ONLINE LEARNING IN STRUCTURED MARKOV DECISION PROCESSES

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

ELECTRICAL AND ELECTRONICS ENGINEERING

By

Nima Akbarzadeh

July 2017

ONLINE LEARNING IN STRUCTURED MARKOV DECISION PROCESSES
By Nima Akbarzadeh
July 2017

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Cem Tekin(Advisor)

Serdar Yüksel

Umut Orguner

Approved for the Graduate School of Engineering and Science:

Ezhan Karaşan
Director of the Graduate School

ABSTRACT

ONLINE LEARNING IN STRUCTURED MARKOV DECISION PROCESSES

Nima Akbarzadeh
M.S. in Electrical and Electronics Engineering
Advisor: Cem Tekin
July 2017

This thesis proposes three new multi-armed bandit problems, in which the learner proceeds in a sequence of rounds where each round is a Markov Decision Process (MDP). The learner's goal is to maximize its cumulative reward without any a priori knowledge on the state transition probabilities. The first problem considers an MDP with sorted states and a *continuation action* that moves the learner to an adjacent state; and a *terminal action* that moves the learner to a terminal state (goal or dead-end state). In this problem, a round ends and the next round starts when a terminal state is reached, and the aim of the learner in each round is to reach the goal state. First, the structure of the optimal policy is derived. Then, the regret of the learner with respect to an *oracle*, who takes optimal actions in each round is defined, and a learning algorithm that exploits the structure of the optimal policy is proposed. Finally, it is shown that the regret either increases logarithmically over rounds or becomes bounded. In the second problem, we investigate the personalization of a clinical treatment. This process is modeled as a goal-oriented MDP with dead-end states. Moreover, the state transition probabilities of the MDP depends on the context of the patients. An algorithm that uses the rule of optimism in face of uncertainty is proposed to maximize the number of rounds in which the goal state is reached. In the third problem, we propose an online learning algorithm for optimal execution in the limit order book of a financial asset. Given a certain amount of shares to sell and an allocated time to complete the transaction, the proposed algorithm dynamically learns the optimal number of shares to sell at each time slot of the allocated time. We model this problem as an MDP, and derive the form of the optimal policy.

Keywords: Online Learning, Markov Decision Process, Multi-armed Bandits, Reinforcement Learning, Dynamic Programming, Clinical Decision Making, Limit Order Book.

ÖZET

ÖZEL YAPILI MARKOV KARAR SÜREÇLERİNDE ÇEVRİMİÇİ ÖĞRENME

Nima Akbarzadeh
Elektrik ve Elektronik Mühendisliği, Yüksek Lisans
Tez Danışmanı: Cem Tekin
Temmuz 2017

Bu tez öğrenicinin sıralı turlarla hareket ettiği üç yeni çok kollu haydut problemi sunmaktadır. Her tur birer Markov Karar Süreci (MKS) olarak modellenmiştir. Öğrencinin amacı durum geçiş olasılıkları üzerinde herhangi ön bilgi olmadan toplam ödülü maksimize etmektir. İlk problem, sıralı durumların, öğreniciyi komşu bir duruma hareket ettiren *devam eylemleri*nin ve öğreniciyi amaç veya çıkmaz duruma götüren *sonlandırma eylemlerinin* olduğu bir MKSdir. Bu problemde, terminal duruma gelindiğinde tur sona erer ve bir sonraki tura geçilir. Her bir turda öğrenicinin hedefi amaç durumuna erişmektir. Öncelikle, en iyi poliçenin yapısı türetilmiştir. Sonrasında, öğrenicinin her turda en uygun aksiyonları alan kahin poliçeye göre pişmanlığı tanımlanmış ve en uygun poliçenin yapısından faydalanan bir öğrenme algoritması önerilmiştir. Son olarak, pişmanlığın tur sayısına göre logaritmik olarak arttığı veya sınırlı olduğu gösterilmiştir. İkinci problemde, kişiselleştirilmiş klinik tedaviler incelenmiştir. Bunlar amaç odaklı çıkmaz durumlu MKS olarak modellenmiştir. Bununla birlikle, MKS'nin durum geçiş olasılıkları hastanın bağlamıyla ilintilidir. Amaç durumuna erişen tur sayısını belirsizlik karşısında iyimserlik kuralını kullanarak maksimize eden bir algoritma geliştirilmiştir. Üçüncü problemde, limitli emir kitabında eniyi hisse satışı problemi ele alınmştır. Belirli miktardaki hissenin belirli bir süre içerisinde satılması gerektiğinde, algoritma, bu sürenin zaman aralıklarında satması gereken en uygun hisse sayısını dinamik olarak öğrenir. Bu problem bir MKS olarak modellenmiş ve en iyi poliçenin formu türetilmiştir.

Anahtar sözcükler: Çevrimiçi örenme, Markov Karar Süreci, Çok Kollu Haydutlar, Pekiştirmeli Örenme, Dinamik Programlama, Klinik Karar Verme, Limitli Emir Kitabı.

Acknowledgement

First of all, I would like to thank my advisor, Assist. Prof. Dr. Cem Tekin, for his excellent guidance throughout the M.Sc. study in Bilkent University. Without his continuous support, this accomplishment would not have been possible.

Next, I would like to thank my thesis defense jury members, Assoc. Prof. Dr. Serdar Yüksel and Assoc. Prof. Dr. Umut Orguner for reviewing my thesis and providing insightful comments.

In addition, I would like to thank our research group members who supported and helped me within the past two years. Also, I like to thank my Iranian friends who were like my brothers and sisters here in Turkey. I would like to thank my international friends in Bilkent University for being aside me and having fun together.

Last but not the least, I want to thank my family members for their relentless and spiritual love and support from the beginning years of my life up to now. They have always inspired me to stay motivated and overcome the difficulties.

Part of the research presented in this thesis (Chapters 2 and 3) is carried out within the scope of TUBITAK 2232 Program (Project no: 116C043).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis introduces three new Multi-armed bandit (MAB) problems, where each round involves a structured Markov Decision Process (MDP). Each round includes multiple decision epochs, where a learner takes an action upon observing the state of the system, which results in a new state or ends the current round. Prior works proposed algorithms for achieving a specific objective, e.g. maximizing the reward, reaching certain states etc., for which the learner needs to select a sequence of actions while observing the states of the system. However, in order to find the optimal mapping from the states to the actions, the learner needs to know all the parameters (state transition probabilities) of the MDP, which is not practical. While learning algorithms that achieve performance comparable to the optimal policy computed using the full knowledge of the state transition probabilities are proposed in the literature [1,2], these works assume that the underlying MDP is irreducible and focus on the notion of average reward optimality. Hence, the learning speed of these algorithms strictly depend on the size of the state space and the action set, which makes them impractical except for MDPs with small to moderate sizes. In this thesis, we depart from this line of work and focus on structured MDPs, for which we show that by knowing the structure (form of the optimal policy) of the MDP, the learner can learn much faster than prior works.

The contributions of this thesis are listed as follows:

- We design online learning algorithms for three structured MDPs in which the state transition probabilities are unknown a priori to the learner.
- For two of these, we obtain the form of the optimal policy and use this result to efficiently learn the optimal policy through repeated interaction with the environment.
- We derive regret bounds for one of the problems. Interestingly, based on the problem parameters, this regret bound is either finite or grows logarithmically in the number of rounds.
- We provide numerical analysis for our algorithms, which shows that they have superior performance compared to their competitors.

In the following sections, we review the literature on MDPs, MAB problem, and reinforcement and online learning. Then, we give a general overview of each problem and summarize our contributions to each problem.

1.1 Markov Decision Process

Markov Decision Process (MDP) is an essential tool in modeling decision problems in dynamically changing environments. In an MDP, the system contains a set of states, a set of actions, a set of rewards/costs associated with the set of states or the tuple of states and actions. In this model, actions cause state transitions, and at each state transition the learner may receive a reward or equivalently pay a cost. The reward/cost can be random or deterministic. In addition, the state transition caused by an action can be random or deterministic. If it is random, then there is a probability distribution over the state space given the state and the action selected in that state. MDPs have multifarious applications in economics, robotics, communications and health-care [3–6].

An MDP has an objective function that is required to be optimized. One possible objective is to maximize the long term average reward or to minimize the long term average cost [7, 8]. Another possible objective is to maximize the possibility of reaching a goal state (or a set of goal states) while minimizing the cost accumulated over the rounds before reaching the goal state. This type of MDP is called the goal-oriented MDP or the stochastic shortest path (SSP) problem [9].

In an MDP, a (deterministic) policy is a mapping from the state space into the action space. Different policies may result in different sequence of actions. The optimal policy is the one which optimizes the objective function by taking the optimal action in each state of the system. Therefore, given any state of the system, the learner seeks the optimal action. When the MDP parameters and the objective function is defined, the optimal or near-optimal solutions can be obtained by methods which are based on value iteration, heuristic search or dynamic programming [7, 8]. For SSP problems, the optimal solution exists if at least one *proper policy* can be found. Proper policy is the policy that reaches the goal state with probability one.

An interesting variation of the SSP problem, in which a proper policy does not necessarily exist is the SSP problem with Dead-end states (D) [10]. Kolobov *et. al.* has proposed various numerical solutions in order to find the optimal policy for problems of this kind [11, 12]. The objective function of these problems are defined as maximizing the probability of hitting G while avoiding D. The techniques developed for these problems require the knowledge of the state transition probabilities in computing the optimal policy.

An important application of MDPs is medical treatment recommendation [6]. In this application, the state space represents the patient health status or the degree of the illness, and therapies correspond to actions. A possible objective function is that the patient recovers from the illness, which is indicated by reaching to a "healthy" state. Another important application of MDPs is finance, where the state represents the market condition or the inventory level of the trader [3]. Investment options or the amount of shares to be traded can be regarded as possible actions. The objective function can be the profit made over a fixed period of time.

1.2 Multi-armed Bandits

MABs are used to model a variety of problems that involve sequential decision making under uncertainty. For instance, in telecommunications MAB can be used to model opportunistic spectrum access [13–15], in medical decision making MAB can be used to model clinical trials [16] and in recommendation systems MAB can be used to model web advertising [17]. In the classical MAB problem [18–20], the learner selects an action from the action set at each round, and receives a random reward that results from the selected action.[1] The learner's objective is to minimize its long-term expected loss (or maximize its long-term expected reward) by learning to select actions that have small losses (or large rewards). This task is not trivial due to the fact that the learner is unaware of the reward distribution of the actions beforehand. Numerous order-optimal index-based learning rules have been developed for the standard MAB problems in prior works [19, 22, 23]. These rules act myopically by choosing the action with the maximum index in each round. Moreover, the index of each action only depends on the past observations collected from that action.

Over the past two decades many variations of the MAB problem have been introduced and many different learning algorithms are proposed to solve these problems. The solutions include the celebrated Gittins index for Bayesian bandits [24], upper confidence bound policies (Normalized UCB, KL-UCB, UCB-1, UCB-2) [19, 22, 23], greedy policies [19] and posterior sampling [20, 25]. In all these problems, the goal is to balance *exploitation* and *exploration*, where exploitation means selecting actions with the goal of maximizing the reward based

[1]It is noteworthy to mention that in the MAB problem, when an action is taken in a round, the learner only observes the reward of the chosen action, and does not receive any feedback about the reward of the other actions. This is called *bandit feedback* [21]

on the current knowledge, and exploration means selecting actions with the goal of acquiring more information. An optimal algorithm needs to balance these two phases to maximize the learner's reward. Usually in MAB problems, an alternative performance metric, which is called the regret, is used to asses the performance of the learner. The regret simply measures the loss of the learner due to not knowing the reward distributions of the actions beforehand. For instance, in a MAB problem with K stochastic arms, the regret is defined as the difference between the total (expected) reward of an *oracle* who knows the true problem parameters (the expected reward of each arm) and acts optimally from the beginning, and the total reward of the learning algorithm used by the learner, which is unaware of the true problem parameters beforehand. It is shown that the regret grows logarithmically in the number of rounds for this problem [18]. Therefore, the average loss with respect to the oracle converges to zero, which shows that asymptotically, the learner achieves the same average reward with the oracle. A comprehensive discussion on MAB problems can be found in [21].

1.3 Reinforcement Learning

Situations that require multiple actions to be taken in each round cannot be modeled using conventional MAB. Hence, reinforcement learning in MDPs with bandit feedback is also considered in numerous works where the problem parameters are unknown.

The solution methods discussed in Section 1.1 require the full knowledge of the problem parameters. However, in some applications, such an assumption is impractical. For instance, this problem appears when a new stock enters the stock market or a new drug is introduced for a disease treatment. In these cases, the learner is unaware of the state transition probabilities of the actions. As a result, the learner is unable to calculate and apply the optimal policy from the beginning. Thus, a learning algorithm is required in order to learn the state transition probabilities, while minimzing the loss (maximizing the reward) of the learner.

Some prior works in reinforcement learning assume that the underlying MDP is unknown and ergodic, i.e., it is possible to reach all other states with a positive probability under any policy given any state [1, 2, 26–29], or deterministic[2] [30]. These works adopt the principle of optimism under uncertainty to choose a policy that minimizes the long-run average cost from a set of MDP models that are consistent with the estimated transition probabilities [1, 2] or estimated MDP parameters [26]. An MDP with deterministic state transitions and adversarial rewards is considered in [30]. In [31], the authors consider a sequential decision making problem in an unmodeled environment where the transition probability kernel is unknown. They use a learning method based on the Lempel-Ziv scheme of universal data compression and prediction with the aim of minimizing the long-term average loss. The mentioned related works use the history of observations to calculate the index, and use this index to select the next action that is believed to be the best.

In addition to the above methods, there exists other methods based on Q-learning, which is a model-free reinforcement learning approach. For these methods, the Q-values of state action pairs are updated after each round. In Q-learning, one possible way to balance exploration and exploitation is to use the ϵ-greedy strategy. However, unlike the methods based on the principle of optimism under uncertainty, in general, Q-learning based methods do not come with finite time performance bounds, i.e., they are only guaranteed to be optimal asymptotically. The interested reader may refer to [32] for a comprehensive discussion on Q-learning.

1.4 Contributions of Chapter 2

In Chapter 2 we propose a new online learning problem with a structured MDP model called the *gambler's ruin bandit problem* (GRBP). This model is inspired by the *gambler's ruin problem* (GRP). In the following subsection, we review the GRP.

[2]Deterministic MDP is an MDP with deterministic state transitions.

1.4.1 The Gambler's Ruin Problem

The GRP has been widely used in literature [33–36]. It can be viewed as the evolution of the wealth of a gambler betting on the success of independent trials. In this problem, the states are sorted on a straight line and the terminal states are set to lie at the ends of the line. At each time step, the state moves to the right or left of the current state with a certain probability (based on whether the gambler wins or loses the current trial). Hence, the GRP is a Markov chain. Given the initial state and the parameters, i.e., the number of states and state transition probabilities, the probability of winning (gambler finishing the game with a certain level of wealth) and ruin (gambler finishing the game bankrupt) has been calculated. Some other interesting aspects such as the exact and expected duration of the game has been investigated as well [37].

Numerous modifications have been proposed to the GRP. For instance, [35] considers four possible state transitions instead of two. The two extra state transitions moves the state to one of the terminal states, named as the Windfall state (goal state G) and the Catastrophic state (dead-end state D). The ruin and winning probabilities and the duration of the game are calculated based on these additional outcomes. In another model [38], modifications such as the chance of absorption in states other than G and D and staying in the same state are considered. The ruin and winning probabilities are calculated according to the proposed state transition model.

1.4.2 Problem Overview

Consider the following example which appears in medical treatment administration. Assume that patients arrive to the *intensive care unit* (ICU) sequentially in rounds. The initial health state of each patient is observed at the beginning of each round. We assume the illness degree defines the state. Treatment choices are assumed to be the actions that can be taken by the learner. An action shifts the patient's health state randomly over the state space according to an unknown distribution. This distribution defines the effectiveness of the treatment. For the state space, let *discharge* of the patient be the goal state and *death* of the patient to be the dead-end state. The problem objective is to maximize the expected number of patients that are discharged by learning the optimal treatment policy using the observations gathered from the previous patients. In the example given above, each round corresponds to a goal-oriented Markov Decision Process (MDP) with dead-ends [12] with fully observable states. The learner knows the state space, goal and dead-end states, but does not know the state transition distribution a priori. At each round, the learner chooses a sequence of actions and only observes the state transitions that result from the chosen actions.

Motivated by the health-care application described above, a new MAB problem is proposed in Chapter 2 in which multiple arms are selected in each round until a terminal state is reached. The set of terminal states consists of the *goal state* G and the *dead-end state* D. We assume that the terminal states are absorbing. This means that if they are reached, the current round ends. The remaining non-terminal (transient) states are assumed to be ordered between the goal and dead-end states. In each of the transient states, there are two possible actions: a *continuation action* (action C) that moves the learner randomly over the state space to the adjacent states around the current state; and a *terminal action* (action F) that moves the learner directly into a terminal state.

The problem discussed above is very similar to the classical GRP. The major difference is that a new action (action F) is introduced to the problem, and the learner has to decide the best action among the two available actions in each state. Therefore, we call this new MAB problem the *Gambler's Ruin Bandit Problem* (GRBP). In the GRBP, the system proceeds in a sequence of rounds $\rho \in \{1, 2, \ldots\}$. Each round is modeled as a goal-oriented MDP with a dead-end state (as in Fig. 1.1) where the state transition probabilities of all of the actions are unknown. Starting from a random, non-terminal initial state, the learner chooses a sequence of actions and observes the resulting state transitions until a terminal state is reached. If G is hit, the learner receives a unit reward and if D is hit, no reward is obtained in that round. Unlike other MDP models in which an immediate reward is revealed for each visited state, the reward in the GRBP is only revealed when a round ends. The goal of the learner is to maximize its cumulative expected reward over the rounds. In addition, there is no limit on the time duration of the rounds and the learner can take as many actions as possible until the time when a round ends.

In this problem, the optimal policy is the one which maximizes the probability of hitting the goal state from any given initial state. If the state transition probabilities are known, the optimal policy can be obtained by using value iteration based methods. We assume existence of an *omnipotent oracle* who knows the state transition probabilities and applies the optimal policy from the initial round. We define the regret of the learner by round ρ as the difference in the expected number of times the goal state is reached by the omnipotent oracle and the learner by round ρ.

First, we show that the optimal policy for the GRBP can be computed in a straightforward manner: there exists a threshold state above which it is always optimal to take action C and on or below which it is always optimal to take action F. Then, we propose an online learning algorithm for the learner, and bound its regret for two different regions that the actual state transition probabilities can lie in. The regret is bounded (finite) in one region, while it is logarithmic in the number of rounds in the other region. These bounds are problem specific, in the sense that they are functions of the state transition probabilities. Finally, we illustrate the behavior of the regret as a function of the state transition probabilities through numerical experiments.

Since the set of possible deterministic policies for the GRBP is exponential in the number of states, it is infeasible to use "myopic" algorithms developed for classical MAB problems [19, 23] to directly learn the optimal policy by experimenting different policies over rounds. In this inefficient approach each policy can be considered as a super-arm. In addition, the GRBP model does not fit into the combinatorial models proposed in prior works [39]. Therefore, a new learning methodology that exploits the structure of the GRBP is needed.

The contribution of Chapter 2 is summarized as follows:

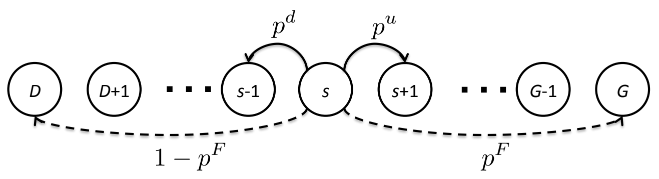

Figure 1.1: State transition model of the GRBP. Only state transitions out of state s are shown. Dashed arrows correspond to possible state transitions by taking action F, while solid arrows correspond to possible state transitions by taking action C. Weights on the arrows correspond to the state transition probabilities. The state transition probabilities for all other non-terminal states are the same as state s.

- We define a new MAB problem, called the GRBP, in which the learner takes a sequence of actions in each round with the objective of reaching the goal state.
- We show that using conventional MAB algorithms such as UCB1 [19] in the GRBP by enumerating all deterministic Markov policies is very inefficient and results in high regret.
- We prove that the optimal policy for the GRBP has a threshold form and the value of the threshold can be calculated in a computationally efficient way.
- We derive problem dependent bounds on the regret of the learner with respect to an omnipotent oracle that acts optimally. Unlike conventional MAB where the problem dependent regret grows at least logarithmically in the number of rounds [18], in the GRBP, regret can be either logarithmic or bounded, based on the values of the state transition probabilities. We explicitly define the condition on the state transition probabilities of the actions for which the regret is bounded.

1.5 Contributions of Chapter 3

1.5.1 Artificial Intelligence in Clinical Decision Making

As recent studies show, there exists healthcare problems, where the standard clinical procedure does not fit to the patients. For instance, a study conducted in the United States identifies treatment quality scores for various diseases, and shows that the treatment quality scores were 75.7% for the breast cancer patients and 45.4% for the diabetes mellitus patients [40]. As the number and complexity of the treatment methods have increased tremendously in the last decades, it became more difficult to match patients with the appropriate treatments. The recent advances in data science allows the development of artificial intelligence systems that provide patient-specific (personalized) diagnosis and treatment [41].

In [6], the treatment process of the patients is modeled as an MDP. In this model, the state transition probabilities of the MDP model is calculated by using the available data. After that, by using the estimated set of state transitions probabilities and backward induction, a policy which minimizes the treatment cost is obtained. Similar to this study, MDPs were also used to model post-operative observation methods of kidney transplantation, rectal cancer patients, and liver transplantation time prediction [42]. Some of the previous studies model the treatment process as a Partially Observable MDP (POMDP) since the patient states were partially observable. The methods used in the above studies form the system model and fix the decision mechanism only according to the training data. Therefore, the model is not updated while test data is being observed. On the contrary, the method presented in this chapter of thesis considers a treatment process where the reward of each therapy is unknown and should be learned over time. There is no training and test phase and the model becomes updated after observing each data instance.

Since each patient's response to a specific treatment might be different, similarities between patients need to be used so that the optimal personalized treatment regimen can be learned. Our method can group patients according to their contexts and learn in a common way for similar patients. With this, we expect the treatment performance to improve as more number of patients is being observed.

1.5.2 Problem Overview

In Chapter 3, we study a patient treatment process which is modeled as a Contextual Goal-oriented with Dead-ends Markov Decision Process (CGDMDP). CGDMDP is a generalization of the Goal Oriented Markov Decision Processes with Dead-ends [12] which is a fully observable MDP. In a CGDMDP, the state transition probabilities depend on the treatment option (the selected action) and the patient's external variables (age, sex, disease history, level of substance, etc.), which we collectively call as the context. Similar to the GRBP, there are goal and dead-end states in CGDMDP. Goal states represents the success and dead-end states represents the failure of the therapy. Similarly, we assume goal and dead-end states are absorbing states and the rest of the states are transient (non-absorbing). The initial state of the patient is one of the transient states. Our aim is to choose the optimal actions such that probability of absorption to the goal states is maximized or equivalently, the chance of being absorbed to a dead-end state is minimized. In literature, this problem is defined as MAXPROB and solved using a variant of value iteration, assuming that the state transition probabilities are known [11]. Our problem is different from the MAXPROB problem since we assume that the state transition probabilities are unknown and are dependent on the patient's context.

The contribution of Chapter 3 is summarized as follows:

- The patient treatment process is modeled as a CGDMDP.
- An algorithm (called CODY) that learns the optimal actions by estimating the state transition probabilities over time is developed.
- A simulator that mimics the actual disease model is created, and the impact

of using the proposed method on the 5-year survival rate for breast cancer is calculated.

The results obtained from this study show that the success rate of the treatments can be improved by using the proposed learning method.

1.6 Contributions of Chapter 4

In Chapter 4, we will discuss the third problem where the goal is to trade efficiently in limit order books. First of all, we give a brief overview on the limit order books.

1.6.1 Limit Order Book

Optimal execution of trades is an important problem in finance [43–46]. Once the decision has been made to sell a certain amount of shares the challenge often lies in how to optimally place this order in the market. The objective is to sell (buy) at the highest (lowest) price possible, while leaving minimal foot-print in the market. Specifically, we consider the selling problem in this chapter.

Our goal is to sell a specific number of shares of a given stock during a fixed time period in a way that minimizes the accumulated cost of the trade. This problem is also called the optimal liquidation problem. In this problem, the traders can specify the volume and the price of shares that they desire to sell in the limit order book (LOB) [3, 47].

Numerous prior works solve this problem using static optimization approaches or dynamic programming [45, 48]. Several other works tackle this problem using a reinforcement learning approach [3, 46, 49].

Reinforcement learning based methods consider various definitions of state, such as the remaining inventory, elapsed time, current spread, signed volume, etc. Actions are defined either as the volume to trade with a market order or as a limit order. A hybrid method is proposed in [46]: firstly, an optimization problem is solved to define an upper bound on the volume to be traded in each time slot, using the Almgren Chriss (AC) model proposed in [45]. Then, a reinforcement learning approach is used to find the best action, i.e., the volume to trade, which is upper-bounded by a relative value obtained in the optimization problem. Another prior work [3] implements the same approach with a different action set and state space. In all of the above works, the authors used Q-learning to find the optimal action for a given state of the system. In [3, 46] the learning problem is separated into training and test phases, where the Q values are only updated in the training phase, and then, these Q values are used in the test phase.

1.6.2 Problem Overview

Unlike prior approaches, we use a model based approach, in which we start with a market model, and then, learn the state transition dynamics of the model in an online manner. For this, we design an algorithm which runs in rounds. Specifically, we separate the state space into private and market variables. The private variable is the inventory level of the available shares to be sold for the remaining time in a round. We define the market variable as the difference in bid price of a time slot from the bid price of the time slot at the beginning of a round. This state model has not been used previously in reinforcement learning approaches. In this problem, the action is defined to be the amount of shares to be sold at a specific price (market order). This amount is upper-bounded by the limit obtained by AC model for each time slot. The algorithm selects actions by estimating the state transition probabilities of the market variables. At the beginning of each round, the state transition probabilities of the market variables are estimated based on the past observations.

We deduce the form of the optimal policy using the mentioned decomposition of the state variables and dynamic programming. Then, we show that the optimal policy for this problem is as follows: there exists a condition for each time slot where if it is satisfied, then the learner sells the maximum limit available for that time slot, otherwise, it does not sell any shares. By using the structure of the optimal policy, the number of actions to be learned is reduced and dynamic programming method is not implemented in every round to obtain the estimated policy. Hence, both optimization and learning speed up.

The contribution of Chapter 4 is summarized as follows:

- We propose a new model for LOB trade execution with private and market states.
- We show that the optimal policy has a special structure: At each time slot, the learner may decide to sell the suggested amount of shares by AC model or do nothing.
- We propose an online learning algorithm that greedily exploits the estimated optimal policy. Unlike other reinforcement learning based approaches [2, 50], this algorithm does not need explorations to learn the state transition probabilities.
- We show that the proposed algorithm provides significant performance improvement over other learning algorithms for LOB in real-world datasets.

Chapter 2

Gambler's Ruin Bandit Problem

The contents of this chapter has appeared in [51].

2.1 Problem Formulation

2.1.1 Definition of the GRBP

In the GRBP, the system is composed of a finite set of states. Let $\mathcal{S} := \{D, 1, \ldots, G\}$ denote the state space where integer $D = 0$ denotes the *dead-end* state and G denotes the *goal* state. Let $\tilde{\mathcal{S}} := \{1, \ldots, G-1\}$ be the set of *initial* (starting) states. The system operates in rounds and let $\rho = 1, 2, \ldots$ be the index of the round. The initial state of each round is drawn from a probability distribution $q(s), s \in \tilde{\mathcal{S}}$ over the set of initial states $\tilde{\mathcal{S}}$. The current round ends and the next round starts when the learner hits state D or G. Because of this, D and G are called *terminal states.* All other states are called non-terminal states. That is why we assume that $q(0) = q(G) = 0$ (no action is required for a round which starts in state 0 or G). Each round is divided into multiple time slots. The learner takes an action in each time slot from the action set $\mathcal{A} := \{C, F\}$ with the aim of reaching G. Here, C denotes the continuation action and F is the terminal action. Action C moves the learner one state to the right or to the left of the current state according to Fig. 1.1. Action F moves the learner directly to one of the terminal states. Possible outcomes of each action is shown in Fig. 1.1. Actions are taken in transient states. Let s_t^ρ denote the state at the beginning of the t^{th} time slot of round ρ. The state transition probabilities for action C are given by

$$\Pr(s_{t+1}^\rho = s+1 | s_t^\rho = s, C) = p^u$$
$$\Pr(s_{t+1}^\rho = s-1 | s_t^\rho = s, C) = p^d$$

where $t \geq 1,\ s \in \tilde{\mathcal{S}}$ and $p^u + p^d = 1$. The state transition probabilities for action F are given by

$$\Pr(s_{t+1}^\rho = G | s_t^\rho = s) = p^F,$$
$$\Pr(s_{t+1}^\rho = D | s_t^\rho = s) = 1 - p^F$$

where $s \in \tilde{\mathcal{S}}, t \geq 1$ and $0 < p^F < 1$. State transition probabilities are independent of time. If the state transition probabilities are known, each round can be modeled as an MDP and an optimal policy can be found by dynamic programming methods such as value iteration [8, 52].

2.1.2 Value Functions, Rewards and the Optimal Policy

Let $\pi = (\pi_1, \pi_2, \ldots)$ be the sequence of actions taken by policy π, where $\pi_t : \tilde{\mathcal{S}} \to \mathcal{A}$ is a mapping from state space to the action space for any $t \geq 1$. π represents a deterministic Markov policy. It is a stationary policy if $\pi_t = \pi_{t'}$ for all t and t' which means the policy does not change over rounds and the mapping is independent from time. For this case we will simply use $\pi : \tilde{\mathcal{S}} \to \mathcal{A}$ to denote a stationary deterministic Markov policy. As the time horizon is not finite and the state transition probabilities are not time-variant, it is sufficient to search for the optimal policy within the set of stationary deterministic Markov policies. The set of stationary deterministic Markov policies is denoted by Π. Let $V^\pi(s)$ denote the probability of reaching G by using policy π given that the system is in state s. $V^\pi(s)$ may also be called as the value function of policy π in state s.

Let $Q^\pi(s,a)$ denote the probability of reaching G by taking action a in state s, and then acting according to policy π afterward. We have

$$\begin{aligned} &Q^\pi(s,C) = p^u V^\pi(s+1) + p^d V^\pi(s-1), \\ &Q^\pi(s,F) = p^F, \ \ s \in \tilde{S}. \end{aligned} \tag{2.1}$$

for $s \in \tilde{\mathcal{S}}$. Then, $V^\pi(s)$, $s \in \tilde{\mathcal{S}}$ can be solved by using the following set of equations:

$$V^\pi(G) = 1, \ V^\pi(D) = 0, \ V^\pi(s) = Q^\pi(s, \pi(s)), \ \forall s \in \tilde{\mathcal{S}}$$

where $\pi(s)$ denotes the action selected by π in state s. The value of policy π (policy value) is defined as

$$V^\pi := \sum_{s \in \tilde{\mathcal{S}}} q(s) V^\pi(s).$$

The optimal policy is denoted by $\pi^* := \arg\max_{\pi \in \Pi} V^\pi$ and the value of the optimal policy is denoted by $V^* := \max_{\pi \in \Pi} V^\pi$. The optimal policy is characterized by Bellman optimality equations:

$$V^*(s) = \max\{p^F V^*(G), p^u V^*(s+1) + p^d V^*(s-1)\}, s \in \tilde{\mathcal{S}}. \tag{2.2}$$

As it is sufficient to search for the optimal policy within stationary deterministic Markov policies. As the possible number of actions that can be selected in each state of the system is two, hence the number of all such policies becomes 2^{G-1}. In Section 2.2, we will prove that the optimal policy for the GRBP has a simple threshold structure, which reduces the number of policies to learn from 2^{G-1} to 2.

2.1.3 Online Learning in the GRBP

As we described in the previous subsection, when the state transition probabilities are known, optimal solution and its probability of reaching the goal can be found by using Bellman optimality equations. When the learner does not know p^u and

p^F (a part of problem parameters which are state transition probabilities), the optimal policy cannot be computed a priori, and hence it has to be learned. We define the learning loss of the learner, who is not aware of the optimal policy a priori, with respect to an oracle, who knows the optimal policy from the initial round. The mathematical definition of the total regret is as follows:

$$\text{Reg}(T) := TV^* - \sum_{\rho=1}^{T} V^{\hat{\pi}_\rho}$$

where $\hat{\pi}_\rho$ denotes the policy that is used by the learner in round ρ. Let $N_\pi(T)$ denote the number of times policy π is used by the learner by round T. For any policy π, let $\Delta_\pi := V^* - V^\pi$ denote the suboptimality gap of that policy (loss of the policy with respect to the optimal policy). The given formulation of regret can be rewritten as

$$\text{Reg}(T) = \sum_{\pi \in \Pi} N_\pi(T) \Delta_\pi. \tag{2.3}$$

In Section 2.3 of this chapter, we will design a learning algorithm that minimize the growth rate of the expected regret, i.e., $\mathbb{E}[\text{Reg}(T)]$. A straightforward way to do this is to use UCB1 algorithm [19] or its variants [23] by taking each policy as an arm. The result below states a logarithmic bound on the expected regret when UCB1 is method is applied.

Theorem 1. *When UCB1 in [19] is used to select the policy to follow at the beginning of each round (with set of arms* Π*), we have*

$$\mathbb{E}[Reg(T)] = 8 \sum_{\pi : V^\pi < V^*} \frac{\log T}{\Delta_\pi} + \left(1 + \frac{\pi^2}{3}\right) \sum_{\pi \in \Pi} \Delta_\pi. \tag{2.4}$$

Proof. See [19]. □

As shown in Theorem 1, the expected regret of UCB-1 depends linearly on the number of suboptimal policies. For the GRBP, the number of policies can be very large. For instance, we have 2^{G-1} different stationary deterministic Markov policies for the defined problem. Thus, it can be inferred that using UCB1 to learn the optimal policy is highly inefficient for the GRBP. The learning algorithm we propose in Section 2.3 exploits a result on the form of the optimal policy that will be derived in Section 2.2 to learn the optimal policy in a fast manner. This learning algorithm calculates an estimated optimal policy using the estimated transition probabilities and the structure of the optimal policy, and hence learns much faster than applying UCB1 naively. Moreover, our learning algorithm can even achieve bounded regret (instead of logarithmic regret) under some special cases which will be discussed later.

2.2 Optimal Policy for the GRBP

In this section, we prove that the optimal policy for the GRBP has a threshold form. The threshold is a state where action F is taken for states equal or lower than the threshold. While for states upper than the threshold, action C is taken. The value of the threshold depends only on the state transition probabilities and the number of states. First, we give the definition of a *stationary threshold* policy. This definition is only specific to the GRBP problem.

Definition 1. *π is a stationary threshold policy if there exists $\tau \in \{0, 1, \ldots, G-1\}$ such that $\pi(s) = C$ for all $s > \tau$ and $\pi(s) = F$ for all $s \leq \tau$. We use π_τ^{tr} to denote the stationary threshold policy with threshold τ. The set of stationary threshold policies is given by $\Pi^{tr} := \{\pi_\tau^{tr}\}_{\tau=\{0,1,\ldots,G-1\}}$.*

Next, we will discuss two lemmas before the theorem in which we obtain the form of the optimal policy. The next lemma constrains the set of policies that the optimal policy lies in.

Lemma 1. *In the GRBP it is always optimal to select action C at $s \in \tilde{\mathcal{S}} - \{1\}$.*

Proof. See Appendix 2.6.1. □

The result that we have obtained in Lemma 1 holds regardless of the set of transition probabilities and the number of states. Lemma 1 leaves out only two candidates for the optimal policy. The first candidate is the policy which selects action C for all transient states, $\forall s \in \tilde{\mathcal{S}}$. The second candidate selects action C in all states except state 1. In state 1, action F will be selected. Hence, the set of the optimal policies is $\{\pi_0^{\text{tr}}, \pi_1^{\text{tr}}\}$. This result, considerably reduces the number of stationary threshold policies which can be candidate optimal policies from 2^{G-1} to 2. Let $r := p^d/p^u$ denote the *failure ratio* of action C. The following lemma gives $V^*(s)$ for $s \in \tilde{\mathcal{S}}$ for the two candidate optimal policies.

Lemma 2. *In the GRBP, the value functions are*
(i)

$$V^*(s) = \begin{cases} \dfrac{1-r^s}{1-r^G}, & \textit{when} \;\; p^u \neq p^d \\ \dfrac{s}{G}, & \textit{when} \;\; p^u = p^d \end{cases}$$

if $\pi^* = \pi_0^{tr}, \forall s \in \tilde{\mathcal{S}}$.
(ii)

$$V^*(s) = \begin{cases} p^F + (1-p^F)\dfrac{1-r^{s-1}}{1-r^{G-1}}, & \textit{when } p^u \neq p^d \\ p^F + (1-p^F)\dfrac{s-1}{G-1}, & \textit{when } p^u = p^d \end{cases}$$

if $\pi^* = \pi_1^{tr}, \forall s \in \tilde{\mathcal{S}}$.

Proof. See Appendix 2.6.2. □

Finally, the form of the optimal policy is given in the following theorem.

Theorem 2. *In the GRBP the optimal policy is* $\pi_{\tau^*}^{tr}$, *where*

$$\tau^* = \begin{cases} \operatorname{sign}(p^F - \dfrac{1-r}{1-r^G}), & \textit{If } p^u \neq p^d \\ \operatorname{sign}(p^F - \dfrac{1}{G}), & \textit{If } p^u = p^d \end{cases}$$

where $\operatorname{sign}(x) = 1$ *if* x *is nonnegative and* 0 *otherwise.*

Proof. See Appendix 2.6.3. □

When $p^u \neq p^d$ ($r \neq 1$), the term $(1-r)/(1-r^G)$ represents probability of hitting G starting from state 1 by always selecting action C. This probability is equal to $1/G$ when $p^u = p^d$ ($r = 1$). As p^F is the probability of hitting G by only taking action F, we compare the mentioned values with p^F to select the optimal policy. As the result, it might be optimal to take the terminal action in state 1 for some cases where we have $p^u > p^F$. The reason behind this is that although the continuation action can move the system state in the direction of the goal state for some time, the long term probability of hitting G by taking the continuation action can be lower than the probability of hitting G by immediately taking the terminal action at state 1.

Equation of the boundary region for which the optimal policy changes from π_0^{tr} to π_1^{tr} is

$$p^F = B(r) := \frac{1-r}{1-r^G} \tag{2.5}$$

when $r \neq 1$. This decision boundary is illustrated in Figure 2.1 for different values of G. We call the region of transition probabilities for which π_0^{tr} is optimal as the *exploration* region, and the region for which π_1^{tr} is optimal as the *no-exploration* region. In exploration region, the optimal policy does not take action F in any state of the system. Therefore, any learning policy that needs to learn how well action F performs, needs to explore action F. On the other hand, in no-exploration region, action F is taken if state 1 is visited. We know that there is a positive probability that action F is taken in a round. A rigorous proof of that is given in Lemma 5. Therefore, there is no need for exploration in this case. As the value of G increases, area of the exploration region decreases due to the fact that probability of hitting the goal state by only taking action C decreases.

In Section 2.3, we define a learning algorithm to learn the optimal policy. This algorithm applies greedy maximization in learning the optimal policy if the estimated optimal policy is π_1^{tr}, while it uses the separation of exploration and exploitation principle with control function $D(\rho)$ if the estimated optimal policy is π_0^{tr} (exploration region) to ensure that action F is taken sufficiently many times to have an accurate estimate about its transition probability. We will show that this algorithm achieves finite regret when the actual transition probabilities lie in the no-exploration region, and will achieve logarithmic regret when the actual transition probabilities lie in the exploration region. The choice of the control function is limited to the functions given in Theorem 3.

2.3 A Greedy Algorithm for the GRBP

In this section, we propose a learning algorithm that minimizes the regret when the state transition probabilities are unknown. The proposed algorithm forms estimates of state transition probabilities based on the history of state transitions, and then, uses these estimates together with the form of the optimal policy obtained in Section 2.2 to calculate an estimated optimal policy at each round.

The learning algorithm for the GRBP is called *Greedy Exploitation with Threshold Based Exploration* (GETBE) and its pseudocode is given in Algorithm 1. Unlike conventional MAB algorithms [18, 19, 23] which require all arms to be sampled at least logarithmically many times, GETBE does not need to sample all policies (arms) logarithmically many times to find the optimal policy with a sufficiently high probability. GETBE achieves this by utilizing the form of the optimal policy derived in the previous section. Although GETBE does not require all policies to be explored, it requires exploration of action F when the estimated optimal policy never selects action F. This *forced* exploration is done to guarantee that GETBE does not get stuck in the suboptimal policy. To illustrate better,

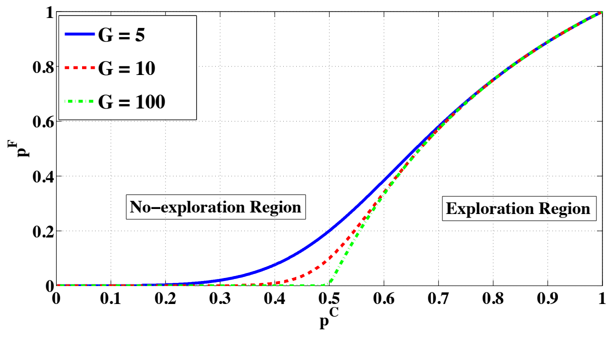

Figure 2.1: The boundary region

assume the true state transition probabilities fall in no-exploration region but the learner does not know this. Hence, when the learner is estimating the true values, the estimated optimal policy may fall into exploration region. When the policy of this region is applied, action F is not taken. Hence, the parameters relative to this action are not updated and thus, there is a chance that the estimated policy does not converge to the optimal one.

GETBE keeps counters $N_F^G(\rho)$, $N_F(\rho)$, $N_C^u(\rho)$ and $N_C(\rho)$: (i) $N_F^G(\rho)$ is the number of times action F is selected and terminal state G is entered upon selection of action F by the beginning of round ρ, (ii) $N_F(\rho)$ is the number of times action F is selected by the beginning of round ρ, (iii) $N_C^u(\rho)$ is the number of times transition from some state s to $s+1$ happened (i.e., the state moved up) after selecting action C by the beginning of round ρ, (iv) $N_C(\rho)$ is the number of times action C is selected by the beginning of round ρ. Let $T_F(\rho)$ and $T_C(\rho)$ represent the number of times action F and action C is selected in round ρ, respectively. Since, action F is a terminal action, it can be selected at most once in each round. However, action C can be selected multiple times in the same round. Let $T_F^G(\rho)$ and $T_C^u(\rho)$ represent the number of times state G is reached after selection of action F and the number of times the state moved up after selection of action C in round ρ, respectively.

At the beginning of round ρ, GETBE forms the transition probability estimates $\hat{p}_\rho^F := N_F^G(\rho)/N_F(\rho)$ and $\hat{p}_\rho^u := N_C^u(\rho)/N_C(\rho)$ that correspond to actions F and C, respectively. Then, it computes the estimated optimal policy $\hat{\pi}_\rho$ by using the form of the optimal policy given in Theorem 2 for the GRBP. If $\hat{\pi}_\rho = \pi_1^{\text{tr}}$, then GETBE operates in *greedy exploitation mode* by acting according to π_1^{tr} for the entire round. Else if $\hat{\pi}_\rho = \pi_0^{\text{tr}}$, then GETBE operates in *triggered exploration mode* and selects action F in the first time slot of that round if $N_F(\rho) < D(\rho)$, where $D(\rho)$ is a non-decreasing *control function* that is an input of GETBE. This control function helps GETBE to avoid getting stuck in the suboptimal policy by forcing the selection of action F, although it is suboptimal according to $\hat{\pi}_\rho$. When $N_F(\rho) \geq D(\rho)$, GETBE employs $\hat{\pi}_\rho$ for the entire round.

At the end of round ρ the values of counters are updated as follows:

$$\begin{aligned} N_F(\rho+1) &= N_F(\rho) + T_F(\rho), \\ N_F^G(\rho+1) &= N_F^G(\rho) + T_F^G(\rho) \\ N_C(\rho+1) &= N_C(\rho) + T_C(\rho), \\ N_C^u(\rho+1) &= N_C^u(\rho) + T_C^u(\rho). \end{aligned} \tag{2.6}$$

These values are used to estimate the transition probabilities that will be used at the beginning of round $\rho + 1$, for which the above procedure repeats. In the analysis of GETBE, we will show that when $N_F(\rho) \geq D(\rho)$, the probability that GETBE selects the suboptimal policy is very small, which implies that the regret incurred is very small.

Algorithm 1 GETBE Algorithm

```
 1: Input: $G, D(\rho)$
 2: Initialize: Take action $C$ and then action $F$ once to form initial estimates: $N_F^G(1), N_F(1) = 1, N_C^u(1), N_C(1) = 1$ (Round(s) to form the initial estimates (at most 2 rounds) are ignored in the regret analysis). $\rho = 1$
 3: while $\rho \geq 1$ do
 4:     Get initial state $s_1^\rho \in \tilde{\mathcal{S}}$, $t = 1$
 5:     $\hat{p}_\rho^F = \frac{N_F^G(\rho)}{N_F(\rho)}$, $\hat{p}_\rho^u = \frac{N_C^u(\rho)}{N_C(\rho)}$, $\hat{r}_\rho = \frac{1-\hat{p}_\rho^u}{\hat{p}_\rho^u}$
 6:     if $\hat{p}_\rho^u = 0.5$ then
 7:         $\hat{\tau}_\rho = \text{sign}(\hat{p}_\rho^F - 1/G)$
 8:     else
 9:         $\hat{\tau}_\rho = \text{sign}(\hat{p}_\rho^F - \frac{1-\hat{r}_\rho}{1-(\hat{r}_\rho)^G})$
10:     end if
11:     while $s_t^\rho \neq G$ or $D$ do
12:         if $(\hat{\tau}_\rho = 0 \ \&\& \ N_F(\rho) < D(\rho)) \ || \ (s_t^\rho \leq \hat{\tau}_\rho)$ then
13:             Select action $F$, observe state $s_{t+1}^\rho$
14:             $T_F(\rho) = T_F(\rho) + 1$, $T_F^G(\rho) = \mathbb{I}(s_{t+1}^\rho = G)$[1]
15:         else
16:             Select action $C$, observe state $s_{t+1}^\rho$
17:             $T_C(\rho) = T_C(\rho) + 1$, $T_C^u(\rho) = T_C^u(\rho) + \mathbb{I}(s_{t+1}^\rho = s_t^\rho + 1)$
18:             $t = t + 1$
19:         end if
20:     end while
21:     Update Counters according to (2.6)
22:     $\rho = \rho + 1$
23: end while
```

[1]$\mathbb{I}(\cdot)$ denotes the indicator function which is 1 if the expression inside evaluates true and 0 otherwise.

2.4 Regret Analysis

In this section, we will bound the (expected) regret of GETBE. We will show that GETBE achieves bounded regret when the true state transition probabilities lie in no-exploration region and logarithmic (in number of rounds) regret when the true state transition probabilities lie in exploration region. Based on Theorem 2, GETBE only needs to learn the optimal policy from the set of policies $\{\pi_0^{\text{tr}}, \pi_1^{\text{tr}}\}$. Using this fact and taking the expectation of (2.3), the expected regret of GETBE can be written as

$$\mathbb{E}[\text{Reg}(T)] = \sum_{\pi \in \{\pi_0^{\text{tr}}, \pi_1^{\text{tr}}\}} \mathbb{E}[N_\pi(T)]\Delta_\pi.$$

Let $\Delta(s) := |V^{\pi_1^{\text{tr}}}(s) - V^{\pi_0^{\text{tr}}}(s)|, s \in \tilde{\mathcal{S}}$ be the suboptimality gap in estimating the probability of hitting G when the system is in state s and let the maximum suboptimality gap be $\Delta_{\max} := \max_{s \in \tilde{\mathcal{S}}} \Delta(s)$. For the suboptimal policy π, we have $\Delta_\pi \leq \Delta_{\max}$. The next lemma gives a closed-form expression for $\Delta(s)$ and $\Delta_{\max}$.

Lemma 3. *We have*

$$\Delta(s) = \begin{cases} \frac{G-s}{G-1}|p^F - \dfrac{1}{G}| & \textit{if } r = 1 \\ \frac{r^{G-1} - r^{s-1}}{r^{G-1} - 1}|p^F - \dfrac{1-r}{1-r^G}| & \textit{if } r \neq 1 \end{cases}$$

and

$$\Delta_{\max} = \begin{cases} |p^F - \dfrac{1}{G}| & \textit{if } r = 1 \\ |p^F - \dfrac{1-r}{1-r^G}| & \textit{if } r \neq 1 \end{cases}$$

Proof. See Appendix 2.6.4. □

The next corollary characterizes the suboptimality gap in terms of $\Delta_{\max}$ when the initial distribution over the states is uniform. The result of this corollary can be used in Theorem 1 in order to obtain a bound the regret that depends on p^F and p^u.

Corollary 1. *If we assume a uniform distribution over the initial states, then the gap used in Theorem 1 is*

$$\Delta_\pi = \begin{cases} \dfrac{G}{2(G-1)}\Delta_{\max} & \text{if } r = 1 \\ \left(\dfrac{r^{G-1}}{r^{G-1}-1} + \dfrac{1}{(1-r)(G-1)}\right)\Delta_{\max} & \text{if } r \neq 1 \end{cases}$$

Proof. If $r = 1$, then we have

$$\begin{aligned} \Delta_\pi &= \sum_{s=1}^{G-1} q(s)\Delta(s) = \frac{1}{G-1}\sum_{s=1}^{G-1}\frac{G-s}{G-1}\Delta_{\max} \\ &= \frac{\Delta_{\max}}{G-1}\left(\frac{G(G-1)}{G-1} - \sum_{s=1}^{G-1}\frac{s}{G-1}\right) \\ &= \frac{\Delta_{\max}}{G-1}\left(G - \frac{\frac{G(G-1)}{2}}{G-1}\right) = \frac{G}{2(G-1)}\Delta_{\max}. \end{aligned}$$

If $r \neq 1$, then we get

$$\begin{aligned} \Delta_\pi &= \sum_{s=1}^{G-1} q(s)\Delta(s) = \frac{1}{G-1}\sum_{s=1}^{G-1}\frac{r^{G-1}-r^{s-1}}{r^{G-1}-1}\Delta_{\max} \\ &= \frac{\Delta_{\max}}{G-1}\left(\frac{(G-1)r^{G-1}}{r^{G-1}-1} - \sum_{s=1}^{G-1}\frac{r^{s-1}}{r^{G-1}-1}\right) \\ &= \frac{\Delta_{\max}}{G-1}\left(\frac{(G-1)r^{G-1}}{r^{G-1}-1} - \frac{\frac{r^{G-1}-1}{r-1}}{r^{G-1}-1}\right) \\ &= \left(\frac{r^{G-1}}{r^{G-1}-1} + \frac{1}{(1-r)(G-1)}\right)\Delta_{\max}. \end{aligned}$$

Finally, the proof is complete. □

Next, we will bound $\mathbb{E}[N_\pi(T)]$ for the suboptimal policy in a series of lemmas. At first, let δ be the minimum Euclidean distance of pair (p^u, p^F) from the boundary region $(x, B(x))$ given in Figure 2.1, where

$$B(x) := \frac{1 - \dfrac{1-x}{x}}{1 - (\dfrac{1-x}{x})^G}.$$

In the next lemma, we give the equation which has to be solved such that the value of δ can be achieved by that.

Lemma 4. *We have*

$$\delta = \sqrt{(x_0 - p^u)^2 + (B(x_0) - p^F)^2}$$

where $x_0 = 1/(1+r_0)$ *and* r_0 *is the positive, real-valued solution of*

$$p^F + \frac{(\frac{1-r^G}{r+1})^2(\frac{1}{r+1} - p^u)}{(G-1)r^G - Gr^{G-1} + 1} = \frac{1-r}{1-r^G}.$$

Proof. See Appendix 2.6.5. □

The value of δ found in Lemma 4 specifies the *hardness* of the GRBP. When δ is small, it is harder to distinguish the optimal policy from the suboptimal policy. If the pair of estimated transition probabilities $(\hat{p}^u_\rho, \hat{p}^F_\rho)$ in round ρ lies within a ball around (p^u, p^F) with radius less than δ, then GETBE will select the optimal policy in that round. The probability that GETBE selects the optimal policy is lower bounded by the probability that the estimated transition probabilities lie in a ball centered at (p^u, p^F) with radius δ. Therefore, if the true state transition probabilities lie close to the boundary, more samples are required to be taken so that we become more confident that the estimated policy is close to the optimal one.

The following lemma gives the probability of selecting each action in a round. It also provides a bound on the number of times an action is selected up to a certain round. We will use the result of this lemma while bounding the regret of GETBE.

Lemma 5. *Assume that the initial distribution over $\tilde{\mathcal{S}}$ is uniform* [2] *and $D(\rho)$ is a sublinear monotonic function in ρ.*
(i) *Let $p_{F,1}$ be the probability of taking action F in round ρ when $\hat{\pi}_\rho = \pi^{tr}_1$ and $p_{C,1}$ be the probability of taking action C at least once in round ρ when $\hat{\pi}_\rho = \pi^{tr}_1$. Then*

$$p_{C,1} = \frac{G-2}{G-1}, \; p_{F,1} = \begin{cases} \frac{G}{2(G-1)} & \textit{if } r = 1 \\ \frac{(G-1)(1-r)-r+r^G}{(G-1)(1-r)(1-r^{G-1})} & \textit{if } r \neq 1 \end{cases}.$$

[2] *We make the uniform initial distribution assumption to obtain a simple analytical form for the selection probabilities. Generalization of our results to arbitrary initial distributions is straightforward.*

(ii) *Let*

$$f_a(\rho) := \begin{cases} 0.5 p_{C,1}\rho, & \text{for } a = C \\ 0.5 p_{F,1}\lceil D(\rho)\rceil, & \text{for } a = F \end{cases}$$

and ρ'_C be the first round in which $0.5 p_{C,1}\rho_D - \sqrt{\rho_D \log \rho}$ becomes positive where $\rho_D = \rho - \lceil D(\rho)\rceil$ and ρ'_F be the first round in which $0.5 p_{F,1}\lceil D(\rho)\rceil - \sqrt{\lceil D(\rho)\rceil \log \rho}$ becomes positive. Then for $a \in \{F, C\}$ we have

$$\mathbb{P}\left(N_a(\rho) \leq f_a(\rho)\right) \leq \frac{1}{\rho^2}, \text{ for } \rho \geq \rho'_a.$$

The control function also has to satisfy the condition that $D(\rho) \geq (1/(p_{F,1})^2)\log \rho$.

Proof. See Appendix 2.6.6. □

Let A_ρ denote the event that a suboptimal policy is selected in round ρ. Let

$$C_\rho := \{|p^u - \hat{p}^u_\rho| \geq \delta/\sqrt{2}\} \cup \{|p^F - \hat{p}^F_\rho| \geq \delta/\sqrt{2}\}.$$

Then, using the union bound, we have

$$\mathbb{E}[\mathbb{I}(A_\rho)] \leq \mathbb{E}[\mathbb{I}(C_\rho)] \leq \mathbb{P}\left(|p^u - \hat{p}^u_\rho| \geq \delta/\sqrt{2}\right) + \mathbb{P}\left(|p^F - \hat{p}^F_\rho| \geq \delta/\sqrt{2}\right).$$

as a result of which we have

$$\mathbb{E}[\sum_{\rho=1}^{T} \mathbb{I}(A_\rho)] = \sum_{\rho=1}^{T} \mathbb{E}[\mathbb{I}(A_\rho)] \leq \sum_{\rho=1}^{T} \sum_{a \in \{F,C\}} \mathbb{P}\left(|p^{\mathrm{a}} - \hat{p}^{\mathrm{a}}_\rho| \geq \delta/\sqrt{2}\right).$$

Let $\mathrm{IR}(T)$ denote the number of rounds by round T in which the estimated transition probabilities lie in the incorrect side of the boundary. We have

$$\begin{aligned} \mathrm{IR}(T) &\leq \mathbb{E}[\sum_{\rho=1}^{T} \mathbb{I}(A_\rho)] \leq \mathbb{E}[\sum_{\rho=1}^{T} \mathbb{I}(A_\rho)] \\ &\leq \sum_{\rho=1}^{T} \sum_{a \in \{F,C\}} \mathbb{P}\left(|p^{\mathrm{a}} - \hat{p}^{\mathrm{a}}_\rho| \geq \delta/\sqrt{2}\right). \end{aligned} \tag{2.7}$$

Finally, we can decompose the regret into two: (i) regret in rounds in which the estimated transition probabilities lie in the incorrect side of the boundary, (ii) regret in rounds in which the estimated transition probabilities lie in the correct side of the boundary and GETBE explores. Let $\mathbb{I}_\rho^{\exp}$ be the indicator function for the exploration event of GETBE. Then,

$$\mathrm{Reg}(T) \leq \left(\mathrm{IR}(T) + \sum_{\rho=1}^{T} \mathbb{I}_\rho^{\exp} \right) \Delta_{\max}. \tag{2.8}$$

In the next theorem, we bound $\mathbb{E}[\mathrm{Reg}(T)]$.

Theorem 3. *Let* $x_1 := \left(1 + \sqrt{(16 p_{F,1}/\delta^2) + 1}\right)/2p_{F,1}$. *Assume that the control function is*

$$D(\rho) = \gamma \log \rho \textit{ where } \gamma > \max\{(x_1)^2, \frac{1}{(p_{F,1})^2}\}.$$

Let $\rho' := \max\{\rho'_C, \rho'_F\}$, *and*

$$w := 2\rho' + 12 + \frac{4}{p_{C,1}\delta^2}.$$

Then, the regret of GETBE is bounded by

$$\mathbb{E}[Reg(T) | \pi^* = \pi_1^{tr}] \leq w$$

and

$$\mathbb{E}[Reg(T) | \pi^* = \pi_0^{tr}] \leq \lceil D(T) \rceil + w\Delta_{\max}.$$

Proof. See Appendix 2.6.7. □

Theorem 3 states that after finite number of rounds, the optimal threshold will be identified with a high probability. Theorem 3 bounds the expected regret of GETBE as a function of time. When $\pi^* = \pi_1^{\mathrm{tr}}$, $\mathrm{Reg}(T) = O(1)$ since both actions will be selected with positive probability by the optimal policy at each round. When $\pi^* = \pi_0^{\mathrm{tr}}$, $\mathrm{Reg}(T) = O(D(T))$ since GETBE forces to explore action F logarithmically many times to avoid getting stuck in a suboptimal policy.

2.5 Numerical Results

We create a synthetic medical treatment selection problem based on [53]. Each state is assumed to be a stage of gastric cancer ($G = 4$, $D = 0$). The goal state is defined as at least three years of survival. Action C is assumed to be chemotherapy and action F is assumed to be surgery. For action C, p^u is determined by using the average survival rates for young and old groups at different stages of cancer given in [53]. For each stage, the survival rate of three years is taken to be the probability of hitting G by taking action C continuously. With this information, we set $p^u = 0.45$. Also, the five-year survival rate of surgery given in [54] (29%) is used to set $p^F = 0.3$.

The regrets shown in Fig. 3 and 4 correspond to different variants of GETBE, named as GETBE-SM, GETBE-PS and GETBE-UCB. Each variant of GETBE updates the state transition probabilities in a different way. GETBE-SM uses the control function together with sample mean estimates of the state transition probabilities as discussed in Algorithm 1. Unlike GETBE-SM, GETBE-UCB and GETBE-PS do not use the control function. GETBE-PS uses posterior sampling from the Beta distribution [25] to sample and update p^F and p^u. The mean of the Beta distribution corresponds to the sample mean estimation. The variance of the distribution decreases as more rounds being played. Therefore, more mass will be assigned to the mean value and hence the algorithm is expected to converge to the true value. In case of GETBE-UCB, an *inflation term* is added to the sample mean estimates of action a which is equal to

$$U_a(\rho) = \sqrt{\frac{2\log(N_F(\rho) + N_C(\rho))}{N_a(\rho)}}. \tag{2.9}$$

As the probabilities should sum up to 1, the estimated transition probability of each action is normalized as follows:

$$\hat{p}^a_\rho = \frac{\dfrac{N^*_a(\rho)}{N_a(\rho)} + U_a(\rho)}{1 + U_a(\rho)}, \ a \in \{F, C\}$$

where $N^*_F(\rho) := N^G_F(\rho)$ and $N^*_C(\rho) := N^u_C(\rho)$.

We compare GETBE with three other algorithms as well. PS-PolSelection and UCB-PolSelection algorithms treat each policy as a super-arm in multi-armed bandit problems. In PS-PolSelection, we use the same posterior sampling method as discussed, however, this time the algorithm is applied over the two candidate optimal policies rather than the state transition probabilities. In UCB-PolSelection, the same inflation term has been used as given in (2.9). For the two latter methods, instead of updating the state transition probabilities, the reward of each policy is updated directly. The final algorithm which is compared with GETBE is NoExplore algorithm. This algorithm computes a new policy at each round by solving the MAXPROB problem in [12] using the sample mean estimates of the state transition probabilities. There is no exploration mechanism for NoExplore.

Initial state distribution is taken to be the uniform distribution. Initial estimates of the transition probabilities are formed by setting $N_F(1) = 1$, $N_F^G(1) \sim$ Unif$[0, 1]$, $N_C(1) = 1$, $N_C^u(1) \sim$ Unif$[0, 1]$. The time horizon is taken to be 5000 rounds, and the control function is set to be $D(\rho) = 10 \log \rho$. Reported results are averaged over 100 iterations.

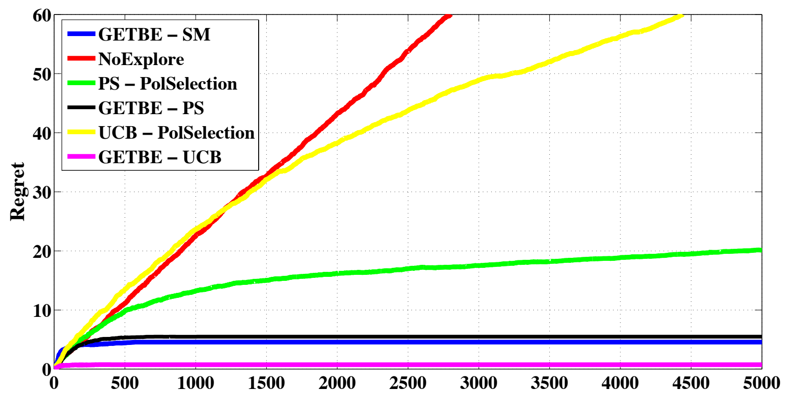

Figure 2.2: Regrets of GETBE and the other algorithms as a function of the number of rounds.

In Fig. 2.2 the regrets of GETBE and other algorithms are shown for p^F and p^u values given above. For this case, the the optimal policy is π_1^{tr}, and all variants of GETBE achieve finite regret, as expected. However, the regrets of UCB-PolSelection and PS-PolSelection increase logarithmically, since they sample each policy logarithmically many times. For this case, the regret of NoExplore algorithm grows linearly, because it cannot update $\hat{p}^F_\rho$ when the estimated optimal policy falls into the exploration region. This shows that NoExplore algorithm has been stuck in the suboptimal region.

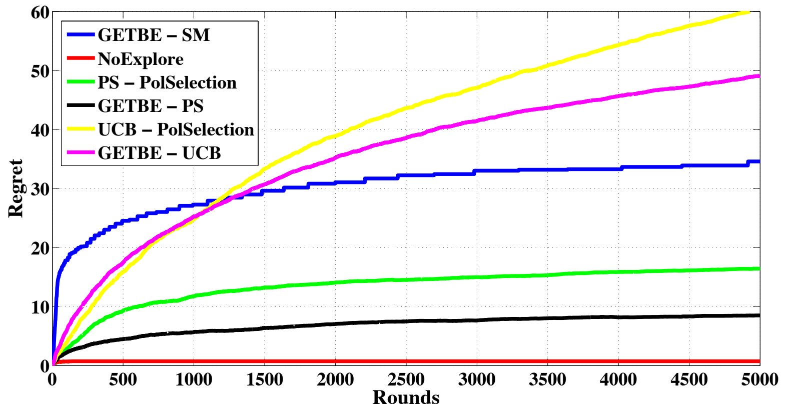

Figure 2.3: Regrets of GETBE and the other algorithms as a function of the number of rounds. when transition pair is in exploration region ($p^u = 0.65, p^F = 0.3, G = 4$).

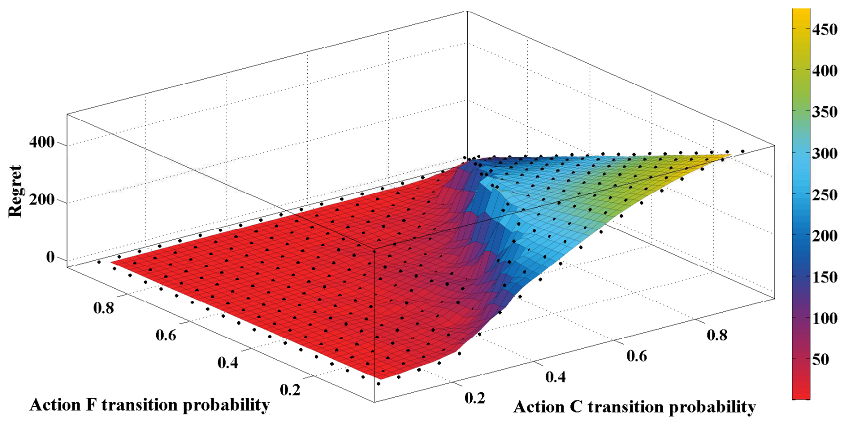

Figure 2.4: Regret of GETBE for different values of p^u, p^F.

Next, we set $p^u = 0.65$ and $p^F = 0.3$, in order to show how the algorithms perform when the optimal policy is π^{tr}_0. The results for this case are given in

Fig. 2.3. As expected, the regret grows logarithmically over the rounds for all variants of GETBE, PS-PolSelection and UCB-PolSelection. On the other hand, NoExplore achieves finite regret because its transition probability estimates are stuck in the optimal region and it does not explore the suboptimal policy.

In order to obtain a general overview about the performance of GETBE, we have plotted the regret of GETBE-SM as a function of p^F and p^u in Fig. 2.4. As the state transition probabilities shift from the no-exploration region to the exploration region the regret increases. A sharp growth in the regret happens near the boundary region. As it is clear from the figure, more cost for exploration (regret) has to be paid if the optimal region in exploration region and the state transition probabilities values are such that (p^u, p^F) is far away from the boundary region.

2.6 Proof of Theorems and Lemmas in Chapter 3

2.6.1 Proof of Lemma 1

By (2.2), for $s \in \tilde{\mathcal{S}} - \{1\}$ we have

$$V^*(s) = \max\{p^F, p^C V^*(s+1) + p^D V^*(s-1)\}.$$

If $V^*(s) = p^F$, this implies that

$$p^C V^*(s+1) + p^D V^*(s-1) \leq p^F \Rightarrow$$
$$V^*(s-1) \leq \frac{p^F - p^C V^*(s+1)}{p^D}.$$

By (2.2), we also have

$$p^F \leq V^*(s), \forall s \in \tilde{\mathcal{S}}. \tag{2.10}$$

Therefore,

$$V^*(s-1) \leq \frac{p^F - p^C V^*(s+1)}{p^D} \leq \frac{p^F - p^C p^F}{p^D} = p^F.$$

Calculations above imply that $V^*(s-1) = p^F$. Hence, we also obtain

$$V^*(s+1) \leq \frac{p^F - p^D V^*(s-1)}{p^C} = p^F.$$

Similar to the previous case, combining the above result with (2.10) results in $V^*(s+1) = p^F$. Consequently, if $V^*(s) = p^F$ for some $s \in \tilde{\mathcal{S}} - \{1\}$, then

$$V^*(s+1) = V^*(s-1) = p^F. \tag{2.11}$$

By (2.11) and considering all of the states in $\tilde{\mathcal{S}} - \{1\}$, if $V^*(s) = p^F$ for some $s \in \tilde{\mathcal{S}} - \{1\}$, then this implies that $V^*(G-1) = p^F$. Since $V^*(G) = 1$, we have

$$\begin{aligned} V^*(G-1) &= \max\{p^F, p^C + p^D p^F\} = p^F \\ &\Rightarrow p^F \geq p^C + p^D p^F \Rightarrow p^F(1-p^D) \geq p^C \\ &\Rightarrow p^F \geq 1 \Rightarrow p^F = 1. \end{aligned}$$

This shows that unless $p^F = 1$, it is suboptimal to select action F in states $\tilde{\mathcal{S}} - \{1\}$. Hence, it is always optimal to select action C at $s \in \tilde{\mathcal{S}} - \{1\}$.

2.6.2 Proof of Lemma 2

Case (i):

For π_1^{tr} we have:

$$\begin{cases} V^{\pi_1^{tr}}(G) = 1 \\ V^{\pi_1^{tr}}(G-1) = p^C V^{\pi_1^{tr}}(G) + p^D V^{\pi_1^{tr}}(G-2) \\ \dots \\ V^{\pi_1^{tr}}(2) = p^C V^{\pi_1^{tr}}(3) + p^D V^{\pi_1^{tr}}(1) \\ V^{\pi_1^{tr}}(1) = p^F \end{cases}$$

$$\Rightarrow \begin{cases} (p^C + p^D) V^{\pi_1^{tr}}(G-1) = p^C + p^D V^{\pi_1^{tr}}(G-2) \\ \dots \\ (p^C + p^D) V^{\pi_1^{tr}}(2) = p^C V^{\pi_1^{tr}}(3) + p^D p^F \end{cases}$$

$$\Rightarrow \begin{cases} p^C(V^{\pi_1^{tr}}(G-1) - 1) = p^D(V^{\pi_1^{tr}}(G-2) - V^{\pi_1^{tr}}(G-1)) \\ \dots \\ p^C(V^{\pi_1^{tr}}(s+1) - V^{\pi_1^{tr}}(s+2)) = p^D(V^{\pi_1^{tr}}(s) - V^{\pi_1^{tr}}(s+1)) \\ \dots \\ p^C(V^{\pi_1^{tr}}(2) - V^{\pi_1^{tr}}(3)) = p^D(p^F - V^{\pi_1^{tr}}(2)) \end{cases}$$

$$\Rightarrow \begin{cases} V^{\pi_1^{tr}}(G-1) - 1 = r^{G-2}(p^F - V^{\pi_1^{tr}}(2)) \\ \dots \\ V^{\pi_1^{tr}}(s) - V^{\pi_1^{tr}}(s+1) = r^{s-1}(p^F - V^{\pi_1^{tr}}(2)) \\ \dots \\ V^{\pi_1^{tr}}(2) - V^{\pi_1^{tr}}(3) = r(p^F - V^{\pi_1^{tr}}(2)) \end{cases} \tag{2.12}$$

We first consider the case $r \neq 1$. Summation of all the terms above results in

$$\begin{aligned}1 - V^{\pi_1^{tr}}(2) &= (V^{\pi_1^{tr}}(2) - p^F)(\sum_{i=1}^{G-2} r^i) \qquad (2.13)\\ \Rightarrow V^{\pi_1^{tr}}(2)(\sum_{i=0}^{G-2} r^i) &= 1 + p^F(\sum_{i=1}^{G-2} r^i)\\ \Rightarrow V^{\pi_1^{tr}}(2)(\sum_{i=0}^{G-2} r^i) &= 1 - p^F + p^F(\sum_{i=0}^{G-2} r^i)\\ \Rightarrow V^{\pi_1^{tr}}(2) &= p^F + \frac{1-p^F}{(\sum_{i=0}^{G-2} r^i)}\\ \Rightarrow V^{\pi_1^{tr}}(2) &= p^F + (1-p^F)\frac{1-r}{1-r^{G-1}}.\end{aligned}$$

Then, to obtain the value for the sth state, we sum up to the $(s-1)$th equation in (2.12):

$$\begin{aligned}V^{\pi_1^{tr}}(s) - V^{\pi_1^{tr}}(2) &= (V^{\pi_1^{tr}}(2) - p^F)(\sum_{i=1}^{s-2} r^i)\\ \Rightarrow V^{\pi_1^{tr}}(s) &= p^F + (V^{\pi_1^{tr}}(2) - p^F)(\sum_{i=0}^{s-2} r^i) \qquad (2.14)\\ \Rightarrow V^{\pi_1^{tr}}(s) &= p^F + (1-p^F)\frac{1-r^{s-1}}{1-r^{G-1}}. \qquad (2.15)\end{aligned}$$

When $r = 1$, continuing from (2.13), we obtain

$$V^{\pi_1^{tr}}(2) = p^F + (1-p^F)\frac{1}{G-1}$$

and

$$V^{\pi_1^{tr}}(s) = p^F + (1-p^F)\frac{s-1}{G-1}.$$

Case (ii):

Since action F is never selected by π_0^{tr}, for this case, standard analysis of the gambler's ruin problem applies. Thus, the probability of hitting G from state s is

$$\frac{1-r^s}{1-r^G} \qquad (2.16)$$

for $r \neq 1$, and s/G for $r = 1$ [55].

2.6.3 Proof of Theorem 2

Since we have found in Lemma 1 that it is always optimal to select action C when the state is in $\{2, \ldots, G-1\}$, to find the optimal policy, it is sufficient to compare the value functions of the two policies for $s = 1$. When $r \neq 1$, this gives $\pi^* = \pi_1^{tr}$ if

$$\frac{1-r}{1-r^G} < p^F$$

and $\pi^* = \pi_0^{tr}$ otherwise.[3] Similarly, when $r = 1$ and $1/G < p^F$, then $\pi^* = \pi_1^{tr}$. Otherwise, $\pi^* = \pi_0^{tr}$. Using these, the value of the optimal threshold is given as

$$\tau^* = \begin{cases} \operatorname{sign}(p^F - \frac{1-r}{1-r^G}) & \text{when } r \neq 1 \\ \operatorname{sign}(p^F - \frac{1}{G}) & \text{when } r = 1 \end{cases}$$

which completes the proof.

2.6.4 Proof of Lemma 3

According to Lemma 2 we have

Case (i) $r = 1$:

$$\begin{aligned} \Delta(s) = |V^{\pi_1^{tr}}(s) - V^{\pi_0^{tr}}(s)| &= |p^F + (1-p^F)\frac{s-1}{G-1} - \frac{s}{G}| \\ &= |p^F(\frac{G-s}{G-1}) + \frac{s-1}{G-1} - \frac{s}{G}| = \frac{G-s}{G-1}|p^F - \frac{1}{G}|. \end{aligned}$$

The above equation is maximized when $s = 1$. Therefore, when $r = 1$,

$$\Delta_{\max} = |p^F - \frac{1}{G}|.$$

[3]When $(1-r)/(1-r^G) = p^F$ both π_1^{tr} and π_0^{tr} are optimal. For this case, we favor π_1^{tr} because it always ends the current round.

Case (ii) $r \neq 1$:

$$\begin{aligned}
\Delta(s) &= |V^{\pi_1^{tr}}(s) - V^{\pi_0^{tr}}(s)| \\
&= |p^F + (1-p^F)\frac{r^{s-1}-1}{r^{G-1}-1} - \frac{r^s-1}{r^G-1}| \\
&= |p^F(\frac{r^{G-1}-r^{s-1}}{r^{G-1}-1}) + \frac{r^{s-1}-1}{r^{G-1}-1} - \frac{r^s-1}{r^G-1}| \\
&= |p^F(\frac{r^{G-1}-r^{s-1}}{r^{G-1}-1}) + \frac{r^s - r^{s-1} + r^{G-1} - r^G}{(1-r^{G-1})(1-r^G)}| \\
&= (\frac{r^{G-1}-r^{s-1}}{r^{G-1}-1})|p^F - \frac{1-r}{1-r^G}|.
\end{aligned}$$

Again, the above equation is maximized when $s = 1$. Therefore, when $r \neq 1$,

$$\Delta_{\max} = |p^F - \frac{1-r}{1-r^G}|.$$

2.6.5 Proof of Lemma 4

We define $L(x) = \sqrt{(x-p^C)^2 + (B(x)-p^F)^2}$ as the distance between any point on the surface to the boundary. In order to find its minimum value, we set

$$\begin{aligned}
\frac{dL(x)}{dx} = 0 &\Rightarrow (x-p^C) + (B(x)-p^F)\frac{dB(x)}{dx} = 0 \\
&\Rightarrow B(x) = p^F + \frac{p^C - x}{\frac{dB(x)}{dx}} \qquad (2.17)
\end{aligned}$$

We use the chain rule to obtain $\frac{dB(x)}{dx}$. We have

$$\frac{dB(x)}{dr} = -\frac{(G-1)r^G - Gr^{G-1} + 1}{(1-r^G)^2}$$

and

$$\frac{dr}{dx} = -\frac{1}{x^2}$$

By the chain rule and replacing $x = 1/(1+r)$ in combination with (2.17), we get

$$B(r) = p^F + \frac{(\frac{1-r^G}{r+1})^2(p^C - \frac{1}{r+1})}{(G-1)r^G - Gr^{G-1} + 1}.$$

From the definition of the boundary region (2.5) we have

$$B(r) = \frac{1-r}{1-r^G}.$$

The value of r_0, and hence, the value of $x_0 = 1/(1+r_0)$ is found by solving

$$p^F + \frac{(\frac{1-r^G}{r+1})^2(p^C - \frac{1}{r+1})}{(G-1)r^G - Gr^{G-1} + 1} - \frac{1-r}{1-r^G} = 0.$$

Define $\tilde{\mathcal{X}}$ as the solution set to the above equation, then the real-valued positive x's from $\tilde{\mathcal{X}}$ are set in function L. Then, the minimum distance to boundary region is given by $\delta = L(x_0)$ in which x_0 is the one which minimizes the function L.

2.6.6 Proof of Lemma 5

The following expressions will be used in the proof:

- $N_0(\rho)$: Number of rounds by ρ for which $\hat{\pi}_\rho = \pi_0^{tr}$.
- $N_1(\rho)$: Number of rounds by ρ for which $\hat{\pi}_\rho = \pi_1^{tr}$.
- $N_{F,1}(\rho)$: Number of rounds by ρ for which action F is taken when $\hat{\pi}_\rho = \pi_1^{tr}$.
- $N_{C,1}(\rho)$: Number of rounds by ρ for which action C is taken when $\hat{\pi}_\rho = \pi_1^{tr}$.
- $n_a(\rho)$: Indicator function of the event that action a is selected for at least once in round ρ.

(i) When $\hat{\pi}_\rho = \pi_1^{tr}$, action C is not taken only if the initial state is 1. Hence,

$$p_{C,1} = 1 - \Pr(s_1^\rho = 1) = 1 - q(1).$$

Let H_1 denote the event that state 1 is reached before state G when $\hat{\pi}_\rho = \pi_1^{tr}$. We have

$$p_{F,1} = \sum_{s=1}^{G-1} \Pr(H_1 | s_1^\rho = s) q(s).$$

When $r = 1$, $p_{F,1}$ is equivalent to the *ruin* probability (probability of hitting the terminal state 1) of a fair gambler's ruin problem over $G-1$ states, where states 1 and G are the terminal states. For this problem, the probability of hitting G from state s is $\frac{s-1}{G-1}$. Hence, probability of hitting state 1 from state s is

$$\Pr(H_1 | s_1^\rho = s) = 1 - \frac{s-1}{G-1} = \frac{G-s}{G-1}.$$

When $r \neq 1$, the problem is equivalent to an unfair gambler's ruin problem with $G-1$ states in which probability of hitting G from state s is $\frac{1-r^{s-1}}{1-r^{G-1}}$. Then, the probability of hitting state 1 from state s becomes

$$\Pr(H_1 | s_1^\rho = s) = 1 - \frac{1-r^{s-1}}{1-r^{G-1}} = \frac{r^{s-1} - r^{G-1}}{1-r^{G-1}}.$$

(ii) Since action C might be selected for more than once in a round, we have $N_a(\rho) \geq n_a(1) + n_a(2) + \cdots + n_a(\rho)$. This holds because in the initialization of GETBE, each action is selected once. Basically, we derive the lower bounds for $N_a(\rho+1)$, but these lower bounds also hold for $N_a(\rho)$ because of the way GETBE is initialized. For a set of rounds $\rho \in \mathcal{T} \subset \{1, \ldots, T\}$, $n_a(\rho)$s are in general not identically distributed. But if $\hat{\pi}_\rho$ is same for all rounds $\rho \in \mathcal{T}$ and if GETBE does not explore in any of the rounds in $\mathcal{T}$, then $n_a(\rho)$s are identically distributed.

First, assume that $N_1(\rho) = k$, $0 \leq k \leq \rho$. Then, the probability that action C is selected at least once in each of these k rounds is $p_{C,1}$. Let j_i denote the index of the round in which the estimated optimal policy is π_1^{tr} for the ith time. The sequence of Bernoulli random variables $n_C(j_i)$, $i = 1, 2 \ldots$ are independent and identically distributed. Hence, the Hoeffding bound can be used to upper-bound the deviation probability of sum of these random variables from the expected sum. Since the estimated optimal policy will be π_0^{tr} for the remaining $\rho - k$ rounds, the number of times action F is selected in all of these rounds will be at most $\lceil D(\rho) \rceil$. Therefore, the probability of taking action C is zero for at most $\lceil D(\rho) \rceil$ rounds. Let $\rho_D := \rho - \lceil D(\rho) \rceil$ and $N'_C(\rho)$ denote the sum of ρ random variables that are drawn from an independent identically distributed Bernoulli random process with parameter $p_{C,1}$. Then,

$$\begin{aligned} N_C(\rho) &\geq \rho - k - \lceil D(\rho) \rceil + \sum_{i=1}^{k} n_C(j_i) \\ &\geq \sum_{i=1}^{\rho - \lceil D(\rho) \rceil} n_C(j_i) = N'_C(\rho - \lceil D(\rho) \rceil) \\ &= N'_C(\rho_D). \end{aligned} \tag{2.18}$$

According to the Hoeffding bound, we have for $z > 0$

$$\Pr\left(N'_C(\rho_D) - \mathbb{E}(N'_C(\rho_D)) \leq -z\right) \leq e^{-2z^2/\rho_D}.$$

When $z = \sqrt{\rho_D \log \rho}$ the above bound becomes

$$\begin{aligned} &\Pr\left(N'_C(\rho_D) \leq \mathbb{E}(N'_C(\rho_D)) - \sqrt{\rho_D \log \rho}\right) \leq \frac{1}{\rho^2} \\ \Rightarrow &\Pr\left(N'_C(\rho_D) \leq p_{C,1}(\rho - \lceil D(\rho) \rceil) - \sqrt{(\rho - \lceil D(\rho) \rceil) \log \rho}\right) \leq \frac{1}{\rho^2}. \end{aligned}$$

Then, by using (2.18) we obtain

$$\Pr\left(N_C(\rho) \leq p_{C,1}(\rho - \lceil D(\rho) \rceil) - \sqrt{(\rho - \lceil D(\rho) \rceil) \log \rho}\right) \leq \frac{1}{\rho^2}.$$

Since ρ'_C is the first round in which $0.5 p_{C,1}\rho - p_{C,1}\lceil D(\rho) \rceil - \sqrt{\rho_D \log \rho}$ becomes positive, on or after ρ'_C, we have $p_{C,1}\rho_D - \sqrt{\rho_D \log \rho} > 0.5 p_{C,1}\rho$. Therefore, we replace $p_{C,1}\rho_D - \sqrt{\rho_D \log \rho}$ with $0.5 p_{C,1}\rho$ and then

$$\Pr\left(N_C(\rho) \leq 0.5 p_{C,1}\rho\right) \leq \frac{1}{\rho^2}, \text{ for } \rho \geq \rho'_C$$

which is equivalent to

$$\Pr\left(N_C(\rho) \leq f_C(\rho)\right) \leq \frac{1}{\rho^2}, \text{ for } \rho \geq \rho'_C. \tag{2.19}$$

Again, assume that $N_1(\rho) = k$. Then, the probability of selecting action F is $p_{F,1}$ in each of these k rounds. Let $\mathcal{R}$ denote the set of the remaining $\rho - k$ rounds. For a round $\rho_r \in \mathcal{R}$, action F is selected only if $N_F(\rho_r) \leq D(\rho_r)$. Among the rounds in $\mathcal{R}$, the number of rounds in which action F is selected is bounded below by $\min\{\rho - k, \lceil D(\rho - k) \rceil\}$. Then, $n_F(j_i)$, $i = 1, 2, \ldots$ is a sequence of i.i.d.

Bernoulli random variables with parameter $p_{F,1}$. From the argument above, we obtain

$$\begin{aligned} N_F(\rho) &\geq \min\{\rho - k, \lceil D(\rho - k) \rceil\} + \sum_{i=1}^{k} n_F(j_i) \\ &\geq \sum_{i=1}^{k+\min\{\rho-k,\lceil D(\rho-k)\rceil\}} n_F(j_i). \end{aligned}$$

When $\min\{\rho - k, \lceil D(\rho - k) \rceil\} = \rho - k$, we have

$$N_F(\rho) \geq \sum_{i=1}^{\rho} n_F(j_i) \geq \sum_{i=1}^{\lceil D(\rho) \rceil} n_F(j_i), \text{ for } \rho \geq \rho'_F.$$

When $\min\{\rho - k, \lceil D(\rho - k) \rceil\} = \lceil D(\rho - k) \rceil$, we have

$$N_F(\rho) \geq \sum_{i=1}^{k+\lceil D(\rho-k) \rceil} n_F(j_i).$$

Next, we will show that $D(\rho - k) + k \geq D(\rho)$ when ρ is sufficiently large. First, $\min\{\rho - k, \lceil D(\rho - k) \rceil\} = \lceil D(\rho - k) \rceil$ implies that

$$\rho \geq \lceil D(\rho - k) \rceil + k \geq D(\rho - k) + k. \tag{2.20}$$

Also, $D(\rho - k) + k \geq D(\rho)$ should imply that

$$\begin{aligned} D(\rho) - D(\rho - k) \leq k &\Leftrightarrow \gamma \log(\rho/(\rho - k)) \leq k \\ \Leftrightarrow \rho/(\rho - k) \leq e^{k/\gamma} &\Leftrightarrow \rho \geq \frac{ke^{k/\gamma}}{e^{k/\gamma} - 1}. \end{aligned} \tag{2.21}$$

Using the results in (2.20) and (2.21), we conclude that $D(\rho - k) + k \geq D(\rho)$ holds when

$$D(\rho - k) + k \geq \frac{ke^{k/\gamma}}{e^{k/\gamma} - 1}. \tag{2.22}$$

By setting $D(\rho) = \gamma \log \rho$ and manipulating (2.22) we get

$$\begin{aligned}
&\gamma \log(\rho - k) + k \geq \frac{ke^{k/\gamma}}{e^{k/\gamma} - 1} \\
&\Leftrightarrow \gamma \log(\rho - k) \geq k(\frac{e^{k/\gamma}}{e^{k/\gamma} - 1} - 1) \\
&\Leftrightarrow \log(\rho - k) \geq \frac{k/\gamma}{e^{k/\gamma} - 1} \\
&\Leftrightarrow \rho - k \geq e^{\frac{k/\gamma}{e^{k/\gamma}-1}} \\
&\Leftrightarrow \rho \geq k + e^{\frac{k/\gamma}{e^{k/\gamma}-1}}.
\end{aligned} \tag{2.23}$$

First, we evaluate the term $h(k) := e^{\frac{k/\gamma}{e^{k/\gamma}-1}}$. We will show that $h(k) \in [1, e]$ for all $k \in \mathbb{R}_+$. By applying L'Hopital's rule we get

$$\lim_{k\to 0} \frac{k/\gamma}{e^{k/\gamma} - 1} = \lim_{k\to 0} \frac{1/\gamma}{(1/\gamma)e^{k/\gamma}} = 1$$

and

$$\lim_{k\to\infty} \frac{k/\gamma}{e^{k/\gamma} - 1} = 0.$$

These two conditions and using the fact that exponential function is continuous we conclude that

$$\lim_{k\to 0} e^{\frac{k/\gamma}{e^{k/\gamma}-1}} = e^{\lim_{k\to 0} \frac{k/\gamma}{e^{k/\gamma}-1}} = e$$

and

$$\lim_{k\to\infty} e^{\frac{k/\gamma}{e^{k/\gamma}-1}} = e^{\lim_{k\to\infty} \frac{k/\gamma}{e^{k/\gamma}-1}} = 1.$$

Next, we will show that $e^{\frac{k/\gamma}{e^{k/\gamma}-1}}$ is decreasing in k. Since $e^{\frac{k/\gamma}{e^{k/\gamma}-1}}$ is a monotonically increasing function of $\frac{k/\gamma}{e^{k/\gamma}-1}$, it is sufficient to show that $\frac{k/\gamma}{e^{k/\gamma}-1}$ is decreasing in k. We have

$$\frac{d}{dk} \frac{k/\gamma}{e^{k/\gamma} - 1} = \frac{\frac{1}{\gamma}(e^{k/\gamma} - 1) - \frac{k}{\gamma}(e^{k/\gamma}/\gamma)}{(e^{k/\gamma} - 1)^2}.$$

The denominator is always positive for $k > 0$. Therefore, we only consider the numerator and write it as

$$\frac{1}{\gamma}(e^{k/\gamma} - 1) - \frac{k}{\gamma}(e^{k/\gamma}/\gamma) = \frac{(\gamma - k)e^{k/\gamma} - \gamma}{\gamma^2}.$$

As the denominator is positive, we only need to show that $(\gamma - k)e^{k/\gamma} - \gamma$ is always negative. The derivative of the above expression is $-(k/\gamma)e^{k/\gamma}$, which is negative for $k > 0$. We also have $(\gamma - k)e^{k/\gamma} - \gamma = 0$ when $k = 0$. These two conditions imply that $(\gamma - k)e^{k/\gamma} - \gamma$ is always negative for $k > 0$, by which we conclude that $e^{\frac{k/\gamma}{e^{k/\gamma}-1}}$ is decreasing in k. Hence, we have

$$1 \leq e^{\frac{k/\gamma}{e^{k/\gamma}-1}} \leq e$$

This implies that $k + e^{\frac{k/\gamma}{e^{k/\gamma}-1}} \leq k + e$. Hence (2.23) holds when $\rho \geq k + e$. This implies that when $k \leq \rho - e$, we have

$$\sum_{i=1}^{k+\lceil D(\rho-k)\rceil} n_F(j_i) \geq \sum_{i=1}^{\lceil D(\rho)\rceil} n_F(j_i).$$

The only cases that are left out are $k = \rho$, $k = \rho - 1$ and $k = \rho - 2$. But we know from the definition of ρ'_F that for $\rho \geq \rho'_F$, $\rho - 2 - \lceil D(\rho)\rceil$ is positive. Hence for these cases we also have

$$\sum_{i=1}^{k+\lceil D(\rho-k)\rceil} n_F(j_i) \geq \sum_{i=1}^{\rho-2} n_F(j_i) \geq \sum_{i=1}^{\lceil D(\rho)\rceil} n_F(j_i).$$

Let $N'_F(\rho)$ denote the sum over $n_F(j_i)$ for ρ rounds. From all of the cases we derived above, we obtain

$$N_F(\rho) \geq \sum_{i=1}^{\lceil D(\rho)\rceil} n_F(j_i) = N'_F(\lceil D(\rho)\rceil) \text{ for } \rho \geq \rho'_F. \tag{2.24}$$

Now, by using Hoeffding bound we have

$$\Pr\left(N'_F(\lceil D(\rho)\rceil) - \mathbb{E}(N'_F(\lceil D(\rho)\rceil)) \leq -z\right) \leq e^{-2z^2/\lceil D(\rho)\rceil}$$

and if $z = \sqrt{\lceil D(\rho)\rceil \log \rho}$ then,

$$\Pr\left(N'_F(\lceil D(\rho)\rceil) < \mathbb{E}(N'_F(\lceil D(\rho)\rceil)) - \sqrt{\lceil D(\rho)\rceil \log \rho}\right) \leq \frac{1}{\rho^2}$$
$$\Pr\left(N'_F(\lceil D(\rho)\rceil) < p_{F,1}\lceil D(\rho)\rceil - \sqrt{\lceil D(\rho)\rceil \log \rho}\right) \leq \frac{1}{\rho^2}.$$

By using (2.24), we get

$$\Pr\left(N_F(\rho) < p_{F,1}\lceil D(\rho)\rceil - \sqrt{\lceil D(\rho)\rceil \log \rho}\right) \leq \frac{1}{\rho^2}. \tag{2.25}$$

Then, by using $D(\rho) = \gamma \log \rho,\ \gamma > 1/p_{F,1}^2$, we have

$$\begin{aligned}
&p_{F,1}\lceil D(\rho)\rceil - \sqrt{\lceil D(\rho)\rceil \log \rho} \\
&= p_{F,1}\lceil \gamma \log \rho\rceil - \sqrt{\lceil \gamma \log \rho\rceil \log \rho} \\
&\geq p_{F,1}\gamma \log \rho - \sqrt{(\gamma \log \rho + 1)\log \rho} \\
&= p_{F,1}\gamma \log \rho - \sqrt{\gamma \log^2 \rho + \log \rho} \\
&\geq p_{F,1}\gamma \log \rho - \sqrt{\gamma \log^2 \rho} - \sqrt{\log \rho} && (2.26) \\
&= (p_{F,1}\gamma - \sqrt{\gamma})\log \rho - \sqrt{\log \rho} && (2.27)
\end{aligned}$$

where (2.26) occurs due to the subadditivity[4] of the square root. Next, we will show that (2.27) becomes positive when ρ is large enough. To do this, we first show that the first term in (2.27) is always positive. This is proven by observing that

$$\gamma > 1/p_{F,1}^2 \Rightarrow p_{F,1}\sqrt{\gamma} - 1 > 0 \Rightarrow p_{F,1}\gamma - \sqrt{\gamma} > 0. \tag{2.28}$$

Since $\log \rho$ increases at a higher rate than $\sqrt{\log \rho}$, it can be shown that $0.5(p_{F,1}\gamma - \sqrt{\gamma})\log \rho - \sqrt{\log \rho}$ will always increase after some round. Since $\lim_{\rho\to\infty} 0.5(p_{F,1}\gamma - \sqrt{\gamma})\log \rho - \sqrt{\log \rho} = \infty$, this term is expected to be positive after some round. From the statement of the lemma, it is known that ρ'_F is greater than or equal to this round. Therefore, for $\rho \geq \rho'_F$, $(p_{F,1}\gamma - \sqrt{\gamma})\log \rho - \sqrt{\log \rho} > 0.5(p_{F,1}\gamma - \sqrt{\gamma})\log \rho$. Using this and (2.27), we obtain

$$\begin{aligned}
p_{F,1}\lceil D(\rho)\rceil - \sqrt{\lceil D(\rho)\rceil \log \rho} &\geq (p_{F,1}\gamma - \sqrt{\gamma})\log \rho - \sqrt{\log \rho} \\
&\geq 0.5(p_{F,1}\gamma - \sqrt{\gamma})\log \rho.
\end{aligned}$$

Then, we use this result and (2.25) to get

$$\Pr\left(N_F(\rho) \leq 0.5(p_{F,1}\gamma - \sqrt{\gamma})\log \rho\right) \leq \frac{1}{\rho^2}, \text{for } \rho \geq \rho'_F$$

which is equivalent to

$$\Pr\left(N_F(\rho) \leq f_F(\rho)\right) \leq \frac{1}{\rho^2}, \text{for } \rho \geq \rho'_F. \tag{2.29}$$

[4]For $a, b > 0$ we have $\sqrt{a} + \sqrt{b} > \sqrt{a+b}$ since $(\sqrt{a} + \sqrt{b})^2 > a + b$.

2.6.7 Proof of Theorem 3

First, we bound $\mathbb{E}[\mathrm{IR}(T)]$. For this, we change the order of summations in (2.7) and we have

$$\mathbb{E}\left[\mathrm{IR}(T)\right] \le \sum_{a\in\{F,C\}} \sum_{\rho=1}^{T} \Pr\left(|p^a - \hat{p}^a_\rho| \ge \delta/\sqrt{2}\right). \tag{2.30}$$

Let $N^*_F(\rho) := N^G_F(\rho)$ and $N^*_C(\rho) := N^u_C(\rho)$. By using the law of total probability, we obtain for $a \in \{F, C\}$

$$\begin{aligned}
&\sum_{\rho=1}^{T} \Pr\left(|p^a - \hat{p}^a_\rho| \ge \frac{\delta}{\sqrt{2}}\right)\\
&= \sum_{\rho=1}^{T}\sum_{n=1}^{\infty} \Pr\left(|p^a - \frac{N^*_a(\rho)}{N_a(\rho)}| \ge \frac{\delta}{\sqrt{2}} | N_a(\rho) = n\right) \Pr\left(N_a(\rho) = n\right)\\
&= \sum_{\rho=1}^{T}\sum_{n=1}^{\infty} \Pr\left(|np^a - N^*_a(\rho)| \ge n\frac{\delta}{\sqrt{2}} | N_a(\rho) = n\right) \Pr\left(N_a(\rho) = n\right). \qquad (2.31)
\end{aligned}$$

For each action, we use the result of Lemma 5 and divide the summation in (2.31) into two summations. Note that the bounds on $N_a(\rho)$ given in Lemma 5 hold when $\rho \ge \rho' = \max\{\rho'_C, \rho'_F\}$. Therefore, we have

$$\begin{aligned}
&\sum_{\rho=1}^{T}\sum_{n=1}^{\infty} \Pr\left(|np^a - N^*_a(\rho)| \ge n\frac{\delta}{\sqrt{2}} | N_a(\rho) = n\right) \Pr\left(N_a(\rho) = n\right)\\
&= \sum_{\rho=1}^{\rho'-1}\sum_{n=1}^{\infty} \Pr\left(|np^a - N^*_a(\rho)| \ge n\frac{\delta}{\sqrt{2}} | N_a(\rho) = n\right) \Pr\left(N_a(\rho) = n\right)\\
&+ \sum_{\rho=\rho'}^{T}\sum_{n=1}^{\infty} \Pr\left(|np^a - N^*_a(\rho)| \ge n\frac{\delta}{\sqrt{2}} | N_a(\rho) = n\right) \Pr\left(N_a(\rho) = n\right)\\
&= \sum_{\rho=\rho'}^{T}\sum_{n=1}^{\infty} \Pr\left(|np^a - N^*_a(\rho)| \ge n\frac{\delta}{\sqrt{2}} | N_a(\rho) = n\right) \Pr\left(N_a(\rho) = n\right)\\
&+ K' \qquad (2.32)
\end{aligned}$$

where

$$K' = \sum_{\rho=1}^{\rho'-1}\sum_{n=1}^{\infty} \Pr\left(|np^a - N^*_a(\rho)| \ge n\frac{\delta}{\sqrt{2}} | N_a(\rho) = n\right) \Pr\left(N_a(\rho) = n\right).$$

Since $\Pr\left(|np^a - N_a^*(\rho)|\geq n\frac{\delta}{\sqrt{2}}\right) \leq 1$ and $\sum\limits_{n=1}^{\infty} \Pr\left(N_a(\rho) = n\right) = 1$, we have

$$\sum_{n=1}^{\infty} \Pr\left(|np^a - N_a^*(\rho)|\geq n\frac{\delta}{\sqrt{2}}|N_a(\rho) = n\right) \Pr\left(N_a(\rho) = n\right) \leq 1.$$

Therefore, an upper bound on K' is ρ'. Next, by using Hoeffding's inequality we get

$$\begin{aligned}
&\sum_{\rho=\rho'}^{T}\sum_{n=1}^{\infty} \Pr\left(|np^a - N_a^*(\rho)|\geq n\frac{\delta}{\sqrt{2}}|N_a(\rho) = n\right) \Pr\left(N_a(\rho) = n\right) \\
&\leq \sum_{\rho=\rho'}^{T}\sum_{n=1}^{\infty} 2e^{-2\frac{(n\delta/\sqrt{2})^2}{n}} \Pr\left(N_a(\rho) = n\right) \\
&= \sum_{\rho=\rho'}^{T}\sum_{n=1}^{\infty} 2e^{-n\delta^2} \Pr\left(N_a(\rho) = n\right) \\
&= \sum_{\rho=\rho'}^{T}\sum_{n=1}^{f_a(\rho)} 2e^{-n\delta^2} \Pr\left(N_a(\rho) = n\right) + \sum_{\rho=\rho'}^{T}\sum_{n=f_a(\rho)+1}^{\infty} 2e^{-n\delta^2} \Pr\left(N_a(\rho) = n\right). \quad (2.33)
\end{aligned}$$

For the first summation in (2.33), we use (2.19) and (2.29) for each action as an upper bound since $n \leq f_a(\rho)$. Therefore,

$$\begin{aligned}
\sum_{\rho=\rho'}^{T}\sum_{n=1}^{f_a(\rho)} 2e^{-n\delta^2} \Pr\left(N_a(\rho) = n\right) &\leq \sum_{\rho=\rho'}^{T} \frac{2e^{-n\delta^2}}{\rho^2} \\
&\leq \sum_{\rho=1}^{\infty} \frac{2}{\rho^2} = \frac{\pi^2}{3}. \quad (2.34)
\end{aligned}$$

For the second summation in (2.33), we first consider $a = C$. We have

$$\begin{aligned}
\sum_{\rho=\rho'}^{T}\sum_{n=f_C(\rho)+1}^{\infty} 2e^{-n\delta^2} \Pr\left(N_C(\rho) = n\right) &\leq \sum_{\rho=\rho'}^{T} 2e^{-f_C(\rho)\delta^2} \sum_{n=f_C(\rho)+1}^{\infty} \Pr\left(N_C(\rho) = n\right) \\
&\leq \sum_{\rho=\rho'}^{T} 2e^{-f_C(\rho)\delta^2} = \sum_{\rho=\rho'}^{T} 2e^{-0.5p_{C,1}\rho\delta^2} \\
&\leq 2e^{-0.5p_{C,1}\rho'\delta^2} \frac{1 - e^{-0.5p_{C,1}\delta^2(T-\rho'+1)}}{1 - e^{-0.5p_{C,1}\delta^2}} \\
&\leq \frac{2e^{-0.5p_{C,1}\rho'\delta^2}}{1 - e^{-0.5p_{C,1}\delta^2}} \leq \frac{2}{1 - e^{-0.5p_{C,1}\delta^2}}. \quad (2.35)
\end{aligned}$$

Next, we consider $a = F$. For this case, we have

$$\begin{aligned}
\sum_{\rho=\rho'}^{T} \sum_{n=f_F(\rho)+1}^{\infty} 2e^{-n\delta^2} \Pr\left(N_F(\rho) = n\right) &\leq \sum_{\rho=\rho'}^{T} 2e^{-f_F(\rho)\delta^2} \sum_{n=f_F(\rho)+1}^{\infty} \Pr\left(N_F(\rho) = n\right) \\
&\leq \sum_{\rho=\rho'}^{T} 2e^{-f_F(\rho)\delta^2} \\
&\leq \sum_{\rho=\rho'}^{T} 2e^{-2\log\rho} \qquad (2.36) \\
&\leq \sum_{\rho=1}^{\infty} \frac{2}{\rho^2} = \frac{\pi^2}{3}. \qquad (2.37)
\end{aligned}$$

The inequality in (2.36) holds due to the fact that $\delta^2 \geq 2\log\rho/f_F(\rho) = 4/(p_{F,1}\gamma - \sqrt{\gamma})$, which follows from $\gamma \geq (x_1)^2$. This can be proved as follows. The term $p_{F,1}\gamma - \sqrt{\gamma}$ is positive because of (2.28). Then, in order to have $\delta^2 \geq 4/(p_{F,1}\gamma - \sqrt{\gamma})$, we must have $p_{F,1}\gamma - \sqrt{\gamma} - 4/\delta^2 \geq 0$. In order to check this, we re-write the left-hand-side of this inequality as a second order polynomial function, which is given by

$$g(x) = ax^2 + bx + c \geq 0$$

where $a = p_{F,1}$, $b = -1$, $c = -4/\delta^2$ and $x = \sqrt{\gamma}$. Since γ is positive, we will find positive values of x for which $g(x)$ is nonnegative. Note also that $g(x)$ is a convex function since its second derivative is $2a$, which is positive. Hence, $g(x)$ is nonnegative for positive values of x that are greater than the largest root of $g(x)$. The roots of $g(x)$ are given as

$$x_1 = \frac{1 + \sqrt{1 + \frac{16p_{F,1}}{\delta^2}}}{2p_{F,1}}, \; x_2 = \frac{1 - \sqrt{1 + \frac{16p_{F,1}}{\delta^2}}}{2p_{F,1}}.$$

It is clear that only x_1 is positive. Thus, $g(x)$ is nonnegative for $x = \sqrt{\gamma} \geq x_1$. This implies that γ has to be greater than $(x_1)^2$ in order to have $\delta^2 \geq \frac{4}{p_{F,1}\gamma - \sqrt{\gamma}}$, which holds for our problem.

Finally, we combine the results of (2.32), (2.33), (2.34), (2.35) and (2.37) and sum the final result over the two actions to get a bound for the expression in

(2.30). This results in

$$\begin{aligned}\mathbb{E}[\mathrm{IR}(T)] &\leq 2(\rho' + \frac{\pi^2}{3}) + \frac{2}{1 - e^{-0.5 p_{C,1}\delta^2}} + \frac{\pi^2}{3} \\ &\leq 2\rho' + \pi^2 + \frac{2}{1 - e^{-0.5 p_{C,1}\delta^2}}. \end{aligned} \tag{2.38}$$

Next, we derive an upper-bound on $1/(1 - e^{-0.5 p_{C,1}\delta^2})$ by using the inequality $e^{-x} \leq 1/(x+1)$ for $x > 0$. Thus, we have

$$1 - e^{-x} \geq \frac{x}{1+x} \Rightarrow \frac{1}{1 - e^{-x}} \leq 1 + \frac{1}{x}.$$

By replacing x with $0.5 p_{C,1}\delta^2$ we have

$$\frac{1}{1 - e^{-0.5 p_{C,1}\delta^2}} \leq 1 + \frac{1}{0.5 p_{C,1}\delta^2}. \tag{2.39}$$

Then, we use (2.39) in (2.38) to obtain

$$\begin{aligned}\mathbb{E}[\mathrm{IR}(T)] \leq 2\rho' + \pi^2 + \frac{2}{1 - e^{-0.5 p_{C,1}\delta^2}} + \frac{\pi^2}{3} &\leq 2\rho' + 2 + \pi^2 + \frac{4}{p_{C,1}\delta^2} \\ &\leq 2\rho' + 12 + \frac{4}{p_{C,1}\delta^2} = w. \end{aligned} \tag{2.40}$$

Next, we derive the regret bounds. Firstly, assume that the optimal policy is π_1^{tr}. Then, the expected number of rounds in which the suboptimal policy is selected is finite and bounded by w (independent of T) given in (2.38). In this case, the exploration is done only when the suboptimal policy is selected and there will be no extra regret term due to exploration. Since the suboptimality gap can be upper bounded by 1 for both when GETBE explores and exploits, we have

$$\mathbb{E}[\mathrm{Reg}(T) | \pi^* = \pi_1^{tr}] \leq w.$$

Secondly, assume that the optimal policy is π_0^{tr}. Similar to the previous case, the expected number of rounds in which the suboptimal policy is selected is at most w. Since the suboptimal policy for this case is π_1^{tr}, GETBE will never explore in rounds in which the suboptimal policy is selected. Hence, the regret incurred by GETBE in a round in which it selects the suboptimal policy is at most $\Delta_{\max}$. On the other hand, GETBE might explore action F when the optimal policy is selected. This results in an additional regret term. Since, the number of explorations of GETBE by round T is bounded by $\lceil D(T) \rceil$, the regret from explorations is bounded by $\lceil D(T) \rceil$. Therefore, we have

$$\mathbb{E}[\mathrm{Reg}(T) | \pi^* = \pi_0^{tr}] \leq w\Delta_{\max} + \lceil D(T) \rceil.$$

Chapter 3

Contextual Goal-oriented with Dead-end State MDP

The contents of this chapter has appeared in [56].

3.1 Problem Formulation

3.1.1 Definition of CGDMDP Model

The CGDMDP can be defined as the tuple of $<\mathcal{X}, \mathcal{S}, T, \mathcal{Z}, \mathcal{D}, \mathcal{G}, \mathcal{A}, \mathcal{P}, s_0>$. Here, $\mathcal{X}$ is the context set where in the breast cancer therapy, it can a combination of race, age, sex, genetic factors and etc. $\mathcal{S} := \{C, 1, 2, \dots, W-1, W\}$ is the set of states which defines the *illness degree* (ID). $C = 0$ is the state where the patient is in its best condition (Clear state) where all symptoms of illness are cleared, while, W is the state where the condition of the patient is worst (such as death). For breast cancer, a related state model is Ivy's model [57]. The initial state of the patient is s_0, which is an element of $\tilde{\mathcal{S}} := \{1, \dots, W-1\}$. T is the target survival time. In real-world applications, T can be interpreted as number of months or years that the patient lives from the start of the disease (patient arrival time).

We define $\mathcal{Z}$ as the set of *global states* which is a set of tuples, where each tuple includes the amount of time passed from the arrival of the patient and the ID:

$$\mathcal{Z} = \{z := (z_1, z_2) = (s, t) : t = 0 \text{ and } s \in \tilde{\mathcal{S}} \text{ or } 1 \leq t \leq T \text{ and } s \in \mathcal{S}\}.$$

$\mathcal{D}$ and $\mathcal{G}$ are the set of dead-end states and the goal states, respectively, which are given as:

$$\mathcal{G} = \{(s, T) \in \mathcal{Z} : s \neq W\}, \ \mathcal{D} = \{(s, t) \in \mathcal{Z} : s = W\}.$$

The states in $\mathcal{D}$ and $\mathcal{G}$ are absorbing, i.e., if they are reached, the patient stays there with probability 1 (current round ends).

The finite action set is denoted by $\mathcal{A}$. For example, for breast cancer, the actions can be surgery, hormone therapy, chemotherapy and combinations of drug types and drug doses used for these therapies. Let $\mathcal{Q}$ be the set of all transition probabilities for all possible contexts, IDs and the actions. $q(x, s, a, s') \in \mathcal{Q}$ denotes the transition probability from state $s \in \mathcal{S}$ to $s' \in \mathcal{S}$ when context is $x \in \mathcal{X}$ and action $a \in \mathcal{A}$ is selected. We assume that if $q(x, s, a, s') > 0$, then $q(x', s, a, s') > 0$, $\forall x' \in \mathcal{X}$. In addition, for any $x \in \mathcal{X}$ and $a \in \mathcal{A}$, we have $q(x, W, a, W) = 1$. Let $J(s, a) := \{s' \in \mathcal{S} : q(x, s, a, s') > 0 \text{ for } x \in \mathcal{X}\}$ and $\mathcal{S}_+(s, a) := \{s' \in J(s, a) : s' < s\}$. $J(s, a)$ represents the set of ID states which are immediately reachable with a positive probability when action a is taken in ID state s, and $\mathcal{S}_+(s, a)$ denotes the subset of ID states in $J(s, a)$ with a higher ID state than the current ID state. Next, based on $q(x, s, a, s')$, we define $p(x, z, a, z')$ which is the transition probability of moving from global state $z = (z_1, z_2) \in \mathcal{Z}$ $(z_2 < T)$ to $z' = (z'_1, z'_2) \in \mathcal{Z}$ when context is $x \in \mathcal{X}$ and action $a \in \mathcal{A}$ is selected. We have

$$p(x, z, a, z') := \begin{cases} q(x, z_1, a, z'_1), & \text{if } z'_2 = z_2 + 1 \\ 0, & \text{if } z'_2 \neq z_2 + 1 \end{cases}. \tag{3.1}$$

Then, we have $\mathcal{P} := \{p(x, z, a, z')\}$.

We assume that $\mathcal{P}$ is unknown to the learner apriori, and hence, the learner cannot directly use MAXPROB [11] or dynamic programming [8] to compute the optimal policy, since these methods require knowledge of the state transition probabilities for every context.

3.1.2 Patient Arrival Model and Performance Metric

The patients arrive sequentially in rounds where $n \in \{1, 2, \ldots\}$ is the index of the round. Each round consists of T time slots, and t is used as the time slot index, where $t \in \{1, \ldots, T\}$. The nth patient is denoted by H_n. The context and the initial state of H_n is denoted by x_n and $s_{n,0}$ respectively. We assume the context of the patient does not change in a round. The global state, ID state of H_n and the action selected for H_n in time slot t is denoted by $z_{n,t}$, $s_{n,t}$ and $a_{n,t}$ respectively. In each round, we want to select the actions such that dead-end states are not reached up to time T given the initial state and the context of the patient. The ultimate goal is to find a strategy by which the number of times patients survive for T time slots is maximized over all rounds.

For this objective, the reward in N rounds is defined as

$$R(N) := \sum_{n=1}^{N} \mathbb{I}(s_{n,T} \neq W)$$

where $\mathbb{I}$ is the indicator function. The reward can be interpreted as the number of rounds in which the patients have not visited W by time slot T.

The optimal solution to this problem can be obtained by using dynamic programming or value iteration methods when all the problem parameters (i.e., the transition probability matrix $\mathcal{P}$) are known. For instance, the MAXPROB algorithm of [11] can be used to obtain the optimal policy of each round, when $\mathcal{P}$ is known in each global state (or dynamic programming can be used directly).

Let the optimal policy of round n be π_n, which is a mapping of $\pi_n : \mathcal{Z} \to \mathcal{A}$. The pseudo-code of the optimal policy is given in Algorithm 2.

3.2 The Learning Algorithm

The learning algorithm we propose in this section is called *Contextual Optimistic DYnamic programming* (CODY). Its pseudo-code is given in Algorithm 3.

Algorithm 2 Optimal Policy (OP)

```
 1: $Input: T, \mathcal{S}, \mathcal{N}, \mathcal{P}$
 2: while $n \leq N$ do
 3:     Get initial state $s_{n,0}$ and context $x_n$
 4:     Calculate $\pi_n$ by using $\mathcal{P}_{x_n} := \{p(x_n, z, a, z')\}$ in Dynamic Programming.
 5:     $t = 0$
 6:     while $t < T$ do
 7:         $z_{n,t} = (t, s_{n,t})$
 8:         $a_{n,t} = \boldsymbol{\pi}_n(z_{n,t})$
 9:         $t = t + 1$
10:     end while
11:     $r_n = \mathbb{I}(s_{n,T} \neq W)$
12:     $n = n + 1$
13: end while
```

CODY separates the context space into M disjoint sets by using similarities between the contexts. To model the similarity, we use the Lipschitz similarity metric:

$$|q(x, s, a, s') - q(x', s, a, s')| \leq L \times dist(x, x'), \ \forall x, x' \in \mathcal{X}, \forall s, s' \in \mathcal{S}, \forall a \in \mathcal{A}. \tag{3.2}$$

(3.2) holds naturally in many applications. For instance, in healthcare, patients with similar environmental factors can respond to the same therapy in a similar way. We assume that the distance metric used in (3.2) is known by CODY, and hence, it can create a partition such that for all contexts inside the same set of the partition, the variation of the transition probabilities is less than some $\epsilon > 0$. Such an assumption is commonly used in the literature [58]. The partition of the context space created by CODY is denoted by $\mathcal{B}$.

CODY keeps the following counters: $N_n(b, s)$ is the number of times ID state s is observed in partition b by the beginning of round n. $N_n(b, s, a)$ is the number of times ID state s is observed in partition b and action a is selected by the beginning of round n. $N_n(b, s, a, s')$ is the number of times ID state s is observed in partition b and action a is selected and ID state s' is observed immediately after state s upon taking a by the beginning of round n.

CODY operates as follows: At the beginning of each round, CODY observes the context x_n and finds the set $b_n \in \mathcal{B}$. CODY estimates the ID state transition probabilities for partition b_n and all actions in the action set, based on the history of ID state transitions and selected actions in that set. We use $\hat{q}_n(b_n, s, a, s')$ to denote the noisy estimate of $q_n(x_n, s, a, s')$ when x_n belongs to set b_n. Basically, CODY sets:

$$\hat{q}_n(b_n, s, a, s') = \begin{cases} \dfrac{N_n(b, s, a, s')}{N_n(b, s, a)}, & \text{if } N_n(b, s, a) > 0 \\ \dfrac{1}{|J(s, a)|}, & \text{if } N_n(b, s, a) = 0 \end{cases}$$

The policy generated by solving MAXPROB (or dynamic programming) using $\{\hat{q}_n(b_n, s, a, s')\}$ may not converge to the optimal policy. As an example, it is known that such a policy would not converge to the optimal policy even in the MAB problem [19], which is much simpler than CGDMDP. Furthermore, it is known that an algorithm that estimates $\{q(x, s, a, s')\}$ with the minimum error does not maximize the reward over all rounds [59]. To maximize the total reward, an algorithm that can tradeoff between the estimation error and reward maximization according to estimated values is required. CODY uses the *optimism under uncertainty* principle, which is a key approach used in solving such problems [1].

The confidence value for round n and the tuple (b, s, a) is denoted by $\phi_n(b, s, a)$, $\forall b \in \mathcal{B}$, $\forall s \in \mathcal{S}$ and $\forall a \in \mathcal{A}$. Then, the confidence set around the estimated value of ID state transition probability is given as follows:

$$[\hat{q}_n(b, s, a, s') - \phi_n(b, s, a),\ \hat{q}_n(b, s, a, s') + \phi_n(b, s, a)].$$

Let $u_n(b_n, s, a, s') := \hat{q}_n(b_n, s, a, s') + \phi_n(b_n, s, a)$, $\forall s' \in \mathcal{S}_+(s, a)$ be the index that corresponds to tuple (b_n, s, a, s'). This index reflects CODY's uncertainty about the estimated state transition probability. Due to the rule of optimism in face of uncertainty, we use $u_n(b_n, s, a, s')$ instead of $\hat{q}_n(b_n, s, a, s')$ where the estimated state transition probabilities are inflated toward the states with better conditions. As the sum of the estimated state transition probabilities for a given action must some up to 1, we normalize the index as follows:

$$u_n(b_n, s, a, s') := \frac{u_n(b_n, s, a, s')}{\sum\limits_{s'' \in J(s,a)} u_n(b_n, s, a, s'')}, \ \forall s' \in \mathcal{S}$$

and we define $\mathcal{U}_n := \{u_n(b_n, s, a, s')\}$. Then, CODY forms the transition probability estimates for the global states for $z_2 < T$, as follows:

$$\hat{p}_n(b_n, z, a, z') := \begin{cases} u_n(b_n, z_1, a, z'_1), & \text{if } z'_2 = z_2 + 1 \\ 0, & \text{if } z'_2 \neq z_2 + 1 \end{cases} \tag{3.3}$$

and uses $\hat{\mathcal{P}}_n := \{\hat{p}(b_n, z, a, z')\}$ in dynamic programming to obtain the estimated policy of round n which is denoted by $\hat{\pi}_n$. The action selected by the estimated policy in time slot t of round n is denoted by $\hat{a}_{n,t}$. Then, in round n, CODY keeps updating the relative counters of that round, i.e., $N_n(b_n, s)$, $N_n(b_n, s, a)$ and $N_n(b_n, s, a, s')$. The updated values of these counters are used to calculate the estimated state transition probabilities and the estimated policies in the following rounds.

3.3 Numerical Results

3.3.1 Simulation Setup

In this part, we discuss the simulation setup that has been implemented to verify the fact that CODY converges asymptotically to the optimal one and can outperform the best fixed algorithm which does not exploit the context information. We consider the survival rate problem in breast cancer patients.

The goal is set to maximizing the five-year survival rate. The patients are assumed to be women suffering from breast cancer. The state transition model is given in Figure 3.1. The set of ID states is $\mathcal{S} = \{C, 1, 2, 3, W\}$. $C = 0$ is the stage of "No Cancer", while $W = 4$ is the death of the patient. States 1, 2 and 3 represent "Local Cancer", "Regional Cancer" and "Distant Cancer", respectively.

Algorithm 3 CODY

```
1: Input : $N, M, J(s, a), \mathcal{S}_+(s, a)$, $\forall s \in \mathcal{S}$ and $\forall a \in \mathcal{A}$.
2: Initialize: Partition $\mathcal{X}$ into $M$ sets and generate $\mathcal{B}$
3: Counter Initialization: $N_n(b, s) = 0$, $\forall b \in \mathcal{B}$, $s \in \mathcal{S} - \{W\}$, $N_n(b, s, a) = 0$, $\forall b \in \mathcal{B}$,
   $s \in \mathcal{S} - \{W\}$, $a \in \mathcal{A}$ and $N_n(b, s, a, s') = 0$, $\forall b \in \mathcal{B}$, $s \in \mathcal{S} - \{W\}$, $a \in \mathcal{A}$, $s' \in \mathcal{S}$.
4: $n = 1$
5: while $n \leq N$ do
6:     Get the initial state $s_{n,0}$ and context $x_n$. Find the set $b_n$ that contains $x_n$
7:     Calculate the values in $\mathcal{U}_n$ using the counters
8:     Find the estimated (optimal) policy $\hat{\pi}_n$ using $\mathcal{U}_n$ and dynamic programming
9:     $t = 0$
10:    while $t < T$ do
11:        $z_{n,t} = (t, s_{n,t})$
12:        $\hat{a}_{n,t} = \hat{\pi}_n(z_{n,t})$ is selected
13:        $N_n(b_n, s_{n,t}) = N_n(b_n, s_{n,t}) + 1$
14:        $N_n(b_n, s_{n,t}, \hat{a}_{n,t}) = N_n(b_n, s_{n,t}, \hat{a}_{n,t}) + 1$
15:        ID state of $s_{n,t+1}$ is observed
16:        $N_n(b_n, s_{n,t}, \hat{a}_{n,t}, s_{n,t+1}) = N_n(b_n, s_{n,t}, \hat{a}_{n,t}, s_{n,t+1}) + 1$
17:        $t = t + 1$
18:    end while
19:    $r_n = \mathbb{I}(s_{n,T} \neq W)$ is obtained
20:    $n = n + 1$
21: end while
```

The action set is $A = \{a, b\}$ where $J(i, a) = \{i + 1, i - 1\}$ and $J(i, b) = \{C, W\}$, $\forall i \in \{1, 2, 3\}$. No action is taken when the ID state is C, and we assume that state 1 might be observed with a fixed probability in the next time slot after C is hit. We take $T = 5$, and each time slot represents one year. We use a two dimensional context space given as $\mathcal{X} = \{(H, F), (nH, F), (H, nF), (nH, nF)\}$ where H, nH, F and nF are used for Hispanic patients, Non-Hispanic patients, patients who have family history of having cancer and patients who does not have family history of having cancer, respectively. We assume the context of each patient is independent from the context of the other patients. [60] provides the distribution over the context space. We assume the context arrival probability distribution is fixed over all rounds and the probability of having context (H, F) is 6%, (H, nF) is 40%, (nH, F) is 9% and (nH, nF) is 45%.

We generate context arrival process according to this distribution for all rounds. The same procedure is also performed for the initial state distribution.

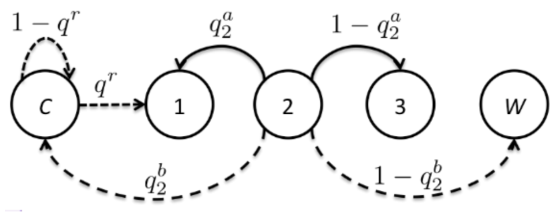

Figure 3.1: State transition model of the CGDMDP. Dashed arrows correspond to possible state transitions by selecting action b, while solid arrows correspond to possible state transitions by selecting action a. Weights on the arrows correspond to state transition probabilities. q_i^a is the probability that the state moves to state $i+1$ when action a is taken and q_i^b is the probability that the state moves to C upon taking action b. q^r is the probability that the patient ID state changes from state C to 1. This figure only shows the state transitions for state 2 when action a is selected. The state transition probabilities for $\{1,3\}$ are the same as state 2.

Based on [61], we assume that the patients arrive at ID states $\{1,2,3\}$ with probabilities $\{0.61, 0.33, 0.06\}$, respectively. The number of patients is set to be 1000, and hence, we have $N = 1000$ number of rounds. The true transition probabilities of each action are given in Appendix 3.4. Next, we set the confidence value according to [19], which is given as

$$\phi_n(b,s,a) = \sqrt{\frac{2\log N_n(b,s)}{N_n(b,s,a)}}.$$

This confidence value is proven to yield order optimal performance for the stochastic MAB problem. We run CODY with three different partitions of the context space.

- $B_1 : \{\{(H,F)\},\{(H,nF)\},\{(nH,F)\},\{(nH,nF)\}\}$ (All possible combinations)
- $B_2 : \{\{(H,F),(H,nF)\},\{(nH,F),(nH,nF)\}\}$ (Combination based on the race)
- $B_3 : \{\{(H,F),(nH,F)\},\{(H,nF),(nH,nF)\}\}$ (Combination based on the family history of having cancer)

3.3.2 Performance of CODY

The performance of CODY is compared with two other algorithms. The first one is OP (pseudo-code is given in Section 3.1). OP uses the true state transition probabilities to calculate the optimal policy for each context. In addition, we compare CODY with the best fixed policy over all time slots which we refer to it as BFP. BFP knows the true state transition probabilities but does not use the context information.

In Table 3.1, the five-year survival percentage of 1000 patients is reported for the three considered algorithms. For CODY, three results are provided, where each result corresponds to a specific partition used by CODY. The improvement in the performance of CODY as a function of rounds is given in Figure 3.2. This shows that the performance of CODY approaches to the performance of OP as the number of rounds increase.

Metric / Methods	OP	BFP	CODY (B_1)	CODY (B_2)	CODY (B_3)
Five-year survival rate	78.00	65.70	74.10	74.75	69.92

Table 3.1: The overall performance of CODY and other algorithms over 1000 patients.

The final average reward per round shows that CODY's performance is 8% better than BFP and only 4% worse than OP. From these results it can be observed that the B_3 partition does not provide enough information about the state transitions while the B_1 and B_2 partitions provide more valuable information about the state transitions.

3.4 Appendix

State transition probabilities are given in the following matrices. The rows correspond to the contexts $\{(H, F), (nH, F), (H, nF), (nH, nF)\}$, respectively, and

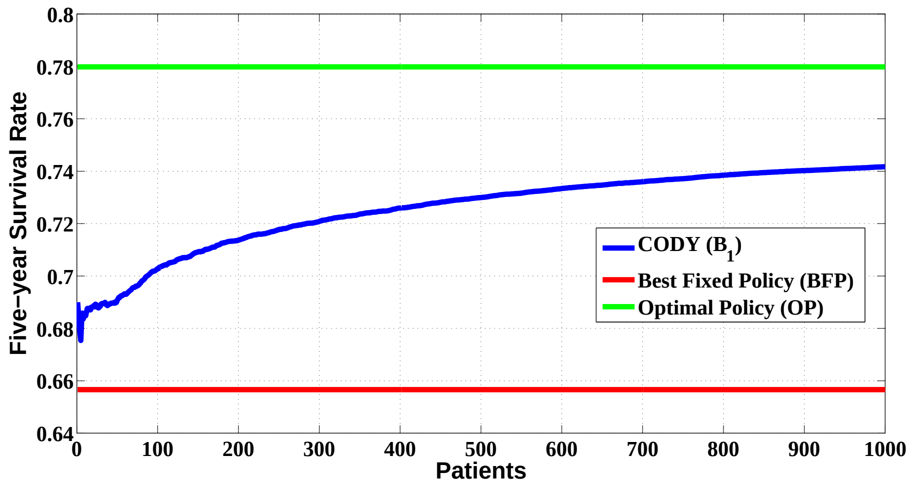

Figure 3.2: The five-year survival rate of the policies averaged over the rounds.

the columns correspond to the ID states $\{1, 2, 3\}$ for action a and b, respectively. For state 0, the transition probability of moving to state 1 is given in q^r vector for each context in $\{(H, F), (nH, F), (H, nF), (nH, nF)\}$, respectively.

$$q^a = \begin{bmatrix} 0.71 & 0.62 & 0.42 \\ 0.76 & 0.69 & 0.49 \\ 0.90 & 0.47 & 0.38 \\ 0.95 & 0.67 & 0.52 \end{bmatrix}, \; q^b = \begin{bmatrix} 0.95 & 0.86 & 0.12 \\ 0.95 & 0.89 & 0.15 \\ 0.94 & 0.88 & 0.57 \\ 0.92 & 0.88 & 0.68 \end{bmatrix}, \; q^r = \begin{bmatrix} 0.38 \\ 0.35 \\ 0.23 \\ 0.15 \end{bmatrix}.$$

Chapter 4

Limit Order Book Trade Marketing

4.1 Problem Formulation

4.1.1 States, Actions, Transitions and Cost

We use $|\mathcal{A}|$ to denote the cardinality of a set $\mathcal{A}$. The system operates in rounds indexed by $\rho \in \{1, 2, \ldots\}$. Each round is composed of L time slots, where L denotes the *maximum execution time*. Time slots are indexed by $l \in \mathcal{L}$. The set of time slots is denoted by $\mathcal{L} := \{1, \ldots, L\}$. The current round ends and a new round begins when the maximum execution time is reached.

***States*:** The system is composed of a finite set of states denoted by $\mathcal{S} := \mathcal{I} \times \mathcal{M}$, where $\mathcal{I}$ denotes the *private state space* and $\mathcal{M}$ denotes the *market state space.*

$\mathcal{I} := \{W_{\min}, \ldots, W_{\max}\}$ is the set of inventory levels of shares at the beginning of each round, where $0 < W_{\min} \leq W_{\max} < \infty$ and $W_{\min}$, $W_{\max}$ are integers. The private state at time slot l of round ρ is denoted by I^l_ρ. We assume that $I^1_\rho = W_\rho$

where $W_\rho \in \mathcal{I}$ is the initial inventory level at round ρ.

Next, we define $\mathcal{M}$. For this, let $p_b(\rho, l)$, $p_a(\rho, l)$ and $p_m(\rho, l)$ be the bid, ask and mid-price (the average between bid and ask price) of time slot l in round ρ. The return of round ρ is defined as

$$\mathrm{Ret}(\rho) := \log(p_m(\rho, L)/p_m(\rho, 1)).$$

The mean of return and volatility (standard deviation of the return) up to round ρ is denoted by μ_ρ and σ_ρ, respectively, which are calculated as follows.

$$\mu_\rho = \frac{1}{\rho-1}\sum_{j=1}^{\rho-1}\mathrm{Ret}(j),\ \sigma_\rho = \sqrt{\frac{1}{\rho-1}\sum_{j=1}^{\rho-1}(\mathrm{Ret}(j)-\mu_\rho)^2}.$$

Based on this, $\mathcal{M}$ is defined as the set of integers that represent the amount of change in the bid price of a time slot in a round from the bid price in the beginning of the round in units of σ_ρ. The market state at time slot l of round ρ is denoted by M_ρ^l. By definition, $M_\rho^1 = 0$ for all rounds. We define the reference price of round ρ as $p_r(\rho)$ and the bid-ask spread in time slot l of round ρ as B_ρ^l as follows.

$$p_r(\rho) := p_m(\rho, 1),\ B_\rho^l := p_a(\rho, l) - p_b(\rho, l)$$

and we have

$$p_b(\rho, l) = p_m(\rho, l) - B_\rho^l/2.$$

In our model, when the market is in state M in time slot l of round ρ, the bid price is modeled as

$$p_b(\rho, l) = p_b(\rho, 1) + M\sigma_\rho.$$

This means that the bid prices are put into discrete values with width σ_ρ, where each one of them is indexed by $M \in \mathcal{M}$. We let $S_\rho^l = (I_\rho^l, M_\rho^l)$ denote the joint state in time slot l of round ρ.

***Actions*:** Actions are defined to be the amount of shares to be traded with a market order[1] [46, 49]. In each round, a sequence of actions is selected with the

[1]A market order to sell is an order to execute a trade at whatever the best prevailing bid price which is a limit order with a price limit of zero at that time.

aim of minimizing the total trade cost. Let a_ρ^l be the action taken at time slot l in round ρ. The number of actions that the user can take in a round is bounded by L. We impose the following assumption on the effect of actions to the market states.

Assumption 1. *It is assumed that the order book is resilient to the trading activities.*

Assumption 1 implies that an action in a time slot has to be chosen such that it does not have an influence on the market state during a round. In practice this means that the market order should not be larger than the depth of the order book at the best bid. This is imposed, for instance, in [45, 46], which effectively prevents taking large actions (large volume of transaction). We assume that the action taken in time slot l of a round cannot be larger than the integer A_l, where A_l, $l \in \mathcal{L}$ is obtained in each round by using the AC model, and hence, we have $\sum_{l=1}^{L} A_l = W_\rho$. Then, we have $a_\rho^l \in \mathcal{A}_l := \{0, \ldots, A_l\}$, $\forall l \in \mathcal{L} - \{L\}$. For $l = L$, the only possible action is to sell the remaining inventory since we require a complete liquidation at the end of a round. Therefore, we have

$$a_\rho^L = I_\rho^L, \ \forall \rho \geq 1.$$

***Transitions*:** Assumption 1 implies that the market state in a round evolves independently from the actions selected by the trader. Hence, the actions only affect the private state, and the market state is modeled as a Markov chain. Let $S' := (I', M')$ and $S := (I, M)$. The state transition probability between time slots l and $l+1$ of round ρ can be written as

$$\begin{aligned} &\Pr(S_\rho^{l+1} = S' | S_\rho^l = S, a_\rho^l = a) = P(M, M')\mathbb{I}(I' = I - a), \\ &\forall S, \forall S' \in \mathcal{S}, \forall a \in \mathcal{A}_l, \forall l \in \mathcal{L} - \{L\}, \forall \rho \geq 1 \end{aligned} \tag{4.1}$$

where $\mathbb{I}(a = b)$ represents the indicator function which is zero when $a \neq b$ and one when $a = b$, and $P(M, M')$ denotes the probability that the market state transitions from M to M'.

Cost: The cost of trade in a round is defined as the *implementation shortfall*, which is given as

$$\mathrm{IS}_\rho := \frac{W_\rho p_r(\rho) - \sum\limits_{l=1}^{L} a_\rho^l p_b(\rho, l)}{W_\rho p_r(\rho)} \tag{4.2}$$

for a sequence of market states $(M_\rho^1, \ldots, M_\rho^L)$, a sequence of actions $(a_\rho^1, \ldots, a_\rho^L)$, an inventory level W_ρ such that $\sum_{l=1}^{L} a_\rho^l = W_\rho$, and a reference price $p_r(\rho)$. The objective is to minimize the accumulated cost in a round. Let $X_\rho := \big(W_\rho, p_r(\rho), \sigma_\rho, B_\rho^1\big)$ be the trade vector of round ρ and let $\mathcal{X}$ be the support of this vector. By using the state definition and B_ρ^1, (4.2) can be re-written as

$$\begin{aligned}
\mathrm{IS}_\rho &= \frac{1}{W_\rho p_r(\rho)} \left[\sum_{l=1}^{L} a_\rho^l \left(p_r(\rho) - p_b(\rho, l) \right) \right] \\
&= \frac{1}{W_\rho p_r(\rho)} \left[\sum_{l=1}^{L} a_\rho^l \left(\frac{B_\rho^1}{2} - M_\rho^l \sigma_\rho \right) \right] \\
&= \sum_{l=1}^{L} C_{X_\rho}(M_\rho^l, a_\rho^l). \qquad (4.3)
\end{aligned}$$

where

$$C_{X_\rho}(M_\rho^l, a_\rho^l) := \frac{1}{W_\rho p_r(\rho)} \left[a_\rho^l \left(\frac{B_\rho^1}{2} - M_\rho^l \sigma_\rho \right) \right]$$

which can be considered as the immediate cost incurred at time slot l of round ρ.

4.1.2 Value Functions and the Optimal Policy

If the state transition probabilities were known in advance, then, the optimal policy can be computed by dynamic programming. In this subsection, we consider this case to gain insight on the form of the optimal policy.

A deterministic Markov policy with time budget L specifies the actions to be taken for each state and trade vector at each time slot. Let $\boldsymbol{\pi} := (\pi_1, \pi_2, \ldots, \pi_L)$ denote such a policy, where $\pi_l : \mathcal{S} \times \mathcal{X} \to \mathcal{A}_l$. We use $\pi_l(M_l, I_l, X)$ to denote the action selected by policy $\boldsymbol{\pi}$ in time slot l when the joint state is (M_l, I_l) in time slot l and the trade vector is X where M_l and I_l represent the market and private variables, respectively. When clear from the context, we will drop the arguments, and represent the action selected by the policy in time slot l by π_l. We replace M_ρ^l and I_ρ^l with M_l and I_l when the round is clear from the context. We define Π denote the set of all deterministic Markov policies with time budget L.

The cost incurred by following policy $\boldsymbol{\pi}$ given trade vector $X \in \mathcal{X}$ is

$$C_X^{\boldsymbol{\pi}} = \sum_{l=1}^{L} C_X \left(M_l, \pi_l(M_l, I_l, X)\right).$$

The optimal policy is the one which minimizes the expected cost of a round which is given as

$$\boldsymbol{\pi}^*(X) := \operatorname*{arg\,min}_{\boldsymbol{\pi} \in \Pi} \mathbb{E}[C_X^{\boldsymbol{\pi}}]$$

where the expectation is taken over the randomness of the market states. The expected cost of the optimal policy given the trade vector X_ρ is denoted by $\mu^*(X)$.

Let $V_l^*(M, I, X)$ denote the expected cost (value function) of policy $\boldsymbol{\pi}^*(X)$ starting from state $S = (M, I)$ at time slot l given X. The Bellman optimality equations [62, 63] are given below: $\forall M \in \mathcal{M},\ \forall I \in \mathcal{I},\ \forall X \in \mathcal{X},\ \forall l \in \mathcal{L} - \{L\}$,

$$\begin{aligned} Q_l^*(M, I, X, a) &:= C_X(M, a) + \mathbb{E}[V_{l+1}^*(M', I - a, X) | M] \\ &= C_X(M, a) + \sum_{M' \in \mathcal{M}} P(M, M') V_{l+1}^*(M', I - a, X). \end{aligned} \tag{4.4}$$

Then, the value function of the optimal policy and optimal action can be obtained $\forall l \in \mathcal{L} - \{L\}$ by

$$\begin{aligned} V_l^*(M, I, X) &= \min_{a \in \mathcal{A}_l} Q_l^*(M, I, X, a) \\ \pi_l^*(M, I, X) &= \operatorname*{arg\,min}_{a \in \mathcal{A}_l} Q_l^*(M, I, X, a). \end{aligned}$$

For the final time slot, we have

$$V_L^*(M, I, X) = C_X(M, I),\ \pi_L^*(M, I, X) = I.$$

In order to obtain the optimal action, these equations need to be solved backwards from time slot L down to 1.

4.2 On the Form of the Optimal Policy

In this section, we show that the optimal policy takes a simple form, which reduces the set of candidates for the optimal action in each time slot to two. Before we discuss the theorem which proves the form of the optimal policy, let us decompose the cost function into two parts as $C_X(M, a) = a g_X(M)$ where $g_X(M)$ is defined to be

$$g_X(M) := \frac{B/2 - M\sigma}{p_r W} \tag{4.5}$$

given the trade vector $X = (W, p_r, \sigma, B)$.

Theorem 4. *Given the LOB model defined in Section 4.1, the optimal action at each time slot is*

$$\pi_l^* = \begin{cases} 0 & \text{if } g_X(M_l) > \mathbb{E}[g_X(M_L)|M_l] \\ A_l & \text{if } g_X(M_l) \leq \mathbb{E}[g_X(M_L)|M_l] \end{cases}, \forall l \in \mathcal{L} - \{L\}$$

and $\pi_L^* = I_L$.

Proof. Let $V_l^\pi(M_l, I_l, X) := \mathbb{E}[\sum_{k=0}^{L-l} C_X M_{l+k}, \pi_{l+k}(M_{l+k}, I_{l+k}, X)|M_l]$ be the value function of policy $\pi \in \Pi$ at time slot l given the triplet (M_l, I_l, X). Using the tower property of the conditional expectation, we obtain

$$\begin{aligned}\mathbb{E}[V_l^\pi(M_l, I_l, X)|M_{l-1}] &= \mathbb{E}\left[\mathbb{E}\left[\sum_{k=0}^{L-l} C_X(M_{l+k}, \pi_{l+k}(M_{l+k}, I_{l+k}, X))\Big|M_l\right]\Big|M_{l-1}\right] \\ &= \sum_{k=0}^{L-l} \mathbb{E}\left[\mathbb{E}\left[C_X(M_{l+k}, \pi_{l+k}(M_{l+k}, I_{l+k}, X))\Big|M_l\right]\Big|M_{l-1}\right] \\ &= \sum_{k=0}^{L-l} \mathbb{E}[C_X(M_{l+k}, \pi_{l+k}(M_{l+k}, I_{l+k}, X))|M_{l-1}].\end{aligned}$$

We use backward induction to prove the theorem. Induction basis consists of time slots L, $L-1$ and $L-2$.

Induction basis:

Since all shares must be sold by the end of a round, in time slot L, the trader must sell all remaining shares I_L. Hence, we have $\pi_L^* = I_L$. We also have

$$\begin{aligned}\mathbb{E}\left[V_L^*(M_L, I_L, X)|M_{L-1}\right] &= \mathbb{E}\left[C_X(M_L, I_L)|M_{L-1}\right] \\ &= \mathbb{E}\left[I_L g_X(M_L)|M_{L-1}\right]\end{aligned}$$

Thus, for π_{L-1}^*, we have

$$\begin{aligned}\pi_{L-1}^*(M_{L-1}, I_{L-1}, X) &= \operatorname*{arg\,min}_{a\in\mathcal{A}_{L-1}}\{C_X(M_{L-1}, a) + \mathbb{E}[V_L^*(M_L, I_{L-1} - a, X)|M_{L-1}]\} \\ &= \operatorname*{arg\,min}_{a\in\mathcal{A}_{L-1}}\{C_X(M_{L-1}, a) + \mathbb{E}\left[C_X(M_L, I_{L-1} - a)|M_{L-1}\right]\} \\ &= \operatorname*{arg\,min}_{a\in\mathcal{A}_{L-1}}\{a g_X(M_{L-1}) + \mathbb{E}[(I_{L-1} - a) g_X(M_L)|M_{L-1}]\} \\ &= \operatorname*{arg\,min}_{a\in\mathcal{A}_{L-1}}\{a g_X(M_{L-1}) + \mathbb{E}[-a g_X(M_L)|M_{L-1}]\} \\ &= \operatorname*{arg\,min}_{a\in\mathcal{A}_{L-1}}\{a\left(g_X(M_{L-1}) - \mathbb{E}[g_X(M_L)|M_{L-1}]\right)\}.\end{aligned}$$

Hence,

$$\begin{cases} g_X(M_{L-1}) > \mathbb{E}[g_X(M_L)|M_{L-1}] \Rightarrow \pi_{L-1}^* = 0 \\ g_X(M_{L-1}) \leq \mathbb{E}[g_X(M_L)|M_{L-1}] \Rightarrow \pi_{L-1}^* = A_{L-1} \end{cases} \Rightarrow \pi_{L-1}^* \in \{0, A_{L-1}\}. \quad (4.6)$$

We also have

$$\begin{aligned}\pi_{L-2}^*(M_{L-2}, I_{L-2}, X) &= \operatorname*{arg\,min}_{a\in\mathcal{A}_{L-2}}\{C_X(M_{L-2}, a) + \mathbb{E}[V_{L-1}^*(M_{L-1}, I_{L-2} - a, X)|M_{L-2}]\} \\ &= \operatorname*{arg\,min}_{a\in\mathcal{A}_{L-2}}\{C_X(M_{L-2}, a) + \mathbb{E}[\pi_{L-1}^*(M_{L-1}, I_{L-2} - a, X) g_X(M_{L-1})|M_{L-2}] \\ &+ \mathbb{E}[\pi_L^* g_X(M_L)|M_{L-2}]\} \\ &= \operatorname*{arg\,min}_{a\in\mathcal{A}_{L-2}}\{C_X(M_{L-2}, a) + \mathbb{E}[\pi_{L-1}^*(M_{L-1}, I_{L-2} - a, X) g_X(M_{L-1})|M_{L-2}] \\ &+ \mathbb{E}[(I_{L-2} - a - \pi_{L-1}^*(M_{L-1}, I_{L-2} - a, X)) g_X(M_L)|M_{L-2}]\}.\end{aligned}$$

From (4.6), we know that π_{L-1}^* only depends on the market statistics. It does not depend on the inventory level, and hence, the action selected in time slot $L-2$. Therefore, we have

$$\begin{aligned}\pi_{L-2}^*(M_{L-2}, I_{L-2}, X) &= \operatorname*{arg\,min}_{a\in\mathcal{A}_{L-2}}\{a g_X(M_{L-2}) + \mathbb{E}[-a g_X(M_L)|M_{L-2}]\} \\ &= \operatorname*{arg\,min}_{a\in\mathcal{A}_{L-2}}\{a(g_X(M_{L-2}) - \mathbb{E}[g_X(M_L)|M_{L-2}])\}.\end{aligned}$$

Thus,

$$\begin{cases} g_X(M_{L-2}) > \mathbb{E}[g_X(M_L)|M_{L-2}] \Rightarrow \pi^*_{L-2} = 0 \\ g_X(M_{L-2}) \leq \mathbb{E}[g_X(M_L)|M_{L-2}] \Rightarrow \pi^*_{L-2} = A_{L-2} \end{cases} \Rightarrow \pi^*_{L-2} \in \{0, A_{L-2}\}.$$

Induction step:

Fix $l \in \{1, \dots, L-3\}$. We will prove that if $\pi^*_{l+k} \in \{0, A_{l+k}\}$, $\forall k \in \{1, \dots, L-l-1\}$, where π^*_{l+k}'s only depend on the market statistics, then $\pi^*_l \in \{0, A_l\}$. We have

$$\begin{aligned}
\pi^*_l(M_l, I_l, X) &= \underset{a\in\mathcal{A}_l}{\arg\min}\{C_X(M_l, a) + \mathbb{E}[V^*_{l+1}(M_{l+1}, I_{l+1}, X)|M_l]\} \\
&= \underset{a\in\mathcal{A}_l}{\arg\min}\left\{C_X(M_l, a) + \sum_{k=1}^{L-l} \mathbb{E}[C_X(M_{l+k}, \pi^*_{l+k}(M_{l+k}, I_{l+k}, X))|M_l]\right\}. \\
&= \underset{a\in\mathcal{A}_l}{\arg\min}\left\{a g_X(M_l) + \sum_{k=1}^{L-l-1} \mathbb{E}[\pi^*_{l+k}(M_{l+k}, I_{l+k}, X) g_X(M_{l+k})|M_l]\right. \\
&\left. + \mathbb{E}\left[\left(I_l - a - \sum_{k=1}^{L-l-1} \pi^*_{l+k}\right) g_X(M_L) \Big| M_l\right]\right\}
\end{aligned}$$

where the last equality holds since

$$\pi^*_L = I_L = I_l - a - \sum_{k=1}^{L-l-1} \pi^*_{l+k}.$$

Since by the induction assumption π^*_{l+k}, $k \in \{1, \dots, L-l-1\}$ only depends on the market statistics, they are all independent from a. Therefore, we have

$$\begin{aligned}
\pi^*_l(M_l, I_l, X) &= \underset{a\in\mathcal{A}_l}{\arg\min}\{a g_X(M_l) + \mathbb{E}[-a g_X(M_L)|M_l]\} \\
&= \underset{a\in\mathcal{A}_l}{\arg\min}\{a\left(g_X(M_l) - \mathbb{E}[g_X(M_L)|M_l]\right)\}
\end{aligned}$$

from which we obtain

$$\begin{cases} g_X(M_l) > \mathbb{E}[g_X(M_L)|M_l] \Rightarrow \pi^*_l = 0 \\ g_X(M_l) \leq \mathbb{E}[g_X(M_L)|M_l] \Rightarrow \pi^*_l = A_l \end{cases} \Rightarrow \pi^*_l \in \{0, A_l\}.$$

This proves that $\pi^*_l \in \{0, A_l\}$, $\forall l \in \{1, \dots, L-1\}$.

$\square$

The theorem shows that the optimal action at each time slot depends on the current market state and the distribution of the market state at the final time slot given the current state. The trader may decide to sell all of the available limit at the current time slot or save the shares up to the final time slot. The intuitive reason behind the result is that we have a linear cost function in a and $g_X(M)$. If the expected market state is greater than the current market state, we desire to wait and sell the maximum allowed amount of shares to sell in the current time slot in the final time slot. The reason for this is that, the final time slot is the only time slot where we can sell more than the pre-defined limit. Thus, the set of candidate optimal actions is given as $A_l^* := \{0, A_l\}, \forall l \in \mathcal{L} - \{L\}$. Therefore, the learning problem reduces to learning the best of these two actions in each time slot. In addition, the number of candidate optimal policies reduces to 2^{L-1}.

4.3 The Learning Algorithm

In this section, we propose a learning algorithm that learns the optimal policy by exploiting the form of the optimal policy given in the previous section by learning the state transition probabilities of the Markov chain of the market variables. This algorithm is named as *Greedy exploitation in Limit Order Book Execution* (GLOBE) and its pseudo-code is given in Algorithm 4.

By round ρ, GLOBE observes $(\rho - 1)(L - 1)$ state transitions. Let $N_\rho(M, M')$ denote the number of occurrences of a state transition from market state M to M' and $N_\rho(M)$ denote the number of times market state M is visited by round ρ. The estimate of the transition probability from state M to M' used at round

Algorithm 4 GLOBE

```
1: Input: $L$, $\mathcal{S}$
2: Initialize: $\rho = 1$, $N_1(M, M') = N_1(M) = 0$, $\hat{P}_\rho(M, M') = 0$, $\forall M, M' \in \mathcal{M}$
3: while $\rho \geq 1$ do
4:
       $$\hat{P}_\rho(M, M') = \frac{N_\rho(M, M') + \mathbb{I}(N_\rho(M) = 0)}{N_\rho(M) + |\mathcal{M}|\mathbb{I}(N_\rho(M) = 0)}$$
5:     Update $\sigma_\rho$ and temporary price impact parameter of the AC model based
       on the past observations
6:     Observe $X_\rho = (W_\rho, p_r(\rho), \sigma_\rho, B^1_\rho)$
7:     Compute $A_l$ [45,46], $\forall l \in \mathcal{L}$
8:     $I^1_\rho = W_\rho$, $l = 1$
9:     while $l < L$ do
10:        Observe market state $M^l_\rho$
11:        Compute $\hat{g}_{X_\rho}(M^l_\rho) := \mathbb{E}[g_{X_\rho}(M_L)|M^l_\rho]$ using $\hat{P}_\rho(M, M')$, $M, M' \in \mathcal{M}$
12:        $a^l_\rho = A_l \times \mathbb{I}(g_{X_\rho}(M^l_\rho) \leq \hat{g}_{X_\rho}(M^l_\rho))$
13:        Calculate $C_{X_\rho}(M^l_\rho, a^l_\rho)$
14:        $I^{l+1}_\rho = I^l_\rho - a^l_\rho$
15:        $l = l + 1$
16:    end while
17:    $a^L_\rho = I^L_\rho$
18:    $\rho = \rho + 1$
19:    Compute $N_\rho(M)$ and $N_\rho(M, M')$, $\forall M, M' \in \mathcal{M}$
20: end while
```

ρ is denoted by $\hat{P}_\rho(M, M')$. These variables are calculated as follows:

$$N_\rho(M, M') = \sum_{\rho_i=1}^{\rho} \sum_{l_i=1}^{l-1} \mathbb{I}(M^{l_i}_{\rho_i} = M)\mathbb{I}(M^{l_i+1}_{\rho_i} = M'),$$

$$N_\rho(M) = \sum_{\rho_i=1}^{\rho} \sum_{l_i=1}^{l-1} \mathbb{I}(M^{l_i}_{\rho_i} = M) = \sum_{M' \in \mathcal{M}} N_\rho(M, M'),$$

$$\hat{P}_t(M, M') = \frac{N_\rho(M, M')}{N_\rho(M)}, \text{ for } N_\rho(M) > 0.$$

After observing the state of each time slot l, it implements the rule given in Theorem 4 to decide on whether to sell A_l or 0, by finding the expected market state in the final time slot of the round using $\hat{P}_\rho(M, M')$s. This allows GLOBE to operate fast without solving the Bellman optimality equations for all actions in each round. The above procedure repeats in each round.

4.4 Numerical Analysis

Our numerical analysis is based on four real-world datasets that contain the order book data for Apple, Amazon, Google, Intel and Microsoft shares traded on NASDAQ to illustrate some aspects of the performance of GLOBE algorithm and other modified version of it with respect to state-of-are algorithm.[2]

In this problem, the trader wants to sell W_ρ number of shares in round ρ at the best price using market orders. We take the difference in the bid prices as the only market variable and the private variable is assumed to be the inventory level. The number of states varies from dataset to dataset based on the volatility scale. We assume that when the market is at state M in time slot l of round ρ, we have

$$p_b(\rho, 1) + (M - 0.5)\sigma_\rho \leq p_b(\rho, l) \leq p_b(\rho, 1) + (M + 0.5)\sigma_\rho.$$

To reduce the number of market states, we use $20\sigma_\rho$ instead of σ_ρ in the market state definition and the above inequalities, for which GLOBE is shown to work well. The initial inventory level of each round is drawn uniformly at random from $[10, 100]$. The time horizon is approximately 6 hours and 30 minutes. Each data instance for each time slot is created by taking the average of the mid/bid/ask prices for every 10 second interval. Then, the dataset is divided into rounds, where each round consists of $L = 4$ consecutive time slots.

The volatility parameter used in the AC model is updated in an online manner as we observe more data. Furthermore, similar to [46], we set the permanent price impact parameter to 0. In addition, we set $\lambda = 0.1$ in the AC model given in [45], and the temporary price impact parameter is calculated and updated according to [45] in an online manner as well. Next, we describe the algorithms that we compare GLOBE against:[3]

(1) Equal Shares (EQ): In this method, we divide the shares equally among the time slots and at each time slot of round ρ, we sell $\lfloor W_\rho / L \rceil$ [48] except the

[2]`https://lobsterdata.com/info/DataSamples.php`.

[3]The results of all of the Q-learning based methods are averaged over 50 iterations.

ending time slot where the remaining inventory is sold (this fact happens due to discretization and rounding W_ρ/L into an integer value).

(2) Almgren Chriss (AC): This algorithm is defined in [45]. We use the same algorithm, however, the volatility and temporary price impact parameters are updated after each round. In addition, we round the suggested number of shares to be sold in each time slot to an integer value. Therefore, in the final time slot, we sell of the remaining inventory.

(3) Q-Exp: This is a Q-learning based method, which uses the state space defined in [46] and the action space defined in this chapter. It uses the ϵ-greedy policy [32] to explore the actions with probability of 0.5 or exploit with probability of 0.95. In this method, the market state space is the combination of bid-ask spread and bid volumes, where each variable consists of 10 different states. The result is averaged over 100 iterations.

(4) Q-Mat: This is the method proposed in [46], but with the action space defined as in this chapter. This method uses the first half of the dataset as the training set to calculate the Q values and build a Q-matrix for all combination of market and inventory states, actions and time slot. The rest of the data is used as the test set. In this method, the market state space is the combination of bid-ask spread and bid volumes as given in Q-Exp method definition. The result is averaged over 100 iterations as well.

(5, 6, 7) Q-Exp+, Q-Mat+ and GLOBE+: These are almost the same as Q-Exp, Q-Mat and GLOBE, where the only difference is that the set of available actions in time slot l ranges from 0 to $2A_l$.

For each method, we calculate the Averaged Cost Per Round (ACPR) at the end of each round, which is given as

$$\mathrm{ACPR}_\rho = \sum_{i=1}^{\rho} \frac{\mathrm{IS}_i}{\rho}. \tag{4.7}$$

Then, we compare ACPR_R of each method (alg) against the AC model by round R starting from the first round in the test set, using a performance metric similar

Method / Dataset	GOOG	AMZN	INTC	MSFT
EQ	0.35	-0.39	0.409	0.327
Q-Exp	-3.974	-2.677	-4.705	-2.833
Q-Mat	-5.429	-5.087	-7.491	-4.68
Q-Exp+	-0.009	-0.955	0.237	0.002
Q-Mat+	-2.913	-3.089	-7.454	-2.979
GLOBE	-0.611	0.501	1.477	0.443
GLOBE+	4.763	3.793	8.516	5.452

Table 4.1: RC of the algorithms at the end of the time horizon with respect to the AC model calculated over the test set.

to the one used in [64], which we call it as the Relative averaged Cost per round (RC), given as

$$\mathrm{RC}_R(alg) := \frac{\mathrm{ACPR}_R(AC) - \mathrm{ACPR}_R(alg)}{|\mathrm{ACPR}_R(AC)|} \times 100.$$

In Table 4.1 we report the RC of the algorithms for the second half of the dataset (test dataset)[4]. We observe that GLOBE outperforms other methods in three of the datasets while GLOBE+ outperforms other methods in all of the datasets with a much larger margin. We also would like to mention that the poor result of GLOBE in the GOOG dataset is possibly due to the limited number of samples available in these datasets. We expect the performance to improve, when the algorithm learns through larger datasets.

In addition, we also show the standard deviation of ACPR over the test set for all of the algorithms in Table 4.2. The table shows that GLOBE and GLOBE+ algorithms are among the methods with the lowest standard deviation in general.

In addition, the ACPR value for all rounds in the test set is depicted for the four mentioned datasets in Figures 4.1, 4.2, 4.3 and 4.4. As it is clear from the figures, GLOBE rand GLOBE+ algorithms has lower costs in general with respect to other algorithms.

[4]This is done in order to have a fair comparison between the models.

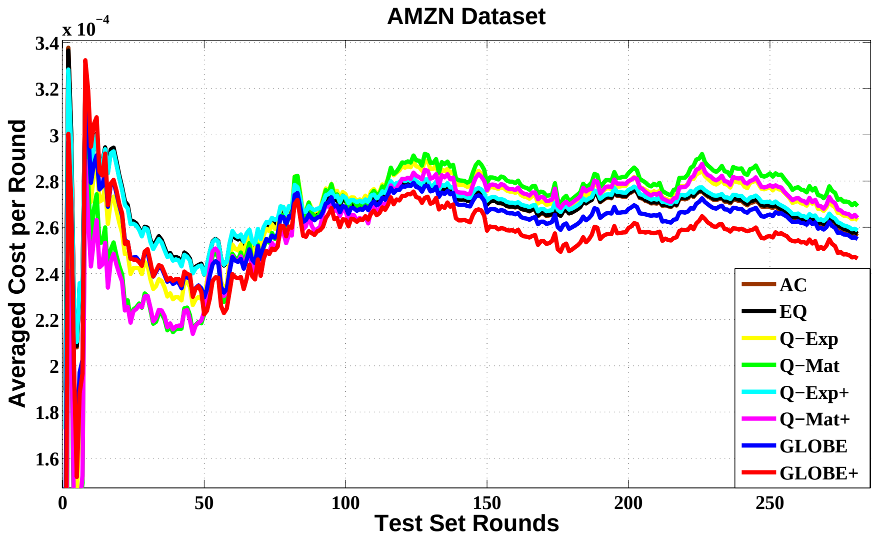

Figure 4.1: This figure illustrates the average cost per round of all of the algorithms over the test set for AMZN dataset.

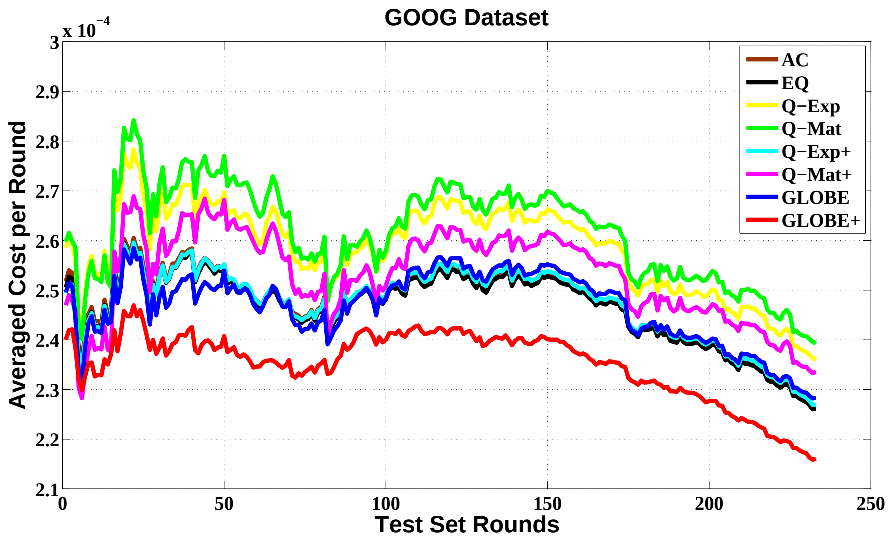

Figure 4.2: This figure illustrates the average cost per round of all of the algorithms over the test set for GOOG dataset.

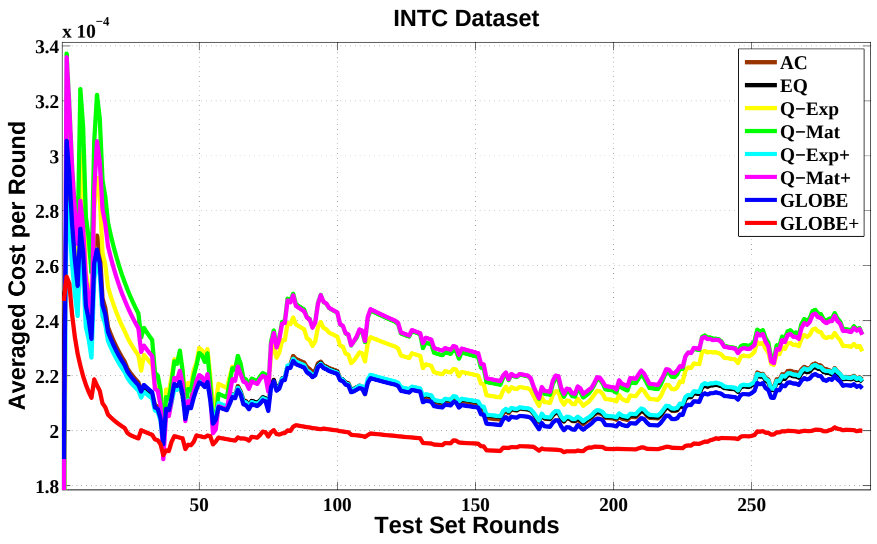

Figure 4.3: This figure illustrates the average cost per round of all of the algorithms over the test set for INTC dataset.

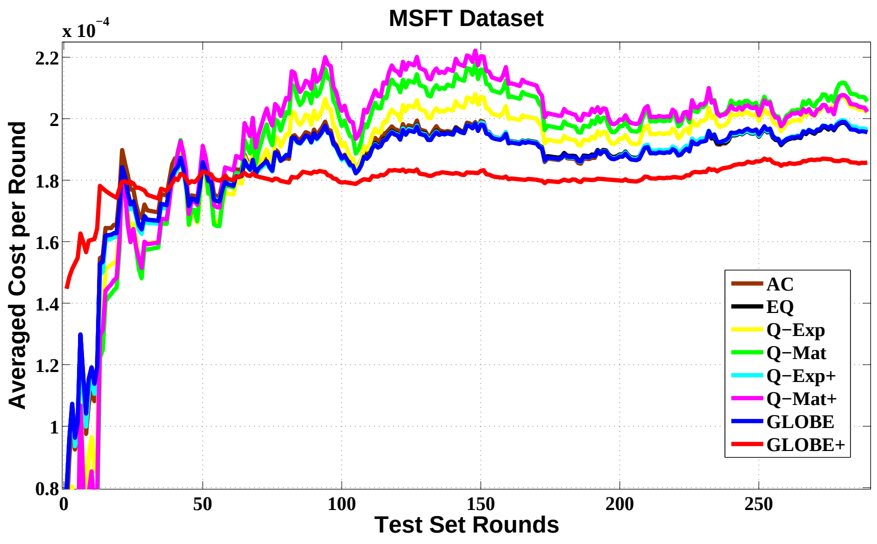

Figure 4.4: This figure illustrates the average cost per round of all of the algorithms over the test set for MSFT dataset.

Method / Dataset	GOOG	AMZN	INTC	MSFT
AC	0.166	0.223	0.188	0.192
EQ	0.167	0.224	0.201	0.207
Q-AC	0.266	0.392	0.316	0.338
Q-Mat	0.233	0.299	0.277	0.277
Q-AC+	0.207	0.294	0.226	0.247
Q-Mat+	0.219	0.279	0.268	0.264
GLOBE	0.19	0.262	0.199	0.199
GLOBE+	0.118	0.317	0.068	0.067

Table 4.2: Standard deviation of the costs of the algorithms incurred over the test set. All of the values are multiplied by 10^3.

Chapter 5

Conclusion

We study the problem of learning in structured Markov decision processes when some problem parameters (such as the state transition probabilities) are unknown. We propose a series of online learning algorithms that learn the unknown parameters on-the-fly to maximize a certain objective.

In Chapter 2, we introduce the GRBP, which is inspired by the gambler's ruin problem. We prove that the optimal policy is a threshold policy. Then, we introduce an algorithm named GETBE that exploits the form of the optimal policy to learn fast. We also prove that the regret of GETBE is either finite or $O(\log(n))$ depending on the unknown problem parameters.

In Chapter 3, we introduce the contextual goal-oriented with dead-end states MDP. This MDP is parameterized by a context variable, which is assumed to change from round to round. We propose a learning algorithm called CODY that performs context space partitioning to exploit similarities between the contexts, and implements dynamic programming with estimated state transition probabilities to approximate the optimal policy. Finally, we provide a healthcare application of CODY and analyze its performance numerically.

In Chapter 4, we discuss the online learning problem in limit order book trade execution. We consider a selling problem where a number of shares have to be sold within a certain time span. Similar to the first problem, we derive the form of optimal policy and propose an online learning algorithm called GLOBE, which exploits the form of the optimal policy to speed up learning. We show on a financial dataset that GLOBE is significantly better than Q-learning based methods proposed in prior works.

Bibliography

[1] A. Tewari and P. Bartlett, "Optimistic linear programming gives logarithmic regret for irreducible MDPs," *Advances in Neural Information Processing Systems*, vol. 20, pp. 1505–1512, 2008.

[2] T. Jaksch, R. Ortner, and P. Auer, "Near-optimal regret bounds for reinforcement learning," *Journal of Machine Learning Research*, vol. 11, no. Apr, pp. 1563–1600, 2010.

[3] Y. Nevmyvaka, Y. Feng, and M. Kearns, "Reinforcement learning for optimized trade execution," in *Proc. 23rd Int. Conf. Machine Learning*, pp. 673–680, 2006.

[4] A. Bayoumi and M. Bennewitz, "Learning optimal navigation actions for foresighted robot behavior during assistance tasks," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 207–212, May 2016.

[5] R. Gillani and A. Nasir, "Incorporating artificial intelligence in shopping assistance robot using Markov decision process," in *2016 International Conference on Intelligent Systems Engineering (ICISE)*, pp. 94–99, Jan 2016.

[6] C. C. Bennett and K. Hauser, "Artificial intelligence framework for simulating clinical decision-making: A Markov decision process approach," *Artificial Intelligence in Medicine*, vol. 57, no. 1, pp. 9–19, 2013.

[7] R. A. Howard, *Dynamic programming and Markov processes.* The MIT press, 1960.

[8] D. P. Bertsekas, *Dynamic programming and optimal control*, vol. 1. Athena Scientific Belmont, MA, 1995.

[9] D. P. Bertsekas and J. N. Tsitsiklis, “An analysis of stochastic shortest path problems,” *Mathematics of Operations Research*, vol. 16, no. 3, pp. 580–595, 1991.

[10] F. Teichteil-Konigsbuch, “Stochastic safest and shortest path problems,” in *AAAI*, 2012.

[11] A. Kolobov, M. Mausam, D. S. Weld, and H. Geffner, “Heuristic search for generalized stochastic shortest path MDPs,” in *21st International Conference on Automated Planning and Scheduling*, 2011.

[12] A. Kolobov, Mausam, and D. Weld, “A theory of goal-oriented MDPs with dead ends,” in *UAI*, pp. 438–447, 2012.

[13] S. H. A. Ahmad, M. Liu, T. Javidi, Q. Zhao, and B. Krishnamachari, “Optimality of myopic sensing in multichannel opportunistic access,” *IEEE Trans. Inf. Theory*, vol. 55, no. 9, pp. 4040–4050, 2009.

[14] Y. Liu, M. Liu, and S. H. A. Ahmad, “Sufficient conditions on the optimality of myopic sensing in opportunistic channel access: A unifying framework,” *IEEE Trans. Inf. Theory*, vol. 60, no. 8, pp. 4922–4940, 2014.

[15] C. Tekin and M. Liu, “Online learning of rested and restless bandits,” *IEEE Trans. Inf. Theory*, vol. 58, no. 8, pp. 5588–5611, 2012.

[16] S. S. .Villar, J. Bowden, and J. Wason, “Multi-armed bandit models for the optimal design of clinical trials: Benefits and challenges,” *Statistical Science*, vol. 30, no. 2, pp. 199–215, 2015.

[17] C. Tekin and M. van der Schaar, “RELEAF: An algorithm for learning and exploiting relevance,” *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 4, pp. 716–727, 2015.

[18] Lai, T. L., Robbins, and Herbert, “Asymptotically efficient adaptive allocation rules,” *Advances in Applied Mathematics*, vol. 6, no. 1, pp. 4–22, 1985.

[19] P. Auer, Cesa-bianchi, N. o, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, pp. 235–256, 2002.

[20] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3/4, pp. 285–294, 1933.

[21] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and non-stochastic multi-armed bandit problems," *Foundations and Trends in Machine Learning*, vol. 5, no. 1, pp. 1–122, 2012.

[22] A. Garivier and O. Cappé, "The KL-UCB algorithm for bounded stochastic bandits and beyond," in *COLT*, pp. 359–376, 2011.

[23] P. Auer and R. Ortner, "UCB revisited: Improved regret bounds for the stochastic multi-armed bandit problem," *Periodica Mathematica Hungarica*, vol. 61, no. 1-2, pp. 55–65, 2010.

[24] Gittins, J.C., Jones, and D.M., "A dynamic allocation index for the sequential design of experiments," *Progress in Statistics Gani, J. (ed.)*, pp. 241–266, 1974.

[25] S. Agrawal and N. Goyal, "Analysis of Thompson sampling for the multi-armed bandit problem," *Journal of Machine Learning Research*, vol. 23, no. 39, pp. 285–294, 2012.

[26] A. Gopalan and S. Mannor, "Thompson sampling for learning parameterized Markov decision processes," in *COLT*, pp. 861–898, 2015.

[27] R. Ortner, "Online regret bounds for Markov decision processes with deterministic transitions."

[28] P. Guan, M. Raginsky, and R. M. Willett, "Online Markov decision processes with Kullback–Leibler control cost," *IEEE Transactions on Automatic Control*, vol. 59, no. 6, pp. 1423–1438, 2014.

[29] G. Neu, A. Gyorgy, C. Szepesvari, and A. Antos, "Online Markov decision processes under bandit feedback," *IEEE Transactions on Automatic Control*, vol. 59, pp. 676–691, March 2014.

[30] O. Dekel and E. Hazan, "Better rates for any adversarial deterministic MDP," in *ICML*, vol. 28, pp. 675–683, 2013.

[31] V. F. Farias, C. C. Moallemi, B. Van Roy, and T. Weissman, "Universal reinforcement learning," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2441–2454, 2010.

[32] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, vol. 1. MIT Press Cambridge, 1998.

[33] G. Harik, E. Cant-Paz, D. E. Goldberg, and B. L. Miller, "The gambler's ruin problem, genetic algorithms, and the sizing of populations," *Evolutionary Computation*, vol. 7, pp. 231–253, Sept 1999.

[34] L. Takacs, "On the classical ruin problems," *J. Amer. Statisistical Association*, vol. 64, pp. 889–906, 1969.

[35] B. Hunter, A. C. Krinik, C. Nguyen, J. M. Switkes, and H. F. von Bremen, "Gambler's ruin with catastrophe and windfalls," *Statistical Theory and Practice*, vol. 2, no. 2, pp. 199–219, 2008.

[36] W. Beyer and M. Waterman, "Symmetries for conditioned ruin problems," *Mathematics Magazine*, vol. 50, no. 1, pp. 42–45, 1977.

[37] W. Feller, *An introduction to probability theory and its applications*, vol. 1. John Wiley & Sons New York, 1968.

[38] T. van Uem, "Maximum and minimum of modified gambler's ruin problem, arxiv:1301.2702," 2013.

[39] N. Cesa-Bianchi and G. Lugosi, "Combinatorial bandits," *Journal of Computer and System Sciences*, vol. 78, no. 5, pp. 1404–1422, 2012.

[40] E. A. McGlynn, S. M. Asch, J. Adams, J. Keesey, J. Hicks, A. DeCristofaro, and E. A. Kerr, "The quality of health care delivered to adults in the United States," *New England Journal of Medicine*, vol. 348, no. 26, pp. 2635–2645, 2003.

[41] V. L. Patel, E. H. Shortliffe, M. Stefanelli, P. Szolovits, M. R. Berthold, R. Bellazzi, and A. Abu-Hanna, "The coming of age of artificial intelligence in medicine," *Artificial Intelligence in Medicine*, vol. 46, no. 1, pp. 5–17, 2009.

[42] F. A. Sonnenberg and J. R. Beck, "Markov models in medical decision making: a practical guide," *Medical Decision Making*, vol. 13, no. 4, pp. 322–338, 1993.

[43] D. Palguna and I. Pollak, "Non-parametric prediction in a limit order book," in *Proc. IEEE Global Conf. Signal and Information Processing (GlobalSIP)*, pp. 1139–1139, 2013.

[44] Y. Feng, D. P. Palomar, and F. Rubio, "Robust optimization of order execution," *IEEE Transactions on Signal Processing*, vol. 63, no. 4, pp. 907–920, 2015.

[45] R. Almgren and N. Chriss, "Optimal execution of portfolio transactions," *Journal of Risk*, vol. 3, pp. 5–40, 2001.

[46] D. Hendricks and D. Wilcox, "A reinforcement learning extension to the Almgren-Chriss framework for optimal trade execution," in *Proc. IEEE Conf. Computational Intelligence for Financial Engineering & Economics (CIFEr)*, pp. 457–464, 2014.

[47] R. Cont and A. De Larrard, "Price dynamics in a Markovian limit order market," *SIAM Journal on Financial Mathematics*, vol. 4, no. 1, pp. 1–25, 2013.

[48] D. Bertsimas and A. W. Lo, "Optimal control of execution costs," *Journal of Financial Markets*, vol. 1, no. 1, pp. 1–50, 1998.

[49] A. A. Sherstov and P. Stone, "Three automated stock-trading agents: A comparative study," in *Int. Workshop Agent-Mediated Electronic Commerce*, pp. 173–187, 2004.

[50] P. Auer and R. Ortner, "Logarithmic online regret bounds for undiscounted reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 19, p. 49, 2007.

[51] N. Akbarzadeh and C. Tekin, "Gambler's ruin bandit problem," in *54th Annual Allerton Conference on Communication, Control, and Computing*, pp. 1236–1243, Sept 2016.

[52] R. Bellman and R. E. Kalaba, *Dynamic programming and modern control theory*, vol. 81. New York: Academic Press, 1965.

[53] T. Isobe, K. Hashimoto, J. Kizaki, M. Miyagi, K. Aoyagi, K. Koufuji, and K. Shirouzu, "Characteristics and prognosis of gastric cancer in young patients," *International Journal of Oncology*, vol. 30, no. 1, pp. 43–49, 2013.

[54] "http://www.cancer.org/cancer/stomachcancer/detailedguide/stomach-cancer-survival-rates."

[55] M. El-Shehawey, "On the gamblers ruin problem for a finite Markov chain," *Statistics & Probability Letters*, vol. 79, no. 14, pp. 1590–1595, 2009.

[56] C. Tekin and N. Akbarzadeh, "Tıbbi karar süreçleri için Markov modeller ve öğrenme algoritmaları," in *Uluslararası Katılımlı 16. Üretim Araştırmaları Sempozyumu*, pp. 1091–1096, Oct 2016.

[57] P. M. Pardalos and H. E. Romeijn, *Handbook of optimization in medicine*, vol. 26. Springer Science & Business Media, 2009.

[58] A. Slivkins, "Contextual bandits with similarity information," *Journal of Machine Learning Research*, vol. 15, pp. 2533–2568, 2014.

[59] S. Bubeck, R. Munos, and G. Stoltz, "Pure exploration in multi-armed bandits problems," in *International Conference on Algorithmic Learning Theory*, pp. 23–37, Springer, 2009.

[60] M. Wenten, F. D. Gilliland, K. Baumgartner, and J. M. Samet, "Associations of weight, weight change, and body mass with breast cancer risk in hispanic and non-hispanic white women," *Annals of epidemiology*, vol. 12, no. 6, pp. 435–444, 2002.

[61] E. B. T. F. S. H. T. M.-S. M. M. C. P. A. P. P. M. G. S. A. S. S. S. T. T. L. T. E. M. T. D. W. D. American Cancer Society, Barrera, "Cancer facts and figures for Hispanics/Latinos 2015-2017," pp. 1–48, 2015.

[62] R. Bellman, "Dynamic programming and stochastic control processes," *Information and Control*, vol. 1, no. 3, pp. 228–239, 1958.

[63] D. P. Bertsekas, *Dynamic programming and optimal control*, vol. 1. Athena Scientific Belmont, MA, 1995.

[64] D. Palguna and I. Pollak, "Mid-price prediction in a limit order book," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 6, pp. 1083–1092, 2016.