose CLRMR, an online learning algorithm. We prove that, compared to a genie that knows the Markov transition matrices and uses the single-best structure at all times, CLRMR yields regret that is polynomial in the number of edges and nearly-logarithmic in time.

6.2 Problem Formulation

We consider a system with \(N\) edges predefined by some application, where time is slotted and indexed by \(n\). For each edge \(i\) (\(1 \leq i \leq N\)), there is an associated state that evolves

\(^1\)This chapter is based in part on [38].
as a discrete-time, finite-state, aperiodic, irreducible Markov chain\(^2\) \(\{X^i(n), n \geq 0\}\) with unknown parameters\(^3\). We denote the state space for the \(i\)-th Markov chain by \(S^i\). We assume these \(N\) Markov chains are mutually independent. The reward obtained from state \(x\) (\(x \in S^i\)) of Markov chain \(i\) is denoted as \(r^i_x\). Denote by \(\pi^i_x\) the steady state distribution for state \(x\). The mean reward obtained on Markov chain \(i\) is denoted by \(\mu^i\). Then we have \(\mu^i = \sum_{z \in S^i} r^i_z \pi^i_z\). The set of all mean rewards is denoted by \(\mu = \{\mu^i\}\).

At each decision period \(n\) (also referred to interchangeably as time slot), an \(N\)-dimensional action vector \(a(n)\), representing an arm, is selected under a policy \(\phi(n)\) from a finite set \(\mathcal{F}\). We assume \(a_i(n) \geq 0\) for all \(1 \leq i \leq N\). When a particular \(a(n)\) is selected, the value of \(r^i_{x_i(n)}\) is observed, only for those \(i\) with \(a_i(n) \neq 0\). We denote by \(\mathcal{A}_{a(n)} = \{i : a_i(n) \neq 0, 1 \leq i \leq N\}\) the index set of all \(a_i(n) \neq 0\) for an arm \(a\). We treat each \(a(n) \in \mathcal{F}\) as an arm. The reward is defined as:

\[
R^{a(n)}(n) = \sum_{i \in \mathcal{A}_{a(n)}} a_i(n)r^i_{x_i(n)}
\]

(6.1)

where \(x_i(n)\) denotes the state of a Markov chain \(i\) at time \(n\).

When a particular arm \(a(n)\) is selected, the rewards corresponding to non-zero components of \(a(n)\) are revealed, i.e., the value of \(r^i_{x_i(n)}\) is observed for all \(i\) such that \(a_i(n) \neq 0\).

\(^2\)We also refer Markov chain \(\{X^i(n), n \geq 0\}\) and Markov chain \(i\) interchangeably.

\(^3\)Alternatively, for Markov chain \(\{X^i(n), n \geq 0\}\), it suffices to assume that the multiplicative symmetrization of the transition probability matrix is irreducible.
The state of the Markov chain evolves *restlessly*, i.e., the state will continue to evolve independently of the actions. We denote by \( P_i = (p_{x,y}^i)_{x,y \in S^i} \) the transition probability matrix for the Markov chain \( i \). We denote by \( (P_i)' = \{(p_i')_{x,y}\}_{x,y \in S^i} \) the adjoint of \( P_i \) on \( l_2(\pi) \), so \( (p_i')_{x,y} = p_{y,x}^i \pi_y^i / \pi_x^i \). Denote \( \hat{P}_i = (P_i)' P \) as the *multiplicative symmetrization* of \( P_i \). For aperiodic irreducible Markov chains, \( \hat{P}_i \)'s are irreducible [28].

A key metric of interest in evaluating a given policy \( \phi \) for this problem is *regret*, which is defined as the difference between the expected reward that could be obtained by the best-possible static action, and that obtained by the given policy. It can be expressed as:

\[
\mathcal{R}^\phi(n) = n \gamma^* - \mathbb{E}^\phi \left[ \sum_{t=1}^{n} R^\phi(t) \right] \\
= n \gamma^* - \mathbb{E}^\phi \left[ \sum_{t=1}^{n} \sum_{i \in A(t)} a_i(t) r_i^x(x_i(t)) \right]
\]

(6.2)

where \( \gamma^* = \max_{a \in A} \sum_{i \in A(a)} a_i \mu_i \) is the expected reward of the optimal arm. For the rest of the chapter, we use \( * \) as the index indicating that a parameter is for an optimal arm. If there is more than one optimal arm, \( * \) refers to any one of them. We denote by \( \gamma^a \) the expected reward of arm \( a \), so \( \gamma^a = \sum_{j=1}^{\left| A_a \right|} a_{p_j} \mu_{p_j} \).

For this combinatorial multi-armed bandit problem with restless Markovian rewards, our goal is to design policies that perform well with respect to regret. Intuitively, we would like the regret \( \mathcal{R}^\phi(n) \) to be as small as possible. If it is sublinear with respect to time \( n \), the time-averaged regret will tend to zero.
6.3 Policy Design

For the above combinatorial MAB problem with restless rewards, we have two challenges here for the policy design:

(1) A straightforward idea is to apply RCA in [75], or RUCB in [57] directly and naively, and ignore the dependencies across the different arms. However, we note that RCA and RUCB both require the storage and computation time that are linear in the number of arms. Since there could be exponentially many arms in this formulation, it is highly unsatisfactory.

(2) Unlike our prior work on combinatorial MAB with rested rewards, for which the transitions only occur each time the Markov chains are observed, the policy design for the restless case is much more difficult, since the current state while starting to play a Markov chain depends not only on the transition probabilities, but also on the policy.

To deal with the first challenge, we want to design a policy which more efficiently stores observations from the correlated arms, and exploits the correlations to make better decisions. Instead of storing the information for each arm, our idea is to use two 1 by \( N \) vectors to store the information for each Markov chain. Then an index for each each arm is calculated, based on the information stored for underlying components. This index is used for choosing the arm to be played each time when a decision needs to be made.

To deal with the second challenge, for each arm \( a \) we note that the multidimensional Markov chain \( \{X^a(n), n \geq 0\} \) defined by underlying components as \( X^a(n) = (X^i(n))_{i \in A_a} \) is aperiodic and irreducible. Instead of utilizing the actual sample path of
Table 6.1: Notation for Algorithm 7.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>number of resources</td>
</tr>
<tr>
<td>$a$</td>
<td>vectors of coefficients, defined on set $\mathcal{F}$; we map each $a$ as an arm</td>
</tr>
<tr>
<td>$A_a$</td>
<td>${ i : a_i \neq 0, 1 \leq i \leq N }$</td>
</tr>
<tr>
<td>$t$</td>
<td>current time slot</td>
</tr>
<tr>
<td>$t_2$</td>
<td>number of time slots in SB2 up to the current time slot</td>
</tr>
<tr>
<td>$b$</td>
<td>number of blocks up to the current time slot</td>
</tr>
<tr>
<td>$m_i^t$</td>
<td>number of times that Markov chain $i$ has been observed during SB2 up to the current time slot</td>
</tr>
<tr>
<td>$\bar{z}_i^t$</td>
<td>average (sample mean) of all the observed values of Markov chain $i$ during SB2 up to the current time slot</td>
</tr>
<tr>
<td>$\zeta^i$</td>
<td>state that determine the regenerative cycles for Markov chain $i$</td>
</tr>
<tr>
<td>$x_i$</td>
<td>the observed state when Markov Chain $i$ is played; $(x_i)_{i \in A_a}$ is the observed state vector if arm $a$ is played</td>
</tr>
</tbody>
</table>

\[ (x_i)_{i \in A_a} = (\zeta^i)_{i \in A_a} \text{ means } x_i = \zeta^i, \forall i. \]

CLRMR operates in blocks. Figure 6.1 illustrates one possible realization of this Algorithm 7. At the beginning of each block, an arm $a$ is picked and within one block, this algorithm always play the same arm. For each Markov chain $\{X^i(n)\}$, we specify a state $\zeta^i$ at the beginning of the algorithm as a state to mark the regenerative cycle. Then, for the multidimensional Markov chain $\{X^a(n)\}$ associated with this arm, the state $(\zeta^i)_{i \in A_a}$ is used to define a regenerative cycle for $\{X^a(n)\}$. All observations, we only take the observations from a regenerative cycle for Markov chains and discard the rest in its estimation of the index.

Our proposed policy, which we refer to as Combinatorial Learning with Restless Markov Reward (CLRMR), is shown in Algorithm 7. Table 6.1 summerizes the notation we use for CLRMR algorithm. For Algorithm 7, $(x_i)_{i \in A_a} = (\zeta^i)_{i \in A_a}$ means $x_i = \zeta^i, \forall i.$
Algorithm 7 Combinatorial Learning with Restless Markov Reward (CLRMR)

1: // INITIALIZATION
2: $t = 1$, $t_2 = 1$; $\forall i = 1, \ldots, N$, $m_i^1 = 0$, $\bar{z}_i^2 = 0$;
3: for $b = 1$ to $N$ do
4: $t := t + 1$, $t_2 := t_2 + 1$; Play any arm $a$ such that $b \in A_a$; denote $(x_i)_{i \in A_a}$ as the observed state vector for arm $a$;
5: $\forall i \in A_{a(n)}$, let $\zeta^i$ be the first state observed for Markov chain $i$ if $\zeta^i$ has never been set; $\bar{z}_2^i := \frac{z_1^i m_2^i + r_i^1}{m_2^i + 1}$, $m_2^i := m_2^i + 1$;
6: while $(x_i)_{i \in A_a} \neq (\zeta^i)_{i \in A_a}$ do
7: $t := t + 1$, $t_2 := t_2 + 1$; Play arm $a$; denote $(x_i)_{i \in A_a}$ as the observed state vector;
8: $\forall i \in A_{a(n)}$, $\bar{z}_2^i := \frac{z_1^i m_2^i + r_i^1}{m_2^i + 1}$, $m_2^i := m_2^i + 1$;
9: end while
10: end for
11: // MAIN LOOP
12: while 1 do
13: // SB1 STARTS
14: $t := t + 1$; Play an arm $a$ which maximizes
15: $\max_{a \in A} \sum_{i \in A_a} a_i \left( \bar{z}_2^i + \sqrt{L \ln t_2 / m_2^i} \right)$; (6.3)
16: where $L$ is a constant.
17: Denote $(x_i)_{i \in A_a}$ as the observed state vector;
18: while $(x_i)_{i \in A_a} \neq (\zeta^i)_{i \in A_a}$ do
19: $t := t + 1$, $t_2 := t_2 + 1$; Play arm $a$ and denote $(x_i)_{i \in A_a}$ as the observed state vector;
20: end while
21: // SB2 STARTS
22: $t_2 := t_2 + 1$;
23: $\forall i \in A_{a(n)}$, $\bar{z}_2^i := \frac{z_1^i m_2^i + r_i^1}{m_2^i + 1}$, $m_2^i := m_2^i + 1$;
24: while $(x_i)_{i \in A_a} \neq (\zeta^i)_{i \in A_a}$ do
25: $t := t + 1$, $t_2 := t_2 + 1$;
26: Play an arm $a$ and denote $(x_i)_{i \in A_a}$ as the observed state vector;
27: end while
28: // SB3 IS THE LAST PLAY IN THE WHILE LOOP. THEN A BLOCK COMPLETES.
29: $b := b + 1$, $t := t + 1$;
30: end while
Each block is broken into three sub-blocks denoted by SB1, SB2 and SB3. In SB1, the selected arm $a$ is played until the state $(\zeta_i^a)_{i \in A_a}$ is observed. Upon this observation we enter a regenerative cycle, and continue playing the same arm until $(\zeta_i^a)_{i \in A_a}$ is observed again. SB2 includes all time slots from the first visit of $(\zeta_i^a)_{i \in A_a}$ up to but excluding the second visit to $(\zeta_i^a)_{i \in A_a}$. SB3 consists a single time slot with the second visit to $(\zeta_i^a)_{i \in A_a}$. SB1 is empty if the first observed state is $(\zeta_i^a)_{i \in A_a}$. So SB2 includes the observed rewards for a regenerative cycle of the multidimensional Markov chain $\{X^a(n)\}$ associated with arm $a$, which implies that SB2 also includes the observed rewards for one or more regenerative cycles for each underlying Markov chain $\{X^i(n)\}, i \in A_a$.

The key to the algorithm 7 is to store the observations for each Markov chain instead of the whole arm, and utilize the observations only in SB2 for them, and virtually assemble them (highlighted with thick lines in Figure 6.1). Due to the regenerative nature of the Markov chain, by putting the observations in SB2, the sample path has exactly the same statics as given by the transition probability matrix. So the problem is tractable.

LLR policy requires storage linear in $N$. We use two 1 by $N$ vectors to store the information for each Markov chain after we play the selected arm at each time slot in SB2. One is $(\bar{z}_2^i)_{1 \times N}$ in which $\bar{z}_2^i$ is the average (sample mean) of observed values in
SB2 up to the current time slot (obtained through potentially different sets of arms over
time). The other one is \((m^1_2)_{1 \times N}\) in which \(m^1_2\) is the number of times that \(\{X^i(n)\}\) has
been observed in SB2 up to the current time slot.

Line 1 to line 10 are the initialization, for which each Markov chain is observed at
least once, and \(\zeta^i\) is specified as the first state observed for \(\{X^i(n)\}\).

After the initialization, at the beginning of each block, CLRMR selects the arm which
solves the maximization problem as in (6.3). It is a deterministic linear optimal problem
with a feasible set \(\mathcal{F}\) and the computation time for an arbitrary \(\mathcal{F}\) may not be polynomial
in \(N\). But, as we show in Section 6.5, there exist many practically useful examples with
polynomial computation time.

\section{Analysis of Regret}

We summarize some notation we use in the description and analysis of our CLRMR
policy in Table 6.2.

We first show in Theorem 7 an upper bound on the total expected number of plays of
suboptimal arms.

\textbf{Theorem 7.} When using any constant \(L \geq 56(H + 1)S^2_{\max}r^2_{\max}\tilde{a}^2_{\max}/\epsilon_{\min}\), we have

\[
\sum_{a: \gamma^a < \gamma^*} (\gamma^* - \gamma^a)\mathbb{E}[T^a(n)] \leq Z_1 \ln n + Z_2
\]
$H \triangleq \max_a |A_a|$. Note that $H \leq N$

$a(\tau) \triangleq$ the arm played in time $\tau$

$b(n) \triangleq$ number of completed blocks up to time $n$

$t(b) \triangleq$ time at the end of block $b$

$t_2(b) \triangleq$ total number of time slots spent in SB2 up to block $b$

$B^a(b) \triangleq$ total number of blocks within the first $b$ blocks in which arm $a$ is played

$m_2^i(t_2(b)) \triangleq$ total number of time slots Markov chain $i$ is observed during SB2 up to block $b$

$z^i_{t_2}(s) \triangleq$ the mean reward from Markov chain $i$ when it is observed for the $s$-th time of only those times played during SB2

$T(n) \triangleq$ time at the end of the last completed block

$T^a(n) \triangleq$ total number of time slots arm $a$ is played up to time $T(n)$

$m_i(s) \triangleq$ number of times that state $x$ occurred when Markov chain $i$ has been observed $s$ times

$Y_1^i(j) \triangleq$ vector of observed states from SB1 of the $j$-th block for playing Markov chain $i$

$Y_2^i(j) \triangleq$ vector of observed states from SB2 of the $j$-th block for playing Markov chain $i$

$Y^i(j) \triangleq$ vector of observed states from the $j$-th block for playing Markov chain $i$

$\hat{\pi}_i^x \triangleq \max \{\pi_i^x, 1 - \pi_i^x\}$

$\hat{\pi}_\max \triangleq \max_{i, x \in S^i} \hat{\pi}_i^x$

$\pi_\min \triangleq \min_{i, x \in S^i} \pi_i^x$

$\pi_\max \triangleq \max_{i, x \in S^i} \pi_i^x$

$\epsilon \triangleq$ eigenvalue gap, defined as $1 - \lambda_2$, where $\lambda_2$ is the second largest eigenvalue of the multiplicative symmetrization of $P^i$

$\epsilon_\min \triangleq \min_i \epsilon^i$

$S_\max \triangleq \max_{i} |S^i|$

$r_\max \triangleq \max_{i, x \in S^i} r_i^x$

$a_\max \triangleq \max_{i \in A_a, a \in F^i} a_i$

$\Delta_a \triangleq \gamma^a - \gamma^a$

$\Delta_{\min} \triangleq \min_{\gamma^a \leq \gamma^a} \Delta_a$
\[ \Delta_{\text{max}} = \max_{\gamma^a \leq \gamma^*} \Delta_a \]
\[ \{X^a(n)\}: \text{multidimensional Markov chain defined} \]
\[ \text{by } X^a(n) = (X^i(n))_{i \in A_a} \]
\[ \zeta^a:\quad (\zeta^i)_{i \in A_a}, \text{state vector that determines} \]
\[ \text{the regenerative cycles for } \{X^a(n)\} \]
\[ \Pi^a_z:\quad \text{steady state distribution for state } z \text{ of } \{X^a(n)\} \]
\[ \Pi^a_{\text{min}}:\quad \min_{z \in S_a} \Pi^a_z \]
\[ \Pi^a_{\text{min}}:\quad \min_{a, z \in S_a} \Pi^a_z \]
\[ M^a_{z_1, z_2}: \text{mean hitting time of state } z_2 \text{ starting} \]
\[ \text{from an initial state } z_1 \text{ for } \{X^a(n)\} \]
\[ M^a_{\text{max}}: \max_{z_1, z_2 \in S_a} M^a_{z_1, z_2} \]
\[ \gamma_{\text{max}}': \max_{\gamma^a \leq \gamma^*} \gamma^a \]

Table 6.2: Notation for regret analysis.

where

\[ Z_1 = \Delta_{\text{max}} \left( \frac{1}{\Pi_{\text{min}}} + M_{\text{max}} + 1 \right) \frac{4NLH^2a^2_{\text{max}}}{\Delta_{\text{min}}^2} \]
\[ Z_2 = \Delta_{\text{max}} \left( \frac{1}{\Pi_{\text{min}}} + M_{\text{max}} + 1 \right) \left( N + \frac{\pi NH S_{\text{max}}}{3\pi_{\text{min}}} \right) \]

To prove Theorem 7, we use the inequalities as stated in Theorem 3.3 from [53] and a theorem from [20].

**Lemma 6** (Theorem 3.3 from [53]). Consider a finite-state, irreducible Markov chain \(\{X_t\}_{t \geq 1}\) with state space \(S\), matrix of transition probabilities \(P\), an initial distribution \(q\) and stationary distribution \(\pi\). Let \(N_q = \left\| \left( \frac{q_x}{\pi_x}, x \in S \right) \right\|_2\). Let \(\hat{P} = P'P\) be the multiplicative symmetrization of \(P\) where \(P'\) is the adjoint of \(P\) on \(l_2(\pi)\). Let \(\epsilon = 1 - \lambda_2\), where \(\lambda_2\) is the second largest eigenvalue of the matrix \(\hat{P}\). \(\epsilon\) will be referred to as the
eigenvalue gap of $\hat{P}$. Let $f: S \to \mathcal{R}$ be such that $\sum_{y \in S} \pi_y f(y) = 0$, $\|f\|_\infty \leq 1$ and $0 < \|f\|_2^2 \leq 1$. If $\hat{P}$ is irreducible, then for any positive integer $n$ and all $0 < \delta \leq 1$

$$
P \left( \frac{\sum_{t=1}^n f(X_t)}{n} \geq \delta \right) \leq N_q \exp \left[ -\frac{n\delta^2}{28} \right].$$

**Lemma 7.** If $\{X_n\}_{n \geq 0}$ is a positive recurrent homogeneous Markov chain with state space $S$, stationary distribution $\pi$ and $\tau$ is a stopping time that is finite almost surely for which $X_\tau = x$ then for all $y \in S$,

$$E \left[ \sum_{t=0}^{\tau-1} I(X_t = y) | X_0 = x \right] = E[\tau | X_0 = x] \pi_y.$$

**Proof of Theorem 7.** We introduce $\tilde{B}^i(b)$ as a counter for the regret analysis to deal with the combinatorial arms. After the initialization period, $\tilde{B}^i(b)$ is updated in the following way: at the beginning of any block when a non-optimal arm is chosen to be played, find $i$ such that $i = \arg \min_{j \in A_u(b)} m_2^j$ (i the index of the elements which are among the ones that have been observed least in SB2 in the non-optimal arm). If there is only one such arm, $\tilde{B}^i(b)$ is increased by 1. If there are multiple such arms, we arbitrarily pick one, say $i'$, and increment $\tilde{B}^{i'}$ by 1. Based on the above definition of $\tilde{B}^i(b)$, each time a non-optimal arm is chosen to be played at the beginning of a block, exactly one element in $(\tilde{B}^i(b))_{1 \times N}$ is incremented by 1. So the summation of all counters in $(\tilde{B}^i(b))_{1 \times N}$ equals the total number of blocks in which we have played non-optimal arms,
\[
\sum_{a : \gamma^a < \gamma^*} E[B^a(b)] = \sum_{i=1}^{N} E[\tilde{B}^i(b)]. \tag{6.4}
\]

We also have the following inequality for \(\tilde{B}^i(b)\):

\[
\tilde{B}^i(b) \leq m_i^2(t(b - 1)), \forall 1 \leq i \leq N, \forall b. \tag{6.5}
\]

Denote by \(c_{t,s} \sqrt{\frac{L \ln t}{s}}\). Denote by \(\tilde{I}^i(b)\) the indicator function which is equal to 1 if \(\tilde{B}^i(b)\) is added by one at block \(b\). Let \(l\) be an arbitrary positive integer. Then we can get the upper bound of \(E[\tilde{B}^i(b)]\) shown in (6.6),

\[
E[\tilde{B}^i(b)] = \sum_{\beta=N+1}^{b} \mathbb{P}\{\tilde{I}^i(\beta) = 1\} \leq l + \sum_{\beta=N+1}^{b} \mathbb{P}\{\tilde{I}^i(\beta) = 1, \tilde{B}^i(\beta - 1) \geq l\}
\leq l + \sum_{\beta=N+1}^{b} \mathbb{P}\{\sum_{k \in A_{a^*}} a_k^* g_{t_2(\beta-1),m_i^2(t(\beta-1))}^k \}
\leq \sum_{j \in A_{a(\beta)}} a_j^i(b) g_{t_2(\beta-1),m_i^2(t(\beta-1))}^j, \tilde{B}^i(\beta - 1) \geq l\}. \tag{6.6}
\]

where \(g_{t,s}^i = \varepsilon_2^i(s) + c_{t,s}\) and \(a(\beta)\) is defined as a non-optimal arm picked at block \(\beta\) when \(\tilde{I}^i(\beta) = 1\). Note that \(m_i^1 = \min_j \{m_j^1: \forall j \in A_{a(\beta)}\}\). We denote this arm by \(a(\beta)\) since at each block that \(\tilde{I}^i(\beta) = 1\), we could get different arms.

Note that \(l \leq \tilde{B}^i(\beta - 1)\) implies,

\[
l \leq \tilde{B}^i(\beta - 1) \leq m_i^2(t(\beta - 1)), \forall j \in A_{a(\beta)}\}. \tag{6.7}
\]
So we can further derive the upper bound of $E[\tilde{B}_i(b)]$ shown in (6.8), where $h_j$ ($1 \leq j \leq |A_{a*}|$) represents the $j$-th element in $A_{a*}$; $p_j$ ($1 \leq j \leq |A_{a(\beta)}|$) represents the $j$-th element in $A_{a(\beta)}$ or $A_{a(t)}$. $A_{a(\tau)}$ represents the arm played in the $\tau$-th time slots counting only in SB2.

$$E[\tilde{B}_i(b)] \leq l + \sum_{\beta = N+1}^{b} \mathbb{P}\left\{ \min_{0 < s_{h_1}, \ldots, s_{h_1} | A_{a*} | < t_2(\beta - 1)} \sum_{j=1}^{|A_{a*}|} a_{h_j}^* g_{t_2(\beta - 1), s_{h_j}} \right\}$$

$$\leq \sum_{\beta = N+1}^{b} \max_{t_2(l) \leq s_{p_1}, \ldots, s_{p_1} | A_{a(\beta)} | < t_2(\beta - 1)} \sum_{j=1}^{|A_{a(\beta)}|} a_{p_j}(\beta) g_{t_2(\beta - 1), s_{p_j}}$$

$$\leq l + \sum_{\beta = N+1}^{b} \sum_{s_{h_1} = 1}^{t_2(\beta - 1)} \sum_{s_{h_1} | A_{a*} | = 1}^{t_2(\beta - 1)} \sum_{s_{p_1} = t_2(l)}^{t_2(\beta - 1)} \sum_{s_{p_1} | A_{a(\beta)} | = l}^{t_2(l)}$$

$$\mathbb{P}\left\{ \sum_{j=1}^{|A_{a*}|} a_{h_j}^* g_{t_2(\beta - 1), s_{h_j}} \leq \sum_{j=1}^{|A_{a(\beta)}|} a_{p_j}(\beta) g_{t_2(\beta - 1), s_{p_j}} \right\}$$

$$\leq l + \sum_{\tau=1}^{t_2(b)} \sum_{s_{h_1} = 1}^{\tau - 1} \sum_{s_{h_1} | A_{a*} | = 1}^{\tau - 1} \sum_{s_{p_1} = t}^{\tau - 1} \sum_{s_{p_1} | A_{a(\beta)} | = l}^{\tau - 1}$$

$$\mathbb{P}\left\{ \sum_{j=1}^{|A_{a*}|} a_{h_j}^* g_{t_2, s_{h_j}} \leq \sum_{j=1}^{|A_{a(\tau)}|} a_{p_j}(\tau) g_{t_2, s_{p_j}} \right\}$$

(6.8)

Note that,

$$\mathbb{P}\left\{ \sum_{j=1}^{|A_{a*}|} a_{h_j}^* g_{t_2, s_{h_j}} \leq \sum_{j=1}^{|A_{a(\tau)}|} a_{p_j}(\tau) g_{t_2, s_{p_j}} \right\}$$

$$= \mathbb{P}\left\{ \sum_{j=1}^{|A_{a*}|} a_{h_j}^* (\bar{z}_2(s_{h_j}) + c_{\tau, s_{h_j}}) \leq \sum_{j=1}^{|A_{a(\tau)}|} a_{p_j}(\tau) (\bar{z}_2(s_{p_j}) + c_{\tau, s_{p_j}}) \right\}$$

(6.9)
\( \sum_{j=1}^{\left| A_{a^*} \right|} a^*_h \bar{z}^j_h (s_{h_j}) \leq \gamma^* - \sum_{j=1}^{\left| A_{a^*} \right|} a^*_h c_{r, s_{h_j}}, \quad (6.10) \)

\( \sum_{j=1}^{\left| A_{a^{(r)}} \right|} a_{p_j} (\tau) \bar{z}_{p_j}^j (s_{p_j}) \geq \gamma^{a(\tau)} + \sum_{j=1}^{\left| A_{a^{(r)}} \right|} a_{p_j} (\tau) c_{r, s_{p_j}}, \quad (6.11) \)

\( \gamma^* < \gamma^{a(\tau)} + 2 \sum_{j=1}^{\left| A_{a^{(r)}} \right|} a_{p_j} (\tau) c_{r, s_{p_j}} \quad (6.12) \)

Now we show the upper bound on the probabilities of inequalities (6.10), (6.11) and (6.12) separately. We first find an upper bound on the probability of (6.10):

\[
\Pr\left\{ \sum_{j=1}^{\left| A_{a^*} \right|} a^*_h \bar{z}^j_h (s_{h_j}) \leq \gamma^* - \sum_{j=1}^{\left| A_{a^*} \right|} a^*_h c_{r, s_{h_j}} \right\}
\]

\[
= \Pr\left\{ \sum_{j=1}^{\left| A_{a^*} \right|} a^*_h \bar{z}^j_h (s_{h_j}) \leq \sum_{j=1}^{\left| A_{a^*} \right|} a^*_h \mu^j - \sum_{j=1}^{\left| A_{a^*} \right|} a^*_h c_{r, s_{h_j}} \right\}
\]

\[
\leq \sum_{j=1}^{\left| A_{a^*} \right|} \Pr\left\{ a^*_h \bar{z}^j_h (s_{h_j}) \leq a^*_h (\mu^j - c_{r, s_{h_j}}) \right\}
\]

\[
= \sum_{j=1}^{\left| A_{a^*} \right|} \Pr\left\{ \bar{z}^j_h (s_{h_j}) \leq \mu^j - c_{r, s_{h_j}} \right\}.
\]

\( \forall 1 \leq j \leq \left| A_{a^*} \right|, \)

\[
\Pr\left\{ \bar{z}^j_h (s_{h_j}) \leq \mu^j - c_{r, s_{h_j}} \right\}
\]

\[
= \Pr\left\{ \sum_{x \in S_{h_j}^{h_j}} \left( \frac{r^j_x m^j_x (s_{h_j})}{s_{h_j}} - r^j_x \pi^j_x \right) \leq \sum_{x \in S_{h_j}^{h_j}} \left( \frac{c_{r, s_{h_j}}}{|S_{h_j}|} \right) \right\}
\]

\[
\leq \sum_{x \in S_{h_j}^{h_j}} \Pr\left\{ \frac{r^j_x m^j_x (s_{h_j})}{s_{h_j}} - r^j_x \pi^j_x \leq - \frac{c_{r, s_{h_j}}}{|S_{h_j}|} \right\}
\]

105
\[
\begin{align*}
&= \sum_{x \in S_{hj}} \mathbb{P}\{r_x^{h_j} m_x^{h_j}(sh_j) - sh_j r_x^{h_j} \pi_x^{h_j} \leq -\frac{sh_j c_r}{|S_{hj}|}\} \\
&= \sum_{x \in S_{hj}} \mathbb{P}\{r_x^{h_j}(sh_j - \sum_{y \neq x} m_y^{h_j}(sh_j)) - r_x^{h_j} sh_j (1 - \sum_{y \neq x} \pi_y^{h_j}) \leq -\frac{sh_j c_r}{|S_{hj}|}\} \\
&= \sum_{x \in S_{hj}} \mathbb{P}\{\sum_{y \neq x} m_y^{h_j}(sh_j) - \sum_{y \neq x} \pi_y^{h_j} \geq \frac{sh_j c_r}{r_x^{h_j} |S_{hj}|}\} \\
&= \sum_{x \in S_{hj}} \mathbb{P}\{\sum_{t=1}^{s_{h_j}} 1(Y_t^{h_j} \neq x) - sh_j (1 - \pi_x^{h_j}) \geq \frac{sh_j c_r}{r_x^{h_j} |S_{hj}|}\} \\
&\leq \sum_{x \in S_{hj}} N_{q_x^{h_j}} \tau^{-\frac{l_x^{h_j}}{28(|S_{hj}| r_x^{h_j})^2}} \quad \text{(6.13)} \\
&\leq \frac{|S_{hj}|}{\tau} \pi_{\min}^{-\frac{l_x^{h_j}}{28 |S_{hj}|^2 r_x^{h_j}}} \quad \text{(6.14)}
\end{align*}
\]

where (6.13) follows from Lemma 6 by letting

\[
\delta = \frac{sh_j c_r}{r_x^{h_j} |S_{hj}|}, \quad f(Y_t^i) = \frac{1(Y_t^i \neq x) - (1 - \pi_x^i)}{\hat{\pi}_x^i}.
\]

\(1(a)\) is the indicator function defined to be 1 when the predicate \(a\) is true, and 0 when it is false. \(\hat{\pi}_x^i\) is defined as \(\hat{\pi}_x^i = \max\{\pi_x^i, 1 - \pi_x^i\}\) to guarantee \(\|f\|_\infty \leq 1\). We note that when \(\delta > 1\) the deviation probability is zero, so the bound still holds.

(6.14) follows from the fact that for any \(q_x^i\),

\[
N_{q_x^i} = \left\| \frac{q_x^i}{\pi_x^i}, x \in S^i \right\|_2 \leq \sum_{x=1}^{|S^i|} \left\| \frac{q_x^i}{\pi_x^i} \right\|_2 \leq \sum_{x=1}^{|S^i|} \|q_x^i\|_2 \pi_{\min} = \frac{1}{\pi_{\min}}.
\]

106
Note that all the quantities in computing the indices and the probabilities above come from SB2. Got for every SB2 in a block, the quantities begin with state $\zeta^a$ and end with a return to $\zeta^a$. So for each underlying Markov chain $\{X^i(n)\}$, $i \in A_a$, the quantities got begin with state $\zeta^i$ and end with a return to $\zeta^i$. Note that for all $i$, Markov chain $\{X^i(n)\}$ could be played in different arms, but the quantities got always begin with state $\zeta^i$ and end with a return to $\zeta^i$. Then by the strong Markov property, the process at these stopping times has the same distribution as the original process. Connecting these intervals together we form a continuous sample path which can be viewed as a sample path generated by a Markov chain with transition matrix identical to the original arm. This is the reason why we can apply Lemma 6 to this Markov chain.

Therefore,

$$\mathbb{P}\left\{ \sum_{j=1}^{\left|A_{a^*}\right|} a^*_{h_j} z^2_{h_j} (s_{h_j}) \leq \gamma^* - \sum_{j=1}^{\left|A_{a^*}\right|} a^*_{h_j} c_{\tau,s_{h_j}} \right\} \leq \frac{HS_{\max}}{\pi_{\min}} - \frac{\lambda_{\min}}{2S_{\max}^2 \pi_{\min}^2 S_{\max}^2}$$

(6.15)

With a similar derivation, we have

$$\mathbb{P}\left\{ \sum_{j=1}^{\left|A_{a^*}\right|} a_{p_j} (\tau) z^2_{p_j} (s_{p_j}) \geq \gamma^{a(\tau)} + \sum_{j=1}^{\left|A_{a^*}\right|} a_{p_j} (\tau) c_{\tau,s_{p_j}} \right\} \leq \sum_{j=1}^{\left|A_{a^*}\right|} \mathbb{P}\left\{ a_{p_j} (\tau) z^2_{p_j} (s_{p_j}) \geq a_{p_j} (\tau) \mu_{p_j} + a_{p_j} (\tau) c_{\tau,s_{p_j}} \right\} \leq \sum_{j=1}^{\left|A_{a^*}\right|} \sum_{x \in S_{p_j}} \mathbb{P}\left\{ r_{p_j} m_{x} (s_{p_j}) - s_{p_j} r_{p_j} \pi_{x} \geq \frac{S_{p_j} c_{\tau,s_{p_j}}}{|S_{p_j}|} \right\}$$
\[
\begin{align*}
\sum_{j=1}^{s_{pj}} \sum_{x \in S_{pj}} P\left\{ \frac{1}{\hat{\pi}_x^{pj} S_{pj}} \sum_{t=1}^{s_{pj}} 1(Y_{jt}^i = x) - \pi_x^{pj} \geq s_{pj} C_{\tau,spj} \right\} \\
\leq \sum_{j=1}^{s_{pj}} \sum_{x \in S_{pj}} N_{pj}^{pj} \tau - \frac{L_{\pi}^{pj}}{28(|S_{pj}|^2 + \pi_{x}^{pj})^2}
\end{align*}
\] (6.16)

\[
\leq \frac{H_{S_{max}}}{\pi_{min}} \tau - \frac{L_{\pi}^{min}}{28S_{max}^2 \pi_{max}^2}
\] (6.17)

where (6.16) follows from Lemma 6 by letting

\[
\delta = \frac{s_{pj} C_{\tau,spj}}{1_x^{pj}|S_{pj}|}, \quad f(Y_{jt}^i) = 1(Y_{jt}^i = x) - \pi_x^{pj} \hat{\pi}_x^{pj} S_{pj}
\]

Note that when \( l \geq \left\lceil \frac{4L \ln t_2(b)}{(\frac{\Delta_{a(\tau)}}{\pi_{max}})^2} \right\rceil \), (6.12) is false for \( \tau \), which gives,

\[
\gamma^* - \gamma_{a(\tau)} - 2 \sum_{j=1}^{s_{pj}} \frac{1}{s_{pj}} \sqrt{L \pi^{pj} \ln t_2(b)}
\]

\[
\geq \gamma^* - \gamma_{a(\tau)} - H_{a_{max}} \sqrt{4L \pi^{pj} \ln t_2(b)}
\]

\[
\geq \gamma^* - \gamma_{a(\tau)} - H_{a_{max}} \sqrt{\frac{4L \ln t_2(b)}{4L \ln t_2(b)} \left( \frac{\Delta_{a(\tau)}}{H_{a_{max}}} \right)^2}
\]

\[
\geq \gamma^* - \gamma_{a(\tau)} - \Delta_{a(\tau)} = 0.
\] (6.19)

Hence, when we let \( l \geq \left\lceil \frac{4LH_{a_{max}}^2 \ln t_2(b)}{\Delta_{a_{min}}^2} \right\rceil \), (6.12) is false for all \( a(\tau) \). Therefore, we have (6.20).
\[ E[\tilde{B}^i(b)] \leq \left[ \frac{4LH^2a^2_{\max} \ln t_2(b)}{\Delta^2_{\min}} \right] \\
+ \sum_{\tau=1}^{t_2(b)} \sum_{\tau-1}^{s_{h_1}=1} \cdots \sum_{\tau-1}^{s_{h_1,A^1}=1} \sum_{\tau-1}^{s_{p_1}=1} \cdots \sum_{\tau-1}^{s_{p_i,A^i(b)}=1} \frac{2HS_{\max}}{\pi_{\min}} \frac{L_{\min}}{28S_{\max}^2r_{\max}^2} \] (6.20)

Following (6.20),

\[ E[\tilde{B}^i(b)] \leq \frac{4LH^2a^2_{\max} \ln n}{\Delta^2_{\min}} + 1 + \frac{HS_{\max}}{\pi_{\min}} \sum_{\tau=1}^{\infty} 2\tau^{-2} \frac{L_{\min}}{28S_{\max}^2r_{\max}^2} \] (6.21)

\[ = \frac{4LH^2a^2_{\max} \ln n}{\Delta^2_{\min}} + 1 + \frac{HS_{\max}}{\pi_{\min}} \sum_{\tau=1}^{\infty} 2\tau^{-2} \] (6.22)

\[ = \frac{4LH^2a^2_{\max} \ln n}{\Delta^2_{\min}} + 1 + \frac{\pi HS_{\max}}{3\pi_{\min}} \]

(6.22) follows since \( L \geq 56(H + 1)S_{\max}^2r_{\max}^2\tilde{\pi}_{\max}^2/\epsilon_{\min} \). According to (6.4),

\[ \sum_{a: \gamma_a < \gamma^*} E[B^a(b)] = \sum_{i=1}^{N} E[\tilde{B}^i(b)] \leq \frac{4NLH^2a^2_{\max} \ln n}{\Delta^2_{\min}} + N + \frac{\pi NH S_{\max}}{3\pi_{\min}} \] (6.23)

Note that the total number of plays of arm \( a \) at the end of block \( b(n) \) is equal to the total number of plays of arm \( a \) during SB2s (the regenerative cycles of visiting state \( \zeta^a \)) plus the total number of plays before entering the regenerative cycles plus one more play resulting from the last play of the block which is state \( \zeta^a \). So we have,

\[ E[T^a(n)] \leq \left( \frac{1}{\Pi_{\min}^a} + M_{\max}^a + 1 \right) E[B^a(b(n))] \].

109
Therefore,

\[
\sum_{a: a < \gamma^*} (\gamma^* - \gamma^a) E[T^a(n)] \\
\leq \Delta_{\max} \sum_{a: a < \gamma^*} \left( \frac{1}{\Pi_{\min}} + M_{\max} + 1 \right) E[B^a(b(n))] \\
\leq \Delta_{\max} \left( \frac{1}{\Pi_{\min}} + M_{\max} + 1 \right) \sum_{a: a < \gamma^*} E[B^a(b(n))] \\
\leq Z_1 \ln n + Z_2
\]

where

\[
Z_1 = \Delta_{\max} \left( \frac{1}{\Pi_{\min}} + M_{\max} + 1 \right) \frac{4NLH^2 a_{\max}^2}{\Delta_{\min}^2}, \\
Z_2 = \Delta_{\max} \left( \frac{1}{\Pi_{\min}} + M_{\max} + 1 \right) \left( N + \frac{\pi N HS_{\max}}{3\pi_{\min}} \right)
\]
where

\[ Z_3 = Z_1 + Z_5 \frac{4NLH^2a_{\text{max}}^2}{\Delta_{\text{min}}^2} \]

\[ Z_4 = Z_2 + \gamma^* \left( \frac{1}{\pi_{\text{min}}} + M_{\text{max}} + 1 \right) + Z_5 (N + \frac{\pi N H S_{\text{max}}}{3\pi_{\text{min}}}) \]

and

\[ Z_5 = \gamma'_{\text{max}} \left( \frac{1}{\Pi_{\text{min}}} + M_{\text{max}} + 1 - \frac{1}{\pi_{\text{max}}} \right) + \gamma^* M_{\text{max}} \]

Proof. Denote the expectations with respect to policy CLRMR given \( \zeta \) by \( E_{\zeta} \). Then the regret is bounded as,

\[ R_{\text{CLRMR}}(n) = \gamma^* E_{\zeta} [T(n)] - E_{\zeta} \left[ \sum_{t=1}^{T(n)} \sum_{i \in A_{a(t)}} a_i(t) r_i x_i(t) \right] \]

\[ + \gamma^* E_{\zeta} [n - T(n)] - E_{\zeta} \left[ \sum_{t=T(n)+1}^{n} \sum_{i \in A_{a(t)}} a_i(t) r_i x_i(t) \right] \]

\[ \leq \left( \gamma^* E_{\zeta} [T(n)] - \sum_a \gamma^a E_{\zeta} [T^a(n)] \right) + \gamma^* E_{\zeta} [n - T(n)] \]

\[ + \sum_a \gamma^a E_{\zeta} [T^a(n)] - E_{\zeta} \left[ \sum_{t=1}^{T(n)} \sum_{i \in A_{a(t)}} a_i(t) r_i x_i(t) \right] \]

\[ \leq Z_1 \ln n + Z_2 + \gamma^* \left( \frac{1}{\Pi_{\text{min}}} + M_{\text{max}} + 1 \right) \]

\[ + \left( \sum_a \gamma^a E_{\zeta} [T^a(n)] - E_{\zeta} \left[ \sum_{t=1}^{T(n)} \sum_{i \in A_{a(t)}} a_i(t) r_i x_i(t) \right] \right) \tag{6.27} \]

where (6.27) follows from Theorem 7 and \( E_{\zeta} [n - T(n)] \leq \frac{1}{\Pi_{\text{min}}} + M_{\text{max}} + 1 \).
Note that:

\[
\sum_{a} \gamma^a \mathbb{E}_{\zeta}[T^a(n)] - \mathbb{E}_{\zeta}\left[ \sum_{t=1}^{T(n)} \sum_{i \in A_{a(t)}} a_i(t) r_i^x(t) \right] \\
\leq \gamma^* \mathbb{E}_{\zeta}[T^*(n)] + \sum_{a: \gamma^a < \gamma^*} \gamma^a \mathbb{E}_{\zeta}[T^a(n)] \\
- \sum_{i \in A_{a^*}, y \in S_i} a_i^* r_i^y \mathbb{E}_{\zeta}\left[ \sum_{j} \sum_{Y_i^j \in Y^i(j)} 1(Y_i^j = y) \right] \\
- \sum_{a: \gamma^a < \gamma^*} \sum_{i \in A_a} \sum_{y \in S_i} a_i r_i^y \mathbb{E}_{\zeta}\left[ \sum_{j} \sum_{Y_i^j \in Y^i_2(j)} 1(Y_i^j = y) \right]
\]

(6.28)

where the inequality above comes from counting only in \(Y^i_2(j)\) instead of \(Y^i(j)\) in (6.28).

Then applying Lemma 7 to (6.28), we have

\[
\mathbb{E}_{\zeta}\left[ \sum_{j} \sum_{Y_i^j \in Y^i_2(j)} 1(Y_i^j = y) \right] = \frac{\pi_i^y}{\pi_i^y} \mathbb{E}_{\zeta}[B^a(b(n))].
\]

So,

\[
- \sum_{a: \gamma^a < \gamma^*} \sum_{i \in A_a} \sum_{y \in S_i} a_i r_i^y \mathbb{E}_{\zeta}\left[ \sum_{j} \sum_{Y_i^j \in Y^i_2(j)} 1(Y_i^j = y) \right] \\
\leq - \sum_{a: \gamma^a < \gamma^*} \frac{\gamma^a}{\pi_{\max}} \mathbb{E}_{\zeta}[B^a(b(n))].
\]

(6.29)

Also note that:

\[
\sum_{a: \gamma^a < \gamma^*} \gamma^a \mathbb{E}_{\zeta}[T^a(n)] \leq \sum_{a: \gamma^a < \gamma^*} \gamma^a \left( \frac{1}{\pi_{\min}^a} + M_{\max}^a + 1 \right) \mathbb{E}_{\zeta}[B^a(b(n))].
\]

(6.30)
Inserting (6.29) and (6.30) into (6.28), we get,

\[
\sum_a \gamma^a \mathbb{E}_\xi[T^a(n)] - \mathbb{E}_\xi[\sum_{t=1}^{T(n)} \sum_{i \in A_{\alpha(t)}} a_i(t) r_{x_i(t)}^i]
\leq \gamma^* \mathbb{E}_\xi[T^*(n)] + \sum_{a : \gamma^a < \gamma^*} \gamma^a \left( \frac{1}{\Pi_{\min}^a} + M_{\max}^a + 1 - \frac{1}{\pi_{\max}} \right) \mathbb{E}_\xi[B^a(b(n))]
\]

\[- \sum_{i \in A_{\alpha}^*} \sum_{y \in S^i} a_i^* r_y^i \mathbb{E}_\xi[\sum_j \sum_{Y_t^i \in Y^i(j)} 1(Y_t^i = y)]
\]

\[= Q^*(n) + \sum_{a : \gamma^a < \gamma^*} \gamma^a \left( \frac{1}{\Pi_{\min}^a} + M_{\max}^a + 1 - \frac{1}{\pi_{\max}} \right) \mathbb{E}_\xi[B^a(b(n))],
\]

where

\[Q^*(n) = \gamma^* \mathbb{E}_\xi[T^*(n)] - \sum_{i \in A_{\alpha}^*} \sum_{y \in S^i} a_i^* r_y^i \mathbb{E}_\xi[\sum_j \sum_{Y_t^i \in Y^i(j)} 1(Y_t^i = y)].
\]

We now consider the upper bound for \(Q^*(n)\). We note that the total number of time slots for playing all suboptimal arms is at most logarithmic, so the number of time slots in which the optimal arm is not played is at most logarithmic. We could then combine the successive blocks in which the best arm is played, and denote by \(\tilde{Y}^*(j)\) the \(j\)-th combined block. Denote \(\tilde{b}^*\) as the total number of combined blocks up to block \(b\). Each combined block \(\tilde{Y}^*\) starts after dis-continuity in playing the optimal arm, so \(\tilde{b}^*(n)\) is less than or equal to total number of completed blocks in which the best arm is not played up to time \(n\). Thus,

\[
\mathbb{E}_\xi[\tilde{b}^*(n)] \leq \sum_{a : \gamma^a < \gamma^*} \mathbb{E}_\xi[B^a(b(n))].
\]
Each combined block $\bar{Y}^*$ consists of two sub-blocks: $\bar{Y}^*_1$ which contains the state vectors for the optimal arm visited from beginning of $\bar{Y}^*$ (empty if the first state is $\zeta^*$) to the state right before hitting $\zeta^*$ and sub-block $\bar{Y}^*_2$ which contains the rest of $\bar{Y}^*$ (a random number of regenerative cycles). Denote the length of $\bar{Y}^*_1$ by $|\bar{Y}^*_1|$ and the length of $\bar{Y}^*_2$ by $|\bar{Y}^*_2|$. We denote $\bar{Y}^*_2(j)$ by the states for Markov chain $i$ for all $i \in A_\ast$ in $\bar{Y}^*_2$.

Therefore we get the upper bound for $Q^*(n)$ as

$$Q^*(n) = \gamma^* \mathbb{E}_\zeta[T^*(n)] - \sum_{i \in A_\ast} \sum_{y \in S_i} a_i^* r_i^y \mathbb{E}_\zeta[\sum_{Y^i_t \in Y^i_t(j)} B^*(b(n)) \sum_{Y^i_t = y}]$$

$$\leq \sum_{i \in A_\ast} \sum_{y \in S_i} a_i^* r_i^y \mathbb{E}_\zeta[\sum_{j=1} B^*_i(n)]$$

$$- \sum_{i \in A_\ast} \sum_{y \in S_i} a_i^* r_i^y \mathbb{E}_\zeta[\sum_{j=1} Y^i_t(j) \sum_{Y^i_t = y}]$$

$$+ \sum_{i \in A_\ast} \sum_{y \in S_i} \gamma^* \mathbb{E}_\zeta[|\bar{Y}^*_1|]$$

$$\leq \gamma^* M_{\max}^* \sum_{a: \gamma^a < \gamma^*} \mathbb{E}_\zeta[B^a(b(n))]$$

where the inequality in (6.33) comes from counting only the rewards obtained in sub-block $\bar{Y}^*_2$ in (6.32). Also, note that based on Lemma 7, (6.33) equals (6.34), and therefore we have the inequality (6.36). Hence, $\forall \zeta$,

$$R_{CLRM}^\zeta(n) \leq Z_1 \ln n + Z_2 + \gamma^*(\frac{1}{\pi_{\min}} + M_{\max} + 1)$$

$$+ \sum_{a: \gamma^a < \gamma^*} \gamma^a (M_{\max}^a + 1) \mathbb{E}_\zeta[B^a(b(n))] + \gamma^* M_{\max}^* \sum_{a: \gamma^a < \gamma^*} \mathbb{E}_\zeta[B^a(b(n))]$$
\[ \leq Z_1 \ln n + Z_2 + \gamma^* \left( \frac{1}{\pi_{\min}} + M_{\max} + 1 \right) \]
\[ + (\gamma'_{\max} \left( \frac{1}{\Pi_{\min}} + M_{\max} + 1 - \frac{1}{\pi_{\max}} \right) + \gamma^* M^*_{\max}) \mathbb{E}_\xi[B^a(b(n))] \]
\[ \leq Z_3 \ln n + Z_4, \quad (6.37) \]

where (6.37) follows from Theorem 7 and (6.23), and

\[ Z_3 = Z_1 + Z_5 \frac{4NLH^2 a_{\max}^2}{\Delta_{\min}^2} \]
\[ Z_4 = Z_2 + \gamma^* \left( \frac{1}{\Pi_{\min}} + M_{\max} + 1 \right) + Z_5 (N + \frac{\pi N H S_{\max}}{3\pi_{\min}}). \]

\[ Z_5 \text{ is defined as} \]
\[ Z_5 = \gamma'_{\max} \left( \frac{1}{\Pi_{\min}} + M_{\max} + 1 - \frac{1}{\pi_{\max}} \right) + \gamma^* M^*_{\max}. \]

Theorem 8 shows when we use a constant \( L \geq 56(H + 1) S^2_{\max} r^{2}_{\max} \hat{\pi}_{\max}^2 / \epsilon_{\min} \), the regret of Algorithm 7 is upper-bounded uniformly over time \( n \) by a function that grows as \( O(N^3 \ln n) \). However, when \( S_{\max}, r_{\max}, \hat{\pi}_{\max} \) or \( \epsilon_{\min} \) (or the bound of them) are unknown, the upper bound of regret can not be guaranteed to grow logarithmically in \( n \).

When no knowledge about the system is available, we extend the CLRMR policy to achieve a regret bounded uniformly over time \( n \) by a function that grows as
\( O(N^3L(n) \ln n) \), using any arbitrarily slowly diverging non-decreasing sequence \( L(n) \) in Algorithm 7. Since \( L(n) \) could grow arbitrarily slowly, this modified version of CLRMR, named CLRMR-LN, could achieve a regret arbitrarily close to the logarithmic order. We present our analysis in Theorem 9.

**Theorem 9.** When using any arbitrarily slowly diverging non-decreasing sequence \( L(n) \) (i.e., \( L(n) \to \infty \) as \( n \to \infty \)), and replacing (6.3) in Algorithm 7 accordingly with

\[
\max_{a \in \mathcal{A}} a_i \left( z^i_2 + \sqrt{\frac{L(n(t_2)) \ln t_2}{m^i_2}} \right)
\]

(6.38)

where \( n(t_2) \) is the time when total number of time slots spent in SB2 is \( t_2 \), the expected regret under this modified version of CLRMR, named CLRMR-LN policy, is at most

\[
\mathcal{R}^{CLRMR-LN}_\text{(n)} \leq Z_6 L(n) \ln n + Z_7
\]

(6.39)

where \( Z_6 \) and \( Z_7 \) are constants.

**Proof.** Replacing \( c_{t,s} \) with \( \sqrt{\frac{L(n(t)) \ln t}{s}} \), and replacing \( L \) with \( L(n(t_2(b))) \) or \( L(n(\tau)) \) accordingly in the proof of Theorem 7, (6.4) to (6.21) still stand.

\( L(n(\tau)) \) is a diverging non-decreasing sequence, so there exists a constant \( \tau' \) such that for all \( \tau \geq \tau' \), \( L(n(\tau)) \geq 56(H + 1)S^2_{\text{max}} r^2_{\text{max}} \hat{r}^2_{\text{max}} / \epsilon_{\text{min}} \), which implies

\[
\frac{L(n(\tau)) \epsilon_{\text{min}}}{28S^2_{\text{max}} r^2_{\text{max}} \hat{r}^2_{\text{max}}} \leq \tau^{-2}.
\]

Thus, we have,
\[
\mathbb{E}[\tilde{B}^i(b)] \leq \frac{4L(n(t_2(b))))H^2a_{\max}^2\ln n}{\Delta_{\min}^2} + 1 + \frac{HS_{\max}}{\pi_{\min}} \sum_{\tau=1}^{\infty} 2\tau^{-2} + Z_8 \\
\leq \frac{4L(n)H^2a_{\max}^2\ln n}{\Delta_{\min}^2} + 1 + \frac{\pi HS_{\max}}{3\pi_{\min}} + Z_8 \tag{6.40}
\]

where
\[
Z_8 = \frac{HS_{\max}}{\pi_{\min}} \sum_{\tau=1}^{\tau' - 1} 2\tau - \frac{L_\min - 56HS_{\max}^2a_{\max}^2}{2\pi S_{\max}^2r_{\max}^2a_{\max}^2} \tag{6.41}
\]

Then we can accordingly have,
\[
\sum_{a: \gamma^a < \gamma^*} (\gamma^* - \gamma^a)\mathbb{E}[T^a(n)] \leq Z_9 L(n) \ln n + Z_2 + \Delta_{\max} \left(\frac{1}{\Pi_{\min}} + M_{\max} + 1\right)NZ_8.
\]

where
\[
Z_9 = \Delta_{\max} \left(\frac{1}{\Pi_{\min}} + M_{\max} + 1\right) \frac{4NH^2a_{\max}^2}{\Delta_{\min}^2}. \tag{6.42}
\]

So,
\[
\mathcal{R}^{CLRM-LN}(n) \leq Z_9 L(n) \ln n + Z_7, \tag{6.43}
\]

where
\[
Z_6 = Z_9 + Z_5 \frac{4NH^2a_{\max}^2}{\Delta_{\min}^2}
\]
\[ Z_7 = Z_2 + \gamma^* \left( \frac{1}{\Pi_{\min}} + M_{\max} + 1 \right) + \Delta_{\max} \left( \frac{1}{\Pi_{\min}} + M_{\max} + 1 \right) N Z_7 \] (6.44)

\[ + Z_5 \left( N + \frac{\pi N H S_{\max}}{3\pi_{\min}} + N Z_7 \right) . \]

6.5 Applications and Simulation Results

We now present an evaluation of our policy over stochastic versions of two combinatorial network optimization problems of practical interest: stochastic shortest path (for routing), and stochastic bipartite matching (for channel allocation).

6.5.1 Stochastic Shortest Path

In the stochastic shortest path problem, given a graph \( G = (V, E) \), with edge weights \( (D_{ij}) \) stochastically varying with time as restless Markov chains with unknown dynamics, we seek to find a path between a given source \( s \) and destination \( t \) with minimum expected delay. We can apply the CLRMR policy to this problem, with some very minor modifications to the policy and the corresponding regret definition to be applicable to a minimization problem instead of maximization. For clarity, (6.3) in Algorithm 7 should be replaced by

\[ a = \arg \min_{a \in \mathcal{F}} \sum_{i \in A_a} a_i \left( z_i^2 - \sqrt{L \ln t_2} \right) ; \] (6.45)
And the definition of regret should be instead expressed as,

$$\mathcal{R}^\phi(n) = \mathbb{E}^\phi\left[\sum_{t=1}^{n} R^\phi(t)(t)\right] - n\eta^*.$$  

(6.46)

where $\eta^*$ represents the minimum cost, which is cost of the optimal arm.

For the stochastic shortest path problems, each path between $s$ and $t$ is mapped to an arm. Although the number of paths could grow exponentially with the number of Markov chains, $|E|$. CLRMR efficiently solves this problem with polynomial storage $|E|$ and regret scaling as $O(|E|^3 \log n)$.

Also, since there exist polynomial time algorithms such as Dijkstra’s algorithm [29] and Bellman-Ford algorithm [18, 32] for shortest path, we can apply these algorithms to solve (6.45) with edge cost $\bar{z}_2^i - \sqrt{\frac{L \ln t_2}{m_2}}$.

![Example graphs](image)

(a) A graph with 15 links and 96 acyclic paths between $s$ and $t$. (b) A graph with 19 links and 260 acyclic paths between $s$ and $t$.

Figure 6.2: Two example graphs for stochastic shortest path routing.

We show the numerical simulation results with two example graphs in Figure 6.2.
Figure 6.2(a) is a graph with 15 links, and there are 96 acyclic paths between $s$ and $t$. We assume each link has two states with the delay 0.1 on good links, and 1 on bad links. Table 6.3 summarizes the transition probabilities on each link.

<table>
<thead>
<tr>
<th>Link</th>
<th>$p_{01}$, $p_{10}$</th>
<th>Link</th>
<th>$p_{01}$, $p_{10}$</th>
<th>Link</th>
<th>$p_{01}$, $p_{10}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>e.1</td>
<td>0.9, 0.1</td>
<td>e.6</td>
<td>0.8, 0.2</td>
<td>e.11</td>
<td>0.8, 0.3</td>
</tr>
<tr>
<td>e.2</td>
<td>0.3, 0.9</td>
<td>e.7</td>
<td>0.2, 0.7</td>
<td>e.12</td>
<td>0.2, 0.7</td>
</tr>
<tr>
<td>e.3</td>
<td>0.2, 0.7</td>
<td>e.8</td>
<td>0.3, 0.8</td>
<td>e.13</td>
<td>0.8, 0.1</td>
</tr>
<tr>
<td>e.4</td>
<td>0.2, 0.7</td>
<td>e.9</td>
<td>0.1, 0.9</td>
<td>e.14</td>
<td>0.4, 0.8</td>
</tr>
<tr>
<td>e.5</td>
<td>0.3, 0.9</td>
<td>e.10</td>
<td>0.3, 0.6</td>
<td>e.15</td>
<td>0.1, 0.8</td>
</tr>
</tbody>
</table>

Table 6.3: Transition probabilities.

Figure 6.2(b) shows another graph with 19 links, which has only 4 links more than the graph in Figure 6.2(a). But there are $4 \times (1 + P(4, 1) + P(4, 2) + P(4, 2) + P(4, 4)) = 260$ acyclic paths in the graph between $s$ and $t$, much more than the number of paths in 6.2(a). We use $P(N, M)$ here to denote the number of permutations that arrange $M$ out of $N$ choices. We again assume each link has two states with the delay 0.1 on good links, and 1 on bad links. Table 6.4 summarizes the transition probabilities on each link.

<table>
<thead>
<tr>
<th>Link</th>
<th>$p_{01}$, $p_{10}$</th>
<th>Link</th>
<th>$p_{01}$, $p_{10}$</th>
<th>Link</th>
<th>$p_{01}$, $p_{10}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>e.1</td>
<td>0.2, 0.8</td>
<td>e.8</td>
<td>0.3, 0.8</td>
<td>e.15</td>
<td>0.1, 0.8</td>
</tr>
<tr>
<td>e.2</td>
<td>0.3, 0.9</td>
<td>e.9</td>
<td>0.1, 0.9</td>
<td>e.16</td>
<td>0.8, 0.1</td>
</tr>
<tr>
<td>e.3</td>
<td>0.2, 0.7</td>
<td>e.10</td>
<td>0.9, 0.1</td>
<td>e.17</td>
<td>0.2, 0.7</td>
</tr>
<tr>
<td>e.4</td>
<td>0.7, 0.1</td>
<td>e.11</td>
<td>0.3, 0.8</td>
<td>e.18</td>
<td>0.9, 0.1</td>
</tr>
<tr>
<td>e.5</td>
<td>0.3, 0.9</td>
<td>e.12</td>
<td>0.2, 0.7</td>
<td>e.19</td>
<td>0.3, 0.8</td>
</tr>
<tr>
<td>e.6</td>
<td>0.2, 0.7</td>
<td>e.13</td>
<td>0.8, 0.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>e.7</td>
<td>0.2, 0.8</td>
<td>e.14</td>
<td>0.4, 0.8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6.4: Transition probabilities.
(a) Simulation results for stochastic short path problem in Figure 6.2(a) with $L = 1324$.

(b) Simulation results for stochastic short path problem in Figure 6.2(b) with $L = 1512$.

(c) Simulation results for stochastic short path problem in Figure 6.2(b) with $L = 2$ (logarithmic scale for the y-axis).

Figure 6.3: Normalized regret $\frac{\mathcal{R}(n)}{\ln n}$ vs. $n$ time slots.
Figure 6.3 shows the simulation results for the graphs in Figure 6.2. In Figure 6.3(a) and 6.3(b), we let $L = 1324$ and $L = 1512$ respectively, such that $L \geq 56(H + 1)S_{\text{max}}^2 \hat{r}_{\text{max}}^2 / \epsilon_{\text{min}}$. We let $L = 2$ for Figure 6.3(c). We can see that in these three cases of both graphs, our proposed CLRMR performs better than the naive application of RCA. We can also see that under both policies, the regret grows logarithmically in time.

We note that as the number of links increases from 15 to 19, so the number of paths increase much faster, from 96 to 260, the gap between the RCA policy and our CLRMR policy is a lot higher.

Another observation is that when $L$ varies from 1512 to 2 as shown in 6.3(c), CLRMR and RCA also seems to yield logarithmic regret over time, and the performance is in fact much better than $L = 1512$. Note that in the proof of Theorem 7, when $L < 56(H + 1)S_{\text{max}}^2 \hat{r}_{\text{max}}^2 / \epsilon_{\text{min}}$, we have $-\frac{L\epsilon_{\text{min}} - 56HS_{\text{max}}^2 \hat{r}_{\text{max}}^2 \hat{\pi}_{\text{max}}^2}{28S_{\text{max}}^2 \hat{r}_{\text{max}}^2 \hat{\pi}_{\text{max}}^2} > -2$. This implies $\sum_{\tau = 1}^{\infty} 2\tau \frac{L\epsilon_{\text{min}} - 56HS_{\text{max}}^2 \hat{r}_{\text{max}}^2 \hat{\pi}_{\text{max}}^2}{28S_{\text{max}}^2 \hat{r}_{\text{max}}^2 \hat{\pi}_{\text{max}}^2}$ does not converge anymore and thus we could not bound $E[\hat{B}_i^i(b)]$ any more. Empirically, however, in 6.3(c) the case when $L < 1512$ also seems to yield logarithmic regret over time and the performance is in fact better than that of $L \geq 1512$, since the non-optimal arms are played less when $L$ is smaller. However, this may possibly be due to the fact that the cases in which $\hat{B}_i^i(b)$ grows faster than $\ln(t)$ only happens with very small probability. The smaller $L$ is, the greater this probability is.
6.5.2 Stochastic Bipartite Matching for Channel Allocation

As a second application, we consider an application in a cognitive radio networks where \( M \) secondary users interfering with each other need to be allocated to \( Q \) non-conflicting orthogonal channels. We assume that, due to geographic dispersion, each user may see different primary user occupancy behavior in each channel. The availability of spectrum opportunities on each user-channel combination \((i, j)\) over a decision period is modeled as a restless two-state Markov chain. It is easy to show that applying CLRMR to this problem yields storage linear in \( MQ \), and a regret bound that scales as

\[
O(\min\{M, Q\}^2MQ \log n),
\]

following Theorem 8.

The computation time of CLRMR is also polynomial, since there are various algorithms to solve the different variations in the maximum weighted matching problems, such as the Hungarian algorithm for the maximum weighted bipartite matching [50] and Edmonds’s matching algorithm [31] for a general maximum matching.

We show simulation results of our CLRMR algorithm and the naive RCA algorithm for opportunistic spectrum access with two scenarios: (i) a system consisting \( Q = 7 \) orthogonal channels and \( M = 4 \) secondary users and (ii) a system consisting \( Q = 9 \) orthogonal channels and \( M = 5 \) secondary users. The transition probability matrices used for these two scenarios are presented in Table 6.5 and 6.6.
Table 6.5: Transition probabilities $p_{01}, p_{10}$ for each user-channel pair.

<table>
<thead>
<tr>
<th></th>
<th>ch.1</th>
<th>ch.2</th>
<th>ch.3</th>
<th>ch.4</th>
<th>ch.5</th>
<th>ch.6</th>
<th>ch.7</th>
</tr>
</thead>
<tbody>
<tr>
<td>u.1</td>
<td>0.5,0.6</td>
<td>0.2,0.7</td>
<td>0.3,0.9</td>
<td>0.8,0.1</td>
<td>0.2,0.8</td>
<td>0.2,0.6</td>
<td>0.2,0.9</td>
</tr>
<tr>
<td>u.2</td>
<td>0.2,0.7</td>
<td>0.2,0.9</td>
<td>0.1,0.8</td>
<td>0.3,0.7</td>
<td>0.3,0.6</td>
<td>0.1,0.7</td>
<td>0.8,0.2</td>
</tr>
<tr>
<td>u.3</td>
<td>0.7,0.1</td>
<td>0.2,0.7</td>
<td>0.1,0.8</td>
<td>0.2,0.8</td>
<td>0.5,0.6</td>
<td>0.2,0.8</td>
<td>0.2,0.7</td>
</tr>
<tr>
<td>u.4</td>
<td>0.2,0.6</td>
<td>0.2,0.8</td>
<td>0.1,0.8</td>
<td>0.4,0.6</td>
<td>0.9,0.2</td>
<td>0.1,0.7</td>
<td>0.2,0.8</td>
</tr>
</tbody>
</table>

Table 6.6: Transition probabilities $p_{01}, p_{10}$ for each user-channel pair.

<table>
<thead>
<tr>
<th></th>
<th>ch.1</th>
<th>ch.2</th>
<th>ch.3</th>
<th>ch.4</th>
<th>ch.5</th>
<th>ch.6</th>
<th>ch.7</th>
<th>ch.8</th>
<th>ch.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>u.1</td>
<td>0.5,0.6</td>
<td>0.2,0.7</td>
<td>0.2,0.9</td>
<td>0.8,0.1</td>
<td>0.2,0.7</td>
<td>0.3,0.7</td>
<td>0.2,0.9</td>
<td>0.2,0.7</td>
<td>0.1,0.9</td>
</tr>
<tr>
<td>u.2</td>
<td>0.3,0.8</td>
<td>0.1,0.9</td>
<td>0.2,0.8</td>
<td>0.3,0.7</td>
<td>0.3,0.6</td>
<td>0.2,0.8</td>
<td>0.4,0.7</td>
<td>0.2,0.8</td>
<td>0.9,0.2</td>
</tr>
<tr>
<td>u.3</td>
<td>0.8,0.1</td>
<td>0.2,0.7</td>
<td>0.3,0.7</td>
<td>0.2,0.8</td>
<td>0.5,0.6</td>
<td>0.2,0.7</td>
<td>0.2,0.7</td>
<td>0.2,0.8</td>
<td>0.1,0.9</td>
</tr>
<tr>
<td>u.4</td>
<td>0.3,0.9</td>
<td>0.2,0.8</td>
<td>0.2,0.9</td>
<td>0.4,0.6</td>
<td>0.9,0.2</td>
<td>0.2,0.9</td>
<td>0.2,0.9</td>
<td>0.2,0.9</td>
<td>0.2,0.9</td>
</tr>
<tr>
<td>u.5</td>
<td>0.5,0.6</td>
<td>0.2,0.7</td>
<td>0.3,0.9</td>
<td>0.2,0.7</td>
<td>0.5,0.5</td>
<td>0.2,0.7</td>
<td>0.8,0.1</td>
<td>0.3,0.9</td>
<td>0.3,0.9</td>
</tr>
</tbody>
</table>

The simulation results are shown and compared in Figure 6.4. We can see that for scenario (i), there are $P(7, 4) = 840$ matchings, while only $7 = 28$ Markov chains. For scenario (ii), as the number of channels and users increases a to 9 and 5, the number of matchings are much higher ($P(9, 5) = 15120$), which is about 336 times higher. So the storage as well as regret of the naive RCA policy grow much faster than CLRMR policy, as the results indicated in 6.4(a) and 6.4(b). For these two simulations, we pick the value of $L$ as 922 and 1135 such that $L \geq 56(H + 1)S^2_{max}r^2_{max}\hat{\pi}^2_{max}/\epsilon_{min}$. We also show the simulation results when $L$ varies from 1135 to 2 in 6.4(c). Again, we see that the performance seems to improve in practice with smaller $L$ values, even if it is not be theoretically guaranteed.

124
(a) $N = 7$ channels, $M = 4$ secondary users, $L = 922$.

(b) $N = 9$ channels, $M = 5$ secondary users, $L = 1135$.

(c) $N = 9$ channels, $M = 5$ secondary users, $L = 2$ (logarithmic scale for the $y$-axis).

Figure 6.4: Normalized regret $\frac{R(n)}{\ln n}$ vs. $n$ time slots.
6.6 Summary

We have presented CLRMR, a provably efficient online learning policy for stochastic combinatorial network optimization with restless Markovian rewards. This algorithm is widely applicable to many networking problems of interest, as illustrated by our simulation based evaluation of the policy over two different problems: stochastic shortest path and stochastic maximum weight bipartite matching.
Chapter 7

Learning for Stochastic Water-Filling

7.1 Overview

The classic water-filling algorithm is deterministic and requires perfect knowledge of the channel gain to noise ratios. In this Chapter\(^1\), we consider how to do power allocation over stochastically time-varying (i.i.d.) channels with unknown gain to noise ratio distributions. We adopt an online learning framework based on stochastic multi-armed bandits. We consider two variations of the problem, one in which the goal is to find a power allocation to maximize \(\sum_i \mathbb{E} \left[ \log(1 + SNR_i) \right]\), and another in which the goal is to find a power allocation to maximize \(\sum_i \log(1 + \mathbb{E}[SNR_i])\). For the first problem, we propose a \textit{cognitive water-filling} algorithm that exploits the linear structure of this problem, that we call CWF1. We show that CWF1 obtains a regret that grows polynomially in the number of channels and logarithmically in time. It therefore asymptotically achieves the optimal time-averaged rate that can be obtained when the gain distributions

\(^1\)This chapter is based in part on [34].
are known. For the second problem, we present an algorithm called CWF2, which is, to our knowledge, the first algorithm in the literature on stochastic multi-armed bandits to exploit non-linear dependencies between the arms. We prove that the number of times CWF2 picks the incorrect power allocation is bounded by a function that is polynomial in the number of channels and logarithmic in time, implying that its frequency of incorrect allocation asymptotically tends to zero.

7.2 Problem Formulation

We define the stochastic version of the classic communication theory problem of power allocation for maximizing rate over parallel channels (water-filling) as follows.

We consider a system with \( N \) channels, where the channel gain-to-noise ratios are unknown random processes \( X_i(n), 1 \leq i \leq N \). Time is slotted and indexed by \( n \). We assume that \( X_i(n) \) evolves as an i.i.d. random process over time (i.e., we consider block fading), with the only restriction that its distribution has a finite support. Without loss of generality, we normalize \( X_i(n) \in [0, 1] \). We do not require that \( X_i(n) \) be independent across \( i \). This random process is assumed to have a mean \( \theta_i = E[X_i] \) that is unknown to the users. We denote the set of all these means by \( \Theta = \{\theta_i\} \).

At each decision period \( n \) (also referred to interchangeably as a time slot), an \( N \)-dimensional action vector \( a(n) \), representing a power allocation on these \( N \) channels, is selected under a policy \( \phi(n) \). We assume that the power levels are discrete, and we can put any constraint on the selections of power allocations such that they are from a finite
set $\mathcal{F}$ (i.e., the maximum total power constraint, or an upper bound on the maximum allowed power per subcarrier). We assume $a_i(n) \geq 0$ for all $1 \leq i \leq N$. When a particular power allocation $a(n)$ is selected, the channel gain-to-noise ratios corresponding to nonzero components of $a(n)$ are revealed, i.e., the value of $X_i(n)$ is observed for all $i$ such that $a_i(n) \neq 0$. We denote by $\mathcal{A}_a(n) = \{i : a_i(n) \neq 0, 1 \leq i \leq N\}$ the index set of all $a_i(n) \neq 0$ for an allocation $a$.

We adopt a general formulation for water-filling, where the sum rate \(^2\) obtained at time $n$ by allocating a set of powers $a(n)$ is defined as:

$$ R_{a(n)}(n) = \sum_{i \in \mathcal{A}_a(n)} f_i(a_i(n), X_i(n)). \quad (7.1) $$

where for all $i$, $f_i(a_i(n), X_i(n))$ is a nonlinear continuous increasing sub-additive function in $X_i(n)$, and $f_i(a_i(n), 0) = 0$ for any $a_i(n)$. We assume $f_i$ is defined on $\mathbb{R}^+ \times \mathbb{R}^+$.

Our formulation is general enough to include as a special case of the rate function obtained from Shannon’s capacity theorem for AWGN, which is widely used in communication networks:

$$ R_{a(n)}(n) = \sum_{i=1}^{N} \log(1 + a_i(n)X_i(n)) $$

\(^2\)We refer to rate and reward interchangeably in this chapter.
In the typical formulation there is a total power constraint and individual power constraints, the corresponding constraint is

\[ \mathcal{F} = \{ a : \sum_{i=1}^{N} a_i \leq P_{\text{total}} \land 0 \leq a_i \leq P_i, \forall i \} , \]

where \( P_{\text{total}} \) is the total power constraint and \( P_i \) is the maximum allowed power per channel.

Our goal is to maximize the expected sum-rate when the distributions of all \( X_i \) are unknown, as shown in (7.2). We refer to this objective as \( \mathcal{O}_1 \).

\[ \max_{a \in \mathcal{F}} \mathbb{E}[\sum_{i \in \mathcal{A}_a} f_i(a_i, X_i)] \]  

(7.2)

Note that even when \( X_i \) have known distributions, this is a hard combinatorial non-linear stochastic optimization problem. In our setting, with unknown distributions, we can formulate this as a multi-armed bandit problem, where each power allocation \( a(n) \in \mathcal{F} \) is an arm and the reward function is in a combinatorial non-linear form. The optimal arms are the ones with the largest expected reward, denoted as \( \mathcal{O}^* = \{ a^* \} \). For the rest of the chapter, we use \( \ast \) as the index indicating that a parameter is for an optimal arm. If more than one optimal arm exists, \( \ast \) refers to any one of them.

We note that for the combinatorial multi-armed bandit problem with linear rewards where the reward function is defined by \( R_{a(n)}(n) = \sum_{i \in \mathcal{A}_{a(n)}} a_i(n)X_i(n) \), \( a^* \) is a solution to a deterministic optimization problem because \( \max_{a \in \mathcal{F}} \mathbb{E}[\sum_{i \in \mathcal{A}_a} a_iX_i] = \max_{a \in \mathcal{F}} \sum_{i \in \mathcal{A}_a} a_i\mathbb{E}[X_i] \). 

130
Different from the combinatorial multi-armed bandit problem with linear rewards, \( a^* \) here is a solution to a stochastic optimization problem, i.e.,

\[
\mathbf{a}^* \in \mathcal{O}^* = \{ \hat{a} : \hat{a} = \arg \max_{a \in \mathcal{F}} E[\sum_{i \in A_a} f_i(a_i, X_i)] \}. \tag{7.3}
\]

We evaluate policies for \( O_1 \) with respect to regret, which is defined as the difference between the expected reward that could be obtained by a genie that can pick an optimal arm at each time, and that obtained by the given policy. Note that minimizing the regret is equivalent to maximizing the expected rewards. Regret can be expressed as:

\[
\mathfrak{R}^\phi(n) = n R^* - E[\sum_{t=1}^n R_{\phi(t)}(t)], \tag{7.4}
\]

where \( R^* = \max_{a \in \mathcal{F}} E[\sum_{i \in A_a} f_i(a_i, X_i)] \), the expected reward of an optimal arm.

Intuitively, we would like the regret \( \mathfrak{R}^\phi(n) \) to be as small as possible. If it is sub-linear with respect to time \( n \), the time-averaged regret will tend to zero and the maximum possible time-averaged reward can be achieved. Note that the number of arms \( |\mathcal{F}| \) can be exponential in the number of unknown random variables \( N \).

We also note that for the stochastic version of the water-filling problems, a typical way in practice to deal with the unknown randomness is to estimate the mean channel gain to noise ratios first and then find the optimized allocation based on the mean values. This approach tries to identify the power allocation that maximizes the power-rate equation applied to the mean channel gain-to-noise ratios. We refer to maximizing this as the
sum-pseudo-rate over averaged channels. We denote this objective by $O_2$, as shown in (7.5).

$$
\max_{a \in F} \sum_{i \in A_a} f_i(a_i, E[X_i])
$$

(7.5)

We would also like to develop an online learning policy for $O_2$. Note that the optimal arm $a^*$ of $O_2$ is a solution to a deterministic optimization problem. So, we evaluate the policies for $O_2$ with respect to the expected total number of times that a non-optimal power allocation is selected. We denote by $T_a(n)$ the number of times that a power allocation is picked up to time $n$. We denote $r_a = \sum_{i \in A_a} f_i(a_i, E[X_i])$. Let $T_{non}^\phi(n)$ denote the total number of times that a policy $\phi$ select a power allocation $r^a < r^{a^*}$. Denote by $1_\phi^a(t)$ the indicator function which is equal to 1 if $a$ is selected under policy $\phi$ at time $t$, and 0 else. Then

$$
E[T_{non}^\phi(n)] = n - E[\sum_{t=1}^n 1_\phi^a(t^*) = 1] 
$$

(7.6)

$$
= \sum_{r_a < r^{a^*}} E[T_a(n)].
$$

### 7.3 Online Learning for Maximizing the Sum-Rate

We first present in this section an online learning policy for stochastic water-filling under object $O_1$. 

7.3.1 Policy Design

A straightforward, naive way to solve this problem is to use the UCB1 policy proposed [13]. For UCB1, each power allocation is treated as an arm, and the arm that maximizes 
\[ \hat{Y}_k + \sqrt{\frac{2 \ln n}{m_k}} \] 
will be selected at each time slot, where \( \hat{Y}_k \) is the mean observed reward on arm \( k \), and \( m_k \) is the number of times that arm \( k \) has been played. This approach essentially ignores the underlying dependencies across the different arms, and requires storage that is linear in the number of arms and yields regret growing linearly with the number of arms. Since there can be an exponential number of arms, the UCB1 algorithm performs poorly on this problem.

We note that for combinatorial optimization problems with linear reward functions, an online learning algorithm LLR has been proposed in Chapter 4 as an efficient solution. LLR stores the mean of observed values for every underlying unknown random variable, as well as the number of times each has been observed. So the storage of LLR is linear in the number of unknown random variables, and the analysis in Chapter 4 shows LLR achieves a regret that grows logarithmically in time, and polynomially in the number of unknown parameters.

However, the challenge with stochastic water-filling with objective \( O_1 \), where the expectation is outside the non-linear reward function, directly storing the mean observations of \( X_i \) will not work.

To deal with this challenge, we propose to store the information for each \( a_i, X_i \) combination, i.e., \( \forall 1 \leq i \leq N, \forall a_i \), we define a new set of random variables \( Y_{i,a_i} = \)
\( f_i(a_i, X_i) \). So now the number of random variables \( Y_{i,a_i} \) is \( \sum_{i=1}^{N} |B_i| \), where \( B_i = \{ a_i : a_i \neq 0 \} \). Note that \( \sum_{i=1}^{N} |B_i| \leq PN \).

Then the reward function can be expressed as

\[
R_a = \sum_{i \in A_a} Y_{i,a_i}, \quad (7.7)
\]

Note that (7.7) is in a combinatorial linear form.

For this redefined MAB problem with \( \sum_{i=1}^{N} |B_i| \) unknown random variables and linear reward function (7.7), we propose the following online learning policy CWF1 for stochastic water-filling as shown in Algorithm 8.

**Algorithm 8** Online Learning for Stochastic Water-Filling: CWF1

1: // INITIALIZATION
2: If \( \max_a |A_a| \) is known, let \( L = \max_a |A_a| \); else, \( L = N \);
3: for \( n = 1 \) to \( N \) do
4: Play any arm \( a \) such that \( n \in A_a \);
5: \( \forall i \in A_a, \forall a_i \in B_i, Y_{i,a_i} := \frac{\bar{Y}_{i,a_i} m_i + f_i(a_i, X_i)}{m_i + 1} \);
6: \( \forall i \in A_a, m_i := m_i + 1 \);
7: end for
8: // MAIN LOOP
9: while 1 do
10: \( n := n + 1 \);
11: Play an arm \( a \) which solves the maximization problem

\[
\sum_{i \in A_a} (\bar{Y}_{i,a_i} + \sqrt{\frac{(L + 1) \ln n}{m_i}}); \quad (7.8)
\]

12: \( \forall i \in A_a, \forall a_i \in B_i, Y_{i,a_i} := \frac{\bar{Y}_{i,a_i} m_i + f_i(a_i, X_i)}{m_i + 1} \);
13: \( \forall i \in A_a, m_i := m_i + 1 \);
14: end while
To have a tighter bound of regret, different from the LLR algorithm, instead of storing the number of times that each unknown random variables $Y_{i,a_i}$ has been observed, we use a 1 by $N$ vector, denoted as $(m_i)_{1 \times N}$, to store the number of times that $X_i$ has been observed up to the current time slot.

We use a 1 by $\sum_{i=1}^{N} |B_i|$ vector, denoted as $(\bar{Y}_{i,a_i})_{1 \times \sum_{i=1}^{N} |B_i|}$ to store the information based on the observed values. $(\bar{Y}_{i,a_i})_{1 \times \sum_{i=1}^{N} |B_i|}$ is updated in as shown in line 12. Each time an arm $a(n)$ is played, $\forall i \in A_{a(n)}$, the observed value of $X_i$ is obtained. For every observed value of $X_i$, $|B_i|$ values are updated: $\forall a_i \in B_i$, the average value $\bar{Y}_{i,a_i}$ of all the values of $Y_{i,a_i}$ up to the current time slot is updated. CWF1 policy requires storage linear in $\sum_{i=1}^{N} |B_i|$.

### 7.3.2 Analysis of Regret

**Theorem 10.** The expected regret under the CWF1 policy is at most

$$\left[ \frac{4L^2(L+1)N \ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3} LN \right] \Delta_{\max}. \quad (7.9)$$

where $\Delta_{\min} = \min_{a \neq a^*} R^* - \mathbb{E}[R_a]$, $\Delta_{\max} = \max_{a \neq a^*} R^* - \mathbb{E}[R_a]$. Note that $L \leq N$.

**Proof.** Let $C_{t,m_i}$ denote $\sqrt{\frac{(L+1)\ln t}{m_i}}$. We introduce $\tilde{T}_i(n)$ as a counter after the initialization period. It is updated in the following way:

At each time slot after the initialization period, one of the two cases must happen: (1) an optimal arm is played; (2) a non-optimal arm is played. In the first case, $(\tilde{T}_i(n))_{1 \times N}$
won’t be updated. When an non-optimal arm \( a(n) \) is picked at time \( n \), there must be at least one \( i \in A_a \) such that \( i = \arg \min_{j \in A_a} m_j \). If there is only one such arm, \( \widetilde{T}_i(n) \) is increased by 1. If there are multiple such arms, we arbitrarily pick one, say \( i' \), and increment \( \widetilde{T}_{i'} \) by 1.

Each time when a non-optimal arm is picked, exactly one element in \( (\widetilde{T}_i(n))_{1 \times N} \) is incremented by 1. This implies that the total number that we have played the non-optimal arms is equal to the summation of all counters in \( (\widetilde{T}_i(n))_{1 \times N} \). Therefore, we have:

\[
\sum_{a: a \neq a^*} \mathbb{E}[T_a(n)] = \sum_{i=1}^{N} \mathbb{E}[\widetilde{T}_i(n)]. \tag{7.10}
\]

Also note for \( \widetilde{T}_i(n) \), the following inequality holds:

\[
\widetilde{T}_i(n) \leq m_i(n), \forall 1 \leq i \leq N. \tag{7.11}
\]

Denote by \( \bar{T}_i(n) \) the indicator function which is equal to 1 if \( \widetilde{T}_i(n) \) is added by one at time \( n \). Let \( l \) be an arbitrary positive integer. Then:

\[
\widetilde{T}_i(n) = \sum_{t=N+1}^{n} 1\{\bar{T}_i(t) = 1\} \leq l + \sum_{t=N+1}^{n} 1\{\bar{T}_i(t) = 1, \bar{T}_i(t-1) \geq l\} \tag{7.12}
\]

where \( 1(x) \) is the indicator function defined to be 1 when the predicate \( x \) is true, and 0 when it is false. When \( \bar{T}_i(t) = 1 \), a non-optimal arm \( a(t) \) has been picked for which
\[ m_i = \min_j \{ m_j : \forall j \in A_{a(t)} \} \]. We denote this arm as \( a(t) \) since at each time that \( \tilde{I}_i(t) = 1 \), we could get different arms.

We denote by \( \overline{Y}_{i,a_i,m_i} \) the average (sample mean) of all the observed values of \( Y_{i,a_i} \) when the corresponding \( X_i \) is observed \( m_i \) times. Let \( \mathbb{E}[Y_{i,a_i}] \) denote \( \mathbb{E}[f_i(a_i, X_i)] \).

Then we have,

\[
\tilde{I}_i(n) \leq l + \sum_{t=N+1}^{n} 1 \{ \sum_{j \in A_{a^*}} (\overline{Y}_{j,a^*_j,m_j(t-1)} + C_{t-1,m_j(t-1)}) \sum_{j \in A_{a(t)}} (\overline{Y}_{j,a_j(t),m_j} + C_{t,m_j(t)} - C_{t-1,m_j(t-1)}) \}
\]

\[
\leq l + \sum_{t=N}^{n} 1 \{ \sum_{j \in A_{a^*}} (\overline{Y}_{j,a^*_j,m_j(t)} + C_{t,m_j(t)}) \}
\]

\[
\leq \sum_{j \in A_{a(t)}} (\overline{Y}_{j,a_j(t),m_j} + C_{t,m_j(t)}), \tilde{I}_i(t) \geq l \}.
\]

Note that \( l \leq \tilde{I}_i(t) \) implies,

\[
l \leq \tilde{I}_i(t) \leq m_j(t), \forall j \in A_{a(t)}. \]

Then,

\[
\tilde{I}_i(n) \leq l + \sum_{t=N+1}^{n} 1 \{ \sum_{j \in A_{a^*}} (\overline{Y}_{j,a^*_j,m_j(t-1)} + C_{t-1,m_j(t-1)}) \}
\]

\[
\leq \sum_{j \in A_{a(t)}} (\overline{Y}_{j,a_j(t),m_j} + C_{t-1,m_j(t-1)}), \tilde{I}_i(t-1) \geq l \}
\]
\[ l + \sum_{t=N}^{n} 1\{ \sum_{j \in A_{a^*}} (\mathbf{Y}_{j,a_j^*,m_j}(t) + C_{t,m_j}(t)) \leq \sum_{j \in A_{a(t)}} (\mathbf{Y}_{j,a_j(t),m_j}(t) + C_{t,m_j}(t)), \bar{T}_i(t) \geq l \}. \]

where \( h_j (1 \leq j \leq |A_{a^*}|) \) represents the \( j \)-th element in \( A_{a^*} \) and \( p_j (1 \leq j \leq |A_{a(t)}|) \) represents the \( j \)-th element in \( A_{a(t)} \).

\[ \sum_{j=1}^{\left| A_{a^*} \right|} (\mathbf{Y}_{h_j,a_{h_j}^*,m_{h_j}} + C_{t,m_{h_j}}) \leq \sum_{j=1}^{\left| A_{a(t)} \right|} (\mathbf{Y}_{p_j,a_{p_j}(t),m_{p_j}} + C_{t,m_{p_j}}) \]

means that at least one of the following must be true:

\[ \sum_{j=1}^{\left| A_{a^*} \right|} \mathbf{Y}_{h_j,a_{h_j}^*,m_{h_j}} \leq R^* - \sum_{j=1}^{\left| A_{a^*} \right|} C_{t,m_{h_j}}, \quad (7.15) \]

\[ \sum_{j=1}^{\left| A_{a(t)} \right|} \mathbf{Y}_{p_j,a_{p_j}(t),m_{p_j}} \geq R_{a(t)} + \sum_{j=1}^{\left| A_{a(t)} \right|} C_{t,m_{p_j}}. \quad (7.16) \]

\[ R^* < R_{a(t)} + 2 \sum_{j=1}^{\left| A_{a(t)} \right|} C_{t,m_{p_j}}. \quad (7.17) \]

Now we find the upper bound for \( P\{ \sum_{j=1}^{\left| A_{a^*} \right|} \mathbf{Y}_{h_j,a_{h_j}^*,m_{h_j}} \leq R^* - \sum_{j=1}^{\left| A_{a^*} \right|} C_{t,m_{h_j}} \}. \) We have:

\[ P\{ \sum_{j=1}^{\left| A_{a^*} \right|} \mathbf{Y}_{h_j,a_{h_j}^*,m_{h_j}} \leq R^* - \sum_{j=1}^{\left| A_{a^*} \right|} C_{t,m_{h_j}} \} \]

\[ = P\{ \sum_{j=1}^{\left| A_{a^*} \right|} \mathbf{Y}_{h_j,a_{h_j}^*,m_{h_j}} \leq \sum_{j=1}^{\left| A_{a^*} \right|} E[Y_{h_j,a_{h_j}^*}] - \sum_{j=1}^{\left| A_{a^*} \right|} C_{t,m_{h_j}} \} \]

\[ \leq \sum_{j=1}^{\left| A_{a^*} \right|} P\{ \mathbf{Y}_{h_j,a_{h_j}^*,m_{h_j}} \leq E[Y_{h_j,a_{h_j}^*}] - C_{t,m_{h_j}} \}. \]
\forall 1 \leq j \leq |\mathcal{A}_a|$, applying the Chernoff-Hoeffding bound stated in Lemma 1 in Chapter 2, we could find the upper bound of each item in the above equation as,

\[
\Pr \{ \sum_{j=1}^{m_{h_j}} Y_{h_j, a^*_h, m_{h_j}} - C_{t, m_{h_j}} \leq R^* - \sum_{j=1}^{m_{h_j}} C_{t, m_{h_j}} \} \\
\leq e^{-2 \frac{(L+1) \ln t}{m_{h_j}}} \\
= e^{-2(L+1) \ln t} \\
= t^{-2(L+1)}.
\]

Thus,

\[
\Pr \{ \sum_{j=1}^{m_{h_j}} Y_{h_j, a^*_h, m_{h_j}} \leq R^* - \sum_{j=1}^{m_{h_j}} C_{t, m_{h_j}} \} \\
\leq |\mathcal{A}_a| t^{-2(L+1)} \tag{7.18}
\]

\[
\leq Lt^{-2(L+1)}.
\]

Similarly, we can get the upper bound of the probability for inequality (7.16):

\[
\Pr \{ \sum_{j=1}^{m_{p_j}} Y_{p_j, a_{p_j}(t), m_{p_j}} \geq R_{a(t)} + \sum_{j=1} C_{t, m_{p_j}} \} \leq Lt^{-2(L+1)}. \tag{7.19}
\]
Note that for \( l \geq \left\lceil \frac{4(L+1) \ln n}{(\Delta_{a(t)})^2} \right\rceil \),

\[
R^* - R_{a(t)} - 2 \sum_{j=1}^{\left| A_{a(t)} \right|} C_{t,m_p} \\
= R^* - R_{a(t)} - 2 \sum_{j=1}^{\left| A_{a(t)} \right|} \sqrt{\frac{(L+1) \ln t}{m_p}} \\
\geq R^* - R_{a(t)} - L \sqrt{\frac{4(L+1) \ln n}{l}} \\
\geq R^* - R_{a(t)} - L \sqrt{\frac{4(L+1) \ln n}{4(L+1) \ln n \left( \frac{\Delta_{a(t)}}{L} \right)^2}} \\
\geq R^* - R_{a(t)} - \Delta_{a(t)} = 0. 
\]

(7.20)

Equation (7.39) implies that condition (7.15) is false when \( l = \left\lceil \frac{4(L+1) \ln n}{(\Delta_{a(t)})^2} \right\rceil \). If we let \( l = \left\lceil \frac{4(L+1) \ln n}{(\Delta_{a(t)})^2} \right\rceil \), then (7.15) is false for all \( a(t) \).

Therefore,

\[
\mathbb{E}[\overline{T}_i(n)] \leq \left\lceil \frac{4(L+1) \ln n}{(\Delta_{\min})^2} \right\rceil + \sum_{t=1}^{\infty} \left( \sum_{m_{A_i} = 1}^{t} \cdots \sum_{m_{A_{a(t)}} = 1}^{t} \sum_{m_p = 1}^{t} \cdots \sum_{m_p = 4L^2t^{-2L+1}}^{t} 2Lt^{-2(L+1)} \right) \\
\leq \frac{4L^2(L+1) \ln n}{(\Delta_{\min})^2} + 1 + L \sum_{t=1}^{\infty} 2t^{-2} \\
\leq \frac{4L^2(L+1) \ln n}{(\Delta_{\min})^2} + 1 + \frac{\pi^2}{3} L. 
\]

(7.21)
So under CWF1 policy, we have:

\[
R_n^\phi(\Theta) = R^* n - \mathbb{E}^{\phi} \left[ \sum_{t=1}^{n} R_{\phi(t)}(t) \right] \\
= \sum_{a: R_a < R^*} \Delta_a \mathbb{E}[T_a(n)] \\
\leq \Delta_{max} \sum_{a: R_a < R^*} \mathbb{E}[T_a(n)] \\
= \Delta_{max} \sum_{i=1}^{N} \mathbb{E}[\hat{T}_i(n)] \\
\leq \left[ \sum_{i=1}^{N} \frac{4L^2(L+1) \ln n}{(\Delta_{min})^2} + N + \frac{\pi^2}{3} LN \right] \Delta_{max} \\
\leq \left[ \frac{4L^2(L+1)N \ln n}{(\Delta_{min})^2} + N + \frac{\pi^2}{3} LN \right] \Delta_{max}.
\]

(7.22)

**Remark 4.** For CWF1 policy, although there are \( \sum_{i=1}^{N} |B_i| \) random variables, the upper bound of regret remains \( O(N^4 \log n) \), which is the same as LLR, as shown by Theorem 2 in Chapter 4. Directly applying LLR algorithm to solve the redefined MAB problem in (7.7) will result in a regret that grows as \( O(P^4 N^4 \log n) \).

**Remark 5.** Algorithm 8 will even work for rate functions that do not satisfy subadditivity.

**Remark 6.** We can develop similar policies and results when \( X_i \) are Markovian rewards as in Chapter 5 and Chapter 6.
7.4 Online Learning for Sum-Pseudo-Rate

We now show our novel online learning algorithm CWF2 for stochastic water-filling with object $O_2$. Unlike CWF1, CWF2 exploits non-linear dependencies between the choices of power allocations and requires lower storage. Under condition where the power allocation that maximize $O_2$ also maximize $O_1$, we will see through simulations that CWF2 has better regret performance.

7.4.1 Policy Design

Our proposed policy CWF2 for stochastic water filling with objective $O_2$ is shown in Algorithm 9.

**Algorithm 9 Online Learning for Stochastic Water-Filling: CWF2**

1: // INITIALIZATION
2: If $\max_a |A_a|$ is known, let $L = \max_a |A_a|$; else, $L = N$;
3: for $n = 1$ to $N$ do
4: Play any arm $a$ such that $n \in A_a$;
5: $\forall i \in A_a, \overline{X}_i := \frac{X_{m_i} + X_i}{m_i + 1}, m_i := m_i + 1$;
6: end for
7: // MAIN LOOP
8: while 1 do
9: $n := n + 1$;
10: Play an arm $a$ which solves the maximization problem

$$\max_{a \in \mathcal{F}} \sum_{i \in A_a} \left( f_i(a, \overline{X}_i) + f_i(a, \sqrt{\frac{(L + 1) \ln n}{m_i}}) \right) ; \quad (7.23)$$

11: $\forall i \in A_{a(n)}, \overline{X}_i := \frac{X_{m_i} + X_i}{m_i + 1}, m_i := m_i + 1$;
12: end while
We use two 1 by \( N \) vectors to store the information after we play an arm at each time slot. One is \((\overline{X}_i)_{1\times N}\) in which \(\overline{X}_i\) is the average (sample mean) of all the observed values of \(X_i\) up to the current time slot (obtained through potentially different sets of arms over time). The other one is \((m_i)_{1\times N}\) in which \(m_i\) is the number of times that \(X_i\) has been observed up to the current time slot. So CWF2 policy requires storage linear in \(N\).

### 7.4.2 Analysis of Regret

**Theorem 11.** Under the CWF2 policy, the expected total number of times that non-optimal power allocations are selected is at most

\[
\mathbb{E}[T^\phi_{\text{non}}(n)] \leq \frac{N(L + 1) \ln n}{B_{\text{min}}^2} + N + \frac{\pi^2}{3}LN, \tag{7.24}
\]

where \(B_{\text{min}}\) is a constant defined by \(\delta_{\text{min}}\) and \(L\); \(\delta_{\text{min}} = \min_{a, r_a < r^*} (r^* - r_a)\).

**Proof.** We will show the upper bound of the regret in three steps: (1) introduce a counter \(\tilde{T}_i(n)\) (defined as below) and show its relationship with the upper bound of the regret; (2) show the upper bound of \(\mathbb{E}[\tilde{T}_i(n)]\); (3) show the upper bound of \(\mathbb{E}[T^\phi_{\text{non}}(n)]\).

1. **The counter \(\tilde{T}_i(n)\)**

After the initialization period, \((\tilde{T}_i(n))_{1\times N}\) is introduced as a counter and is updated in the following way: at any time \(n\) when a non-optimal power allocation is selected, find \(i\) such that \(i = \arg\min_{j \in \mathcal{A}_a(n)} m_j\). If there is only one such power allocation, \(\tilde{T}_i(n)\) is increased by 1. If there are multiple such power allocations, we arbitrarily pick one, say
and increment $\tilde{T}_i$ by 1. Based on the above definition of $\tilde{T}_i(n)$, each time when a non-optimal power allocation is selected, exactly one element in $(\tilde{T}_i(n))_{1 \times N}$ is incremented by 1. So the summation of all counters in $(\tilde{T}_i(n))_{1 \times N}$ equals to the total number that we have selected the non-optimal power allocations, as below:

$$\sum_{a: R_a < R^*} E[T_a(n)] = \sum_{i=1}^{N} E[\tilde{T}_i(n)].$$

(7.25)

We also have the following inequality for $\tilde{T}_i(n)$:

$$\tilde{T}_i(n) \leq m_i(n), \forall 1 \leq i \leq N.$$  

(7.26)

(2) Show the upper bound of $E[\tilde{T}_i(n)]$

Let $C_{l,m_i}$ denote $\sqrt{(L+1)\ln l/m_i}$. Denote by $\tilde{I}_i(n)$ the indicator function which is equal to 1 if $\tilde{T}_i(n)$ is added by one at time $n$. Let $l$ be an arbitrary positive integer. Then, we could get the upper bound of $E[\tilde{T}_i(n)]$ as shown in (7.27), where $a(t)$ is defined as a non-optimal power allocation picked at time $t$ when $\tilde{I}_i(t) = 1$. Note that $m_i = \min_j \{m_j : \forall j \in A_{a(t)}\}$. We denote this power allocation by $a(t)$ since at each time that $\tilde{I}_i(t) = 1$, we could get different selections of power allocations.

$$E[\tilde{T}_i(n)] = \sum_{t=N+1}^{n} P\{\tilde{I}_i(t) = 1\} \leq l + \sum_{t=N+1}^{n} P\{\tilde{I}_i(t) = 1, \tilde{T}_i(t-1) \geq l\}$$
\[ \leq l + \sum_{t=N+1}^{n} P\left\{ \sum_{j \in A_{a(t)}} \left( f_j(a_j^*, \mathbf{X}_{j,m_j(t-1)}) + f_j(a_j^*, C_{t-1,m_j(t-1)}) \right) \right\} \]

\begin{equation}
\leq \sum_{j \in A_{a(t)}} \left( f_j(a_j(t), \mathbf{X}_{j,m_j(t-1)}) + f_j(a_j(t), C_{t-1,m_j(t-1)}) \right), \tilde{T}_i(t-1) \geq l \right\}. \tag{7.27}
\end{equation}

Note that \( l \leq \tilde{T}_i(t-1) \) implies \( l \leq \tilde{T}_i(t-1) \leq m_j(t-1), \forall j \in A_{a(t)} \). So we could get an upper bound of \( E[\tilde{T}_i(n)] \) as shown in (7.28), (7.29), (7.30), (7.31), where \( h_j (1 \leq j \leq \lvert A_{a*} \rvert) \) represents the \( j \)-th element in \( A_{a*} \); \( p_j (1 \leq j \leq \lvert A_{a(t)} \rvert) \) represents the \( j \)-th element in \( A_{a(t)} \); \( r^* = \sum_{j=1}^{\lvert A_{a*} \rvert} f_{h_j}(a_{h_j}^*, \theta_{h_j}) = \sum_{i \in A_{a*}} f_i(a_i, \theta_i) \); \( r_{a(t)} = \sum_{j=1}^{\lvert A_{a(t)} \rvert} f_{p_j}(a_{p_j}(t), \theta_{p_j}) = \sum_{i \in A_{a}} f_i(a_i, \theta_i) \).

\[ E[\tilde{T}_i(n)] \leq l + \sum_{t=N+1}^{n} P\left\{ \min_{0 < m_{h_1}, \ldots, m_{h_{\lvert A_{a*} \rvert}} < t} \sum_{j=1}^{\lvert A_{a*} \rvert} \left( f_{h_j}(a_{h_j}^*, \mathbf{X}_{h_j,m_{h_j}}) + f_{h_j}(a_{h_j}^*, C_{t-1,m_{h_j}}) \right) \right\} \]

\[ \leq \max_{t \leq m_{p_1}, \ldots, m_{p_{\lvert A_{a(t)} \rvert}} < t} \sum_{j=1}^{\lvert A_{a(t)} \rvert} \left( f_{p_j}(a_{p_j}(t), \mathbf{X}_{p_j,m_{p_j}}) + f_{p_j}(a_{p_j}(t), C_{t-1,m_{p_j}}) \right) \}

\[ \leq l + \sum_{t=2}^{\infty} \sum_{m_{h_1}=1}^{t-1} \sum_{m_{h_2}=1}^{t-1} \sum_{m_{h_3}=1}^{t-1} \cdots \sum_{m_{h_{\lvert A_{a*} \rvert}}=1}^{t-1} \sum_{m_{p_1}=t}^{\infty} \cdots \sum_{m_{p_{\lvert A_{a(t)} \rvert}}=t}^{\infty} \sum_{j=1}^{\lvert A_{a*} \rvert} \left( f_{h_j}(a_{h_j}^*, \mathbf{X}_{h_j,m_{h_j}}) + f_{h_j}(a_{h_j}^*, C_{t-1,m_{h_j}}) \right) \]

\[ \sum_{j=1}^{\lvert A_{a(t)} \rvert} \left( f_{p_j}(a_{p_j}(t), \mathbf{X}_{p_j,m_{p_j}}) + f_{p_j}(a_{p_j}(t), C_{t-1,m_{p_j}}) \right) \} \tag{7.28} \]

\footnote{These equations are on the next page due to the space limitations.}
So we have,

\[ E[\hat{T}_i(n)] \leq l + \sum_{t=2}^{\infty} \sum_{m_{h_1}=1}^{t-1} \ldots \sum_{m_{h_{l-1}|A^*}=1}^{t-1} \sum_{m_{p_1}=l}^{t-1} \sum_{m_{p_{l-1}|A_{a(t)}}=l}^{t-1} \]

\[ \sum_{j=1}^{0} f_{h_j}(a_{h_j}^*, \overline{X}_{h_j,m_{h_j}}) \leq r^* - \sum_{j=1}^{0} f_{h_j}(a_{h_j}^*, C_{t-1,m_{h_j}}). \] (7.29)

\[ \sum_{j=1}^{0} f_{p_j}(a_{p_j}(t), \overline{X}_{p_j,m_{p_j}}) \geq r_{a(t)} + \sum_{j=1}^{0} f_{p_j}(a_{p_j}(t), C_{t-1,m_{p_j}}). \] (7.30)

\[ r^* < r_{a(t)} + 2 \sum_{j=1}^{0} f_{p_j}(a_{p_j}(t), C_{t-1,m_{p_j}}) \} \] (7.31)

Now we show the upper bound of the probabilities for inequalities (7.29), (7.30) and (7.31) separately. We first find the upper bound of the probability for (7.29), as shown in (7.33).

\[ \sum_{j=1}^{0} \{ f_{h_j}(a_{h_j}^*, \overline{X}_{h_j,m_{h_j}}) \leq r^* - \sum_{j=1}^{0} f_{h_j}(a_{h_j}^*, C_{t-1,m_{h_j}}) \} \]

\[ = \sum_{j=1}^{0} \{ f_{h_j}(a_{h_j}^*, \overline{X}_{h_j,m_{h_j}}) + f_{h_j}(a_{h_j}^*, C_{t-1,m_{h_j}}) \} \leq \sum_{j=1}^{0} f_{h_j}(a_{h_j}^*, \theta_{h_j}) \}

\[ \leq \sum_{j=1}^{0} \{ f_{h_j}(a_{h_j}^*, \overline{X}_{h_j,m_{h_j}}) + f_{h_j}(a_{h_j}^*, C_{t-1,m_{h_j}}) \} \leq f_{h_j}(a_{h_j}^*, \theta_{h_j}) \}

\[ \leq \sum_{j=1}^{0} \{ f_{h_j}(a_{h_j}^*, \overline{X}_{h_j,m_{h_j}} + C_{t-1,m_{h_j}}) \} \leq f_{h_j}(a_{h_j}^*, \theta_{h_j}) \] (7.32)

\[ = \sum_{j=1}^{0} \{ \overline{X}_{h_j,m_{h_j}} + C_{t-1,m_{h_j}} \leq \theta_{h_j} \} \] (7.33)
Equation (7.32) holds because of lemma 1. So \( \forall j \),

\[
f_{h_j}(a_{h_j}^*, \overline{X}_{h_j,m_{h_j}} + C_{t-1,m_{h_j}}) \leq f_{h_j}(a_{h_j}^*, \overline{X}_{h_j,m_{h_j}}) + f_{h_j}(a_{h_j}^*, C_{t-1,m_{h_j}}). \tag{7.34}
\]

(7.33) holds because \( \forall i \), \( f_i(a_i, X_i) \) is a non-decreasing function in \( X_i \) for any \( X_i \geq 0 \).

In (7.33), \( \forall 1 \leq j \leq |A_{a^*}| \), applying the Chernoff-Hoeffding bound stated in Lemma 1, we could find the upper bound of each item as,

\[
P\{ \overline{X}_{h_j,m_{h_j}} + C_{t-1,m_{h_j}} \leq \theta_{h_j} \} \leq e^{-2 \cdot \frac{1}{m_{h_j}} \cdot (m_{h_j})^2 \cdot \frac{(L+1) \ln(t-1)}{m_{h_j}}} = (t-1)^{-2(L+1)}.
\]

Thus,

\[
P\{ \sum_{j=1}^{A_{a^*}} f_{h_j}(a_{h_j}^*, \overline{X}_{h_j,m_{h_j}}) \leq r^* - \sum_{j=1}^{A_{a^*}} f_{h_j}(a_{h_j}^*, C_{t-1,m_{h_j}}) \}
\]

\[
\leq |A_{a^*}| t^{-2(L+1)} \leq L(t-1)^{-2(L+1)}. \tag{7.35}
\]

Now we can get the upper bound of the probability for inequality (7.30), as shown in (7.36).

\[
P\{ \sum_{j=1}^{A_{a(t)}} f_{p_j}(a_{p_j}(t), \overline{X}_{p_j,m_{p_j}}) \geq r_{a(t)} + \sum_{j=1}^{A_{a(t)}} f_{p_j}(a_{p_j}(t), C_{t-1,m_{p_j}}) \}
\]

\[
= P\{ \sum_{j=1}^{A_{a(t)}} f_{p_j}(a_{p_j}(t), \overline{X}_{p_j,m_{p_j}}) \geq \sum_{j=1}^{A_{a(t)}} \left( f_{p_j}(a_{p_j}(t), \theta_{p_j}) + f_{p_j}(a_{p_j}(t), C_{t-1,m_{p_j}}) \right) \}
\]

\[
\leq \sum_{j=1}^{A_{a(t)}} P\{ f_{p_j}(a_{p_j}(t), \overline{X}_{p_j,m_{p_j}}) \geq f_{p_j}(a_{p_j}(t), \theta_{p_j}) + f_{p_j}(a_{p_j}(t), C_{t-1,m_{p_j}}) \}
\]

147
\[
\sum_{j=1}^{\left|A_{a(t)}\right|} P\{f_{p_j}(a_{p_j}(t), X_{p_j,m_{p_j}}) \geq f_{p_j}(a_{p_j}(t), \theta_{p_j} + C_{t-1,m_{p_j}})\} = \sum_{j=1}^{\left|A_{a(t)}\right|} P\{X_{p_j,m_{p_j}} \geq \theta_{p_j} + C_{t-1,m_{p_j}}\} \leq L(t - 1)^{-2(L+1)}. \tag{7.36}
\]

Equation (7.36) holds, following a similar reasoning as used to derive (7.35).

For all \(i\) and given any \(a_i\), since \(f_i(a_i, x)\) is an increasing, continuous function in \(x\), we could find a constant \(B_i(a_i)\) such that

\[
f_i(a_i, B_i(a_i)) = \frac{\delta_{\text{min}}}{2L}. \tag{7.37}
\]

Denote \(B_{\text{min}}(a) = \min_{i \in A_a} B_i(a_i)\). Then \(\forall i \in A_a\), we have

\[
f_i(a_i, B_{\text{min}}(a)) \leq \frac{\delta_{\text{min}}}{2L}. \tag{7.38}
\]

Note that for \(l \geq \left\lceil \frac{(L+1) \ln n}{B_{\text{min}}^2(a(t))} \right\rceil\),

\[
r^* - r_{a(t)} - 2 \sum_{j=1}^{\left|A_{a(t)}\right|} f_{p_j}(a_{p_j}(t), C_{t-1,m_{p_j}}) = r^* - r_{a(t)} - 2 \sum_{j=1}^{\left|A_{a(t)}\right|} f_{p_j}(a_{p_j}(t), \sqrt{(L + 1) \ln(t - 1) \over m_{p_j}})
\geq r^* - r_{a(t)} - 2 \sum_{j=1}^{\left|A_{a(t)}\right|} f_{p_j}(a_{p_j}(t), \sqrt{(L + 1) \ln n \over l}) \geq r^* - r_{a(t)} - 2 \sum_{j=1}^{\left|A_{a(t)}\right|} f_{p_j}(a_{p_j}(t), \sqrt{(L + 1) \ln n \over l}). \tag{7.39}
\]
\[ \geq r^* - r_{a(t)} = 2 \sum_{j=1}^{|A_{a(t)}|} f_{p_j}(a_{p_j}(t), B_{\min}(a(t))) \]

\[ \geq \delta_{a(t)} - 2 \sum_{j=1}^{|A_{a(t)}|} \frac{\delta_{\min}}{2L} \geq \delta_{a(t)} - \delta_{\min} \geq 0. \] 

(7.40)

So (7.31) is false when \( l \geq \left\lceil \frac{(L+1) \ln n}{B_{\min}^2(a(t))} \right\rceil \). We denote \( B_{\min} = \min_{a \in \mathcal{A}} B_{\min}(a(t)) \), and let \( l \geq \left\lceil \frac{(L+1) \ln n}{B_{\min}^2} \right\rceil \), then (7.31) is false for all \( a(t) \).

Therefore, we get the upper bound of \( \mathbb{E}[\tilde{T}_i(n)] \) as in (7.41).

\[ \mathbb{E}[\tilde{T}_i(n)] \leq \left\lceil \frac{(L+1) \ln n}{B_{\min}^2} \right\rceil + \sum_{t=2}^{\infty} \left( \sum_{m_{h_1}=1}^{t-1} \cdots \sum_{m_{p_1}=1}^{t-1} \sum_{m_{p_{|A_{a(t)}|}=l}}^{t-1} 2L(t-1)^{-2(L+1)} \right) \leq \frac{(L+1) \ln n}{B_{\min}^2} + 1 + L \sum_{t=1}^{\infty} 2t^{-2} \leq \frac{(L+1) \ln n}{B_{\min}^2} + 1 + \frac{\pi^2}{3} L. \] 

(7.41)

(3) Upper bound of \( \mathbb{E}[T_{\text{non}}^\phi(n)] \)

\[ \mathbb{E}[T_{\text{non}}^\phi(n)] = \sum_{a : R_a < R^*} \mathbb{E}[T_a(n)] = \sum_{i=1}^{N} \mathbb{E}[\tilde{T}_i(n)] \leq \frac{N(L+1) \ln n}{B_{\min}^2} + N + \frac{\pi^2}{3} LN. \] 

(7.42)
Remark 7. CWF2 can be used to solve the stochastic water-filling with objective $O_1$ as well if $\exists a^* \in O^*$, such that $\forall a \notin O^*$,

\[
\sum_{i \in A_{a^*}} f_i(a_i, \theta_i) > \sum_{j \in A_a} f_j(a_j, \theta_j). \quad (7.43)
\]

Then the regret of CWF2 is at most

\[
\mathcal{R}_{CWF2}(n) \leq \left[ \frac{N(L + 1) \ln n}{B_{\text{min}}^2} + N + \frac{\pi^2}{3} LN \right] \Delta_{\text{max}}, \quad (7.44)
\]

7.5 Applications and Simulation Results

7.5.1 Numerical Results for CWF1

![Figure 7.1: Normalized regret $\mathcal{R}(n) / \log n$ vs. $n$ time slots.](image)
We now show the numerical results for CWF2 policy. We consider a OFDM system with 4 subcarriers. We assume the bandwidth of the system is 4 MHz, and the noise density is $-80$ dBw/Hz. We assume Rayleigh fading with parameter $\sigma = (2, 0.8, 2.80, 32)$ for 4 subcarriers. We consider the following objective for our simulation:

\[
\begin{align*}
\text{max} & \quad \mathbb{E} \left[ \sum_{i=1}^{N} \log(1 + a_i(n)X_i(n)) \right] \\
\text{s.t.} & \quad \sum_{i=1}^{N} a_i(n) \leq P_{\text{total}}, \forall n
\end{align*}
\]

\[
\begin{align*}
a_1(n) \in \{0, 10, 20, 30\}, \forall n \\
a_2(n) \in \{0, 10, 20, 30\}, \forall n \\
a_3(n) \in \{0, 10, 20, 30, 40\}, \forall n \\
a_4(n) \in \{0, 10, 20\}, \forall n
\end{align*}
\]

where $P_{\text{total}} = 60$ mW (17.8 dBm). The unit for above power constraints from (7.47) to (7.50) is mW. Note that (7.46) to (7.50) define the constraint set $\mathcal{F}$.

For this scenario, there are 140 different choices of power allocations, and the optimal power allocation can be calculated as $(20, 20, 20, 0)$.

We compare the performance of our proposed CWF1 policy with UCB1 policy and LLR policy, as shown in Figure 7.1. As we can see from 7.1, naively applying UCB1 and LLR policy results in a worse performance than CWF1, since the UCB1 policy cannot exploit the underlying dependencies across arms, and LLR policy does not utilize the observations as efficiently as CWF1 does.
7.5.2 Numerical Results for CWF2

We show the simulation results of CWF2 using the same system as in 7.5.1.

We consider the following objective for our simulation:

\[
\max \left[ \sum_{i=1}^{N} \log (1 + a_i(n)\mathbb{E}[X_i(n)]) \right]
\]

s.t. \(a \in \mathcal{F}\) \hspace{1cm} (7.51)

where \(\mathcal{F}\) is same as in 7.5.1.

For this scenario, we assume Rayleigh fading with parameter \(\sigma = (1.23, 1.0, 0.55, 0.95)\) for 4 subcarriers. And the optimal power allocation can be calculated as \((20, 20, 0, 20)\).

![Figure 7.2: Numerical results of \(\mathbb{E}[\tilde{T}_i(n)]/\log n\) and theoretical bound.](image)

Figure 7.2 shows the simulation results of the total number of times that non-optimal power allocations are chosen by running CWF2 up to 30 million time slots. We also
show the theoretical upper bound in figure 7.2. In this case, we see that the theoretical upper bound is quite loose and the algorithm does much better in practice.

For this setting, we note that (7.43) is satisfied, since \((20, 20, 0, 20)\) also maximizes (7.45). So as stated in Remark 7, CWF2 can also be used to solve stochastic water filling with \(O_1\), with regret that grows logarithmically in time and polynomially in the number of channels.

We show a comparison of the UCB1 policy, LLR policy, CWF1 policy and CWF2 policy under this setting in Figure 7.3. We can see that CWF2 performs the best by far since it incorporate a way to exploit non-linear dependencies across arms, and learn more efficiently.

![Figure 7.3: Normalized regret \(\frac{\Omega(n)}{\log n}\) vs. \(n\) time slots.](image)
7.6 Summary

We have considered the problem of optimal power allocation over parallel channels with stochastically time-varying gain-to-noise ratios for maximizing information rate (stochastic water-filling) in this work. We approached this problem from the novel perspective of online learning. The crux of our approach is to map each possible power allocation into arms in a stochastic multi-armed bandit problem. The significant new challenge imposed here is that the reward obtained is a non-linear function of the arm choice and the underlying unknown random variables. To our knowledge there is no prior work on stochastic MAB that explicitly treats such a problem.

We first considered the problem of maximizing the expected sum rate. For this problem we developed the CWF1 algorithm. Despite the fact that the number of arms grows exponentially in the number of possible channels, we show that the CWF1 algorithm requires only polynomial storage and also yields a regret that is polynomial in the number of power levels per channel and the number of channels, and logarithmic in time.

We then considered the problem of maximizing the sum-pseudo-rate, where the pseudo rate for a stochastic channel is defined by applying the power-rate equation to its mean SNR \((\log(1 + E[SNR]))\). The justification for considering this problem is its connection to practice (where allocations over stochastic channels are made based on estimated mean channel conditions). Albeit sub-optimal with respect to maximizing the expected sum-rate, the use of the sum-pseudo-rate as the objective function is a more tractable approach. For this problem, we developed a new MAB algorithm that we call
CWF2. This is the first algorithm in the literature on stochastic MAB that exploits non-linear dependencies between the arm rewards. We have proved that the number of times this policy uses a non-optimal power allocation is also bounded by a function that is polynomial in the number of channels and power-levels, and logarithmic in time.

Our simulations results show that the algorithms we develop are indeed better than naive application of classic MAB solutions. We also see that under settings where the power allocation for maximizing the sum-pseudo-rate matches the optimal power allocation that maximizes the expected sum-rate, CWF2 has significantly better regret-performance than CWF1.

Because our formulations allow for very general classes of sub-additive reward functions, we believe that our technique may be much more broadly applicable to settings other than power allocation for stochastic channels. We would therefore like to identify and explore such applications in future work.
Chapter 8

Decentralized Learning for Opportunistic Spectrum Access

8.1 Overview

The fundamental problem of multiple secondary users contending for opportunistic spectrum access over multiple channels in cognitive radio networks has been formulated recently as a decentralized multi-armed bandit (D-MAB) problem. In a D-MAB problem there are \( M \) users and \( N \) arms (channels) that each offer i.i.d. stochastic rewards with unknown means so long as they are accessed without collision. The goal is to design distributed online learning policies that incur minimal regret. We consider two related problem formulations in this chapter\(^1\). First, we consider the setting where the users have a prioritized ranking, such that it is desired for the \( K \)-th-ranked user to learn to access the arm offering the \( K \)-th highest mean reward. For this problem, we present

\(^1\)This chapter is based in part on [33].
DLP, the first distributed policy that yields regret that is uniformly logarithmic over time without requiring any prior assumption about the mean rewards. Second, we consider the case when a fair access policy is required, i.e., it is desired for all users to experience the same mean reward. For this problem, we present DLF, a distributed policy that yields order-optimal regret scaling with respect to the number of users and arms, better than previously proposed policies in the literature. Both of our distributed policies make use of an innovative modification of the well-known UCB1 policy for the classic multi-armed bandit problem that allows a single user to learn how to play the arm that yields the $K$-th largest mean reward.

### 8.2 Problem Formulation

We consider a cognitive system with $N$ channels (arms) and $M$ decentralized secondary users (players). The throughput of $N$ channels are defined by random processes $X_i(n)$, $1 \leq i \leq N$. Time is slotted and denoted by the index $n$. We assume that $X_i(n)$ evolves as an i.i.d. random process over time, with the only restriction that its distribution have a finite support. Without loss of generality, we normalize $X_i(n) \in [0, 1]$. We do not require that $X_i(n)$ be independent across $i$. This random process is assumed to have a mean $\theta_i = E[X_i]$, that is unknown to the users and distinct from each other. We denote the set of all these means as $\Theta = \{\theta_i, 1 \leq i \leq N\}$.

At each decision period $n$ (also referred to interchangeably as time slot), each of the $M$ decentralized users selects an arm only based on its own observation histories under
a decentralized policy. When a particular arm $i$ is selected by user $j$, the value of $X_i(n)$ is only observed by user $j$, and if there is no other user playing the same arm, a reward of $X_i(n)$ is obtained. Else, if there are multiple users playing the same arm, then we assume that, due to collision, at most one of the conflicting users $j'$ gets reward $X_i(n)$, while the other users get zero reward. This interference assumption covers practical models in networking research, such as the perfect collision model (in which none of the conflicting users derive any benefit) and CSMA with perfect sensing (in which exactly one of the conflicting user derives benefit from the channel). We denote the first model as $M_1$ and the second model as $M_2$.

We denote the decentralized policy for user $j$ at time $n$ as $\phi_j(n)$, and the set of policies for all users as $\phi = \{\phi_j(n), 1 \leq j \leq M\}$. We are interested in designing decentralized policies, under which there is no information exchange among users, and analyze them with respect to regret, which is defined as the gap between the expected reward that could be obtained by a genie-aided perfect selection and that obtained by the policy. We denote $O_M^*$ as a set of $M$ arms with $M$ largest expected rewards. The regret can be expressed as:

$$R^\phi(\Theta; n) = n \sum_{i \in O_M^*} \theta_i - \mathbb{E}^\phi[\sum_{t=1}^n S_{\phi(t)}(t)]$$

(8.1)

where $S_{\phi(t)}(t)$ is the sum of the actual reward obtained by all users at time $t$ under policy $\phi(t)$, which could be expressed as:

$$S_{\phi(t)}(t) = \sum_{i=1}^N \sum_{j=1}^M X_i(t) I_{i,j}(t),$$

(8.2)
where for $M_1$, $I_{i,j}(t)$ is defined to be 1 if user $j$ is the only user to play arm $i$, and 0 otherwise; for $M_2$, $I_{i,j}(t)$ is defined to be 1 if user $j$ is the one with the smallest index among all users playing arm $i$ at time $t$, and 0 otherwise. Then, if we denote $V_{i,j}^\phi(n) = E[\sum_{t=1}^n I_{i,j}(t)]$, we have:

$$E^\phi[\sum_{t=1}^n S_{\phi(t)}(t)] = \sum_{i=1}^N \sum_{j=1}^M \theta_i E[V_{i,j}^\phi(n)]$$ (8.3)

Besides getting low total regret, there could be other system objectives for a given D-MAB. We consider two in this paper. In the prioritized access problem, we assume that each user has information of a distinct allocation order. Without loss of generality, we assume that the users are ranked in such a way that the $m$-th user seeks to access the arm with the $m$-th highest mean reward. In the fair access problem, users are treated equally to receive the same expected reward.

### 8.3 Selective Learning of the $K$-th Largest Expected Reward

We first propose a general policy to play an arm with the $K$-th largest expected reward $(1 \leq K \leq N)$ for classic multi-armed bandit problem with $N$ arms and one user, since the key idea of our proposed decentralized policies running at each user in section 8.4 and 8.5 is that user $m$ will run a learning policy targeting an arm with $m$-th largest expected reward.
Our proposed policy of learning an arm with $K$-th largest expected reward is shown in Algorithm 10.

**Algorithm 10** Selective learning of the $K$-th largest expected rewards (SL($K$))

1: // INITIALIZATION
2: for $t = 1$ to $N$ do
3: Let $i = t$ and play arm $i$;
4: $\hat{\theta}_i(t) = X_i(t)$;
5: $m_i(t) = 1$;
6: end for
7: // MAIN LOOP
8: while 1 do
9: $t = t + 1$;
10: Let the set $O_K$ contains the $K$ arms with the $K$ largest values in (8.4)
11: $\hat{\theta}_i(t - 1) + \sqrt{\frac{2 \ln t}{m_i(t - 1)}}$; (8.4)
12: Play arm $k$ in $O_K$ such that
13: $k = \arg \min_{i \in O_K} \hat{\theta}_i(t - 1) - \sqrt{\frac{2 \ln t}{m_i(t - 1)}}$; (8.5)
14: $\hat{\theta}_k(t) = \frac{\hat{\theta}_k(t-1)m_k(t-1)+X_k(t)}{m_k(t-1)+1}$,
15: $m_k(t) = m_k(t - 1) + 1$;
16: end while

We use two 1 by $N$ vectors to store the information after we play an arm at each time slot. One is $(\hat{\theta}_i)_{1 \times N}$ in which $\hat{\theta}_i$ is the average (sample mean) of all the observed values of $X_i$ up to the current time slot (obtained through potentially different sets of arms over time). The other one is $(m_i)_{1 \times N}$ in which $m_i$ is the number of times that $X_i$ has been observed up to the current time slot.
Note that while we indicate the time index in Algorithm 10 for notational clarity, it is not necessary to store the matrices from previous time steps while running the algorithm. So SL($K$) policy requires storage linear in $N$.

**Remark:** SL($K$) policy generalizes UCB1 in [13] and presents a general way to pick an arm with the $K$-th largest expected rewards for a classic multi-armed bandit problem with $N$ arms (without the requirement of distinct expected rewards for different arms).

Now we present the analysis of the upper bound of regret, and show that it is linear in $N$ and logarithmic in time. We denote $A_K$ as the set of arms with $K$-th largest expected reward. Note that Algorithm 10 is a general algorithm for picking an arm with the $K$-th largest expected reward for the classic multi-armed bandit problems, where we allow multiple arms with $K$-th largest expected reward, and all these arms retreated as optimal arms. The following theorem holds for Algorithm 10.

**Theorem 12.** Under the policy specified in Algorithm 10, the expected number of times that we pick any arm $i \notin A_K$ after $n$ time slots is at most:

$$\frac{8 \ln n}{\Delta_{K,i}} + 1 + \frac{2\pi^2}{3}. \quad (8.6)$$

where $\Delta_{K,i} = |\theta_K - \theta_i|$, $\theta_K$ is the $K$-th largest expected reward.

**Proof.** Denote $T_i(n)$ as the number of times that we pick arm $i \notin A_K$ at time $n$. Denote $C_{t,m_i}$ as $\sqrt{(L+1)\ln t}/m_i$. Denote $\hat{\theta}_{i,m_i}$ as the average (sample mean) of all the observed values.
of $X_i$ when it is observed $m_i$ time. $\mathcal{O}_K^*$ is denoted as the set of $K$ arms with $K$ largest expected rewards.

Denote by $I_i(n)$ the indicator function which is equal to 1 if $T_i(n)$ is added by one at time $n$. Let $l$ be an arbitrary positive integer. Then, for any arm $i$ which is not a desired arm, i.e., $i \notin A_K$:

$$T_i(n) = 1 + \sum_{t=N+1}^{n} 1\{I_i(t)\} \leq l + \sum_{t=N+1}^{n} 1\{I_i(t), T_i(t-1) \geq l\}$$

$$\leq l + \sum_{t=N+1}^{n} (1\{I_i(t), \theta_i < \theta_K, T_i(t-1) \geq l\} + 1\{I_i(t), \theta_i > \theta_K, T_i(t-1) \geq l\})$$

(8.7)

where $1(x)$ is the indicator function defined to be 1 when the predicate $x$ is true, and 0 when it is false.

Note that for the case $\theta_i < \theta_K$, arm $i$ is picked at time $t$ means that there exists an arm $j(t) \in \mathcal{O}_K^*$, such that $j(t) \notin \mathcal{O}_K$. This means the following inequality holds:

$$\hat{\theta}_{j(t),T_j(t)}(t-1) + C_{t-1,T_j(t)}(t-1) \leq \hat{\theta}_{i,T_i(t-1)}(t-1) + C_{t-1,T_i(t-1)},$$

(8.8)

Then, we have

$$\sum_{t=N+1}^{n} 1\{I_i(t), \theta_i < \theta_K, T_i(t-1) \geq l\}$$

$$\leq \sum_{t=N+1}^{n} 1\{\hat{\theta}_{j(t),T_j(t)}(t-1) + C_{t-1,T_j(t)}(t-1) \leq \hat{\theta}_{i,T_i(t-1)}(t-1) + C_{t-1,T_i(t-1)}, T_i(t-1) \geq l\}$$

(8.9)
\[
\sum_{t=N+1}^{n} \mathbb{I}\left\{ \min_{0 \leq m_j(t) < t} \hat{\theta}_{j(t),m_j(t)} + C_{t-1,m_j(t)} \leq \max_{l \leq m_i < t} \hat{\theta}_{i,m_i} + C_{t-1,m_i} \right\}
\]
\[
\leq \sum_{t=1}^{\infty} \sum_{m_j(t)=1}^{t-1} \sum_{m_i=l}^{t-1} \mathbb{I}\left\{ \hat{\theta}_{j(t),m_j(t)} + C_{t,m_j(t)} \leq \hat{\theta}_{i,m_i} + C_{t,m_i} \right\}
\]
\[
\hat{\theta}_{j(t),m_j(t)} + C_{t,m_j(t)} \leq \hat{\theta}_{i,m_i} + C_{t,m_i} \text{ implies that at least one of the following must be true:}
\]
\[
\hat{\theta}_{j(t),m_j(t)} \leq \theta_j(t) - C_{t,m_j(t)}, \tag{8.10}
\]
\[
\hat{\theta}_{i,m_i} \geq \theta_i + C_{t,m_i}, \tag{8.11}
\]
\[
\theta_j(t) < \theta_i + 2C_{t,m_i}. \tag{8.12}
\]

Applying the Chernoff-Hoeffding bound [70], we could find the upper bound of (8.10) and (8.11) as,
\[
\mathbb{P}\{ \hat{\theta}_{j(t),m_j(t)} \leq \theta_j(t) - C_{t,m_j(t)} \} \leq e^{-4 \ln t} = t^{-4}, \tag{8.13}
\]
\[
\mathbb{P}\{ \hat{\theta}_{i,m_i} \geq \theta_i + C_{t,m_i} \} \leq e^{-4 \ln t} = t^{-4} \tag{8.14}
\]

For \( l \geq \left\lceil \frac{8 \ln n}{\Delta_{K,i}} \right\rceil \),
\[
\theta_j(t) - \theta_i - 2C_{t,m_i} \geq \theta_K - \theta_i - 2\sqrt{\frac{2\Delta_{K,i}^2 \ln t}{8 \ln n}} \geq \theta_K - \theta_i - \Delta_{K,i} = 0, \tag{8.15}
\]
so (8.12) is false when \( l \geq \left\lceil \frac{8 \ln n}{\Delta_{K,i}} \right\rceil \). Note that for the case \( \theta_i > \theta_K \), when arm \( i \) is picked at time \( t \), there are two possibilities: either \( \mathcal{O}_K = \mathcal{O}_K^* \), or \( \mathcal{O}_K \neq \mathcal{O}_K^* \). If \( \mathcal{O}_K = \mathcal{O}_K^* \), the following inequality holds:
\[ \hat{\theta}_{i,T_i(t-1)} - C_{t-1,T_i(t-1)} \leq \hat{\theta}_{K,T_K(t-1)} - C_{t-1,T_K(t-1)}. \]

If \( O_K \neq O_K^* \), \( O_K \) has at least one arm \( h(t) \notin O_K^* \). Then, we have:

\[ \hat{\theta}_{i,T_i(t-1)} - C_{t-1,T_i(t-1)} \leq \hat{\theta}_{h(t),T_{h(t)}(t-1)} - C_{t-1,T_{h(t)}(t-1)}. \]

So to conclude both possibilities for the case \( \theta_i > \theta_K \), if we denote \( O_{K-1}^* = O_K^* - A_K \), at each time \( t \) when arm \( i \) is picked, there exists an arm \( h(t) \notin O_{K-1}^* \), such that

\[ \hat{\theta}_{i,T_i(t-1)} - C_{t-1,T_i(t-1)} \leq \hat{\theta}_{h(t),T_{h(t)}(t-1)} - C_{t-1,T_{h(t)}(t-1)}. \]  \hspace{1cm} (8.16)

Then similarly, we can have:

\[ \sum_{t=N+1}^{n} 1 \{ I_i(t), \theta_i > \theta_K, T_i(t-1) \geq l \} \]

\[ \leq \sum_{t=1}^{\infty} \sum_{m_i=l}^{t-1} \sum_{m_{h(t)}=l}^{t-1} 1 \{ \hat{\theta}_{i,m_i} - C_{t,m_i} \leq \hat{\theta}_{h(t),m_{h(t)} - C_{t,m_{h(t)}}} \} \]  \hspace{1cm} (8.17)

Note that \( \hat{\theta}_{i,m_i} - C_{t,m_i} \leq \hat{\theta}_{h(t),m_{h(t)} - C_{t,m_{h(t)}}} \) implies one of the following must be true:

\[ \hat{\theta}_{i,m_i} \leq \theta_i - C_{t,m_i}, \]  \hspace{1cm} (8.18)

\[ \hat{\theta}_{h(t),m_{h(t)}} \geq \theta_h(t) + C_{t,m_{h(t)}}, \]  \hspace{1cm} (8.19)

\[ \theta_i < \theta_h(t) + 2C_{t,m_i}. \]  \hspace{1cm} (8.20)
We again apply the Chernoff-Hoeffding bound and get $P\{\hat{\theta}_{i,m_i} \leq \theta_i - C_{t,m_i}\} \leq t^{-4}$,

$P\{\hat{\theta}_{h(t),m_{h(t)}} \geq \theta_{h(t)} + C_{t,m_{h(t)}}\} \leq t^{-4}$.

Also note that for $l \geq \left\lceil \frac{8 \ln n}{\Delta_{K,i}^2} \right\rceil$,

$$\theta_i - \theta_{h(t)} - 2C_{t,m_i} \geq \theta_i - \theta_K - \Delta_{K,i} \geq 0, \quad (8.21)$$

so (8.20) is false.

Hence, we have

$$E[T_i(n)] \leq \left\lceil \frac{8 \ln n}{\Delta_{K,i}^2} \right\rceil$$

$$+ \sum_{t=1}^{\infty} \sum_{m_{j(t)}=1}^{t-1} \sum_{m_i=1}^{t-1} (P\{\hat{\theta}_{j(t),m_{j(t)}} \leq \theta_{j(t)} - C_{t,m_{j(t)}}\} + P\{\hat{\theta}_{i,m_i} \geq \theta_i + C_{t,m_i}\})$$

$$+ \sum_{t=1}^{\infty} \sum_{m_i=\left\lceil \frac{8 \ln n}{\Delta_{K,i}^2} \right\rceil}^{t-1} \sum_{m_{h(t)}=1}^{t-1} (P\{\hat{\theta}_{i,m_i} \leq \theta_i - C_{t,m_i}\} + P\{\hat{\theta}_{h(t),m_{h(t)}} \geq \theta_{h(t)} + C_{t,m_{h(t)}}\})$$

$$\leq \frac{8 \ln n}{\Delta_{K,i}^2} + 1 + 2\sum_{t=1}^{\infty} \sum_{m_{j(t)}=1}^{t-1} \sum_{m_i=1}^{t-1} 2t^{-4}$$

$$\leq \frac{8 \ln n}{\Delta_{K,i}^2} + 1 + \frac{2\pi^2}{3}.$$

The definition of regret for the above problem is different from the traditional multi-armed bandit problem with the goal of maximization or minimization, since our goal now is to pick the arm with the $K$-th largest expected reward and we wish we could
minimize the number of times that we pick the wrong arm. Here we give two definitions of the regret to evaluate the SL($K$) policy.

**Definition 1.** We define the regret of type 1 at each time slot as the absolute difference between the expected reward that could be obtained by a genie that can pick an arm with $K$-th largest expected reward, and that obtained by the given policy at each time slot. Then the total regret of type 1 by time $n$ is defined as sum of the regret at each time slot, which is:

$$R_{\phi 1}^{\phi}(\Theta; n) = \sum_{t=1}^{n} |\theta_{K} - E^{\phi}[S_{\phi(t)}(t)]|$$  \hspace{1cm} (8.22)

**Definition 2.** We define the total regret of type 2 by time $n$ as the absolute difference between the expected reward that could be obtained by a genie that can pick an arm with $K$-th largest expected reward, and that obtained by the given policy after $n$ plays, which is:

$$R_{\phi 2}^{\phi}(\Theta; n) = |n\theta_{K} - E^{\phi}[\sum_{t=1}^{n} S_{\phi(t)}(t)]|$$  \hspace{1cm} (8.23)

Here we note that $\forall n$, $R_{\phi 2}^{\phi}(\Theta; n) \leq R_{\phi 1}^{\phi}(\Theta; n)$ because $|n\theta_{K} - E^{\phi}[\sum_{t=1}^{n} S_{\phi(t)}(t)]| = |n\theta_{K} - \sum_{t=1}^{n} E^{\phi}[S_{\phi(t)}(t)]| \leq \sum_{t=1}^{n} |\theta_{K} - E^{\phi}[S_{\phi(t)}(t)]|$.  

**Corollary 1.** The expected regret under both definitions is at most

$$\sum_{i : i \notin A_{k}} \left( \frac{8 \ln n}{\Delta_{K,i}} \right) + \left( 1 + \frac{2\pi^2}{3} \right) \sum_{i : i \notin A_{k}} \Delta_{K,i}.$$  \hspace{1cm} (8.24)
Proof. Under the SL($K$) policy, we have:

\[
\mathcal{R}_2^\phi(\Theta; n) \leq \mathcal{R}_1^\phi(\Theta; n) \\
= \sum_{t=1}^n |\theta_k - \mathbb{E}[S_{\phi(t)}(t)]| \\
= \sum_{i:i \notin A_k} \Delta_{K,i} \mathbb{E}[T_i(n)] \\
\leq \sum_{i:i \notin A_k} \left( \frac{8 \ln n}{\Delta_{K,i}} \right) + \left( 1 + \frac{2\pi^2}{3} \right) \sum_{i:i \notin A_k} \Delta_{K,i}.
\] (8.25)

Corollary 1 shows the upper bound of the regret of SL($K$) policy. It grows logarithmical in time and linearly in the number of arms.

### 8.4 Distributed Learning with Prioritization

We now consider the distributed multi-armed bandit problem with prioritized access. Our proposed decentralized policy for $N$ arms with $M$ users is shown in Algorithm 11.

In this algorithm, line 2 to 6 is the initialization part, for which user $m$ will play each arm once to have the initial value in $\left( \hat{\theta}_i^m \right)_{1 \times N}$ and $\left( m_i^m \right)_{1 \times N}$. Line 3 ensures that there will be no collisions among users. Similar as in Algorithm 10, we indicate the time index for notational clarity. Only two $1$ by $N$ vectors, $\left( \hat{\theta}_i^m \right)_{1 \times N}$ and $\left( m_i^m \right)_{1 \times N}$, are used by user $m$ to store the information after we play an arm at each time slot. We denote $o_m^*$ as
Algorithm 11 Distributed Learning Algorithm with Prioritization for $N$ Arms with $M$ Users Running at User $m$ (DLP)

```plaintext
1: // INITIALIZATION
2: for $t = 1$ to $N$ do
3:   Play arm $k$ such that $k = ((m + t) \mod N) + 1$;
4:   $\hat{\theta}^m_k(t) = X_k(t)$;
5:   $m^m_k(t) = 1$;
6: end for
7: // MAIN LOOP
8: while 1 do
9:   $t = t + 1$;
10:   Play an arm $k$ according to policy SL($m$) specified in Algorithm 10;
11:   $\hat{\theta}^m_k(t) = \frac{\hat{\theta}^m_k(t-1)m^m_k(t-1) + X_k(t)}{m^m_k(t-1) + 1}$;
12:   $m^m_k(t) = m^m_k(t-1) + 1$;
13: end while
```

the index of arm with the $m$-th largest expected reward. Note that $\{o^*_m\}_{1 \leq m \leq M} = O^*_M$.

Denote $\Delta_{i,j} = |\theta_i - \theta_j|$ for arm $i, j$. We now state the main theorem of this section.

**Theorem 13.** The expected regret under the DLP policy specified in Algorithm 11 is at most

$$
\sum_{m=1}^{M} \sum_{i \neq o^*_m} \left( \frac{8 \ln n}{\Delta^2_{o^*_m,i}} + 1 + \frac{2\pi^2}{3} \right) \theta_{o^*_m} + \sum_{m=1}^{M} \sum_{h \neq m} \left( \frac{8 \ln n}{\Delta^2_{o^*_h,o^*_m}} + 1 + \frac{2\pi^2}{3} \right) \theta_{o^*_m}. \quad (8.26)
$$

**Proof.** Denote $T_{i,m}(n)$ the number of times that user $m$ pick arm $i$ at time $n$.

For each user $m$, the regret under DLP policy can arise due to two possibilities: (1) user $m$ plays an arm $i \neq o^*_m$, (2) other user $h \neq m$ plays arm $o^*_m$. In both cases, collisions
may happen, resulting a loss which is at most $\theta_{o^*_m}$. Considering these two possibilities, the regret of user $m$ is upper bounded by:

$$\mathcal{R}^\phi(\Theta; m; n) \leq \sum_{i \neq o^*_m} \mathbb{E}[T_{i,m}(n)]\theta_{o^*_m} + \sum_{h \neq m} \mathbb{E}[T_{o^*_m,h}(n)]\theta_{o^*_m}$$  \hspace{1cm} (8.27)

From Theorem 12, $T_{i,m}(n)$ and $T_{o^*_m,h}(n)$ are bounded by

$$\mathbb{E}[T_{i,m}(n)] \leq \frac{8\ln n}{\Delta_{o^*_m,i}^2} + 1 + \frac{2\pi^2}{3}.$$ \hspace{1cm} (8.28)

$$\mathbb{E}[T_{o^*_m,h}(n)] \leq \frac{8\ln n}{\Delta_{o^*_h,o^*_m}^2} + 1 + \frac{2\pi^2}{3}.$$ \hspace{1cm} (8.29)

So,

$$\mathcal{R}^\phi(\Theta; m; n) \leq \sum_{i \neq o^*_m} \left( \frac{8\ln n}{\Delta_{o^*_m,i}^2} + 1 + \frac{2\pi^2}{3} \right)\theta_{o^*_m} + \sum_{h \neq m} \left( \frac{8\ln n}{\Delta_{o^*_h,o^*_m}^2} + 1 + \frac{2\pi^2}{3} \right)\theta_{o^*_m}$$  \hspace{1cm} (8.30)

The upper bound for regret is:

$$\mathcal{R}^\phi(\Theta; n) = \sum_{m=1}^{M} \mathcal{R}^\phi(\Theta; m; n) \leq \sum_{m=1}^{M} \sum_{i \neq o^*_m} \left( \frac{8\ln n}{\Delta_{o^*_m,i}^2} + 1 + \frac{2\pi^2}{3} \right)\theta_{o^*_m} + \sum_{m=1}^{M} \sum_{h \neq m} \left( \frac{8\ln n}{\Delta_{o^*_h,o^*_m}^2} + 1 + \frac{2\pi^2}{3} \right)\theta_{o^*_m}$$  \hspace{1cm} (8.31)
If we define $\Delta_{\min} = \min_{1 \leq i \leq N, 1 \leq j \leq M} \Delta_{i,j}$, and $\theta_{\max} = \max_{1 \leq i \leq N} \theta_i$, we could get a more concise (but looser) upper bound as:

$$R^\phi(\Theta; n) \leq M(N + M - 2)\left(\frac{8 \ln n}{\Delta_{\min}^2} + 1 + \frac{2\pi^2}{3}\right)\theta_{\max}.$$  \hspace{1cm} (8.32)

Theorem 13 shows that the regret of our DLP algorithm is uniformly upper-bounded for all time $n$ by a function that grows as $O(M(N + M) \ln n)$.

### 8.5 Distributed Learning with Fairness

For the purpose of fairness consideration, secondary users should be treated equally, and there should be no prioritization for the users. In this scenario, a naive algorithm is to apply Algorithm 11 directly by rotating the prioritization as shown in Figure 8.1. Each user maintains two $1 \times N$ vectors $(\hat{\theta}^m_{j,i})_{1 \times N}$ and $(m^m_{j,i})_{1 \times N}$, where the $j$-th row stores only the observation values for the $j$-th prioritization vectors. This naive algorithm is shown in Algorithm 12.

![Figure 8.1: Illustration of rotating the prioritization vector.](image-url)
Algorithm 12: A Naive Algorithm for Distributed Learning Algorithm with Fairness (DLF-Naive) Running at User $m$.

1: At time $t$, run Algorithm 11 with prioritization $K = ((m + t) \mod M) + 1$, then update the $K$-th row of $(\hat{\theta}^m)_{M \times N}$ and $(m^m)_{M \times N}$ accordingly.

We can see that the storage of Algorithm 12 grows linear in $MN$, instead of $N$. And it does not utilize the observations under different allocation order, which will result a worse regret as shown in the analysis of this section. To utilize all the observations, we propose our distributed learning algorithm with fairness (DLF) in Algorithm 13.

Algorithm 13: Distributed Learning Algorithm with Fairness for $N$ Arms with $M$ Users Running at User $m$ (DLF)

1: // INITIALIZATION
2: for $t = 1$ to $N$ do
3: Play arm $k$ such that $k = ((m + t) \mod N) + 1$;
4: $\hat{\theta}^m_k(t) = X_k(t)$;
5: $m^m_k(t) = 1$;
6: end for
7: // MAIN LOOP
8: while 1 do
9: $t = t + 1$;
10: $K = ((m + t) \mod M) + 1$;
11: Play an arm $k$ according to policy SL($K$) specified in Algorithm 10;
12: $\hat{\theta}^m_k(t) = \frac{\hat{\theta}^m_k(t-1)m^m_k(t-1)+X_k(t)}{m^m_k(t-1)+1}$;
13: $m^m_k(t) = m^m_k(t-1) + 1$;
14: end while

Same as in Algorithm 11, only two 1 by $N$ vectors, $(\hat{\theta}^m)_{1 \times N}$ and $(m^m)_{1 \times N}$, are used by user $m$ to store the information after we play an arm at each time slot.

Line 11 in Algorithm 13 means user $m$ play the arm with the $K$-th largest expected reward with Algorithm 10, where the value of $K$ is calculated in line 10 to ensure the
desired arm to pick for each user is different, and the users play arms from the estimated largest to the estimated smallest in turns to ensure the fairness.

**Theorem 14.** The expected regret under the DLF-Naive policy specified in Algorithm 12 is at most

\[
\sum_{o_m^* \in \mathcal{O}_n^*} \sum_{m=1}^M \sum_{i \neq o_m^*} \left( \frac{8 \ln \left[ n/M \right]}{\Delta_{o_m^*,i}^2} + 1 + \frac{2\pi^2}{3} \right) \theta_{o_m^*} + \sum_{o_m^* \in \mathcal{O}_n^*} \sum_{m=1}^M \sum_{h \neq m} \left( \frac{8 \ln \left[ n/M \right]}{\Delta_{o_h^*,o_m^*}^2} + 1 + \frac{2\pi^2}{3} \right) \theta_{o_m^*}.
\]

(8.33)

**Proof.** Theorem 14 is a direct conclusion from Theorem 13 by replacing \( n \) with \( \lceil n/M \rceil \), and then take the sum over all \( M \) best arms which are played in the algorithm.

The above theorem shows that the regret of the DLF-Naive policy grows as \( O(M^2(N+M) \ln n) \).

**Theorem 15.** The expected regret under the DLF policy specified in Algorithm 13 is at most

\[
M \sum_{i=1}^N \left( \frac{8 \ln n}{\Delta_{\min,i}^2} + 1 + \frac{2\pi^2}{3} \right) \theta_{\max} + M(M-1) \sum_{i \in \mathcal{O}_M^*} \left( \frac{8 \ln n}{\Delta_{\min,i}^2} + 1 + \frac{2\pi^2}{3} \right) \theta_i,
\]

(8.34)

where \( \Delta_{\min,i} = \min_{1 \leq m \leq M} \Delta_{o_m^*,i} \).

**Proof.** Denote \( K_m^*(t) \) as the index of the arm with the \( K \)-th (got by line 10 at time \( t \) in Algorithm 13 running at user \( m \)) largest expected reward. Denote \( Q_{m_i}^n(n) \) as the number of times that user \( m \) pick arm \( i \neq K_m^*(t) \) for \( 1 \leq t \leq n \).
We notice that for any arbitrary positive integer \( l \) and any time \( t \), \( Q_i^m(t) \geq l \) implies \( m_i(t) \geq l \). So (8.7) to (8.21) in the proof of Theorem 12 still hold by replacing \( T_i(n) \) with \( Q_i^m(n) \) and replacing \( K \) with \( K_m^*(t) \). Note that since all the channels are with different rewards, there is only one element in the set \( A_{K_m^*(t)} \).

To find the upper bound of \( \mathbb{E}[Q_i^m(n)] \), we should let \( l \) to be \( l \geq \left\lceil \frac{8 \ln n}{\Delta_{\text{min},i}^2} \right\rceil \) such that (8.12) and (8.20) are false for all \( t \). So we have,

\[
\mathbb{E}[Q_i^m(n)] \\
\leq \left[ \frac{8 \ln n}{\Delta_{\text{min},i}^2} \right] + \sum_{t=1}^{\infty} \sum_{m_j(t)=1}^{t-1} \sum_{m_i(t)=1}^{t-1} (P\{\hat{\theta}_{j(t),m_j(t)} \leq \theta_j(t) - C_{t,m_j(t)}\})
+ P\{\hat{\theta}_{i,m_i} \geq \theta_i + C_{t,m_i}\})
\]
\[
+ \sum_{t=1}^{\infty} \sum_{m_i(t)=\left(\frac{8 \ln n}{\Delta_{K_m^*(t),i}^2}\right)}^{t-1} \sum_{m_h(t)=1}^{t-1} (P\{\hat{\theta}_{i,m_i} \leq \theta_i - C_{t,m_i}\})
+ P\{\hat{\theta}_{h(t),m_h(t)} \geq \theta_h(t) + C_{t,m_h(t)}\})
\]
\[
\leq \frac{8 \ln n}{\Delta_{\text{min},i}^2} + 1 + 2 \sum_{t=1}^{\infty} \sum_{m_j(t)=1}^{t-1} \sum_{m_i(t)=1}^{t-1} 2t^{-4}
\]
\[
\leq \frac{8 \ln n}{\Delta_{\text{min},i}^2} + 1 + \frac{2\pi^2}{3}.
\]

Hence for user \( m \), we could calculate the upper bound of regret considering the two possibilities as in the proof of Theorem 13 as:

\[
\mathcal{R}^\phi(\Theta, m; n) \leq \sum_{i=1}^{N} Q_i^m(n)\theta_{\text{max}} + \sum_{h \neq m} \sum_{i \in \mathcal{O}_{K_m^*}^h} Q_h^m(n)\theta_i
\] (8.36)
So the upper bound for regret for $m$ users is:

\[
\mathcal{R}^\phi(\Theta; n) = \sum_{m=1}^{M} \mathcal{R}^\phi(\Theta, m; n)
\leq M \sum_{i=1}^{N} \left( \frac{8 \ln n}{\Delta_{\min, i}^2} + 1 + \frac{2\pi^2}{3} \right) \theta_{\max} + M(M - 1) \sum_{i \in \mathcal{O}_M^*} \left( \frac{8 \ln n}{\Delta_{\min, i}^2} + 1 + \frac{2\pi^2}{3} \right) \theta_i
\]

(8.37)

To be more concise, we could also write the above upper bound as:

\[
\mathcal{R}^\phi(\Theta; n) \leq M(N + M(M - 1)) \left( \frac{8 \ln n}{\Delta_{\min}^2} + 1 + \frac{2\pi^2}{3} \right) \theta_{\max}.
\]

(8.38)

**Theorem 16.** When time $n$ is large enough such that

\[
\frac{n}{\ln n} \geq \frac{8(N + M)}{\Delta_{\min}^2} + (1 + \frac{2\pi^2}{3}) N + M,
\]

(8.39)

the expected regret under the DLF policy specified in Algorithm 13 is at most

\[
M \sum_{i \notin \mathcal{O}_M^*} \left( \frac{8 \ln n}{\Delta_{\min, i}^2} + 1 + \frac{2\pi^2}{3} \right) \theta_{\max} + M^2(1 + \frac{2\pi^2}{3}) \theta_{\max} + M(M - 1)(1 + \frac{2\pi^2}{3}) \sum_{i \in \mathcal{O}_M^*} \theta_i.
\]

(8.40)

**Proof.** The inequality (8.35) implies that the total number of times that the desired arms are picked by user $m$ at time $n$ is lower bounded by $n - \sum_{i=1}^{N} \left( \frac{8 \ln n}{\Delta_{\min, i}^2} + 1 + \frac{2\pi^2}{3} \right)$. Since all
the arms with $M$ largest expected rewards are picked in turn by the algorithm, $\forall i \in O^*_M$, we have
\[
m_i(n) \geq \frac{1}{M} \left( n - \sum_{i=1}^{N} \frac{8 \ln n}{\Delta_{\min,i}^2} + 1 + \frac{2\pi^2}{3} \right).
\tag{8.41}
\]
where $m_i(n)$ refers to the number of times that arm $i$ has been observed up to time $n$ at user $m$. (For the purpose of simplicity, we omit $m$ in the notation of $m_i$.)

Note that when $n$ is big enough such that $\frac{n}{\ln n} \geq \frac{8(N+M)}{\Delta_{\min}} + (1 + \frac{2\pi^2}{3})N + M$, we have,
\[
m_i(n) \geq \frac{1}{M} \left( n - \sum_{i=1}^{N} \frac{8 \ln n}{\Delta_{\min,i}^2} + 1 + \frac{2\pi^2}{3} \right) \geq \frac{8 \ln n}{\Delta_{\min}}.
\tag{8.42}
\]

When (8.42) holds, both (8.12) and (8.20) are false. Then $\forall i \in O^*_M$, when $n$ is large enough to satisfy (8.42),
\[
\mathbb{E}[Q^m_i(n)] = \sum_{t=N+1}^{n} \mathbb{1}\{I_i(t)\}
= \sum_{t=N+1}^{n} (\mathbb{1}\{I_i(t), \theta_i < \theta_K\} + \mathbb{1}\{I_i(t), \theta_i > \theta_K\})
\leq \sum_{t=1}^{\infty} \sum_{m_{j(t)}=1}^{t-1} \sum_{m_i=1}^{\left\lceil (8 \ln n)/\Delta_{\min}^2 \right\rceil} (\mathbb{P}\{\hat{\theta}_{j(t),m_{j(t)}} \leq \theta_{j(t)} - C_{t,m_{j(t)}}\} + \mathbb{P}\{\hat{\theta}_{i,m_i} \geq \theta_i + C_{t,m_i}\})
+ \sum_{t=1}^{\infty} \sum_{m_i=1}^{\left\lceil (8 \ln n)/\Delta_{\min}^2 \right\rceil} \sum_{m_{h(t)}=1}^{t-1} (\mathbb{P}\{\hat{\theta}_{i,m_i} \leq \theta_i - C_{t,m_i}\} + \mathbb{P}\{\hat{\theta}_{h(t),m_{h(t)}} \geq \theta_{h(t)} + C_{t,m_{h(t)}}\})
\leq 1 + 2 \sum_{t=1}^{\infty} \sum_{m_{j(t)}=1}^{t-1} \sum_{m_i=1}^{t-1} 2t^{-4} \leq 1 + \frac{2\pi^2}{3}.
\tag{8.43}
\]
So when (8.42) is satisfied, a tighter bound for the regret in (8.34) is:

\[ R^\phi(\Theta; n) \leq M \sum_{i \in \mathcal{O}_M^*} \left( \frac{8 \ln n}{\Delta_{\min,i}^2} + 1 + \frac{2\pi^2}{3} \right) \theta_{\max} + M^2 \left(1 + \frac{2\pi^2}{3}\right) \theta_{\max} + M(M - 1) \left(1 + \frac{2\pi^2}{3}\right) \sum_{i \in \mathcal{O}_M^*} \theta_i. \]  

(8.44)

We could also write a concise (but looser) upper bound as:

\[ R^\phi(\Theta; n) \leq M(N - M) \left( \frac{8 \ln n}{\Delta_{\min}} + 1 + \frac{2\pi^2}{3} \right) \theta_{\max} + M^3 \left(1 + \frac{2\pi^2}{3}\right) \theta_{\max}. \]  

(8.45)

Comparing Theorem 14 with Theorem 15 and Theorem 16, if we define \( C = \frac{8(N + M)}{\Delta_{\min}^2} + (1 + \frac{2\pi^2}{3})N + M \), we can see that the regret of the naive policy DLF-Naive grows as \( O(M^2(N + M) \ln n) \), while the regret of the DLF policy grows as \( O(M(N + M^2) \ln n) \) when \( \frac{n}{\ln n} < C \), \( O(M(N - M) \ln n) \) when \( \frac{n}{\ln n} \geq C \). So when \( n \) is large, the regret of DLF grows as \( O(M(N - M) \ln n) \).

We also note that the following theorem has been shown in [8] on the lower bound of regret under any distributed policy.
Theorem 17 (Proposition 1 from [8]). The regret of any distributed policy $\phi$ is lower-bounded by

$$R^\phi(\Theta; n) \geq \sum_{m=1}^{M} \sum_{i \in O_m} \Delta_{\min, i} \mathbb{E}[Q^m_i].$$

(8.46)

Lai and Robbins [52] showed that for any uniformly good policy, the lower bound of $Q^m_i$ for a single user $i$ grows as $\Omega(\ln n)$. So DLF is a decentralized algorithm with finite-time order-optimal regret bound for fair access.

### 8.6 Simulation Results

We present simulation results for the algorithms developed in this work, varying the number of users and channels to verify the performance of our proposed algorithms detailed earlier. In the simulations, we assume channels are in either idle state (with throughput 1) or busy state (with throughput 0). The state of each $N$ channel evolves as an i.i.d. Bernoulli process across time slots, with the parameter set $\Theta$ unknown to the $M$ users.

Figure 8.2 shows the simulation results averaged over 50 runs using the three algorithms, DLP, DLF-Naive, and DLF, and the regrets are compared. Figure 8.2(a) shows the simulations for $N = 4$ channels, $M = 2$ users, with $\Theta = (0.9, 0.8, 0.7, 0.6)$. In Figure 8.2(b), we have $N = 5$ channels, $M = 3$ users, and $\Theta = (0.9, 0.8, 0.7, 0.6, 0.5)$. Figure 8.2(c) shows the simulations for $N = 7$ channels, and $M = 4$ users, with $\Theta = (0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3)$. 

177
(a) $N = 4$ channels, $M = 2$ secondary users, $\Theta = (0.9, 0.8, 0.7, 0.6)$.

(b) $N = 5$ channels, $M = 3$ secondary users, $\Theta = (0.9, 0.8, 0.7, 0.6, 0.5)$.

(c) $N = 7$ channels, $M = 4$ secondary users, $\Theta = (0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3)$.

Figure 8.2: Normalized regret $\frac{R(n)}{\ln n}$ vs. $n$ time slots.
As expected, DLF has the least regret, since one of the key features of DLF is that it does not favor any one user over another. The chance for each user to use any one of the $M$ best channels are the same. It utilizes its observations on all the $M$ best channels, and thus makes less mistakes for exploring. DLF-Naive not only has the greatest regret, also uses more storage. DLP has greater regret than DLF since user $m$ has to spend time on exploring the $M-1$ channels in the $M$ best channels expect channel $k \neq o_m^*$. Not only this results in a loss of reward, this also results in the collisions among users. To show this fact, we present the number of times that a channel is accessed by all $M$ users up to time $n = 10^6$ in Figure 8.3.

Figure 8.2 also explores the impact of increasing the number of channels $N$, and secondary users $M$ on the regret experienced by the different policies with the minimum distance between arms $\Delta_{\text{min}}$ fixed. It is clearly that as the number of channels and secondary users increases, the regret, as well as the regret gap between different algorithms increases.
In Figure 8.4, we compare the normalized regret $\frac{R_n}{\ln n}$ of DLF logarithm and the TDFS algorithm proposed by Liu and Zhao [59, 60], in a system with $N = 4$ channels and $M = 2$ secondary users. $\Theta = (0.9, 0.8, 0.7, 0.6)$. The results are got by averaging 50 runs up to half million time slots. We can see that compared with TDFS, our proposed DLF algorithm not only has a better theoretical upper bound of regret, it also performs better for practical use. Also, TDFS only works for problems with single-parameterized distribution. We don’t have this requirement for DFS. Besides, the storage of TDFS is not polynomial.

### 8.7 Summary

The problem of distributed multi-armed bandits is a fundamental extension of the classic online learning framework that finds application in the context of opportunistic spectrum access for cognitive radio networks. We have made two key algorithmic contributions to
this problem. For the case of prioritized users, we presented DLP, the first distributed policy that yields logarithmic regret over time without prior assumptions about the mean arm rewards. For the case of fair access, we presented DLF, a policy that yields order-optimal regret scaling in terms of the numbers of users and arms, which is also an improvement over prior results. Through simulations, we have further shown that the overall regret is lower for the fair access policy.
Chapter 9

Conclusions and Open Questions

In this dissertation, we have presented a new class of combinatorial multi-armed bandits and policies to exploit the structures of the dependencies to improve the cost of learning compared to prior work for large-scale stochastic network optimization problems in the following settings:

- i.i.d. rewards with linear dependencies;
- rested Markovian rewards with linear dependencies;
- restless Markovian rewards with linear dependencies;
- i.i.d. rewards with nonlinear dependencies;

We have shown that the dependencies can be handled tractably with policies that have provably good performance in terms of regret as well as storage and computation. Our proposed novel policies yield regret that grows polynomially in the number of unknown
parameters, dramatically improving the cost of learning compared to prior work for large-scale stochastic network optimization problems.

Besides these, we have proposed decentralized online learning algorithms running at each user to make a selection among multiple choices for classic multi-armed bandits for both prioritized and fair setting.

While the results in this dissertation have provided useful insights into real-world optimization with unknown variables, there are a number of interesting open questions and extension directions to be explore in the future.

- **Extensions on linear rewards:**

  The first open question is that while we have developed a particular policy LLR with a uniform bound on regret that grows as an $O(N^4 \ln n)$ function in Chapter 4, it is unclear what the lower-bound on regret for this problem is. It could be conjectured to be as low as $O(N \ln n)$; if this were to be true, then it should be possible to develop a more efficient policy than our current LLR policy. Then the upper bounds of regret of MLMR (proposed in Chapter 5) and CLRMR (proposed in Chapter 6) can also be further lowered.

- **Nonlinear rewards:**

  It would be of great interest to see if it is possible to also tackle more general nonlinear reward functions beyond the work in Chapter 7, at least in structured cases that have proved to be tractable in deterministic settings, such as convex functions.
• **Multidimensional rewards:**

There are many problems in communication networks that multiple performance objectives (e.g., delay, throughput, reliability, energy) are needed to be satisfied. It is of interested to find a point on the Pareto-optimal boundary set for such problem instead of a single optimal solution. So, a possible way is to consider rewards obtained at each time as a vector, and then maximize one of these rewards in expectation subject to average constraints on the other elements of the reward vector. An open problem here is to find out if it is possible to solve such a problem by using a convex linear combination of the rewards as a feedback for the learning.

• **Strict regret in special cases:**

In this dissertation, we have derived results for combinatorial MAB with restless and rested Markovian rewards with respect to the regret defined by the optimal arm as a static arm. In other work [26] and [66], we have derived results for the optimal policy for a two-state Bayesian restless multi-armed bandit with identical and non-identical arms with respect to the true regret where the genie will switch arms when playing optimally. Developing efficient learning policies for other cases remains an open problem. An alternative, possibly more efficient, approach to online learning in these kinds of problems might be to use the historical observations of each arm to estimate the $P$ matrix, and use these estimates iteratively in making arm selection decisions at each time. It is, however, unclear at present how to prove regret bounds using such an iterative estimation approach.
Bibliography


