7-2024

Sequential decision learning for social good and fairness

Dexun LI
Singapore Management University, dexunli.2019@phdcs.smu.edu.sg

SEQUENTIAL DECISION LEARNING FOR SOCIAL GOOD AND FAIRNESS

DEXUN LI

SINGAPORE MANAGEMENT UNIVERSITY
2024

Sequential Decision Learning for Social Good and Fairness

by

Dexun Li

Submitted to School of Computing and Information Systems in partial fulfillment of the requirements for the Degree of Doctor of Philosophy in Computer Science

Dissertation Committee:

Pradeep VARAKANTHAM (Supervisor / Chair)
Professor of School of Computing and Information Systems
Singapore Management University

Shih-Fen CHENG
Associate Professor of School of Computing and Information Systems
Singapore Management University

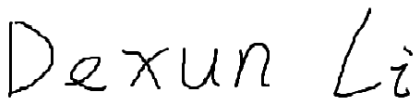
Tien MAI
Assistant Professor of School of Computing and Information Systems
Singapore Management University

Arunesh SINHA
Assistant Professor of Management Science and Information Systems Department
Rutgers University

Singapore Management University
2024

I hereby declare that this PhD dissertation is my original work and it has been written by me in its entirety.
I have duly acknowledged all the sources of information which have been used in this dissertation.

This PhD dissertation has also not been submitted for any degree in any university previously.



Dexun Li

17 July 2024

Sequential Decision Learning for Social Good and Fairness

Dexun Li

Abstract

Sequential decision learning is one of the key research areas in artificial intelligence. Typically, a sequence of events is observed through a transformation that introduces uncertainty into the observations and based on these observations, the recognition process produces a hypothesis of the underlying events. This learning process is characterized by maximizing the sum of the reward signals. However, many real-life problems are inherently constrained by limited resources. Besides, when the learning algorithms are used to inform decisions involving human beings (e.g., Security and justice, health intervention, etc), they may inherit the potential, pre-existing bias in the dataset and exhibit similar discrimination against protected attributes such as race and gender. Therefore, it is essential to ensure fairness constraints are met and budget constraints are not violated when applying sequential decision learning algorithms in real-world scenarios. In this dissertation, we focus on the practical problem of fair sequential decision learning that contributes to the social good, within settings of Restless Multi-Armed Bandits (RMAB) and Reinforcement Learning (RL). In particular, the dissertation is split into two major parts.

In the first part of the work, we consider the RMAB setting. RMAB is an apt model to represent decision learning problems in public health interventions (e.g., tuberculosis, maternal, and child care), anti-poaching planning, sensor monitoring, personalized recommendations, and many more. In the context of public health settings, the problem is characterized by multiple arms (i.e., patients) whose state evolves in an uncertain manner (e.g., medication usage in the case of tuberculosis) and threads moving to ”bad” states have to be steered

to ”good” outcomes through interventions. Due to the limited resources (e.g., public health workers), typically certain individuals, communities, or regions are starved of interventions, which can potentially have a significant negative impact on the individual/community in the long term. To that end, we argue the need to ensure fairness during decision-making (e.g., select arms/patients to give health interventions). We, therefore, combine recent advances in RMAB research with our proposed definition of fairness in the face of uncertainty to develop a scalable and efficient algorithm to learn a policy that can handle fairness constraints without sacrificing significant solution quality. We provide theoretical performance guarantees and validate our approaches on simulated benchmarks.

In the second part of the thesis, we address the sequential decision learning in a reinforcement learning setting, starting with the problem of influence maximization in an unknown social network. The objective is to identify a set of peer leaders within a real-world physical social network who can disseminate information to a large group of people. This approach has found a wide range of applications, including HIV prevention, substance abuse prevention, microfinance adoption, etc. Unlike online social networks, real-world networks are not completely known, and collecting information about the network is costly as it involves surveying multiple people. Specifically, we focus on the problem of the network discovery process for influence maximization with a limited budget (i.e., certain numbers of surveying). Because interactions with the environment in real-world settings are costly, it is crucial for reinforcement learning algorithms to have minimum possible environment interactions, i.e., to be sample efficient. To achieve this, we propose a curriculum-based approach that enhances the sample efficiency of existing RL methods. Our proposed algorithm has been demonstrated to outperform existing approaches in a sample-efficient manner.

We further explore training generally capable RL agents in complex environments. Recent research has highlighted the potential of the Unsupervised

Environment Design (UED), a framework that automatically generates a curriculum of training environments. Agents trained in these environments can develop general capabilities. Specifically, our focus lies on applying UED in scenarios where resources are limited, characterized by a limited number of generated environments and limited training horizons. To this end, we introduce a hierarchical MDP framework, which consists of an upper-level RL teacher agent tasked with generating suitable training environments for a lower-level student agent. The RL teacher can leverage previously discovered environment structures and generate challenging environments at the frontier of the student's capabilities by observing the representation of the student policy. We incorporate an additional fairness reward to accurately guide the environment generation process and leverage recent advances in generative models to minimize the costly collection of experiences required to train the teacher agent. Our proposed method significantly reduces the resource-intensive interactions between agents and environments, and empirical experiments across various domains demonstrate the effectiveness of our approach. Our research can lead to more principled, robust, and widely accepted systems that can be used to assist in training non-expert humans.

Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I am deeply indebted to my supervisor, Prof. Pradeep Varakantham, for his invaluable guidance and profound insights throughout my research journey. His mentorship has significantly shaped the direction of this thesis, my academic perspective, and my career over the past years.

I am also immensely thankful to the members of my thesis committee, Prof. Shih-Fen Cheng, Prof. Tien Mai, and Prof. Arunesh Sinha, for their invaluable time and constructive feedback, which have been crucial in refining this work.

My sincere appreciation goes to Singapore Management University for providing the resources and a conducive research environment essential for completing this thesis. I am grateful to the faculty and staff for their support at every stage of my journey. I am also thankful to my friends and colleagues for their encouragement, stimulating discussions, and shared experiences, which have made this academic pursuit more fulfilling and enjoyable.

I owe a heartfelt thank you to my parents and loved ones for their unwavering love and understanding throughout this challenging yet rewarding journey. Their support has been my cornerstone. I dedicate a special memory to my father, who passed away while I was completing my Ph.D. His support for me never wavered, and I sincerely wish he had been there to see me complete my Ph.D. His memory continues to inspire and guide me every day.

Lastly, I would like to express my deepest appreciation to all those individuals who, directly or indirectly, contributed to this thesis. Your support has been invaluable and is deeply appreciated.

Chapter 1

Introduction

This dissertation is concerned with the study of sequential decision learning in real-world scenarios where the agent faces constraints due to limited resources. This process requires the agent to interact with the environment sequentially amidst uncertainty. The agent iteratively collects information, updating decision parameters based on the cumulative information available, and then makes a decision. The goal is to identify an optimal sequence of actions to perform a specific task. Such systems hold broad potential across various social domains. For example, as shown in Figure 1.1, in the public health intervention domain, they address challenges related to health and hunger, including early-stage diagnosis and optimized food allocation. In the domain of security and justice, they tackle societal challenges by preventing crime and other physical dangers and protecting the world's wildlife and forests from poaching and illegal logging.

However, In many application domains, especially those involving safety-critical or resource-related decision-learning, a common feature is the presence of limited resources, which typically restricts the agent's exploration in the environment during the decision process. These scenarios can lead to budget constraints when deploying algorithms. This necessitates reducing the costly interactions between agents and environments, particularly in the context of reinforcement learning, to achieve sample efficiency. Moreover, when decision-learning algo-

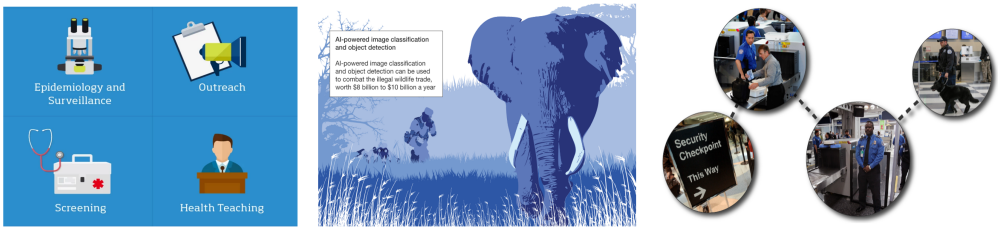


Figure 1.1: The applications of sequential decision-learning in real-world scenarios.

rithms, which use real-world datasets, are applied to inform decisions involving human beings (e.g., health interventions, skill training), they may inherit underlying, pre-existing biases in the dataset and exhibit similar discrimination or sensitivity to protected attributes such as gender. In these cases, it is crucial to consider fairness constraints in methods.

Specifically, we develop and employ algorithmic to address the above challenges through two major settings discussed in Parts II and parts III:

Part II We focus on Restless Multi-Armed Bandits (RMAB)framework for the health intervention problem, particularly in settings where resources are scarce and there is a need to consider fairness constraints for resource allocations.

Part III We explore a reinforcement learning framework for two specific problems, utilizing curriculum-based insights to enhance sample efficiency.

1.1 Motivation for RMAB setting-Part II

Restless Multi-Armed Bandits Process is a generalization of the classical Multi-Armed Bandits (MAB) process, which has been studied since the 1930s [41]. RMAB is a powerful framework for budget-constrained resource allocation tasks in which a decision-maker must select a subset of arms for interventions in each round. Each arm evolves according to an underlying Markov Decision Process (MDP). The overall objective in an RMAB model is to sequentially select arms so as to maximize the expected value of the cumulative rewards collected over all the arms. RMAB is of relevance in public health monitoring

scenarios, recommendation systems and many others. Tracking a patient's health or adherence and intervening at the right time is an ideal problem setting for an RMAB [2, 9, 56], where the patient health/adherence state is represented using an arm. Resource limitation constraint in RMAB comes about due to the severely limited availability of healthcare personnel. By developing practically relevant approaches for solving RMAB within severe resource limitations, RMAB can assist patients in alleviating health issues such as diabetes [66]. hypertension [11], tuberculosis [15, 69], depression [55, 62], etc.

While Whittle index based approaches [56, 46] address the RMAB problem with an infinite time horizon by providing an asymptotically optimal solution and contribute theoretical results, they are susceptible to starving arms, which can have severe repercussions in public health scenarios. Owing to the deterministic selection strategy of picking arms that provide the maximum benefit, in many problems, only a small set of arms typically get picked. As shown in our experimental analysis, where almost 50% of the arms do not get any interventions using the Whittle index approach. While it is an optimal decision, it should be noted that interventions help educate patients or beneficiaries on potential benefits and starvation of such interventions for many patients can result in a lack of proper understanding of the program and reduce its effectiveness in the long run. Thus, there is a need to not starve arms without significantly sacrificing optimality.

Existing works have proposed different notions of fairness in the context of MAB to prevent starvation by enabling the selection of non-optimal arms. Li et al. [50] study a new Combinatorial Sleeping MAB model with Fairness constraints, called CSMAB-F. Their fairness definition requires algorithm to ensure a minimum selection fraction for each arm. Patil et al. [71] introduce similar fairness constraints in the stochastic MAB problem, where they use a pre-specified vector to denote the guaranteed number of selections. Joseph et al. [37] define fairness as saying that a worse arm should not be picked compared to a

better arm, despite the uncertainty on payoffs. Chen et al. [17] form the allocation decision-making problem as the MAB with fairness constraints, where fairness is defined as a minimum rate at which a task or resource is assigned to a user. We build on the notion of fairness for reinforcement learning setting and introduce two different fairness notions for our RMAB setting: a deterministic fairness constraint and a probabilistic fairness constraint. Specifically, the deterministic fairness constraint requires that for any arm (or more generally, for a type of arm), the number of decision epochs since the arm (or the type of arm) was activated last time is upper bounded, while the probabilistic fairness definition requires that an RMAB algorithm never favor an arm probabilistically over another arm, if the long-term cumulative reward of choosing the latter arm is higher. Providing such decision support with a fairness mindset can promote acceptability among community [76, 42].

1.2 Motivation for RL setting-Part III

In contrast to the previous RMAB setting, Reinforcement Learning (RL) is a subset of machine learning in which the agent learns how to behave optimally in an unknown environment. Essentially, the agent must learn what the best action is in each state of the environment over time. The agent in reinforcement learning employs the principle of trial and error to investigate the consequences of actions in a given state. This is accomplished by examining the scalar feedback signal associated with each action. However, interactions between the environment and RL agents are expensive in many real-world applications, limiting our ability to collect data. That raises the necessity for sample-efficient algorithms. Furthermore, fairness concerns highlight the importance of learning risk-sensitive policies in practical applications. In this dissertation, we explore two specific sequential decision-learning problems for social good constrained by limited resources. In both these problems, we employ curriculum-based approaches to

improve sample efficiency and reduce costly interactions.

The first problem of interest is influence maximization, which aims to identify a small subset of nodes in a network that can maximize the diffusion of information. It has found application in HIV prevention, substance abuse prevention, micro-finance adoption, etc. Unlike online social networks, real-world networks are not completely known, and collecting information about network is costly as it involves surveying multiple people. Therefore, in these applications, it is also important to efficiently discover a subset of the network within a limited budget such that selecting peer leaders from this subgraph can help in maximizing the influence in the complete network. The existing work in this direction proposes a reinforcement learning framework to discover a subset of networks within a given budget by leveraging the automatically learned node and graph representations. The set of peer leaders is chosen from the discovered network structure. To learn an efficient policy, the reinforcement learning agent needs to explore the environment which requires multiple interactions with the environment during training. The environment interactions in real-world settings are costly, so it is important for the reinforcement learning algorithms to have minimum possible environment interactions. To address this, we combine recent advances in reinforcement learning in the face of uncertainty to create a scalable algorithm for learning an optimal policy in a sample-efficient manner in order to reduce the number of interactions with the environment. We first translate the problem into a goal-directed learning problem by proposing a novel heuristic to design an appropriate goal for each input state. It then uses insights from past work on Curriculum-guided Hindsight Experience Replay (CHER) to improve sample efficiency. CHER involves replaying each episode with multiple pseudo goals, so the agent can get multiple experiences in a single environment interaction. We conduct experiments on real-world datasets and show that our approach can significantly outperform the existing approach.

The advancements in Reinforcement Learning (RL) have led to significant

successes in various applications, such as game playing [61, 85], robot control [47, 3], and many others. However, training RL agents with general capabilities remains a major challenge due to the millions of experiences required to train an RL agent in each environment, which is both time-consuming and expensive. One promising approach to address this problem is to "shallowly" train an agent on a sequence of tasks or environments [22, 36, 70, 51]. In the second research problem, we focus on designing an adaptive curriculum for the environment generation process to train an RL agent with general capabilities, applicable to real-world human training scenarios. This process involves adaptively generating environment instances/levels at the frontier of the agent's capabilities, which can lead to a more robust agent. However, existing approaches in this area focus primarily on randomly generating environments for open-ended training. The notion of open-ended training requires training an agent across hundreds of thousands of randomly generated environments for hundreds of millions of time steps. This is impractical in scenarios with limited resources, such as the constraints on the number of generated environments. Therefore, it is crucial to design a framework that achieves the same performance of RL agents using fewer generated environments, thus reducing costly interactions between the agents and the environments. We introduce a hierarchical MDP framework for environment design tailored for scenarios with resource constraints. This framework consists of an upper-level RL teacher agent responsible for generating suitable training environments for a lower-level student agent. We incorporate an additional fairness reward to guide the teacher agent to generate suitable environments. Furthermore, we leverage advances in generative models to reduce the time-consuming experience collection process. Our proposed method significantly reduces the resource-intensive interactions between agents and environments and empirical experiments across various domains demonstrate the effectiveness of our approach.

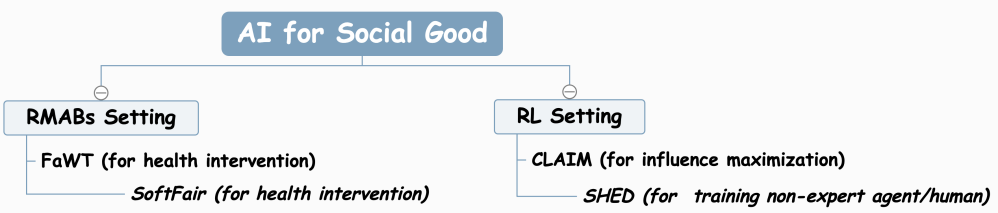


Figure 1.2: The relationship between my works under the AI for Social Good with limited resources initiative. The works are categorized into two main areas: RMABs Setting and RL Setting. Within the RMABs Setting, two parallel works focus on health intervention: FaWT and SoftFair. FaWT considers deterministic fairness constraints while SoftFair deals with probabilistic fairness constraints. The RL Setting includes two parallel works: CLAIM, which focuses on influence maximization, and SHED, which focuses on training non-expert agents or humans.

1.3 Contributions and Outline

The contributions and the outline of this dissertation are as follows:

- Chapter 2 (Restless Multi-Armed Bandits with Fairness Constraints): This chapter covers sequential decision-learning in the Restless Multi-Armed Bandits (RMAB) setting. We formally introduce the finite/infinite horizon RMAB model with a new objective of computing policies that balance the trade-off between maximizing cumulative rewards while giving a reasonable chance for each arm (proportional to their value) to get selected for intervention. We are interested in ensuring that RMAB decision-making is also fair to different arms. We first give two different types of fairness definitions and further develop scalable and efficient algorithms for balancing the trade-off between the goal of having resources uniformly distributed and maximizing cumulative rewards. Our methods leverage recent advances in RAMB research to handle proposed fairness constraints and we provide theoretical performance guarantees for our proposed methods. We also demonstrate the utility of our approaches on simulated benchmarks and show that our proposed fairness objectives can be handled without a significant sacrifice on the performance quality.

- Chapter 3 (Influence Maximization in Unknown Social Networks): We discuss the reinforcement learning setting where an agent interacts with the environment to find an optimal sequence of actions to perform a specific task. We introduce the problem of influence maximization in unknown social networks, which is widely used in applications such as viral marketing [43], rumor control [13], HIV prevention [105, 109], etc. Unlike online social networks, real-world networks are not completely known, and collecting information about the network is costly as it involves surveying multiple people. In this chapter, we focus on the problem of network discovery for influence maximization. The existing work in this direction proposes a reinforcement learning framework. As in real-world settings, the environment interactions are costly, the approach can be improved by reducing the costly environment interactions. We develop a curriculum-based algorithm to improve the sample efficiency of RL methods. Additionally, we conduct experiments on real-world datasets and show that our approach can outperform the existing approaches.

- Chapter 4 (Training Robust Agent with Limited Resources): There has been rapid progress in the deployment of RL systems in the real world. This involves automatically generating a curriculum of training environments, enabling agents trained in these environments to develop general capabilities, i.e., achieving good zero-shot transfer performance. However, existing environment generation approaches focus primarily on the random generation of environments for open-ended agent training. This is impractical in scenarios with limited resources, such as the constraints on the number of generated environments. In this chapter, we introduce a hierarchical MDP framework for environment design under resource constraints. Our proposed method can significantly reduce the resource-intensive interactions between agents and environments and empirical

experiments across various domains demonstrate the effectiveness of our approach. We hope our research can lead to more principled, robust, and widely accepted systems that can be used to assist in training non-expert humans. Figure

- Chapter 5 (Conclusion and Future Work): Finally, we summarize our works and describe future research directions.

Chapter 2

Restless Multi-Armed Bandits

Restless multi-armed bandits (RMAB) is a popular framework for optimizing performance with limited resources under uncertainty. It is an extremely useful model for monitoring beneficiaries (arms) and executing timely interventions using health workers (limited resources) to ensure optimal benefit in public health settings. For instance, RMAB has been used to track patients' health and monitor their adherence in tuberculosis settings, ensure pregnant mothers listen to automated calls about good pregnancy practices, etc. Due to the limited resources, typically certain individuals, communities, or regions are starved of interventions, which can potentially have a significant negative impact on the individual/community in the long term. For example, in the context of public health settings, this would ensure that different people and/or communities are fairly represented while making public health intervention decisions. To that end, we first define two different types of fairness objectives. Then we provide scalable approaches to ensure long-term optimality while satisfying the proposed fairness constraints in RMAB. We also provide theoretical properties and show our proposed methods are asymptotically optimal. Finally, we demonstrate the utility of our approaches on simulated benchmarks and show that the our fairness objectives can be handled without a significant sacrifice on the optimal value. The two works in this Chapter are presented in:

- Li, Dexun, and Pradeep Varakantham. “Avoiding Starvation of Arms in Restless Multi-Armed Bandits.” Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems. 2023.
- Li, Dexun, and Pradeep Varakantham. “Efficient resource allocation with fairness constraints in restless multi-armed bandits.” Uncertainty in Artificial Intelligence. PMLR, 2022.

2.1 Introduction

Picking the right time and manner of limited interventions is a problem of great practical importance in tuberculosis [56], maternal and child care [10, 58], anti-poaching operations [74], cancer detection [46], and many others. All these problems are characterized by multiple arms (i.e., patients, pregnant mothers, regions of a forest) whose state evolves in an uncertain manner (e.g., medication usage in the case of tuberculosis, engagement patterns of mothers on calls related to good practices in pregnancy) and threads moving to “bad” states have to be steered to “good” outcomes through interventions. The key challenge is that the number of interventions is limited due to a limited set of resources (e.g., public health workers, patrol officers in anti-poaching operations). Restless Multi-Armed Bandits (RMAB), a generalization of Multi-Armed Bandits (MAB) that allows non-active bandits to also undergo the Markovian state transition, has become an ideal model to represent the aforementioned problems of interest as it models uncertainty in arm transitions (to capture uncertain state evolution), actions (to represent interventions) and budget constraint (to represent limited resources).

Existing work [56, 10, 57] has focused on developing theoretical insights and practically efficient methods to solve RMAB. At each decision epoch, RMAB methods identify arms that provide the biggest improvement with an intervention.

Such an approach though technically optimal can result in certain arms (or type of arms) getting starved for interventions.

In the case of interventions with regards to public health, RMAB algorithms focus interventions on the top beneficiaries who will improve the objective (public health outcomes) the most. This can result in certain beneficiaries never talking to public health workers and thereby moving to bad states (and potentially also impacting other beneficiaries in the same community) from where improvements can be minor even with intervention and hence never getting picked by RMAB algorithms. As shown in Fig. 2.1, when using the Threshold Whittle index approach proposed by Mate et al. [56], the arm activation probability is lopsided, with 30% of arms getting activated more than 50 times and 50% of the arms are never activated. Such starvation of interventions can result in arms moving to a bad state from where interventions cannot provide big improvements and therefore there is further starvation of interventions for those arms. Such starvation can happen to entire regions or communities, resulting in lack of fair support for beneficiaries in those regions/communities. To avoid such cycles between bad outcomes, there is a need for RMAB algorithms to consider fairness in addition to maximizing expected reward when picking arms. Risk sensitive RMAB [58] considers an objective that targets to reduce such starvation, however, they *do not guarantee* that arms (or types of arms) are picked a minimum number of times.

Recent work in Multi-Armed Bandits (MAB) has presented different notions of fairness. For example, Li et al. [50] study a Combinatorial Sleeping MAB model with Fairness constraints, called CSMAB-F. The fairness constraints ensure a minimum selection fraction for each arm. Patil et al. [71] introduce similar fairness constraints in the stochastic multi-armed bandit problem, where they use a pre-specified vector to denote the guaranteed number of pulls. Joseph et al. [37] define fairness as saying that a worse arm should not be picked compared to a better arm, despite the uncertainty on payoffs. Chen et al. [17]

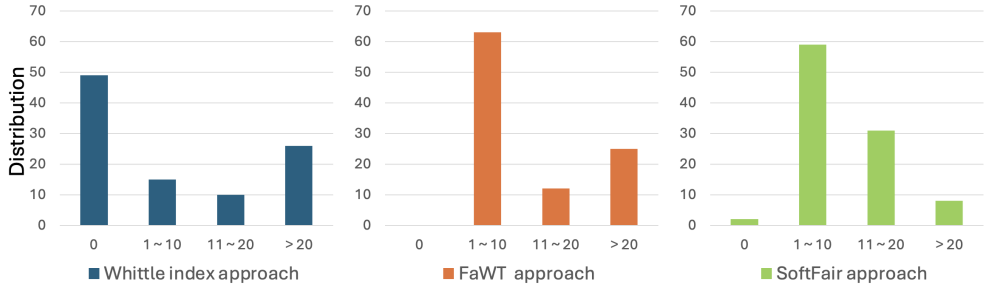


Figure 2.1: The x-axis is the number of times activated, and the y-axis is the percentage of each frequency range. We consider the RMAB with $k = 10$, $N = 100$, $T = 1000$. Left: the result of using the Whittle index algorithm without considering fairness constraints. Middel: the result of *FaWT* with considering deterministic fairness constraints in Section 2.3, and we set $L = 50$, $\eta = 2$. Right: the result of *SoftFair* when considering probabilistic fairness constraints in Section 2.4. As can be noted, without fairness constraints in place, almost 50% of the arms never get activated.

define the fairness constraint as a minimum rate that is required when allocating a task or resource to a user. The above fairness definitions are relevant and we generalize from these to propose two different types of fairness notions for RMAB. Unfortunately, approaches developed for fair MAB cannot be utilized for RMAB, due to uncertain state transitions with passive actions as well.

Contributions: To the best of our knowledge, we are the first one to consider fairness constraints in RMAB. Here are the key contributions:

- We propose two different types of fairness constraints wherein the first one is defined as for any arm (or more generally, for a type of arm), we require that the number of decision epochs since the arm (or the type of arm) was activated last time is upper bounded. This will ensure that every arm (or type of arm) gets activated a minimum number of times, thus generalizing on the fairness notions in MAB described earlier. The second type is the soft fairness objective which entails an algorithm never probabilistically favors one arm over another if the long-term cumulative reward of choosing the latter arm is higher.

- For the first deterministic fairness constraint, we provide a modification to

the Whittle index algorithm that is scalable and optimal while being able to handle both finite and infinite horizon cases. We also provide a model-free learning method to solve the problem when the transition probabilities are not known beforehand.

- For the second probabilistic fairness constraint, we introduce a practically relevant algorithm called *SoftFair*, *SoftFair* enforces the probabilistic (also called "soft") fairness constraint and thereby avoids starvation of interventions for arms. Unlike the well-known Whittle index algorithm that can only solve the infinite horizon setting, *SoftFair* can also easily handle finite horizon RMAB. *SoftFair* provides a trade-off between optimal performance and avoiding intervention starvation for arms. This trade-off is highlighted by the performance bounds and theoretical properties of the *SoftFair* algorithm.

- Experiment results on the generated dataset show that our approaches can be competitive with other policies in terms of expected reward, while significantly reducing the starvation of interventions for arms.

2.2 Related Work

We focus on two threads of relevant research, the first category is related to approaches for solving RMAB, and the second category is related to fairness definitions and related approaches in decision making.

2.2.1 Whittle Index Policy

As one of the most well-studied generalisations of the Multi-Armed Bandit (MAB), RMAB is increasingly used for decision learning problems ranging from wireless broadcast [75, 83], job allocation [35], cancer detection [46], wildlife protection [74], recommender systems [60], and health intervention [44, 10].

Whittle [103] considered the Lagrangian relaxation of the RMAB in which arm selection constraint (number of arms selected $= k$) is enforced on average over the horizon. This policy, referred as the Whittle index policy is asymptotically optimal [102]. Liu and Zhao [53] investigate the application of RMAB in dynamic multichannel access, establish indexability and obtain Whittle index in closed form for both discounted and average reward criteria. In [74], the authors formulate the wildlife protection problem as a RMAB model and present an algorithm that is based on binary search to find Whittle index policy. Mate et al. [56] build a fast algorithm for computing the Whittle index, which provides an order-of-magnitude speedup compared to Qian et al. [74]. Biswas et al. [10] develop a model-free learning method based on Q-learning mechanism and show that it converge to the optimal solution asymptotically. Online RMAB has also raised some attentions in recent years, Wang et al. [98] present a learning policy to construct offline instances in guiding action selection. Xiong et al. [108] propose a generative model based reinforcement learning augmented algorithm toward an index policy.

2.2.2 Fairness in Decision-making

Another line of work that is closely related to ours is the growing body of literature on ensuring fairness in decision making [34, 18], in particular in the domain of resource allocation [50, 17]. For example, ensuring resources are fairly distributed among the arms is an important design concern in wireless communication systems [25]. In the case of beneficiaries, an arm/patient might consider action/participation fair when the participation of a certain patient (i.e., due to receiving an active action) resulted in a greater increase in expected time spent in an adherent state compared to non-participation (i.e., the passive action on the arm/patient) [42]. One widely used fairness notion in MAB literature is to ensure that there is a minimum rate of arm activation for each user (arm) over

time [50, 72]. Joseph et al. [37] introduce the study of fairness in MAB problems, where their fairness notion is defined as not giving preference to a worse arm over a better one. The quality of an action is the expected immediate reward for selecting an action from the current state. However, this notion of fairness can lead to policies favoring short-term rewards and ignoring long-term rewards. Jabbari et al. [34] therefore adapt the fairness notion by defining the quality of an action as its potential long-term reward and generalize it to a reinforcement learning setting.

We generalize these fairness concepts to the RMAB setting and propose deterministic and probabilistic fairness constraints. The deterministic fairness constraint is defined as that for any arm, we require that there is an upper bound for the length of the decision epochs since that arm was last activated. This will ensure that each arm has a minimum number of selections. The second type is a probabilistic fairness constraint, where the algorithm will never probabilistically favor another arm if the long-term cumulative reward of the latter arm is higher.

Discussion of Fairness Choice. It is natural to ask what makes the proposed notion of fairness in this chapter the right one? Our proposed fairness constraint is driven by the flaw of SOTA that a substantial number of arms are never selected, such starvation of intervention results in a huge demand for fairness requirement in the real world. Furthermore, while our proposed algorithm can still be used, our notion of fairness can also be extended to fairness on the group/type of arms (i.e., check if the group fairness requirement is violated, and if so, select the arm with the highest index value in that group/type). One of the most widely used fairness notions is to define a minimum rate that is required when allocating resources to users, and our deterministic fairness constraint can be viewed as a variant of this [17, 50, 71]. Another form of fairness constraint is to require that the algorithm never prefers a worse action over a better one based on the expected immediate reward [37]. This can be seen as a variant of

the Myopic algorithm, while our probabilistic fairness constraint ensures that the probability is proportional to the long-term cumulative reward. There might be many other measures of fairness and it may be impossible to satisfy multiple types of fairness simultaneously (COMPAS Case Study). However, our proposed fairness constraints are the most appropriate forms in the real world. Namely, in the field of medical interventions, we can meet the requirement that everyone will receive or have the probability to receive medical treatment without sacrificing a significant overall performance, while SOTA/Myopic will only favor certain beneficiaries.

In the following sections, we first discuss the deterministic fairness constraint in RMAB, and then move to the probabilistic fairness constraint.

2.3 RMAB with deterministic fairness constraint

2.3.1 Problem Description

In this section, we formally introduce the RMAB problem. There are N independent arms, each of which evolves according to an associated Markov Decision Process (MDP). An MDP is characterized by a tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\}$, where $\mathcal{S}$ represents the state space, $\mathcal{A}$ represents the action space, $\mathcal{P}$ represents the transition function, and r is the state-dependent reward function. Specifically, each arm has a binary-state space: 1 ("good") and 0 ("bad"), with action-dependent transition matrix $\mathcal{P}$ that is potentially different for each arm. Let $a_t^i \in \{0, 1\}$ denote the action taken at time step t for arm i, and $a_t^i = 1(a_t^i = 0)$ is called active (passive) and indicates the arm i being pulled (not pulled). Due to limited resources, at each decision epoch, the decision-maker can activate (or intervene on) at most k out of N arms and receive reward accrued from all arms determined by their states. $\sum_{i=1}^{N} a_t^i = k$ describes this limited resource constraint. Figure 2.2 provides an example of an arm in RMAB.

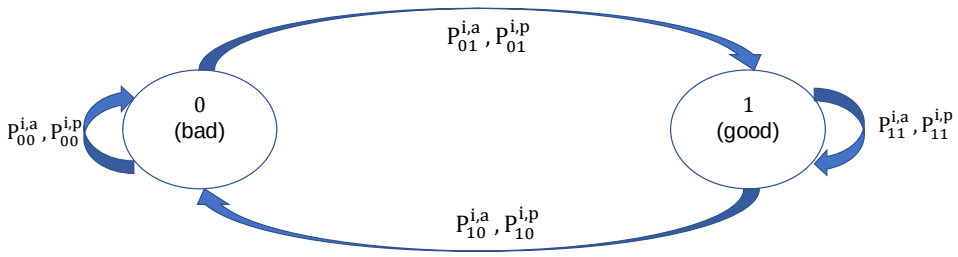


Figure 2.2: a and p denote the active and passive actions on arm i respectively. $P_{s,s'}^{i,a}$ and $P_{s,s'}^{i,p}$ are the transition probabilities from state s to state s' under action a and p respectively for arm i.

The state of arm i evolves according to the transition matrix $P_{s,s'}^{i,a}$ for the active action and $P_{s,s'}^{i,p}$ for the passive action [1]. We follow the setting in Mate et al. [56], when the arm i is activated, the latent state of arm i will be fully observed by the decision-maker. The states of passive arms are unobserved by the decision-maker.

When considering such a partially observable problem, it is sufficient to let the MDP state be the belief state: the probability that the arm is in the "good" state. We need to keep track of the belief state on the current state of the unobserved arm. This can be derived from the decision-maker's partial information which is encompassed by the last observed state and the number of decision time steps since the last activation of the arm. Let $\omega_s^i(u)$ denote the belief state, i.e., the probability that the state of arm i is 1 when it was activated u time steps ago with the observed state s. The belief state in the next time step can be obtained by solving the following recursive equations:

$$\omega_s^i(u+1) = \begin{cases} \omega_s^i(u)P_{1,1}^{i,p} + (1-\omega_s^i(u))P_{0,1}^{i,p} & \text{passive} \\ P_{s',1}^{i,a} & \text{active} \end{cases} \tag{2.1}$$

Where s' is the new state observed for arm i when the active action was taken. The belief state can be calculated in closed form with the given transition probabilities. We let $\omega = \omega_s^i(u+1)$ for ease of explanation when there is no ambiguity.

A policy π maps the belief state vector $\Omega_t = \{\omega_t^1, \cdots, \omega_t^N\}$ at each time

[1] Note that we use $P_{s,s'}^{i,a}$ and $P_{s,s'}^{i,1}$ ($P_{s,s'}^{i,p}$ and $P_{s,s'}^{i,0}$) interchangeably in this thesis

step t for all arms to the action vector, $a_t = \{0,1\}^N$. Here ω_t^i is the belief state for arm i at time step t. We want to design an optimal policy to maximize the cumulative long-term reward over all the arms. One widely used performance measure is the expected discounted reward over the horizon T :

$$\mathbb{E}_\pi[\sum_{t=1}^{T} \beta^{t-1} R_t(\Omega_t, \pi(\Omega_t))|\Omega_0]$$

Here $R_t(\Omega_t, \pi(\Omega_t))$ is the reward obtained in slot t under action $a_t = \pi(\Omega_t)$ determined by policy π, β is the discount factor. As we discussed in the introduction, in addition to maximizing the cumulative reward, ensuring fairness among the arms is also a key design concern for many real-world applications. In order to model the fairness requirement, we introduce constraints that ensure that any arm (or kind of arms) is activated at least η times during any decision interval of length L. The overall optimization problem corresponding to the problem at hand is thus given by:

$$\begin{aligned} &\underset{\pi}{\text{maximize}}\ \mathbb{E}_\pi[\sum_{t=1}^{T} \beta^{t-1} R_t(\Omega_t, \pi(\Omega_t))|\Omega_0] \\ &\text{subject to } \sum_{i}^{N} a_t^i = k, \forall t \in \{1,\ldots,T\} \\ &\qquad \sum_{t=u}^{u+L} a_t^i \geq \eta \quad \forall u \in \{1,\ldots,T-L\}, \forall i \in \{1,\ldots,N\}. \end{aligned} \tag{2.2}$$

η is the minimum number of times an arm should be activated in a decision period of length L. The strength of fairness constraints is thus governed by the combination of L and η. Obviously, this requires $k \times L > N \times (\eta - 1)$ as the fairness constraint should meet the resource constraint. This fairness problem can be formulated at the level of regions/communities by also summing over all the arms, i in a region in the second constraint, i.e.,

$$\sum_{i \in r} \sum_{t=u}^{u+L} a_t^i \geq \eta$$

Our approaches with a simple modification are also applicable to this fairness constraint at the level of regions/communities.

2.3.2 Background: Whittle Index

In this section, we describe the Whittle Index algorithm [103] to solve RMAB. This algorithm at every time step, computes index values (Whittle Index values) for every arm and then activates the arms that have the top "k" index values. Whittle index quantifies how appealing it is to activate a certain arm. This algorithm provides optimal solutions if the underlying RMAB satisfies the indexability property, defined in Definition 1.

Formally[2], the Whittle index of an arm in a belief state ω (i.e., the probability of good state 1) is the minimum subsidy λ such that it is optimal to make the arm passive in that belief state. Let $V_{\lambda,T}(\omega)$ denote the value function for the belief state ω over a horizon T. Then it could be written as

$$V_{\lambda,T}(\omega) = \max\{V_{\lambda,T}(\omega; a = 0), V_{\lambda,T}(\omega; a = 1)\}, \tag{2.3}$$

where $V_{\lambda,T}(\omega; a = 0)$ and $V_{\lambda,T}(\omega; a = 1)$ denote the value function when taking passive and active actions respectively at the first decision epoch followed by optimal policy in the future time steps. Because the expected immediate reward is ω and subsidy for a passive action is λ, we have the value function for passive action as:

$$V_{\lambda,T}(\omega, a = 0) = \lambda + \omega + \beta V_{\lambda,T-1}(\tau^1(\omega)), \tag{2.4}$$

where $\tau^1(\omega)$ is the 1-step belief state update of ω when the passive arm is unobserved for another 1 consecutive slot (see the update rule in Eq. 2.1). Note that ω is also the expected reward associated with that belief state. For an active action, the immediate reward is ω and there is no subsidy. However, the actual

[2]Since we will only be talking about one arm at a time step, we will abuse the notation by not indexing belief, action and value function with arm id or time index.

state will be known and then evolve according to the transition matrix for the next step:

$$V_{\lambda,T}(\omega, a=1) = \omega + \beta(\omega V_{\lambda,T-1}(P^a_{1,1}) + (1-\omega)V_{\lambda,T-1}(P^a_{0,1})). \tag{2.5}$$

Definition 1 *An arm is indexable if the passive set under the subsidy λ given as $\mathcal{P}_\lambda = \{\omega : V_{\lambda,T}(\omega, a=0) \geq V_{\lambda,T}(\omega, a=1)\}$ monotonically increases from $\emptyset$ to the entire state space as λ increases from $-\infty$ to ∞. The RMAB is indexable if every arm is indexable.*

Intuitively, this means that if an arm takes passive action with subsidy λ, it will also take passive action if $\lambda' > \lambda$. Given the *indexability*, $W_T(\omega)$ is the least subsidy, λ that makes it equally desirable to take active and passive actions.

$$W_T(\omega) = \inf_\lambda \{\lambda : V_{\lambda,T}(\omega; a=1) \leq V_{\lambda,T}(\omega; a=0)\} \tag{2.6}$$

Definition 2 *A policy is a threshold policy if there exists a threshold λ_{th} such that the action is passive $a=0$ if $\lambda > \lambda_{th}$ and $a=1$ otherwise.*

Existing efficient methods for solving RMABs derive these threshold policies. We here provide a detailed description of the Fast Whittle Index Computation algorithm introduced in Mate et al. [56], since our method to solve the deterministic fairness constraint is based on it. Mate et al. [56] derived a closed form for computing the Whittle index for both average reward and discounted reward criterion, where the objective could also be written as $\bar{R}^\pi_\lambda = \mathbb{E}\sum_\omega f^\pi(\omega) R_{a^\pi}(\omega)$, where $f^\pi(\omega)$ is defined as the fraction of time spent in each belief state ω induced by policy π and $f^\pi(\omega) \in [0,1]$. Their proposed Whittle index computation algorithm can achieve a 3-order-of-magnitude speedup compared to Qian et al. [74]. In the two-states setting ($s \in \{0,1\}$), they use a tuple $(B_0^{\omega_{th}}, B_1^{\omega_{th}})$ to denote the belief threshold, where $\omega_{th} \in [0,1]$, and $B_0^{\omega_{th}}, B_1^{\omega_{th}} \in 1, \ldots, L$ are the index of the first belief state in each chain where it is optimal to act (i.e., the belief is less

than or equal to ω_{th}). The length is at most L long due to our fairness constraints. This is defined as the forward threshold policy, and Mate et al. [56] used the Markov chain structure to derive the occupancy frequencies for each belief state $\omega_s(t)$, which is as follows,

$$f^{(B_0^{\omega_{th}}, B_1^{\omega_{th}})}(\omega_s(t)) = \begin{cases} a & \text{if } s = 0, t \leq B_0 \\ b & \text{if } s = 1, t \leq B_1 \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

$$a = \left(\frac{B_1\omega_0(B_0)}{1-\omega_1(B_1)} + B_0\right)^{-1}, b = \left(\frac{B_1\omega_0(B_0)}{1-\omega_1(B_1)} + B_0\right)^{-1} \frac{\omega_0(B_0)}{1-\omega_1(B_1)} \tag{2.8}$$

These occupancy frequencies do not depend on the subsidy λ. For the forward threshold policy $(B_0^{\omega_{th}}, B_1^{\omega_{th}})$, they use the $R_\lambda^{B_0^{\omega_{th}}, B_1^{\omega_{th}}}$ to denote the average reward, then can decompose the average reward into the contribution of the state reward and the subsidy λ

$$\begin{aligned} R_\lambda^{(B_0^{\omega_{th}}, B_1^{\omega_{th}})} &= \sum_{\omega \in \mathscr{B}} \omega f^{(B_0^{\omega_{th}}, B_1^{\omega_{th}})}(\omega) \\ &\quad + \omega \left(1 - f^{(B_0^{\omega_{th}}, B_1^{\omega_{th}})}(\omega_1(B_1)) - f^{(B_0^{\omega_{th}}, B_1^{\omega_{th}})}(\omega_0(B_0))\right) \end{aligned} \tag{2.9}$$

Given the definition of the Whittle index λ, this could be interpreted to two corresponding threshold policies being equally optimal. More specifically, for a belief state $\omega_0(B_0)$, the two adjacent threshold polices $\{(B_0^{\omega_{th}}, B_1^{\omega_{th}}), (B_0^{\omega_{th}} + 1, B_1^{\omega_{th}})\}$ would be optimal to be active and passive respectively. recall that the Whittle index is the smallest λ for which the active and the passive actions are both optimal. Thus the subsidy which makes the average reward of those two adjacent polices equal in value must be the Whittle index for the belief state $\omega_0(B_0)$. Formally, this could be calculated through $R_{(\lambda}^{B_0^{\omega_{th}}, B_1^{\omega_{th}})} = R_{(\lambda}^{B_0^{\omega_{th}}, B_1^{\omega_{th}}+1)}$. Similarly, we can obtain the Whittle index for the belief state $\omega_1(B_1)$ through $R_\lambda^{(B_0^{\omega_{th}}, B_1^{\omega_{th}})} = R_\lambda^{(B_0^{\omega_{th}}, B_1^{\omega_{th}}+1)}$. These computations are repeated for every belief states to find the minimum subsidy value while $B_s^{\omega_{th}} \leq L$. The main idea of their approach is shown in Fig 2.3.

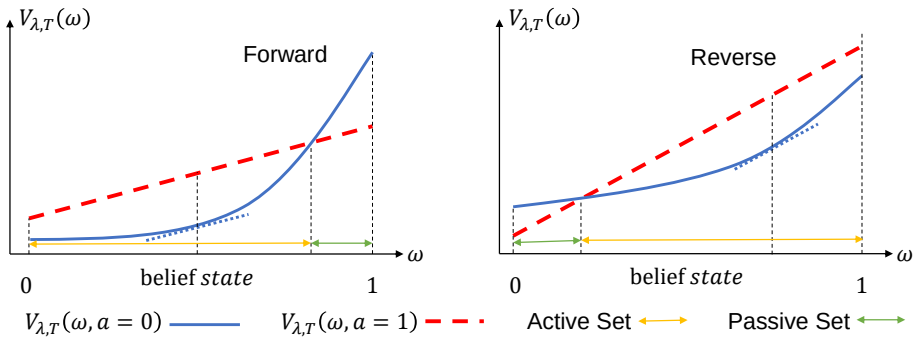


Figure 2.3: The forward and reverse policy

2.3.3 Algorithm for Different Settings

The key advantage of a Whittle index-based approach is scalability without sacrificing solution quality. In this section, we provide Whittle index-based approaches to handle fairness constraints under known transition models in the different settings. We specifically consider partially observable settings.

When we need to consider the partial observability of the state of the RMAB problem, it is sufficient to let the MDP state be the belief state: the probability that the arm is in the "good" state [38]. As a result, the partially observable RMAB has a large number of belief states [56].

Recall that the definition of the Whittle index $W_T(\omega)$ of belief state ω is the smallest λ s.t. it is optimal to make the arm passive in the current state. We can compute the Whittle index value for each arm, and then rank the index value of all N arms and select top k arms at each time step to activate. With fairness constraints, the change to the approach is minimal and intuitive. ***The optimal policy is to choose the arms with the top "k" index values until a fairness constraint is violated for an arm. In that time step, we replace the last arm in top-k with the arm for which fairness constraint is violated.*** We show that this simple change works across the board for the infinite and finite horizon, fully and partially observable settings. We provide the detailed algorithm in Algorithm 1 and also provide sufficient conditions under which the Algorithm 1 is optimal.

We give a visualization of our proposed Whittle index-based approach to

solve the fairness constraint in Figure 2.4. The belief state MDP works as follows: initially, after an action, the state $s \in \{0, 1\}$ of the selected arm is observed. Then the belief state changes to $P^a_{s,1}$ one slot later, which is represented as the blue node at the head of the chain. Subsequent passive actions cause the belief state to evolve according to the initial observation in the same chain. Then if the arm is activated again under the proposed algorithm, it will transit to the head of one of the chains with the probability according to its belief state as shown in the black arrow. If the arm's fairness constraint is not met, i.e., it has not been chosen in the last $L-1$ time slots, it will be activated at the time slot L, and go to the head of one of the chains (as shown by the red dashed arrow).

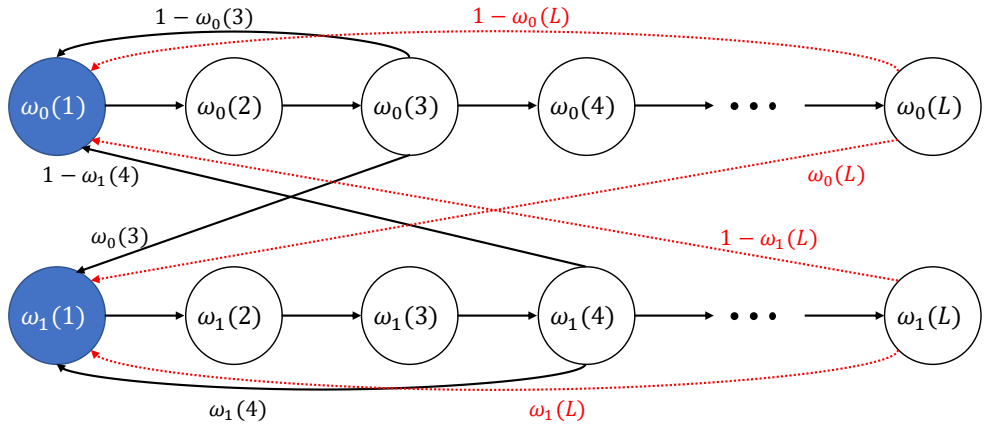


Figure 2.4: Visualization of Whittle index approach with fairness constraints.

Infinite time horizon

We first focus on the infinite time horizon and provide the expression for λ. $V_{\lambda,\infty}(\omega)$ denotes the value that can be accrued from a single-armed bandit process with subsidy λ over an infinite time horizon ($T \to \infty$) if the belief state is ω. Therefore, we have:

$$V_{\lambda,\infty}(\omega) = \max \begin{cases} \lambda + \omega + \beta V_{\lambda,\infty}(\tau^1(\omega)) & \text{passive} \\ \omega + \beta \left(\omega V_{\lambda,\infty}(P^a_{1,1}) + (1-\omega) V_{\lambda,\infty}(P^a_{0,1})\right) & \text{active} \end{cases} \tag{2.10}$$

For any belief state ω, the u-steps belief update $\tau^u(\omega)$ will converge to ω^* as

$u \to \infty$, where $\omega^* = \frac{P^p_{0,1}}{1+P^p_{0,1}-P^p_{1,1}}$. It should be noted that this convergence can happen in two ways depending on the state transition patterns:

- Case 1: Positively correlated channel $(P^p_{1,1} \geq P^p_{0,1})$.

  The belief update process is shown in Figure 2.5. We can see that for the positively correlated case, they have a monotonous belief update process.

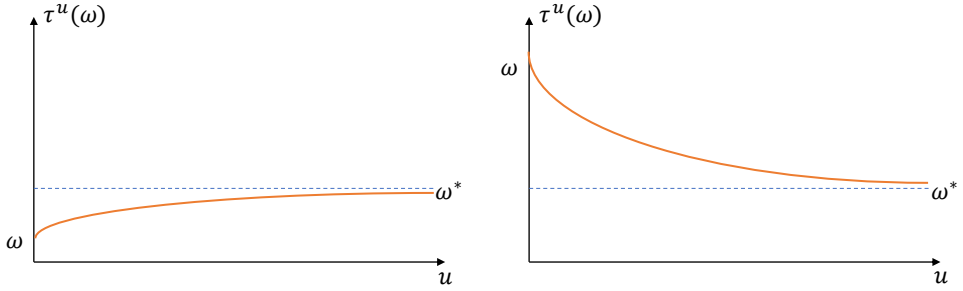


Figure 2.5: The u-step belief update of an unobserved arm $(P^p_{1,1} \geq P^p_{0,1})$

  We first consider the *non-increasing belief process* as indicated in the right graph. Formally, for $\forall u \in \mathbb{N}^+$, we have $\omega(u) \geq \omega(u+1)$ if the initial belief state ω is above the convergence value. Similarly, for the *increasing belief process* shown in the left graph, we have the initial belief state $\omega < \omega^*$.

- Case 2: Negatively correlated channel $(P^p_{1,1} < P^p_{0,1})$.

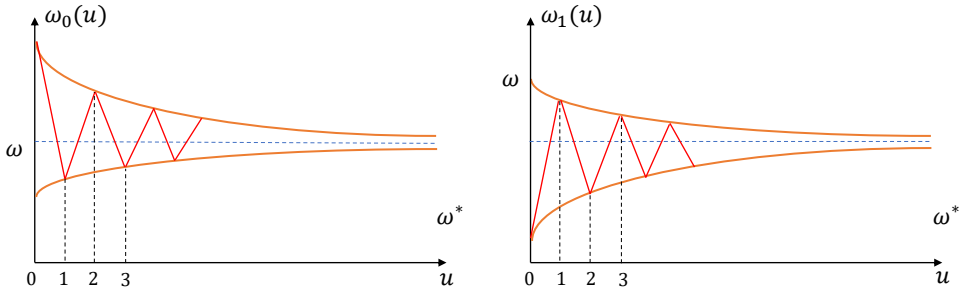


Figure 2.6: The u-step belief update of an unobserved arm $(P^p_{1,1} < P^p_{0,1})$

  The belief state converges to ω^* from the opposite direction as shown in Figure 2.6. This case has similar properties and is less common in the real world because it is more likely to remain in a good state than to move from a bad state to a good state. Therefore, we omit the lengthy discussion.

The belief state transition patterns are of particular importance because in proving optimality of Algorithm 1, the belief evolution pattern for the arm (whose fairness constraint will be violated) plays a crucial role.

Algorithm 1: Fair Whittle Thresholding (FaWT)

Input: Transition matrix $\mathcal{P}$, fairness constraint, η and L, set of belief states $\{\omega^1, \ldots, \omega^N\}$, k

for *time step* t *from 1 to* T **do**
  for *each arm i in* 1 *to* N **do**
    Compute the corresponding Whittle index $TW(\omega^i)$ under the infinite horizon using the *Forward and Reverse Threshold* policy;
    if *the activation frequency* η *for arm* i *will not be satisfied at the end of the period of length* L **then**
      Add arm i to the action set ϕ;
      $k = k - 1$;
  Add arms with top k highest $TW(\cdot)$ (for infinite horizon case) or $W_T(\cdot)$ (for finite horizon case) values to the action set ϕ Decrease the residual time horizon by $T = T - 1$;

Output: Action set ϕ

Theorem 1 *For infinite time horizon ($T \to \infty$) RMAB with Fairness Constraints governed by parameters η and L, Algorithm 1 (i.e., activating arm i at the end of the time period when its fairness constraint is violated) is optimal:*

*1. For $\omega^i \leq \omega^*$ (*increasing belief process*), if*

$$(P_{1,1}^{i,p} - P_{0,1}^{i,p}) \left(1 + \frac{\beta \Delta_3}{1-\beta}\right) \left(1 - \beta(P_{1,1}^{i,a} - P_{0,1}^{i,a})\right) \leq (P_{1,1}^{i,a} - P_{0,1}^{i,a}) \quad (2.11)$$

$\Delta_3 = \min\{(P_{1,1}^{i,p} - P_{0,1}^{i,p}), (P_{1,1}^{i,a} - P_{0,1}^{i,a})\}$.

*2. For $\omega^i \geq \omega^*$ (*non-increasing belief process*), if:*

$$(P_{1,1}^{i,p} - P_{0,1}^{i,p})(1-\beta)\Delta_1 \geq (P_{1,1}^{i,a} - P_{0,1}^{i,a}) \left(1 - \beta(P_{1,1}^{i,a} - P_{0,1}^{i,a})\right) \quad (2.12)$$

$\Delta_1 = \min\{1, 1 + \beta(P_{1,1}^{i,p} - P_{0,1}^{i,p}) - \beta(P_{1,1}^{i,a} - P_{0,1}^{i,a})\}$

Proof Sketch. Consider an arm i that has not been activated for $L-1$ time slots. In such a case, Algorithm 1 will select arm i to activate in the next time step $t = L$. Define the intervention effect of activating arm i as

$$V_{\lambda,\infty}(\omega, a=1) - V_{\lambda,\infty}(\omega, a=0)$$

Following standard practice and for notational convenience, we do not index the intervention effect and value functions with i. Due to independent evolution of arms, moving active action of arm i does not result in a greater value function for other arms according to the Whittle index algorithm, thus it suffices to only consider arm i. Here is the proof flow:

(1) Algorithm 1 optimality requires that the intervention effect at time step $t = L-1$ is smaller than intervention effect at $t = L$. Optimality can be established by requiring the partial derivative of the intervention effect w.r.t. time step t is greater than 0.

(2) However, computing this partial derivative $\frac{\partial(V_{\lambda,\infty}(\omega,a=1)-V_{\lambda,\infty}(\omega,a=0))}{\partial t}$ is difficult because value function expression is complex. We use chain rule to get:

$$\frac{\partial(V_{\lambda,\infty}(\omega, a=1) - V_{\lambda,\infty}(\omega, a=0))}{\partial \omega} \cdot \frac{\partial(\omega)}{\partial(t)}$$

(3) The sign of second term, $\frac{\partial \omega}{\partial t}$ is based on the belief state transition pattern described before this theorem. We then need to consider the sign of the first term, $\frac{\partial(V_{\lambda,\infty}(\omega,a=1)-V_{\lambda,\infty}(\omega,a=0))}{\partial \omega}$.

(4) We can compute this by deriving the bound on $V_{\lambda,\infty}(\omega_1) - V_{\lambda,\infty}(\omega_2), \forall \omega_1, \omega_2$ as well as bounds on $\frac{\partial V_{\lambda,\infty}(\omega)}{\partial \omega}$. See the appendix 6.1 for detailed proof. □

Finite time horizon

Mechanisms developed to handle fairness in infinite-horizon settings can also be applied to finite-horizon settings. Because the algorithm remains unchanged:

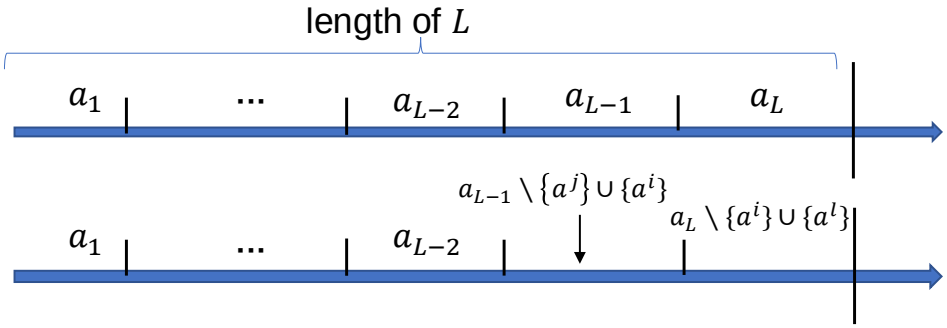


Figure 2.7: The action vector for RMAB is a_t at time step t. Then we move the action a^i that satisfies fairness constraint to earlier slot and replace k-th ranked action a^j. Action a^l is then added according to the index value at the end.

The strategy is to select the arm with the top "k" index value until the arm's fairness constraint is violated. Therefore we omit it in this dissertation.

Uncertainty in Transition Matrix

In most real-world applications [10], there may not be adequate information about all the state transitions. In such cases, we don't know how likely a transition is and thus, we won't be able to use the Whittle index approach directly. We provide a mechanism to apply the Thompson sampling-based learning mechanism for solving RMAB problems without prior knowledge and where it is feasible to get learning experiences. Thompson sampling [92] is an algorithm for online decision problems, and can be applied in MDP [28] as well as Partially Observable MDP [59]. In Thompson sampling, we initially assume that the arm has a prior Beta distribution in the transition probability according to the prior knowledge (if there is no prior knowledge available, we assume a prior $Beta(1,1)$ as this is the uniform distribution on $(0,1)$). We choose Beta distribution because it is a convenient and useful prior option for Bernoulli rewards [1].

In our algorithm, referred to as FaWT-U and provided in Algorithm 2, at each time step, we sample the posterior distribution over the parameters, and then use the Whittle index algorithm to select the arm with the highest index value to play

if the fairness constraint is not violated. We can utilize our observations to update our posterior distribution, because playing the selected arms will reveal their current state. Then, the algorithm takes samples from the posterior distribution and repeats the procedure again.

Algorithm 2: Fair Whittle Thresholding with Uncertainty in transition matrix(FaWT-U)

Input: Posterior $Beta$ distribution over the transition matrix $\mathcal{P}$, fairness constraint, η and L, set of belief states $\{\omega^1, \ldots, \omega^N\}$, budget k

1 **for** *each arm i in* 1 *to* N **do**
2 Sample the transition probability parameters independently from posterior;
3 Compute Whittle indices based on the transition matrix and belief state;
4 **if** *the activation frequency* η *for arm* i *is not satisfied at the end of the period of length* L **then**
5 Add arm i to the action set ϕ;
6 $k = k - 1$;
7 Add the arms with top k index value into ϕ;
8 Play the selected arms and receive the observations;
9 Update the posterior distribution;

Output: Action set ϕ and updated posterior distribution over parameters

We employ the sampled transition probabilities and belief states $\{\omega^1, \ldots, \omega^N\}$, as well as the residual time horizon T as the input to the Whittle index computation (Line 3 in Algorithm 2).

Unknown Transition Matrix

We now tackle the second challenge, in which the transition matrix is completely unknown. In this case, we can take advantage of the model-free learning method to avoid directly using the whittling index policy.

Q-Learning is most commonly used to solve the sequential decision learning problem, which was first introduced by Watkins and Dayan [100] as an early breakthrough in reinforcement learning. It is widely studied for social good [63, 49], and it has also been extensively used in RMAB problems [27, 6, 10] to estimate the expected Q-value, $Q^*(s, a, l)$, of taking action $a \in \{0, 1\}$ after

$l \in \{1, \ldots, L\}$ time slots since last observation $s \in \{0, 1\}$. The off-policy TD control algorithm is defined as

$$Q^{t+1}(s_t, a_t, l_t) \leftarrow Q^t(s_t, a_t, l_t) + \alpha_t(s_t, a_t, l_t) \left[R_{t+1} + \gamma \max_a \left(Q^t(s_{t+1}, a, l_{t+1}) - Q^t(s_t, a_t, l_t) \right) \right] \tag{2.13}$$

Where γ is the discount rate, $\alpha_t(s_t, a_t, l_t) \in [0, 1]$ is the learning rate parameter, i.e., a small $\alpha_t(s_t, a_t, l_t)$ will result in a slow learning process and no update when $\alpha_t(s_t, a_t, l_t) = 0$. While a large $\alpha_t(s_t, a_t, l_t)$ may cause the estimated Q-value to rely heavily on the most recent return, when $\alpha_t(s_t, a_t, l_t) = 1$, the Q-value will always be the most recent return.

We now describe how to use the Whittle index-based Q-Learning mechanism to solve the RMAB problem with fairness constraints. We build on the work by Biswas et al. [10] for fully observable settings. In addition to considering fairness constraints, our model can be viewed as an extension to the partially observable setting. Due to fairness constraints, l can be a maximum of L time steps. Therefore, belief space is also limited. We are able to use the Q-Learning based approach to effectively compute the Whittle index value and this approach is summarized in Algorithm 3,

One typical form of $\alpha_t(s_t, a_t, l_t)$ could be $1/z(s_t, a_t, l_t)$, where $z(s_t, a_t, l_t) = \left(\sum_{u=0}^{t} \mathbb{I}\{s_u = s, a_u = a, l_u = l\} \right) + 1$ for each initial observed state $s \in \{0, 1\}$, action $a \in \{0, 1\}$ and time length since last activation $l \in \{1, \ldots, L\}$ at the time slot u from the beginning. With such mild form of $\alpha_t(s_t, a_t, l_t)$, we now are able to build the theoretical support for the Q-Learning based Whittle index approach.

Theorem 2 *Selecting the highest-ranking arms according to the* $Q_i^*(s, a = 1, l) - Q_i^*(s, a = 0, l)$ *till the budget constraint is met is equivalent to maximizing* $\left\{ \sum_{i=1}^{N} Q_i^*(s, a, l) \right\}$ *over all possible action set* $\{0, 1\}^N$ *such that* $\sum_{i=1}^{N} a_i = k$.

Proof Sketch. A proof based on work by [10] is given in Appendix 6.1.3. □

Theorem 3 *Stability and convergence: The proposed approach converges to the optimal with probability* 1 *under the following conditions:*

Algorithm 3: Fair Whittle Thresholding based Q-Learning(FaWT-Q)

Input: parameter ϵ and k, and $\alpha_t(s_t, a_t, l_t)$, initial observed state set $\{s\}^N$,

for *each arm i in* 1 *to* N **do**
  Initialize the $Q_i(s, a, l) \leftarrow 0$ for each state $s \in \{0, 1\}$, and each action $a \in \{0, 1\}$ and time length $l \in \{1, \ldots, L\}$;
  For each $s \in \{0, 1\}$ and $l \in \{1, \ldots, L\}$ initialize the Whittle index value set $\lambda_i(s, l) \leftarrow 0$;
for *t from* 1 *to* T **do**
  for *arm i in 1 to N* **do**
    if *the fairness constraint is violated* **then**
      Add arm i to the action set ϕ;
      $k = k - 1$;
  With prob ϵ add random k arms to ϕ and with prob $1 - \epsilon$ add arms with top k $\lambda_i(s, l)$ value ;
  Activate the selected arms and receive rewards and observations;
  for *each arm i in* 1 *to* N **do**
    Update the $Q_i^{t+1}(s, a, l)$ according to Eq. 2.13;
    if $i \in \phi$ **then**
      Set $l = 1$ and update s_i according to the received observation;
    else
      Set $l = l + 1$;
    Update the new Q-Learning based Whittle index by $\lambda_i^{t+1}(s, l) = Q_i(s, a = 1, l) - Q_i(s, a = 0, l)$

Output: Action set ϕ

1. The state space and action space are finite;

2. $\sum_{t=1}^{\infty} \alpha_t(s_t, a_t, l_t) = \infty \quad \sum_{t=1}^{\infty} \alpha_t^2(\omega_i(t)) < \infty$

Proof Sketch. The key to the convergence is contingent on a particular sequence of episodes observed in the real process [100]. Detailed proof is given in Appendix 6.1.4. □

2.3.4 Experiment

To the best of our knowledge, we are the first to explore fairness constraints in RMAB, hence the goal of the experiment section is to evaluate the performance of our approach in comparison to existing baselines:

Random: At each round, decision-maker randomly select k arms to activate.

Myopic: Select k arms that maximize the expected reward at the immediate next

round. A myopic policy ignores the impact of present actions on future rewards and instead focuses entirely on predicted immediate returns. Formally, this could be described as choosing the k arms with the largest gap $\Delta\omega_t = (\omega_{t+1}|a_t = 1) - (\omega_{t+1}|a_t = 0)$ at time t.

Constraint Myopic: It is the same as the Myopic when there is no conflict with fairness constraints, but if the fairness constraint is violated, it will choose the arm that satisfies the fairness constraint to play.

Oracle: Algorithm by Qian et al. [74] under the assumption that the states of all arms are fully observable and the transition probabilities are known without considering fairness constraints.

To demonstrate the performance of our proposed methods, we test our algorithms on synthetic domains [56] and provide numerical results averaged over 50 runs.

Average reward value with penalty: In Figure 2.8, we show the average reward $\bar{R}$ at each time step received by an arm over the time interval $T = 1000$ for $N = 50, 100, 200, 500$ and $k = 10\% \times N$ with the fairness constraint $L = 20$, and $\eta = 2$. We will receive a reward of 1 if the state of an arm is $s = 1$, and no reward otherwise. We impose a small penalty of -0.01 if the fairness constraint of an arm is not satisfied. The graph on the left shows the performance of FaWT method when assuming the transition matrix is known. The middle graph is the average reward obtained using the FaWT-U approach when the transition model is not fully available. The right graph illustrates the result of FaWT-Q method when the transition model is unknown. As shown in the figure, our approaches consistently outperform the Random and Myopic baselines, and in addition to satisfying the fairness constraints, they have a near-optimal performance with a small difference gap when compared to the Oracle baseline. Note that the Myopic approach may fail in some cases(shown in [56]), it performs worse than the Random approach.

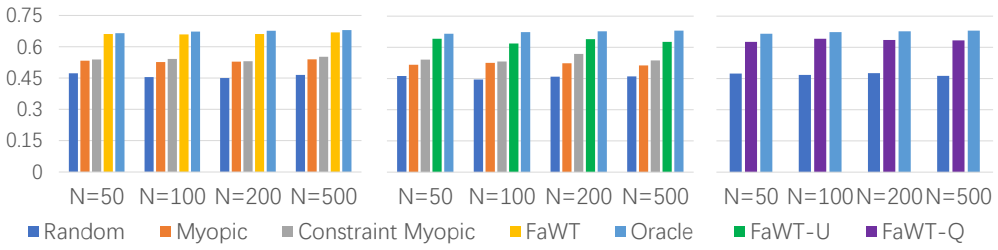


Figure 2.8: Comparison of performance of our approach and baseline approaches

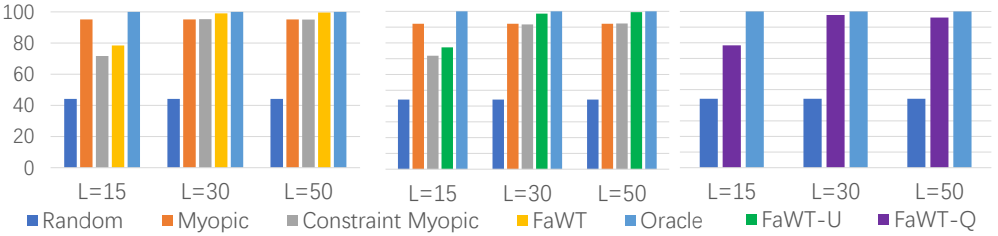


Figure 2.9: Intervention benefit ratio of our approach and baseline approaches without penalty for the violation of the fairness constraint. We set $N = 100$, $k = 10$, $T = 1000$, $\eta = 2$ and $L = \{15, 30, 50\}$.

No penalty for the violation of the fairness constraint: We also investigate the intervention benefit ratio defined as $\frac{\bar{R}_{\text{method}} - \bar{R}_{\text{No intervention}}}{\bar{R}_{\text{Oracle}} - \bar{R}_{\text{No intervention}}} \times 100\%$, where $\bar{R}_{\text{No intervention}}$ denotes the average reward without any intervention involved. Here, we do not employ penalties when the fairness constraint is not satisfied, as we want to evaluate the benefit provided by interventions with our fair policy and policies of other approaches. We provide the intervention benefit ratio for different values of L for all approaches in Figure 2.9. Again, the left graph shows the result of FaWT approach, the middle graph is the result of FaWT-U approach, and the right graph shows the result of FaWT-Q method. Our proposed approaches can achieve a better intervention benefit ratio compared with the baseline when L is 30 and above. However, for L = 15, where there is a strict fairness constraint (i.e., $\frac{k \times L}{(\eta - 1) \times N}$ is close to 1), it has a significant impact on solution quality. The performances of all our approaches improve when the fairness constraint's strength decreases (L increases). Overall, our proposed methods can handle various levels of fairness constraint strength without sacrificing significantly on solution quality. We provide additional experiments in Appendix 6.1.5 to study how the strength of fairness constraints and the intervention level affect average

rewards.

2.4 RMAB with probabilistic fairness constraint

2.4.1 Problem Description

In this section, we delve into RMABs with probabilistic fairness constraints, which is called soft fairness constraint. This constraint ensures that the algorithm does not probabilistically favor one arm over another if the latter arm's long-term cumulative reward is higher. We focus on the finite time horizon setting, which represents a more general scenario. Because algorithms developed within finite time horizon settings can be readily adapted to infinite time horizon settings. Furthermore, we consider a fully observable setting for ease of explanation.

We formally introduce the finite horizon RMAB model, where the new objective of computing policies is to balance the trade-off between maximizing cumulative rewards while giving a reasonable chance for each arm (proportional to their value) to get selected for intervention (The infinite horizon is a simpler case of the finite horizon, so we only discuss the finite-horizon setting here). As indicated earlier, this is a property that is of critical importance in public health settings. RMAB is defined using the following tuple:

$$\langle N, \{\mathcal{M}_i\}_{i \in N}, T, k \rangle$$

There are N independent arms[3] and each arm i evolves according to an associated Markov Decision Process (MDP), $\mathcal{M}_i$ is characterized by the tuple $\{\mathcal{S}_i, \mathcal{A}_i, \mathcal{P}_i, R_i, \gamma\}$:

- $\mathcal{S}_i$ represents the state space. Typically, in public health settings, $\mathcal{S}_i = \{0, 1\}$. 1 represents patient in the "good" state (patient adheres to the health program), and 0 represents patient in the "bad" state (patient not adhering).

[3] in public health settings, the patients or beneficiaries will be the arms

- $\mathcal{A}_i$ represents the action space. $\mathcal{A}_i = \{0, 1\}$ with 1 corresponding to activating or intervening on the arm and 0 action corresponding to not activating the arm or staying passive.

- $\mathcal{P}_i$ represents the action dependent transition dynamics of arm i. Specifically, $P^{a_i}_{s_i,s'_i}$ refers to the probability of transitioning from state s_i to s'_i when the arm i is taking action $a_i \in \{0, 1\}$.

- R_i provides the independent rewards obtained by arm i. We assume a range for these rewards, given by $[R_{min}, R_{max}]$. We use a simple reward function: $R_i(s_i, a_i) = s'_i \in \{0, 1\}$ determined by the next state s'_i obtained by taking action a_i when the observed state is s_i for any arm $i \in [N]$. [4] Note that the expected immediate reward will be $\mathbb{E}[R_i(s_i, a_i)] = P^{a_i}_{s_i,1}$.

- γ is the discount factor.

T is the time horizon. k is the resource capacity constraint that limits the number of arms that can be selected at each time step $t \in [T]$, i.e.,:

$$\sum_{i=1}^{N} a_i^t = k \tag{2.14}$$

Policy, π for the overall RMAB is a mapping from joint states, $\mathbf{s} = [s_1, \ldots, s_N]$ of all arms to joint actions, $\mathbf{a} = [a_1, \ldots, a_N]$. $\pi(\mathbf{s}, \mathbf{a}) \in [0, 1]$ denotes the probability of selecting the joint action $\mathbf{a}$ when the joint state of RMAB is $\mathbf{s}$. Particularly, $\pi_i(s_i, a_i) \in [0, 1]$ denotes the probability of selecting action a_i, with $\sum_{a_i} \pi_i(s_i, a_i) = 1$. We denote the state-action value function for a policy π by

$$Q^\pi(\mathbf{s}, \mathbf{a}) = \mathbb{E}^\pi[\sum_{t=1}^{T} \gamma^{t-1} R^t(\mathbf{s}^t, \mathbf{a}^t)] = \mathbb{E}^\pi[\sum_{t=1}^{T} \gamma^{t-1} \sum_{i=1}^{N} R_i^t(s_i^t, a_i^t)]$$

$Q^\pi(\mathbf{s}, \mathbf{a})$ is the expected cumulative discounted long-term reward over all arms when taking action $\mathbf{a}$ in the joint state $\mathbf{s}$. The objective is to find an optimal

[4] The reward function over RMAB can be written as $R(\mathbf{s}, \mathbf{a}) = \sum_{i=1}^{N} R_i(s_i, a_i)$

policy π^* that can satisfy

$$Q^{\pi^*}(\mathbf{s}, \mathbf{a}) = \max_{\pi} Q^{\pi}(\mathbf{s}, \mathbf{a}) = Q^*(\mathbf{s}, \mathbf{a}),$$

where $Q^*(\mathbf{s}, \mathbf{a})$ is the optimal state-action value function:

$$Q^*(\mathbf{s}, \mathbf{a}) = R(\mathbf{s}, \mathbf{a}) + \gamma \sum_{\mathbf{s}'} \Pr(\mathbf{s}'|\mathbf{s}, \mathbf{a}) \max_{\mathbf{a}'} Q^*(\mathbf{s}', \mathbf{a}') \tag{2.15}$$

Similar to Jabbari et al. [34], we define the fairness using the state-action value function $Q^*(\mathbf{s}, \mathbf{a})$ as follows:

Definition 3 *(Fairness) A stochastic policy, π is fair if for any time step $t \in [T]$, any joint state $\mathbf{s}$ and actions $\mathbf{a}, \mathbf{a}'$, where $\mathbf{a} \neq \mathbf{a}'$:*

$$\pi^t(\mathbf{s}, \mathbf{a}) \geq \pi^t(\mathbf{s}, \mathbf{a}') \textit{ only if } Q^*(\mathbf{s}, \mathbf{a}) \geq Q^*(\mathbf{s}, \mathbf{a}') \tag{2.16}$$

In summary, the objective is to efficiently approximate the maximum cumulative long-term reward while satisfying resource constraints and fairness constraints. Towards this end, the reward maximization problem can be formulated as

$$\begin{aligned} &\underset{\pi}{\text{maximize}}\ \mathbb{E}_{\pi}[\sum_{t=1}^{T} \gamma^{t-1} R^t(\mathbf{s}^t, \mathbf{a}^t)] \\ &\text{such that Equation. 2.14, and Equation. 2.16 are satisfied} \end{aligned} \tag{2.17}$$

We show in Proposition 1 that this fairness notion at the level of joint actions is equivalent to selecting arms with higher probability if their relative importance is higher. The notations that are frequently used in this Chapter are summarized in Table 2.1.

2.4.2 *SoftFair* Approach

In this section, we design a probabilistically fair (as defined in Definition 3) arm selection algorithm, referred to as *SoftFair*. *SoftFair* builds on softmax value iteration [111, 96] in conjunction with Whittle index. Softmax value iteration

Notation	Description
k, n, T	n:number of all competing arms in RMAB, k:number of arms can be selected each round, T: time horizon.
η, L	η is the minimum number of times an arm should be activated in a decision period of length L.
c	c: multiplier parameter.
λ	λ: minimum subsidy such that it is optimal to make the arm passive in current state
s_i, a_i, $\mathbf{s}, \mathbf{a}$	s_i, a_i: state and action of arm i, $\mathbf{s}, \mathbf{a}$: state vector and action vector of RMAB.
ω_s^i	ω_s^i denote the belief state, i.e., the probability that the state of arm i is 1 with the observed state s.
$[n], [T]$	We use [n] to represent the set of integers $\{0, \ldots, n\}$ for $n \in \mathbb{N}$, so as [T].
$Q_{m,t}(s,a)$, $V_{m,t}(s)$	$Q_{m,t}(s,a)$: A state-action value function for the subsidy m and state s when taking action a start at time step t followed by optimal policy using Whittle index based approach in the future time steps; $V_{m,t}(s)$: Value function for the subsidy m and state s start at time step t using Whittle index based approach
$Q_t(\mathbf{s}, \mathbf{a})$, $V_t(\mathbf{s})$	$Q_t(\mathbf{s}, \mathbf{a})$: The state-action value function when taking action $\mathbf{a}$ at time step t with state $\mathbf{s}$ $V_t(s)$: The value function at the time step t with state $\mathbf{s}$.

Table 2.1: Notations

is one of the simplest dynamic programming algorithms, which recursively computes the value function through a point-wise update rule [77].

In order to implement the softmax value iteration method in the RMAB setting, we need to compute the relative value of activating an arm (in comparison to not activating the arm) and compute the probability distribution of selecting an arm using a *softmax* function over the relative value. More specifically, during the ep-th iteration, *SoftFair* identifies the estimated value function of the state of each arm $i \in [N]$ at the time step $t \in [T]$, and calculate the difference of state-action value function between the active and passive action.

$$\begin{aligned} Q_i^{t;ep}(s_i^t, a_i^t) &= R_i^t(s_i^t, a_i^t) + \gamma \sum_{s_i^{t+1}} \Pr(s_i^{t+1} | s_i^t, a_i^t) V_i^{t+1;ep}(s_i^{t+1}) \\ \zeta_i^{t;ep}(s_i^t, a_i^t) &= e^{Q_i^{t;ep}(s_i^t, a_i^t) - V_i^{t;ep}(s_i^t)} \\ \lambda_i^{t;ep} &= \log \zeta_i^{t;ep}(s_i^t, a_i^t = 1) - \log \zeta_i^{t;ep}(s_i^t, a_i^t = 0) \end{aligned} \tag{2.18}$$

Here $V_i^{t;ep}(\cdot)$ is the value function of arm i from time step t till the end of horizon after being updated ep times. Similarly, $Q_i^{t;ep}(s_i^t, a_i^t)$ is the state-action value function of arm i from time step t till the end of horizon during ep-th iteration. Then *SoftFair* maps each arm i's state to a state-specific probability distribution over actions using the following *softmax* expression in the $k = 1$ case.

$$\pi^{t;ep}(\mathbf{s}^t, \mathbf{a}^t = \mathbb{I}_{\{i\}}) = \frac{\exp(c \cdot \lambda_i^{t;ep})}{\sum_{q=1}^{N} \exp(c \cdot \lambda_q^{t;ep})} \tag{2.19}$$

where $\mathbf{a}^t = \mathbb{I}_{\{i\}}$ denotes the joint action [5] to select the arm i while keeping other arms passive, and $\pi^{t;ep}(\mathbf{s}^t, \mathbf{a}^t = \mathbb{I}_{\{i\}})$ denotes the probability that arm i will be selected under the joint state $\mathbf{s}^t$ during the ep-th iteration. $c \in (0, \infty)$ is the multiplier parameter [6] that can adjust the gap between the probabilities of choosing an arm. If $c = \infty$, *SoftFair* becomes the standard optimal Bellman operations [5] (Refer to Equation 2.26). When $c = 0$, each arm has the same probability of being selected, and *SoftFair* can make the resources uniformly distributed. Equation 2.19 shows the probability of selecting an joint action at each time step when $k = 1$.

Unfortunately, this expression does not hold when selecting a subset of arms, i.e., $k > 1$. This is because when the resource constraint $k > 1$, the probability of an arm being selected will also rely on the probability of other arms being selected, henceforth affecting the recursive update of the value function. Let $\mathbf{a}^t = \mathbb{I}_{\{\phi\}}$ denote the action to select arms in the set ϕ. Then, ϕ^- is the set that includes all of the arms except those in set ϕ. After getting the action probability of selecting a single arm, which is the multinomial distribution, formulated as $[\pi^{t;ep}(\mathbf{s}^t, \mathbf{a}^t = \mathbb{I}_{\{1\}}), \pi^{t;ep}(\mathbf{s}^t, \mathbf{a}^t = \mathbb{I}_{\{2\}}), \ldots, \pi^{t;ep}(\mathbf{s}^t, \mathbf{a}^t = \mathbb{I}_{\{N\}})]$. We can then sample from this multinomial distribution without replacement to obtain k arms to activate, which ensures that we meet the resource constraint as well as the fairness constraint. More specifically, we can derive the probability that the arm i is among the k selected arms (active set ϕ), denoted as $\Pr(a_i^t = 1|\mathbf{s}^t)$ [7]. Consider the multinomial distribution, the results of k draws made at random without replacement is a random permutation of all the elements, and this can be computed through the permutation tool. Consequently, we have:

$$\pi^{t;ep}(\mathbf{s}^t, \mathbf{a}^t = \mathbb{I}_{\{\phi\}}) = \Pi_{i\in\phi} \Pr(a_i^t = 1|\mathbf{s}^t) \Pi_{j\in\phi^-} \big(1 - \Pr(a_j^t = 1|\mathbf{s}^t)\big) \quad (2.20)$$

[5] $\mathbb{I}_{\{i\}}$ is the indicator with value 1 at the ith item and value 0 at other places. Equivalently, this means activating arm i while keeping the other arms passive

[6] The updation process of our *Softfair* algorithm will converge to the Bellman Equation 2.26 with an exponential rate in terms of c [88], and c controls the asymptotic performance [45].

[7] Note that $\Pr(a_i = 1|\mathbf{s}) = \pi^{ep}(\mathbf{s}^t, \mathbf{a}^t = \mathbb{I}_{\{i\}}) = \textit{softmax}_c(c \cdot \lambda_i)$ if $k = 1$

For arm i, the value function $V_i^{t;ep}(\cdot)$ at time step t during the ep-th iteration can be written as

$$V_i^{t;ep}(s_i^t) = \sum_{a_i^t \in \{0,1\}} \Pr(a_i^t|\mathbf{s}^t) Q_i^{t;ep}(s_i^t, a_i^t)$$

Please note that $Pr(a_i^t|\mathbf{s}^t)$ is computed based on the sample estimate. The update of value function $V_i^{t;ep}(\cdot)$ during the ep-th iteration for any $t \in [T]$ is:

- For the current state, s_i^t of arm i, the value function is updated in the following way:

$$V_i^{t;ep+1}(s_i^t) = \sum_{a_i^t \in \{0,1\}} \Pr(a_i^t|\mathbf{s}^t) \sum_{s_i^{t+1} \in \{0,1\}} \Pr(s_i^{t+1}|s_i^t, a_i^t) \cdot \Big(R(s_i^t, a_i^t) + \gamma V_i^{t+1;ep}(s_i^{t+1}) \Big) \tag{2.21}$$

- For all other states, s^t of arm i we have

$$V_i^{t;ep+1}(s^t) = V_i^{t;ep}(s^t) \tag{2.22}$$

Similarly, we can also write the equation to update the state-action value function, and we provide it in the Appendix 6.2.1. The overall process of *SoftFair* is summarized in Algorithm 4 and is guaranteed to ensure that an arm is selected in proportion with its λ value, thereby guaranteeing fairness while approximately maximizing the overall value. This guarantee is possible because we can decouple the fairness constraint defined on the joint action to each individual arm. We have the following proposition, which is equivalent to the definition 3.

Proposition 1 *Fairness of a stochastic policy defined in Equation 2.16 can also be stated in terms of arm selection as follows:*

$$Pr(a_i^t = 1|\mathbf{s}^t) \geq Pr(a_j^t = 1|\mathbf{s}^t) \textit{ only if } \lambda_i^t \geq \lambda_j^t \tag{2.23}$$

Algorithm 4: Soft-Fair Value Iteration (*SoftFair*)

Input: Transition matrix P, time horizon T, set of observed states $\mathbf{s}$, resource constraint k, episodes K

$Q(s) \leftarrow 0, \forall s$;
for *iteration* $ep = 1, \ldots, I$ **do**
  Initialize $\mathbf{s}^0 = \{s_1^0, \ldots, s_N^0\}$;
  for *step* $t = 0, \ldots, T$ **do**
    for *arm* $i = 1, \ldots, n$ **do**
      Compute $Q_i^{t;ep}(s_i^t, a_i^t)$ and $\lambda_i^{t;ep}(s_i^t, a_i^t)$ using Equation. 2.18;
      Compute $\pi^{t;ep}(\mathbf{s}^t, \mathbf{a}^t = \mathbb{I}_{\{i\}})$ using Equation. 2.19;
    Sample k arms and add them into action set;
    for *arm* $i = 1, \ldots, n$ **do**
      Compute $\Pr(a_i^t = 1 | \mathbf{s}^t)$;
      Update $V_i^{t;ep}(s)$ using Equation. 2.21 and Equation. 2.22
    Play the arm in the action set, and observe next state $\mathbf{s}^{t+1}$

Output: The value function $V_i(s)$ for arm $i \in [n]$

Intuitively, this implies an arm, i will not be selected with lower probability than that of arm j if λ value of arm i is higher than that of arm j. The proof showing that the proposition is equivalent to the definition 3 is provided in the appendix 6.2.

2.4.3 Analysis of *SoftFair*

In this section, we formally analyze the properties of the *SoftFair* algorithm. We begin by comparing *SoftFair* with the well-known Whittle index algorithm and show why the Whittle index approach is not suitable for our case (Fairness constraint and Finite time horizon), and then provide the performance bound of *SoftFair*.

SoftFair vs. Whittle index based methods

Whittle index policy is known to be the asymptotically optimal solution to RMAB for the *infinite* time horizon. It independently assigns an index value for each arm to measure how attractive it is to activate an arm at a particular state. The index is computed using the concept of a "subsidy" m, which can

be viewed as the opportunity cost of remaining passive, and is rewarded to the arm that is passive, in addition to the usual reward. The Whittle index for an arm i is defined as the infimum value of the subsidy, m that must be offered to the algorithm to make the algorithm indifferent between selecting and not selecting the arm. Consider a single arm $i \in [n]$ where the state is s_i^t at time step $t \in [T]$, let $Q_{m;i}^t(s_i^t, a_i^t = 0)$ and $Q_{m;i}^t(s_i^t, a_i^t = 1)$ denote its active and passive state-action value functions under a subsidy m, respectively. For ease of explanation, we drop the subscript i when there is no ambiguity. The value function of an arm in the state s is

$$V_m^t(s^t) = \max\{Q_m^t(s^t, a^t = 0), Q_m^t(s^t, a^t = 1)\}.$$

The Whittle index $W(s^t)$ for the state s^t can be formally written as:

$$W(s^t) = \inf_m \left\{ m^t : Q_m^t(s^t, a^t = 0) = Q_m^t(s^t, a^t = 1) \right\}. \tag{2.24}$$

After computing the Whittle index for each arm, a policy π will activate those k arms whose current states have the highest indices. In order to use the Whittle index approach, it needs to satisfy a technical condition called *indexability* introduced by Weber and Weiss [102]. The indexability can be expressed in a simple way: Consider an arm with subsidy m, the optimal action is passive, then $\forall m' > m$, the optimal action should remain passive. The RMAB is indexable if every arm is indexable.

However, traditional Whittle index based approaches rely on the assumption of an infinite time horizon, and the performance deteriorates severely when time horizons are finite. Figure 2.10 shows an illustrative example where Whittle index values are low when an arm's residual time horizon is short, and there is a bias in approximating the Whittle index value under the finite time horizon setting using methods proposed in [74, 57]. Often, real-world phenomena are formalized in a finite time horizon setting, which precludes the direct use of Whittle index based methods. We now demonstrate that a phenomenon called

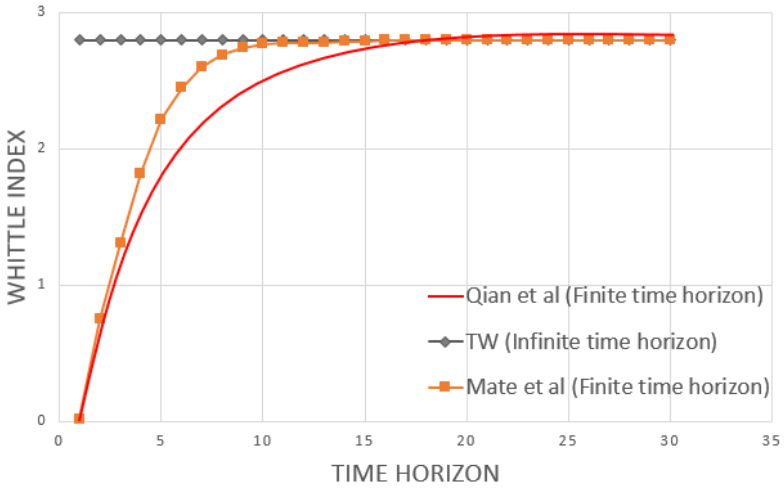


Figure 2.10: Whittle index value as a function of the residual time horizon. Figure taken from Mate et al. [57]. The grey line is the whittle index value in an infinite time horizon setting, and the others are approximated Whittle index values under a finite time horizon to capture the index decay phenomenon.

Whittle index decay [57, 48] exists in our problem. All detailed proofs can be found in the Appendix 6.2.

Theorem 4 *At any time step $t \in [T]$, the Whittle index m^t for arm i under the observed state s_i^t is the value that satisfies the equation $Q_m^t(s_i^t, a_i^t = 0) = Q_m^t(s_i^t, a_i^t = 1)$. The Whittle index will decay as the value of current time step t increases: $\forall t < T : m^t > m^{t+1} \geq m^T = P_{s,1}^1 - P_{s,1}^0$.*

Proof Sketch. Consider the discount reward criterion with the discount factor γ, we can simply compute m^T and m^{T-1} by solving equations $Q_m^T(s_i^T, a_i^T = 0) = Q_m^T(s_i^T, a_i^T = 1)$ and $Q_m^{T-1}(s_i^{T-1}, a_i^{T-1} = 0) = Q_m^{T-1}(s_i^{T-1}, a_i^{T-1} = 1)$. We can find that $m^{T-1} \geq m^T = P_{s,1}^1 - P_{s,1}^0$. Then in order to show $m^t > m^{t+1}$, we first prove a lemma to show value function $V_m^t(s_i^t) > V_m^{t+1}(s_i^t) \geq 0$, and then we can combine this with the definition of m^t to complete the proof. The detailed proof can be found in the appendix 6.2. □

The Whittle index based approach needs to solve the costly finite horizon problem because the index value varies according to the residual time horizon even in the same state, and computing the index value under the finite horizon setting is ($O(|S|^k T)$ time and space complexity [32]. However, as an alternative

method, our *SoftFair* can naturally approximate the optimal value function at arbitrary time steps while requiring less memory space than model-free learning methods such as Q-learning. In addition, the optimal condition for approximating the Whittle index value is difficult to satisfy. For example, Mate et al. [57] demonstrate that their proposed approach is optimal under the condition:

$$P^1_{1,1} - P^1_{0,1} \leq \left(P^0_{1,1} - P^0_{0,1}\right)\left(1 + \gamma(P^1_{1,1} - P^1_{0,1})\right)(1 - \gamma)$$

Intuitively, consider the case where $P^0_{0,1} = P^1_{0,1}$ and $P^0_{1,1} = P^1_{1,1}$ (also considered by Liu and Zhao [53]), this makes such a condition always not satisfied. Furthermore, we will show that the Whittle index based approach fails to address the problem of fair distribution of interventions (the distribution of resources is lopsided). In contrast, our proposed method, *SoftFair* becomes the optimal algorithm when $c \to \infty$ and can control the trade-off between optimal performance and uniform distribution of resources.

Due to the finite time horizon setting in many practical applications, the Whittle index based method can not effectively approximate the whittle index value, and it only concentrates on beneficiaries who can mostly improve the objective in the case of initiatives related to public health. This can result in some beneficiaries never having the opportunity to receive intervention from public health professionals, which may lead to a poor adherence behavior and henceforth a bad state from which improvements may only be marginal even with intervention, preventing them from ever being chosen by the index policy. Refer to Figure 2.1 to get a better picture of the difference between the Whittle index approach and *SoftFair*. We can see that when using the Threshold Whittle index based method proposed by Mate et al. [57], the activation frequency of the arm is extremely unbalanced, with nearly half of the arms never being selected. Such starvation of interventions may escalate to communities. To avoid such cycle between bad outcomes, the RMAB needs to consider fairness

in addition to maximizing cumulative long-term reward when picking arms. We now demonstrate why *SoftFair* can satisfy our proposed fairness constraint while effectively approximating our cumulative reward maximization objective. We begin by providing a theorem showing that *SoftFair* is guaranteed to be optimal when the multiplier parameter $c \to \infty$.

Theorem 5 *Choosing the top k arms according to the λ value in Equation 2.19 ($c \to \infty$) is equivalent to maximizing the cumulative long-term reward.*

Proof Sketch. Because when c approaches infinity, *SoftFair* becomes deterministically choosing the arm with the highest λ value. Let ϕ^* to be the set of actions containing the k arms with the highest-ranking of λ value, we need to show $Q(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi\}}) \geq Q(\mathbf{s}, \mathbf{a}' = \mathbb{I}_{\{\phi'\}})$ for $\forall \phi'$, where ϕ' is the set of any k selected arms, and $\phi' \neq \phi^*$. We first get the expression of $\sum_{i \in \phi} \lambda_i$. Combining the definition of λ in Equation 2.18 with the fact that $\sum_{i \in \phi^*} \lambda_i \geq \sum_{j \in \phi'} \lambda_j$, we add $\sum_{z \notin \phi^* \wedge z \notin \phi'} Q(s_z, a_z = 0)$ on both sides of the inequality function to show $Q(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi\}}) \geq Q(\mathbf{s}, \mathbf{a}' = \mathbb{I}_{\{\phi'\}})$. □

When c approaches infinity, *SoftFair* becomes the optimal policy, but it will suffer from the starvation phenomena. As c gets closer to 0, *SoftFair* can ensure that every arm/beneficiary has roughly the same probability of receiving the intervention, which leads to a uniform distribution of resources. Given these facts, c can control the trade-off between ensuring the fair distribution of resources and the objective of maximizing cumulative rewards. In the subsequent theorem, we demonstrate that *SoftFair* satisfies our proposed fairness constraint.

Theorem 6 *SoftFair is fair under our proposed fairness constraint, and c controls the trade-off between fairness and optimal performance.*

Proof Sketch. Similar to the proof of the Theorem 5, we can see that the value of λ is proportional to the state-action value function. According to the Equation 2.19, the probability of selecting an arm is the softmax function on λ,

and it can be guaranteed that the higher the value of λ, the higher the probability of selecting that arm. Therefore *SoftFair* remains fair under our proposed fairness constraints. The trade-off between ensuring a fair distribution of resources and the objective of maximizing cumulative rewards is governed by c, where a larger c means *SoftFair* prefers arms with a higher value of λ, while a small c means that *SoftFair* tends to ensure that resources are uniformly distributed among the arms.

□

In the next section, we will show how the value of c controls the performance bounds of the *SoftFair* algorithm.

Performance bound of *SoftFair*

For ease of explanation, we investigate the case of $k = 1$ at each time step, and the multi-selection ($k > 1$) can be viewed as the iteration of the case $k = 1$. Let Ψ_{soft} denote our *Soft* operator at time step $t \in [T]$, we ignore the subscript t here, which is

$$\begin{aligned} Q^{ep+1}(\mathbf{s}, \mathbf{a}) &= \Psi_{soft} Q^{ep}(\mathbf{s}, \mathbf{a}) \\ &= \sum_{\mathbf{s}'} \Pr(\mathbf{s}'|\mathbf{s}, \mathbf{a})(R(\mathbf{s}, \mathbf{a}) + \gamma \sum_{\mathbf{a}'} \Pr(\mathbf{a}'|\mathbf{s}') Q^{ep}(\mathbf{s}', \mathbf{a}')) \\ &= R(\mathbf{s}, \mathbf{a}) + \gamma \sum_{\mathbf{s}'} \Pr(\mathbf{s}'|\mathbf{s}, \mathbf{a}) \sum_{\mathbf{a}'} \Pr(\mathbf{a}'|\mathbf{s}') Q^{ep}(\mathbf{s}', \mathbf{a}'). \end{aligned} \tag{2.25}$$

Before we derive the performance bound for *SoftFair*, We first bound the state-action value function in the following lemma.

Lemma 1 *The state-action value function $Q(\mathbf{s}, \mathbf{a})$ is bounded within $[0, \frac{n}{1-\gamma}]$.*

Proof Sketch. The upper bound can be obtained by showing that $\forall(\mathbf{s}, \mathbf{a})$, state-action value during the $ep-$th iteration are bounded through induction. □

Corollary 1 *As we have $R_{max} = n$ and $R_{min} = 0$ of RMAB, we can easily derive that $|Q(\mathbf{s}, \mathbf{a}) - Q(\mathbf{s}, \mathbf{a}')| \leq \frac{n}{1-\gamma}$, for $\forall Q$ and $\forall(\mathbf{s}, \mathbf{a})$.*

Following Song et al. [88], we let $\delta(\mathbf{s}) = \sup_Q \max_{\mathbf{a},\mathbf{a}'} |Q(\mathbf{s},\mathbf{a}) - Q(\mathbf{s},\mathbf{a}')|$ denote the largest distance between state-action value functions. Then we have the following lemma showing the bound on the difference between two state-action value functions.

Lemma 2 *$\forall Q$ and $\forall \mathbf{s}$, Let $\Pr(\cdot|\mathbf{s}) = [\Pr(\mathbf{a} = \mathbb{I}_{\{1\}}|\mathbf{s}), \ldots, \Pr(\mathbf{a} = \mathbb{I}_{\{n\}}|\mathbf{s})]^\top$ and $Q(\mathbf{s},\cdot) = [Q(\mathbf{s},\mathbf{a} = \mathbb{I}_{\{1\}}), \ldots, Q(\mathbf{s},\mathbf{a} = \mathbb{I}_{\{n\}})]^\top$, here the superscript $\top$ denotes the vector transpose. We have $\frac{\delta(\mathbf{s})}{n\exp[c\cdot\delta(\mathbf{s})]} \leq \max_{\mathbf{a}} Q(\mathbf{s},\mathbf{a}) - (\Pr(\cdot|\mathbf{s}))^\top Q(\mathbf{s},\cdot) \leq \frac{n-1}{2+c}$.*

Proof Sketch. We first sort $Q(\mathbf{s},\mathbf{a} = \mathbb{I}_{\{i\}})$ in the ascending order according to the λ value and replace $\Pr(\cdot|\mathbf{s})$ with $Q(\mathbf{s},\cdot)$. We take advantage of the fact that for any two non-negative sequences $\{x_i\}$ and $\{y_i\}$, $\frac{\sum_i x_i}{1+\sum_i y_i} \leq \sum_i \frac{x_i}{1+y_i}$, combine this fact with the difference between state-action value functions for different actions. Through using Taylor series, we can derive the upper and lower bounds. □

Different from *Soft* Operator Ψ_{soft} in Eq. 2.25 ,let Ψ denote the Bellman optimality operator, which we have

$$\begin{aligned} Q^{ep+1}(\mathbf{s},\mathbf{a}) &= \Psi Q^{ep}(\mathbf{s},\mathbf{a}) \\ &= R(\mathbf{s},\mathbf{a}) + \gamma \sum_{\mathbf{s}'} \Pr(\mathbf{s}'|\mathbf{s},\mathbf{a}) \max_{\mathbf{a}'} Q^{ep}(\mathbf{s}',\mathbf{a}') \end{aligned} \tag{2.26}$$

For the optimal state-action value function, we have $\Psi Q^*(\mathbf{s},\mathbf{a}) = Q^*(\mathbf{s},\mathbf{a})$. We have the following theorem showing the performance bound of *SoftFair* compared to the optimal value.

Theorem 7 *Our SoftFair method can achieve the performance bound as $\limsup_{ep\to\infty} V^{ep}(\mathbf{s}) \leq V^*(\mathbf{s})$, where $V^*(\mathbf{s})$ is the optimal value function. More specifically, we have*

$$\limsup_{ep\to\infty} Q^{ep}(\mathbf{s},\mathbf{a}) \leq Q^*(\mathbf{s},\mathbf{a}) \quad \textit{and}$$

$$\liminf_{ep\to\infty} Q^{ep}(\mathbf{s},\mathbf{a}) \geq Q^*(\mathbf{s},\mathbf{a}) - \frac{n-1}{(2+c)(1-\gamma)}$$

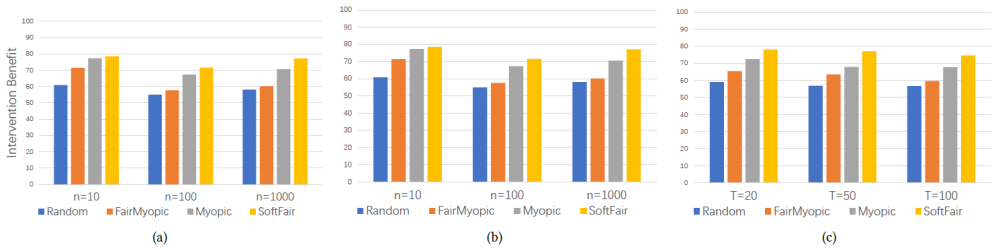


Figure 2.11: Intervention benefit of *SoftFair* is consistently greater than other baselines. (a) We fix $T = 50$, and $k = 10\% * n$, and let $n = \{10, 100, 1000\}$. (b) We fix $T = 50$, and $n = 100$, and let $k = \{5, 10, 20\}$. (c) We fix $n = 100$, and $k = 10$, and let $T = \{20, 50, 100\}$.

Proof. We derive the performance bound through induction based on Lemma 1 and 2. □

Conjecture 1 *For the cause when multiple arms can be pulled at each time step, i.e., $k > 1$, Our SoftFair method can achieve the bound as $\limsup_{ep\to\infty} \Psi^{ep} V^0(\mathbf{s}) \leq V^*(\mathbf{s})$. More specifically, we have*

$$\limsup_{ep\to\infty} Q^{ep}(\mathbf{s}, \mathbf{a}) = \limsup_{ep\to\infty} \Psi^{ep} Q^0(\mathbf{s}, \mathbf{a}) \leq Q^*(\mathbf{s}, \mathbf{a}) \quad \textit{and}$$

$$\liminf_{ep\to\infty} Q^{ep}(\mathbf{s}, \mathbf{a}) \geq Q^*(\mathbf{s}, \mathbf{a}) - \frac{n-k}{(2+c)(1-\gamma)}$$

Given that Lemma 1 and Corollary 1 hold true for $k > 1$, but Lemma 2 changes, and we predict the upper bound will change from $\frac{n-1}{(2+c)(1-\gamma)}$ to $\frac{n-k}{(2+c)(1-\gamma)}$. This leads us to conjecture a new performance bound. Furthermore, when $k = n$, all arms are selected simultaneously, resulting in $\liminf_{ep\to\infty} Q^{ep}(\mathbf{s}, \mathbf{a}) = Q^*(\mathbf{s}, \mathbf{a})$. Therefore, this observation aligns with our conjecture.

2.4.4 Experiments

In this section, we empirically compare our proposed method *SoftFair* to the baselines on both (a) a realistic patient adherence behavior dataset [40] and (b) a synthetic dataset to represent more general settings enforced by structural constraints on transition matrix (more details about the dataset can be found in Appendix 6.2.5). We consider the finite time horizon where reward

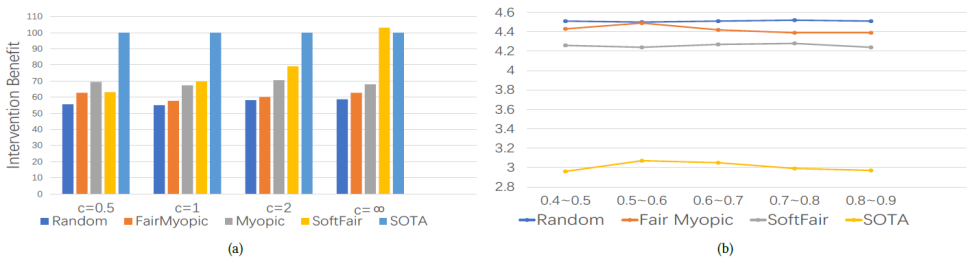


Figure 2.12: (a) The intervention benefit of different multiplier c. Here $c = \infty$ refers to deterministically selecting the top k arm with the highest cumulative rewards. (b) The action entropy of a single process. We investigate the action entropy for different value of $P^1_{0,1}$ range from 0.4 to 0.9 (at 0.45, 0.55, 0.65, 0.75, 0.85, respectively), and $c = 1$.

is the undiscounted sum of arms/beneficiaries in the good state over all time steps and set the following scenario for the simulation: $n = \{10, 100, 1000\}$, $k = \{5\%n, 10\%n, 20\%n\}$, $T = \{20, 50, 100\}$. All results are averaged over 50 simulations. In particular, We compare our method against the following baselines:

- **Random**: At each time step, algorithm randomly select k arms to play. This will ensure that each arm has the same probability of being selected.

- **Myopic**: A myopic policy ignores the impact of present actions on future rewards and instead focuses entirely on the predicted immediate returns. It select k arms that maximize the expected reward at the immediate next time step. Formally, this could be described as choosing the k arms with the largest gap $\Delta^t = P^1_{s,1} - P^0_{s,1}$ at time step t under the observed state s.

- **FairMyopic**: After computing Δ^t for each arm, instead of deterministically selecting the arm with the highest immediate reward, we use the *softmax* function over Δ^t to get the probability of each arm being selected. Then we sample the k arms according to the probability.

- **FaWT**: Algorithm proposed by Li and Varakantham [48]. They ensure that each arm will be selected at least η times during any intervention interval of length L. Since this algorithm requires two predefined and extra parameters,

the intervention interval length L and the minimum selection times during each interval η, it is not feasible to create a fair comparison against other approaches across all settings. However, for one of the settings we are able to provide a direct comparison with *SoftFair* by doing a brute force search for fair parameter values for FaWT.

- **SOTA**: Algorithm proposed by Mate et al. [57] under the assumption that the states of all arms are fully observable and the transition probabilities are known. We use a sigmoid function to approximate the Whittle index value and select arms deterministically for the finite time horizon setting.

We examine policy performance from two perspectives: (a) Intervention benefit (essentially the solution quality): The intervention benefit is defined as $\frac{\bar{R}_{\text{method}}-\bar{R}_{\text{No intervention}}}{\bar{R}_{\text{SOTA}}-\bar{R}_{\text{No intervention}}} \times 100\%$. It calculates the difference between one algorithm's expected cumulative reward and the cumulative reward when no intervention is involved, then normalized by the difference between the asymptotically optimal but fairness-agnostic SOTA algorithm in baselines (100% intervention benefit) and the reward obtained without intervention (0% intervention benefit) and. (b) Action distribution entropy (representative of the fairness): We calculate the selection frequency distribution across all time steps, and then compute its entropy after normalization through: $Entropy = -\sum_{i\in[n]} P(i)\log P(i)$, where $P(i)$ refers to the normalization of the number of times arm i is selected (i.e., the number of times that arm i has been selected divided by $k \cdot T$), and $P(i)\log P(i) = 0$ if an arm is never selected.

Realistic dataset : Obstructive sleep apnea is one of the most prevalent sleep disorder among adults, and continuous positive airway pressure therapy (CPAP) is a highly effective treatment when it is used consistently for the duration of each sleep bout. But non-adherence to CPAP in patients hinders effective treatment for this type of sleep disorder. Similar to [29], we adapt the Markov model

Policy	Intervention benefit	Action entropy
Random	79 ± 13	4.56 ± 0.0056
Myopic	98 ± 3.3	2.67 ± 0.0
FairMyopic	83 ± 11	4.5 ± 0.0089
SoftFair	93 ± 7.6	4.27 ± 0.019

Table 2.2: Resutls for CAPA Adherence dataset with $n = 100, k = 10, T = 80$. The Myopic policy provides the highest and most stable intervention benefit but with low action diversity (has a severe intervention starvation). In contrast, the Random policy has high action diversity but lower and more variable benefits. FairMyopic and SoftFair strike a balance, with SoftFair showing a relatively high and stable benefit along with high action diversity.

of CPAP adherence behavior in [40] to a two-state system with the clinical adherence criteria. We add a small noise to each transition matrix so that the dynamics of each individual arm is different (See more details about the dataset in Appendix 6.2.5).

In table 2.2, we report average results for each algorithm. Myopic method has the best performance, which is caused by the specific structure of the underlying transition matrices, since there is not too much difference between n Markovian models, and in this case the Myopic approach is indeed close to optimal. However, the myopic approach has significantly lower action entropy, which is indicative of overall fairness. Meanwhile, our *SoftFair* provides the right trade-off between intervention benefit and having a varied selection of arms (high action entropy) at each time step.

Synthetic dataset (a) We first test the performance when the number of patients (arms) varies. Figure 2.11a compares the intervention benefit for $n = \{10, 100, 1000\}$ patients and $k = 10\%$ of n. As shown in Figure 2.11a, in addition to satisfying the fairness constraints, our *SoftFair* consistently outperforms the Random, Myopic and FairMyopic baselines. (b) We next compare the intervention benefit when the number of arms n is fixed and the resource constraint k is varied. Specifically, we fix $n = 100$ patients, and let $k = \{5, 10, 20\}$. Figure 2.11b shows that there has been a gradual increase in the intervention

benefit as the k increases. One possible reason is that a larger resource budget k can make the arms with higher cumulative rewards more likely to be selected, thereby reducing the performance gap with the SOTA method. (c) The performance of our method is slightly influenced by the time horizon T. As shown in Figure 2.11c, the common trend is that a smaller T leads to better performance. This means that our method can efficiently solve the RMAB in a finite time horizon, while a larger horizon T will make the convergence slower. Overall, all results demonstrate the our method provides a good trade-off between providing high intervention benefit and preventing starvation for arms (through high action entropy).

Intervention benefit when c changes We investigate the effect of the multiplier parameter c on performance. Formally, a larger c will widen the gap between the probabilities of choosing an arm, leading to better performance as it prefers selecting an arm with a higher cumulative reward. Figure 2.12 (a) reveals that *SoftFair* performs well empirically as c increases, and if we deterministically choose the top k arms based on the value of λ, it achieves the optimal result.

Action entropy comparison We also compare the entropy of the action of a process in the synthetic dataset when $P^1_{0,1}$ ranges from 0.4 to 0.9. As shown in Figure 2.12, the Random policy has the highest value as it requires uniform selection of all arms. Our proposed method, *SoftFair* consistently has a higher action entropy than the SOTA method because we enforce fairness constraints. FairMyopic has a high action entropy value, but it is indeed unfair under our proposed fairness constraints, as it relies on immediate rewards.

***SoftFair* vs. FaWT** We perform a search in the value space of parameters η and L of FaWT and the value space of multiplier c of *SoftFair*, and we use the value of these parameters which makes the values of the action distribution entropy of these two methods close to each other and compare their performance.

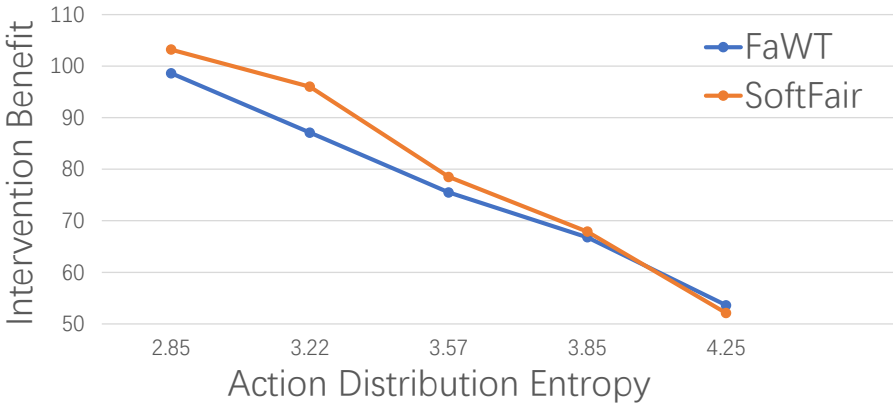


Figure 2.13: Comparison of performance of FaWT and *SoftFair* when their action distribution entropy values are close.

We present the result in Figure 2.13. As shown in the figure, *SoftFair* can better balance the trade-off between the goal of uniform resource distribution and maximizing cumulative rewards. This may be due to the difficulty in satisfying the conditions for optimal performance of FaWT.

Discussion In some real-world applications, state transitions may not be fully available. In this case, we can learn the transition probabilities online using a learning method based on Thompson Sampling. We initially assumed that each arm had a prior beta distribution in transition probabilities based on prior knowledge. If no prior knowledge is available, we assume the prior $Beta(1, 1)$, since this is a uniform distribution over $(0, 1)$. The Beta distribution was chosen because it is a convenient and useful choice for Bernoulli rewards. At each time step, we sample the posterior distribution of the parameters and then use the *SoftFair* algorithm to choose which arm to play. We can use our observations to update our posterior distribution. The algorithm then draws samples from the posterior distribution and repeats the process again. Table 2.3 shows the result for the CAPA Adherence dataset when the transition probability is not available.

Policy	Intervention benefit	Action entropy
Random	79 ± 13	4.56 ± 0.0056
Myopic	94 ± 5.0	3.07 ± 0.0009
FairMyopic	84 ± 10.5	4.49 ± 0.0083
SoftFair	92 ± 8.1	4.21 ± 0.023

Table 2.3: Resutls for CAPA Adherence dataset with $n = 100, k = 10, T = 80$. The Myopic policy provides a high and stable intervention benefit but with low action diversity. The Random policy shows high action diversity but lower and more variable benefits. FairMyopic offers moderate benefits with high diversity, while SoftFair strikes a balance with high benefits and relatively high action diversity.

2.5 Conclusion

In this chapter, we initiate the study of fairness constraints in the context of the Restless Multi-Arm Bandits model, which is of critical importance for adherence problems in public health (e.g., monitoring the adherence of preventive medicine for Tuberculosis, monitoring the engagement of mothers during calls on good practices during pregnancy). We define a fairness metric that encapsulates and generalizes existing fairness definitions employed for both Multi-Arm Bandit and reinforcement learning problems. To tackle the challenges introduced by the objective, we design different computationally efficient algorithms for the proposed different types of fairness constraints. Specifically, contrary to expectations, we are able to provide minor modifications to the existing algorithm for RMAB problems in order to handle the deterministic fairness constraint. For the probabilistic fairness constraint, we integrate the advances in RMAB research and *softmax* value iteration technique to effectively approximate the optimal value function within the proven conditions while having fairness guarantees. We provide theoretical results on how our methods provide the best way to handle fairness without sacrificing solution quality. Our approaches are demonstrated empirically as well on benchmark problems from the literature.

To summarize, addressing fairness constraints within the Restless Multi-

Armed Bandit framework is crucial for ensuring equitable decision learning processes. As we move on to subsequent chapters in the context of reinforcement learning (RL), our focus will expand to the overarching theme of this dissertation: sequential decision learning for social good. Our exploration is dedicated not only to deepening our understanding of optimizing reinforcement learning strategies to benefit social good, but also to improving the sample efficiency of algorithms in real-world applications with limited resources. Through this effort, we strive to provide valuable insights and methods that effectively connect theoretical advances to practical implementations, thereby facilitating the deployment of artificial intelligence systems in the real world.

Chapter 3

Influence Maximization in Unknown Social Networks

In Chapter 2, we presented sequential decision learning within the Restless Multi-Armed Bandit (RMAB) framework. RMAB is a powerful framework for budget-constrained resource allocation tasks in which a decision-maker must select a subset of arms for interventions in each round. We delved into different types of fairness constraints to develop practical solutions for RMAB, particularly in the critical domain of health interventions with the presence of severe resource limitations. In the upcoming two chapters, we shift our focus to sequential decision learning within the Reinforcement Learning (RL) framework. The agent in RL employs the principle of trial and error to investigate the optimal consequences of actions in a given state, guided by the feedback from scalar signals associated with each action. Yet, in many real-world applications, the interaction between the environment and RL agents incurs significant costs, thereby restricting our capacity for data collection and underscoring the need for sample-efficient algorithms. This dissertation investigates two distinct sequential decision learning problems aimed at promoting social good under resource constraints. For both problems, we apply curriculum-based approaches to enhance sample efficiency and minimize costly interactions.

In this chapter, we consider a specific practical problem in the reinforcement learning setting. We focus on the challenge of identifying a small subset of nodes in a network that can maximize the diffusion of information. Recently, it has also found application in HIV prevention, substance abuse prevention, micro-finance adoption, etc., where the goal is to find the set of peer leaders in a real-world physical social network who can disseminate information to a large group of people. Unlike online social networks, real-world networks are not completely known, and collecting information about the network is costly as it involves surveying multiple people. In this work, we focus on the problem of network discovery for influence maximization. The existing work in this direction proposes a reinforcement learning framework. As the environment interactions in real-world settings are costly, so it is important for the reinforcement learning algorithms to have minimum possible environment interactions, i.e, to be sample efficient. In this Chapter, we propose CLAIM - **C**urriculum **L**e**A**rning **P**olicy for **I**nfluence **M**aximization to improve the sample efficiency of RL methods. We conduct experiments on real-world datasets and show that our approach can outperform the current best approach. This work is presented in:

- Li, Dexun, Meghna Lowalekar, and Pradeep Varakantham. "CLAIM: Curriculum learning policy for influence maximization in unknown social networks." Uncertainty in Artificial Intelligence. PMLR, 2021.

3.1 Introduction

Social interactions between people play an important role in spreading information and behavioral changes. The problem of identifying a small set of influential nodes in a social network that can help in spreading information to a large group is termed as influence maximization (IM) [43]. It was widely used in applications such as viral marketing [43], rumor control [13], etc, which use the online

social networks. In addition to these, IM has also found useful applications in domains involving real world physical social networks. Some of these applications include identifying peer leaders in homeless youth networks to spread awareness about HIV [105, 109], identifying student leaders in school networks to disseminate information on substance abuse [95], identifying users who can increase participation in micro-finance [7], etc. In the case of real world social networks, the network information is not readily available and it is generally gathered by individually surveying different people who are part of the network. As conducting such surveys is a time-intensive process requiring substantial efforts from a dedicated team of social work researchers, it is not practically possible to have access to a complete network structure. Therefore, the influence maximization problem in the real world is coupled with the uncertain problem of discovering a network using a limited survey budget (i.e., the number of people who can be queried).

Most of the existing work [105, 104, 109] which addresses real-world influence maximization problems perform network discovery by surveying nodes while exploiting a specific network property such as community structure. CHANGE algorithm [105] is based on the principle of *friendship paradox* and performs network discovery by surveying a random node and one of its neighbour. Each node reveals the information about its neighbors upon querying. The subgraph obtained after querying a limited set of nodes is used to pick a set of influential nodes using an influence maximization algorithm. A recent work by Kamarthi et al. [39] provides a reinforcement learning based approach to automatically train an agent for network discovery. They developed an extension to DQN referred to as Geometric-DQN to learn policies for network discovery by extracting relevant graph properties, which achieves better performance than the existing approaches. As any other reinforcement learning approach, the work by Kamarthi et al. [39] needs to perform multiple interactions with the environment to perform exploration. As in the real world settings, the environment interac-

tions are costly, the approach can be improved by reducing the environment interactions, i.e., by increasing the sample efficiency. This approach employs a myopic heuristic (new nodes discovered) to guide exploration and we employ goal directed learning to provide a forward looking (non-myopic) heuristic.

In this chapter, we propose to model the network discovery problem as a goal-directed reinforcement learning problem. We take the advantage of the *Hindsight Experience Replay* [4] framework which suggests learning from failed trajectories of agent by replaying each episode with a different goal (e.g. the state visited by agent at the end of its failed trajectory) than the one agent was trying to achieve. This helps in increasing sample efficiency as agents can get multiple experiences for learning in a single environment interaction. To further improve the performance, we use the curriculum guided selection scheme proposed by Fang et al. [24] to select the set of episodes for experience replay. While there have been some other works that focus on improving the sample-efficiency [89, 14, 19], most of them are designed for domain-specific applications and unlike our curriculum-guided selection scheme which adaptively controls the exploration-exploitation trade-off by gradually changing the preference on goal-proximity and diversity-based curiosity, they only perform curiosity-driven learning.

Contributions: In summary, following are the main contributions of the this chapter along different dimensions:

- **Problem:** We convert the whole process of network discovery and influence maximization into a goal directed learning problem. Unlike standard goal directed learning problems where the goal state is known, in this problem, the goal state is not given. We provide a novel heuristic to generate goals for our problem setting.

- **Algorithm:** We propose a new approach CLAIM - **C**urriculum **L**e**A**rning Policy for **I**nfluence **M**aximization in unknown social networks which by using

Curriculum guided hindsight experience replay and goal directed Geometric-DQN architecture can learn sample efficient policies for discovering network structure.

- **Experiments:** We perform experiments in social networks from three different domains and show that by using our approach, the total number of influenced nodes can be improved by up to 7.51% over the existing approach.

Notation	Description
$G^* = (V^*, E^*)$	Entire Unknown Graph
S	Set of nodes known initially
$G_t = (V_t, E_t)$	Subgraph of G^* discovered after t queries
$N_{G^*}(u)$	Neighbors of vertex u in graph G^*
$E(X, Y)$	All direct edges that connect a node in set X and a node in set Y
$O(G)$	Set of nodes from graph G selected by influence maximization algorithm O
$I_{G^*}(A)$	Expected Number of nodes influenced in graphG^* on choosing A as the set of nodes to activate

Table 3.1: Notations

3.2 Problem Description

The problem considered in this work involves discovering a subgraph of the unknown network such that the set of peer leaders chosen from the discovered subgraph maximizes the number of people influenced by peer leaders. We now describe both the components of the problem, i.e., network discovery and influence maximization in detail. The notations used in the problem description are defined in Table 3.1.

- **Network Discovery Problem:** The network discovery problem can be described as a sequential decision-making problem where at each step, the agent queries a node from the discovered subgraph. The queried node reveals its neighbors, expanding the discovered subgraph. The process

goes on for a fixed number of steps, determined by the budget constraint. Formally, initially we are given a set of nodes S and the agent can observe all the neighbors of nodes in set S. Therefore, $V_0 = S \cup N_{G^*}(S)$. The agent has a budget of T queries to gather additional information. For $(t+1)^{th}$ query, the agent can choose a node u_t from G_t and observe G_{t+1}.

$$G_{t+1} = (V_t \cup N_{G^*}(u_t), E(G_t) \cup E(N_{G^*}(u_t), \{u_t\})).$$

At the end of network discovery process, i.e., after T queries, we get the final discovered subgraph G_T. This graph is provided as an input to an IM algorithm.

- **Influence Maximization (IM) :** IM is the problem of choosing a set of influential nodes in a social network who can propagate information to maximum nodes. In this chapter, the information propagation over the network is modeled using the Independent Cascade Model (ICM) [43], which is the most commonly used model in the literature. In the ICM, at the start of the process, only the nodes in the set of chosen initial nodes are active. The process unfolds over a series of discrete time steps, where at every step, each newly activated node attempts to activate each of its inactive neighbors and succeeds with some probability p. The process ends when there are no newly activated nodes at the final step. After discovering the subgraph G_T using the network discovery process, we can use any standard influence maximization algorithm to find out the best set of nodes to activate based on the available information. Lowalekar et al. [54] showed the robustness of the well-known greedy approach [43] on medium scale social network instances, which is also served as the oracle in our work.

Overall, given a set of initial nodes S and its observed connections $N_{G^*}(S)$, our task is to find sequence of queries $(u_0, u_1, \ldots, u_{T-1})$ such that G_T maximizes

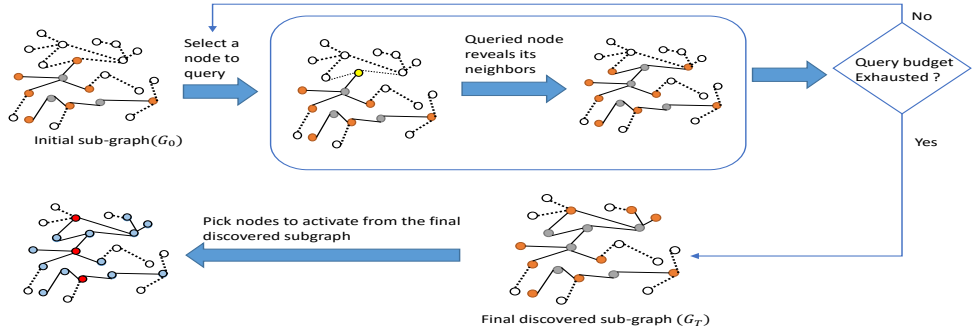


Figure 3.1: Network discovery and influence maximization. Grey: Set of Queried Nodes; Orange: Set of unqueried nodes (In the initial subgraph G_0, grey nodes will represent the set S and orange nodes will represent the set $N_{G^*}(S)$), Yellow: node picked by the agent to query (u_t); Red: nodes selected by influence maximization algorithm in the final discovered subgraph ($O(G_T)$); Blue: other nodes in the final discovered subgraph G_T

$I_{G^*}(O(G_T))$. Figure 3.1 shows the visual representation of the problem.

3.3 Background

In this section, we describe the relevant research, the MDP formulation and the Geometric-DQN architecture used by Kamarthi et al. [39] to solve the network discovery and influence maximization problem.

3.3.1 MDP Formulation

The social network discovery and influence maximization problem can be formally modelled as an MDP.

- **State:** The current discovered graph G_t is the state.
- **Actions:** The nodes yet to be queried in network G_t constitute the action space. So, set of possible actions is $V_t \setminus \{S \cup_{i \leq t} u_i\}$ $\forall t > 0$ and $N_{G^*}(S)$ when $t = 0$.
- **Rewards:** Reward is only obtained at the end of episode, i.e,, after T steps. It is the number of nodes influenced in the entire graph G^* using G_T, i.e.,

$I_{G^*}(O(G_T))$. The episode reward is denoted by R_T, where T is the length of the episode (budget on the number of queries available to discover the network).

Training: To train the agent in the MDP environment, DQN algorithm is used but the original DQN architecture which takes only the state representation as an input and outputs the action values can not be used as the action set is not constant and depends on the current graph. Therefore, both state and action are provided as an input to DQN and it predicts the state action value. The DQN model can be trained using a single or multiple graphs. If we train simultaneously on multiple graphs, then the MDP problem turns out to be Partially observable MDP, as the next state is determined by both the current state and current action as well as the graph we are using. The range of reward values also depends on the size and structure of the graph, therefore, the reward value is normalized when multiple graphs are used for training.

$$R_T = \frac{I_{G^*}(O(G_T))}{OPT(G^*)} \tag{3.1}$$

3.3.2 Geometric-DQN

As described in the previous section, the state is the current discovered graph G_t and actions are the unqueried nodes in the current discovered graph. So, a good vector representation of the current discovered graph is required. It is also important to represent nodes such that it encodes the structural information of the node in the context of the current discovered graph. Figure 3.2 shows the Geometric-DQN architecture which takes the state and action representation as input and outputs the $Q(s, a)$ values. The details about state and action representation are provided below.

- **State representation:** The state is the current graph. and the Geometric-DQN architecture uses Graph Convolutional Networks to generate graph embeddings.

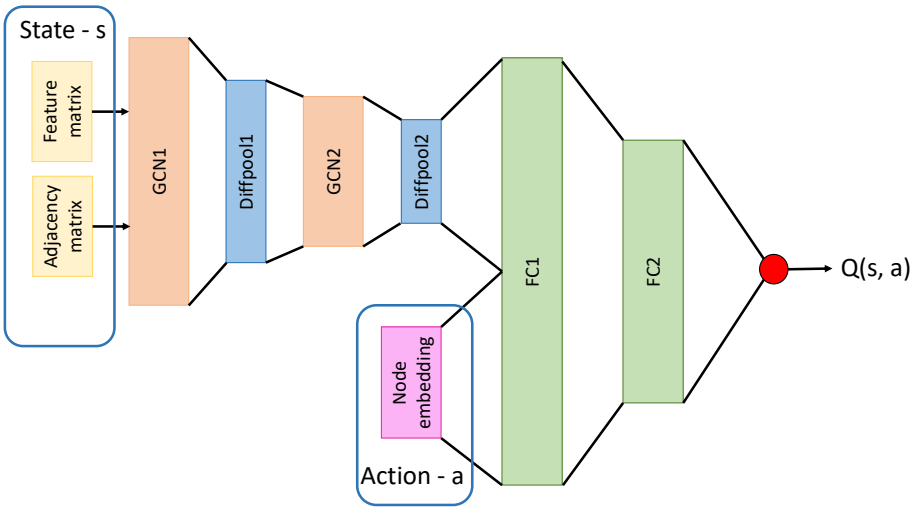


Figure 3.2: Geometric-DQN Architecture. Figure taken from Kamarthi et al. [39]. FC1/FC2 - fully connected layers.

The graph G_t is represented with the adjacency matrix $A_t \in \mathbb{R}^{|V_t| \times |V_t|}$ and a node feature matrix $F_t^{(k-1)} \in \mathbb{R}^{|V_t| \times d}$ in layer $k-1$ where d is the number of features. The node features in the input layer of graph convolution network, i.e., F_t^0 are generated by using random-walk based Deepwalk embeddings[1] [73].

Now, a Graph Convolutional layer derives node features using a transformation function $F^k = M(A, F^{k-1}; W^k)$, where W^k represent the weights of the k^{th} layer. Using the formulation in Ying et al. [110], the transformation function is given by

$$F_t^{(k)} = ReLU(D^{-\frac{1}{2}} \tilde{A} D^{-\frac{1}{2}} F_t^{(k-1)} W^{(k)})$$

where $\tilde{A}$ means adjacency matrix A_t with added self-connections, i.e., $\tilde{A} = A_t + I_n$ (I_n is the identity matrix). $D = \sum_j \tilde{A}_{ij}$. To better represent the global representation of graph, differential pooling is used which learns hierarchical representations of the graph in an end-to-end differentiable manner by iteratively coarsening the graph, using graph convolutional layer as a building block. The output of graph convolutional network is provided as an input to a

[1]Deepwalk learns node representations that are similar to other nodes that lie within a fixed proximity on multiple random walks.

pooling layer.

- **Actions representation:** DeepWalk node embeddings are also utilized for representing actions. We use ϕ to denote the deepwalk embeddings.

Therefore, if G_t is the current graph (state) and u_t is the current node to be queried (action), we represent state as $S_t = (F_t^0, A_t)$ and action as $\phi(u_t)$ which are input to the network as shown in the Figure 3.2.

3.4 Our Approach - CLAIM

In this section, we present our approach CLAIM - **C**urriculum **L**e**A**rning Policy for **I**nfluence **M**aximization in unknown social networks. We first explain how the problem can be translated into a Goal directed learning problem. The advantage of translating the problem into goal directed learning problem is that it allows us to increase sample efficiency by using the Curriculum guided Hindsight experience replay (CHER) [24]. CHER involves replaying each episode with pseudo goals, so the agent can get multiple experiences in a single environment interaction which results in increasing the sample efficiency.

To use goal directed learning in our setting, we first present our novel heuristic to generate goals and the modifications to the MDP formulation for goal directed learning. After that, we present our algorithm to generate a curriculum learning policy using Hindsight experience replay.

3.4.1 Goal Directed Reinforcement Learning

In the Goal Directed or Goal Conditioned Reinforcement Learning [4, 64], an agent interacts within an environment to learn an optimal policy for reaching a certain goal state or a goal defined by a function on the state space in an initially unknown or only partially known state space. If the agent reaches the goal, then

the reinforcement learning method is terminated, and it solves the goal-directed exploration problem.

In these settings, the reward that the agent gets from the environment is also dependent on the goal that the agent is trying to achieve. A goal-conditioned Q-function $Q(s, a, g)$ [80] learns the expected return for the goal g starting from state s and taking action a. Given a state s, action a, next state s' , goal g and corresponding reward r, one can train an approximate Q-function parameterized by θ by minimizing the following Bellman error:

$$\frac{1}{2}||Q_\theta(s, a, g) - (r + \gamma \cdot \max_{a'} Q_{\theta'}(s', a', g)||^2$$

This loss can be optimized using any standard off-policy reinforcement learning method [64].

Generally, in these goal-directed reinforcement learning problems, a set of goal states or goals defined by a function on the state space is given and the agent needs to reach one of the goal states (goals). But in our setting, we do not have an explicit goal state given. To convert the network discovery and influence maximization problem to a goal directed learning problem, we introduce the notion of goals for our problem. We define the goal as the expected long-term reward, i.e., the expected value of the number of nodes that can be influenced in the network, intuitively, the state which can achieve this goal value serves as our goal state. As we have a limited query budget to discover the network, the goal value will be highly dependent on the initial sub-graph. If we use the same value of goal for each start state, for some start states this common goal value will turn out to be a very loose upper bound (or very loose lower bound). Experimentally, we found that if the goal value is too far from the actual value that can be achieved, it negatively affects the speed of learning. So, we design a heuristic to compute a different goal for each start state.

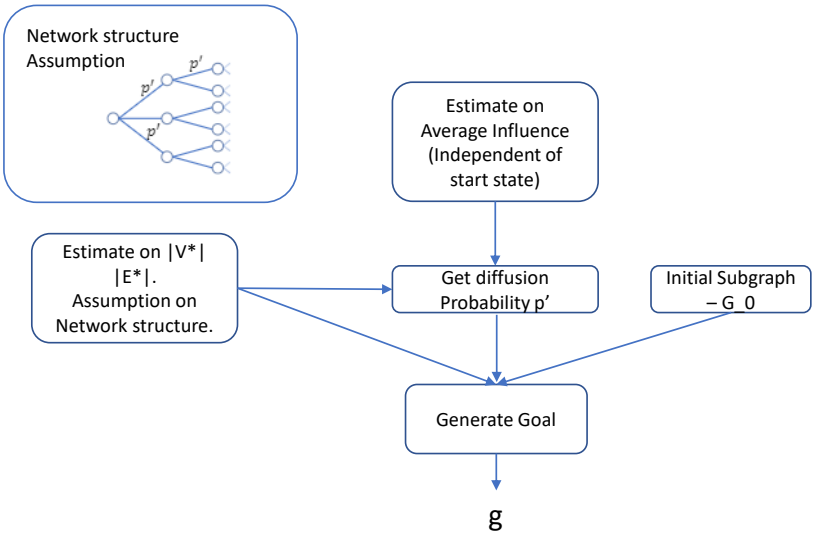


Figure 3.3: Process to generate the goal for each start state

Goal Generation Heuristic

As we need to generate a goal at the start of each episode (i.e., before the agent starts interacting with the environment), we need to compute the goal value without making any queries to the environment. We assume that based on the domain knowledge, agent can get an estimate about the number of nodes ($|\tilde{V}^*|$) and edges ($|\tilde{E}^*|$ in the network and also an estimate about average number of nodes which can be influenced in the network (irrespective of the start state) ($\tilde{I}^*$). We now describe how we use these estimates to design our heuristic to compute the goal value for each start state.

Figure 3.3 represents the steps for our heuristic. As the network is unknown to the algorithm, we assume a network structure and compute the diffusion probability based on the assumed network structure, and estimate the number of nodes, edges, and average influence. By using the computed diffusion probability, given estimates, and assumed network structure, we generate a goal value for a given initial subgraph.

We assume that the network is undirected and uniformly distributed, i.e, each node is connected to $\frac{2*|\tilde{E}^*|}{|\tilde{V}^*|}$ nodes. We also assume a local tree structure as shown

in Figure 3.3 [2] to approximate the actual expected influence within the social network [16, 97]. The root of the tree can be any of the $|S|$ nodes (initially given nodes) and each node will be part of only one of such trees. The influence propagation probability is assumed to be p' and is considered the same for all edges. We now show how the value of p' can be computed based on the network structure assumption and available information.

1. **Computing p':** We find a value of p' such that the expected influence in our tree-structured network is similar to the estimate on the average value of influence $\tilde{I}^*$. To compute the expected influence or expected number of nodes activated in the network, we need to know the number of layers in the tree structure. Therefore, we first compute the number of layers in our assumed tree structure. Let $K_1 = |S| * 2 * \frac{|E^*|}{|V^*|}$, which is the number of nodes at first layer. For subsequent layers, each node will be connected to $2 * \frac{|\tilde{E}^*|}{|\tilde{V}^*|} - 1$ nodes at the layer below it (one edge will be to the node at the above layer). We use r to denote the quantity $2 * \frac{|\tilde{E}^*|}{|\tilde{V}^*|} - 1$. As the total number of nodes in the graph is $|\tilde{V}^*|$, the sum of the number of nodes at all layers should be equal to $|\tilde{V}|^*$. Let L denote the number of layers. Then,

$$|\tilde{V}|^* = |S| + K_1 + K_1 * r + K_1 * r^2 + .. + K_1 * r^{L-1} \tag{3.2}$$

$$\implies \frac{(|\tilde{V}|^* - |S|)}{K_1} = \frac{r^L - 1}{r - 1} \tag{3.3}$$

Solving for L gives $L = \log_r(1 + \frac{|\tilde{V}|^* - |S|) * (r-1)}{K_1})$. Now, we compute the expected number of nodes activated (influenced) in our assumed network with the propagation probability p'. Let J denote the expected number of

[2] These simplified assumptions work well to approximate the influence propagation. We also observe in our experiments that our heuristic outputs a value that is closer to the actual value.

nodes influenced in the network. Then,

$$J = |S| + K_1 * p' + K_1 * r * p'^2 + K_1 * r^2 * p'^3 \tag{3.4}$$
$$+ .. + K_1 * r^{L-1} * p'^L$$
$$\implies \frac{(J - |S|)}{K_1} = \frac{p' * ((p' * r)^L - 1)}{p' * r - 1} \tag{3.5}$$

If our assumed network is similar to the actual network, the value of J should be close to $\tilde{I}^*$, i.e., the average number of nodes influenced in the network. Therefore, to find the value of p', we perform a search in the probability space and use the value of p' which makes J closest to $|\tilde{I}|^*$.

2. **Computing goal value g for a given initial subgraph:** Now, to compute the goal value for a given initial subgraph, we use the p' value computed above. The subgraph is known, i.e., the neighbors of nodes in set S ($N_{G^*}(S)$) are known. Therefore, the number of nodes at the first layer is equal to $N_{G^*}(S)$, i.e., $K_1 = |N_{G^*}(S)|$. For the next layer onwards, we assume a similar tree structure as before with each node connected to $2 * \frac{|\tilde{E}|}{|\tilde{V}|} - 1$ node at the layer below it. Therefore, to compute the goal value, we substitute K_1 as $|N_{G^*}(S)|$ in equations 3.3 and 3.5 to compute the number of layers and influence value. We use the value of p' computed above and solve for J. The J value obtained is the influence value we can achieve for the given subgraph based on the assumptions and available information. We use the value of J as our goal g for the subgraph.

Modifications to the MDP formulation:

The state and action remain the same as before but due to the introduction of goals, the reward function is now parameterized by the goal. Let $R_{t,g}$ denote the reward obtained at timestep t when the goal is g. As we only get episode reward,

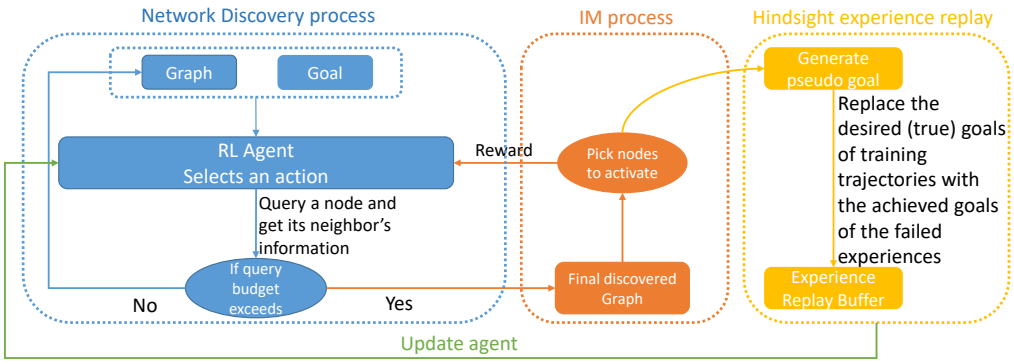


Figure 3.4: Network discovery framework for influence maximization

therefore [3],

$$R_{T-1,g} = \frac{I_{G^*}(O(G_T)) - g}{g} \; and \; R_{t,g} = 0, \forall t \neq T-1 \tag{3.6}$$

3.4.2 Algorithm

In this section, we describe the algorithm used to train the reinforcement learning agent. We use the DQN algorithm and use Curriculum Guided Hindsight Experience Replay to improve the sample efficiency. Algorithm 5 describes the detailed steps. Figure 3.4 provides a visual representation.

We train using multiple training graphs. In each episode, we sample a training graph and then sample an initial set of nodes S. We generate the input state by computing the deepwalk embeddings at each timestep and use $\epsilon-$greedy policy to select the action, i.e., the node to be queried. In step 13, we store the transitions according to standard experience replay where we add the goal as well in the experience buffer.

Steps 14-16 are the ***first set of crucial steps*** to improve the sample efficiency, where as per the Hindsight Experience Replay technique proposed by Andrychowicz et al. [4], we sample pseudo goals and in addition to storing the sample with the actual goal for the episode, we also store each sample by modifying the desired goal (which the agent could not achieve in the failed trajectory) with a pseudo goal g'. The reward with the pseudo goal g' is recomputed as per

[3]Normalizing the reward using the goal stabilizes the learning.

Algorithm 5: Train Network

Input: Train graphs $\mathcal{G} = \{G_1, G_2, ..., G_k\}$, number of episodes N. Query budget T

Initialize DQN Q_θ and target DQN $Q_{\theta'}$ with $\theta = \theta'$ and the Replay Buffer $\mathcal{B}$;

for *episode* $= 1$ *to* N **do**

  $G = sample(\mathcal{G})$, $S = sample(G)$;

  Initialize the subgraph $G_0 = (S \cup N_{G^*}(S), E(S, N_{G^*}(S)))$ and corresponding desired goal g;

  $F_0^0 = DeepWalk(G_0)$ and $S_0 = (F_0^0, A_0)$;

  Get the possible action set $X = N_G(S)$;

  for $t = 0$ *to* $T - 1$ **do**

    With probability ϵ select a random node u_t from X and with probability 1-ϵ select $u_t \leftarrow \max_{u \in X} Q_\theta(S_t, \phi(u), g)$;

    Query node u_t and observe new graph G_{t+1};

    Update the state $F_{t+1}^0 = DeepWalk(G_{t+1})$ and $S_{t+1} = (F_{t+1}^0, A_{t+1})$;

    Update the possible action set X which is the set of nodes not yet queried in G_{t+1};

  for $t = 0$ *to* $T - 1$ **do**

    Store the transition $(S_t, \phi(u_t), R_t, g, S_{t+1}, g)$ in $\mathcal{B}$;

    Sample the additional goals $\mathbb{G}$ for replay ;

    for $g' \in \mathbb{G}$ **do**

      Add the transition $(S_t, \phi(u_t), R_{t,g'}, S_{t+1}, g')$ to replay buffer $\mathcal{B}$;

  for $t = 0$ *to* $T - 1$ **do**

    Sample a minibatch A from the replay buffer $\mathcal{B}$(according to the proximity and diversity scores) ;

    Update the proximity-diversity trade-off parameter $\lambda \leftarrow \gamma \times \lambda$;

    Update Q_θ using the minibatch A;

  Update target network $Q_{\theta'}$ with parameters of Q_θ at regular intervals;

the Equation 3.6.

While there are multiple possible strategies to generate the set of pseudo goals [4], the most common strategy to generate the pseudo goals is to use the goal achieved at the end of the episode. Therefore, in this work, we use g' as $I_{G^*}(O(G_T))$.

Step 18 is the ***second crucial step*** towards improving the sample efficiency where for sampling experiences from the replay buffer, we use a curriculum guided selection process which relies on the goal-proximity and diversity based curiosity [24]. Instead of sampling experiences uniformly, we select a subset of experiences based on the trade-off between goal-proximity and diversity-based curiosity. This plays an important role in guiding the learning process. A large proximity value enforces the training to proceed toward the desired goals, while a large diversity value ensures the exploration of different states and regions in the environment. To sample a subset A of size k for replay from the experience buffer $\mathcal{B}$, the following optimization needs to be solved:

$$\max_{A \subseteq B, |A| \leq k} F(A) = \max_{A \subseteq B, |A| \leq k} (F_{prox}(A) + \lambda F_{div}(A)) \tag{3.7}$$

where B is the uniformly sampled subset of size mk from the buffer $\mathcal{B}$. Let $m = 3$ as Fang et al. [24] does. $F_{prox}(A)$ measures the proximity of the achieved goals g' in A to its desired goal g. The second term $F_{div}(A)$ denotes the diversity of states and regions of the environment in A. The weight λ is used to balance the trade-off between the proximity and the diversity. The trade-off between the two values is balanced such that it enforces a human-like learning strategy, where there is more curiosity in exploration in the earlier stages and later the weight is shifted to the goal-proximity.

In our work, we define proximity as the similarity between goal values and diversity based on the distance between visited states. This is because even though goal values (influence achieved) can be different, the states visited can still be very similar to each other. Formally, to define proximity, we use the

difference between achieved goal g'_i and the desired goal g_i as distance and subtract it from a large constant to get the similarity, i.e.,

$$F_{prox}(A) = \sum_{i \in A} (c - |(g'_i - g_i)|) \tag{3.8}$$

where c is a large number to guarantee $(c - |(g'_i - g_i)|) \geq 0$ for all possible g_i, and g_i is the goal corresponding to experience i in set A. For defining diversity, we need to compute similarity between states, and the Geometric DQN architecture allows us to easily compute this value. Diversity is defined as follows

$$F_{div}(A) = \sum_{j \in B} \max_{i \in A} \{0, sim(s_i^{emb}, s_j^{emb})\} \tag{3.9}$$

where we use s_i^{emb} to denote the embedding vector of the state (representation of the graph in the embedding space) corresponding to the experience i. The embedding vector of the state is the output of the graph convolution and pooling layer (input to FC1) in Figure 3.2. $sim(s_i^{emb}, s_j^{emb})$ denotes the similarity score between the vector representations and is computed by taking the dot product of the vectors.

This definition of diversity is inspired by the facility location function [20, 52] which was also used by Fang et al. [24]. Intuitively, this diversity term is measuring how well the selected experiences in set A can represent the experiences from B. A large diversity score $F_{div}(A)$ indicates that every achieved state in B can find a sufficiently similar state in A. A diverse subset is more informative and thus helps in improving learning.

It has been shown that $F(A)$ defined in equation 3.7 is a monotone non-decreasing submodular function [4] Therefore, even though exactly solving equation 3.7 is NP-hard, due to the submodularity property, a greedy algorithm can provide a solution with an approximation factor $1 - \frac{1}{e}$ [65]. The greedy algorithm picks top k experiences from the buffered experiences B. It will start by taking

[4]It is a weighted sum of a non-negative modular function ($F_{prox}(A)$) and a submodular function ($F_{div}(A)$). Please refer to the paper by Fang et al. [24] for details.

A as an empty set and at each step, it will add the experience i which maximizes the marginal gain. We denote the marginal gain for experience i by $F(i|A)$ and it is given by

$$F(i|A) = F(i \cup A) - F(A) \tag{3.10}$$

Therefore, by using equations 3.7-3.9, we get

$$\begin{aligned} F(i|A) = (c - (|g_i' - g_i|)) + \lambda \sum_{j \in B} \max\{0, (sim(s_i^{emb}, s_j^{emb}) \\ - \max_{l \in A}(sim(s_l^{emb}, s_j^{emb})))\} \end{aligned} \tag{3.11}$$

At the end of each episode, the trade-off coefficient λ is multiplied by a discount rate γ, which produces the continuous shifting of weights from diversity to proximity score. Then effect of $F_{div}(A)$ will go to zero when $\lambda \to 0$.

3.5 Experiments

The goal of the experiment section is to evaluate the performance of our approach CLAIM in comparison to the following state-of-the-art approaches:

- **Random** - At each step, it randomly queries a node from available unqueried nodes.
- **CHANGE** Algorithm by Wilder et al. [105]
- **Geometric-DQN (Baseline)** Algorithm by Kamarthi et al. [39]

Category	Train networks	Test networks
Retweet	copen, occupy	israel, damascus, obama, assad
Animal	plj, rob	bhp, kcs
FSW	zone 1	zone 2, zone 3

Table 3.2: Train and test networks

Dataset: The first network is the Retweet Network from twitter [78]. The second network is Animal Interaction networks which are a set of contact networks of field voles (Microtus agrestis) inferred from mark-recapture data collected

Network category	Retweet networks				Animals networks		FSW networks	
Test networks	israel	damascus$**$	obama$**$	assad$*$	bhp$***$	kcs$**$	zone 2$*$	zone 3
OPT value	113.9	195.8	154.7	134.2	111.9	113.4	20.98	16.40
Random value	31.17	84.71	40.81	69.44	36.80	54.39	13.26	12.31
CHANGE value	32.42	92.41	48.61	69.77	35.87	54.52	12.60	10.51
Geometric DQN	37.33	105.2	52.01	75.12	40.12	60.81	13.65	12.35
CLAIM approach	**38.55**	**113.1**	**54.67**	**77.49**	**42.25**	**64.58**	**13.94**	**12.48**
Improve percent	3.27%	7.51%	5.11%	3.15%	5.31%	6.20%	2.12%	1.05%

Table 3.3: Comparison of influence score of our proposed approach and existing approaches for each test network. For each network, a paired t-test is performed and $*$ indicates statistical significance of better performance at $\alpha = 0.05$ level, $**$ at $\alpha = 0.01$ level, and $***$ at $\alpha = 0.001$ level.

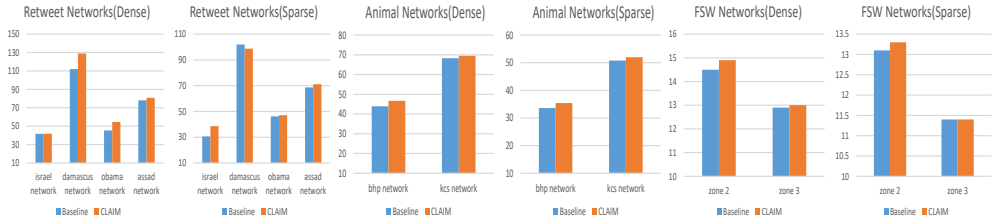


Figure 3.5: Comparison of performance of our approach and baseline approach in dense and sparse network environment.

over 7 years and from four sites [21]. The third network is a real-world physical network between Female Sex Workers (FSW) in a large Asian city divided into multiple zones. This is a confidential dataset physically collected by a non-profit by surveying different female sex workers recently. The goal in FSW networks is to discover the network and select a subset of FSW from the discovered network to be enrolled in the HIV prevention programs. The enrolled FSWs should be such that they can pass on the information (influence) maximum FSWs in the complete network. For each family of networks, we divide them into train and test data as shown in Table 3.2.

Experimental Settings: Our experimental settings are similar to the settings used in Kamarthi et al. [39]. There are 5 nodes in the set S. All nodes in S and their neighbors are known. We have further budget of $T = 5$ queries to discover the network. After getting the final subgraph G_T, we pick 10 nodes to activate using greedy influence maximization algorithm. We use $p = 0.1$ as the diffusion probability for all the edges.

3.5.1 Results

To demonstrate sample efficiency, we measure the performance of our approach against past approaches by the average number of nodes influenced over 100 runs under a fixed number of queries. Here are the key observations:

- **Average influence value:** Table 3.3 shows the comparison of the number of nodes influenced by different algorithms. Each algorithm selects the set of nodes to activate from the discovered graph. As shown in the table, our approach consistently outperforms all existing approaches across different networks. CLAIM learns a better policy in the same number of episodes and hence more sample-efficient. We would like to highlight here that even a small consistent improvement in these settings is very important as it can ensure more life safety (as an example by educating people about HIV prevention).

- **Effect of density of the initial subgraph:** The number of nodes that can be influenced in the graph is highly dependent on the position of the initial subgraph in the whole social network. Therefore, we also test the performance of CLAIM against the baseline approach on the *dense* and *sparse* initial subgraphs (we identify the initial subgraph as dense or sparse based on the ratio of $\frac{|S \cup N_{G^*}(S)|}{|S|}$). We compare the average influence values as shown in Figure 3.5. CLAIM outperforms the baseline in most of the cases, except the sparse case in the damascus network. The reason for this may be that the damascus network is an extremely sparse network, and it has some specific structure property that leads to this result.

- **Ablation Study:** We also present the detailed results for our ablation study over all datasets in Table 3.4. We observe the effect of adding each additional component in CLAIM one by one. First, we add only the goal as a feature to the baseline model. Next, we add the Hindsight

Network category	Retweet				Animals		FSW	
Networks	israel	damascus	obama	assad	bhp	kcs	zone 2	zone 3
Baseline	37.33	105.2	52.01	75.12	40.12	60.81	13.65	12.35
Geometric DQN	36.24	110.5	51.61	73.68	41.59	62.80	13.79	12.32
HER	37.79	109.4	53.51	76.32	42.00	**64.64**	13.81	**12.48**
CLAIM	**38.55**	**113.1**	**54.67**	**77.49**	**42.25**	**64.58**	**13.94**	**12.48**

Table 3.4: Ablation study for each test network

Networks\Method	Geometric DQN	CLAIM
israel	37.11 ± 0.42	38.32 ± 0.32
damascus	104.2 ± 5.22	112.8 ± 4.01
obama	52.15 ± 1.05	54.78 ± 0.78
assad	74.53 ± 1.71	77.45 ± 1.02
bhp	40.24 ± 1.25	42.37 ± 0.81
kcs	59.67 ± 1.91	63.21 ± 1.43
zone 2	13.62 ± 0.01	13.94 ± 0.00
zone 3	12.23 ± 0.01	12.45 ± 0.00

Table 3.5: Stability of our approach compared to the baseline on different sets of 100 runs

Experience Replay and finally, we add the curriculum-guided selection of experiences for replay. These results indicate that a single component can not guarantee a better result for all networks, and we need all three components to improve the performance across multiple datasets.

- **Stability check:** We check the stability of CLAIM by comparing the performance of models trained using different random seeds. We train three models for both baseline and CLAIM. Table 3.5 shows the mean and deviation of influence value for different networks. CLAIM not only achieves a high mean it also provides a low deviation reflecting the stability of the approach.

- **Property insight:** We also explore the properties of the selected nodes to further investigate why CLAIM performs better. We look at *degree centrality measures, closeness centrality measures,* and *betweenness centrality measures* of the nodes queried in the underlying graph. In particular, we conduct experiment using *assad*, a retweet network with sparsely interconnected star-graph. As we can see in Figure 3.6, compared to the

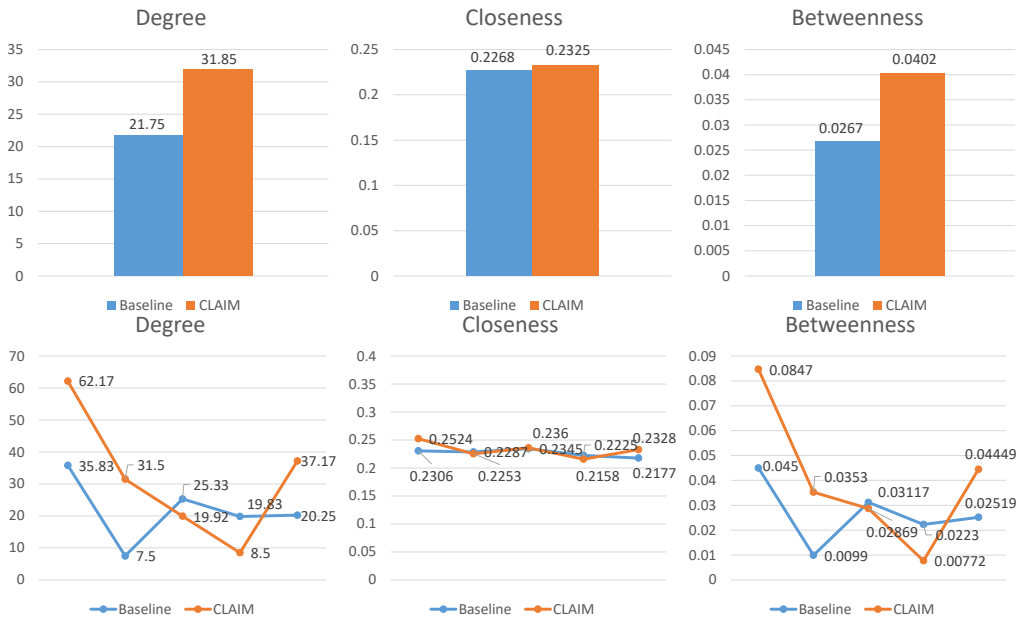


Figure 3.6: Top Graph - Average degree, closeness and betweenness centrality of nodes queried in the full graph by CLAIM and baseline. Bottom Graph - Variation of these properties across timesteps.

baseline approach, on average, CLAIM can recognize nodes with a higher degree, closeness, and betweenness centrality. As a result, CLAIM is able to discover a bigger network. The higher degree centrality, higher closeness centrality, and higher betweenness also show that CLAIM can explore nodes that play an important role in the influence maximization problem. Besides, these values are large at the beginning which means that CLAIM tends to explore a bigger graph first, and then leverage the available information with the learned graph to find complex higher-order patterns in the graphs that enable it to find key nodes during the intermediate timesteps, and finally utilize all the information to expand the discovered graph at the end.

3.6 Discussion

We provide a justification for the choices made in the work.

- **Network structure assumption for goal generation:** As we have no prior

information about the networks except the initial nodes, we need to make some assumptions to compute the goal value. We make the assumption of the network being uniformly distributed and use a tree structure to approximate the information propagation as most networks observed for these problems have similar structures or can be converted in these forms with minimal loss of information.

- **Goodness of heuristic used for goal generation:** Experimentally, we observe that the goal value computed by our heuristic is closer to the actual value. For example, for the training network *copen*, the achieved influence value by the model after training is at most within 20% of the goal value computed using a heuristic. In addition, most of the achieved influence values are much closer and are smaller than the computed goal. In the future, we will investigate different ways to generate a goal with a proven upper bound.

3.7 Conclusion

In this chapter, we propose a sample efficient reinforcement learning approach for network discovery and influence maximization problems with limited budget. Through detailed experiments, we show that our approach outperforms existing approaches on real world datasets. In the next chapter we discuss how to train a generally capable agent that can perform well in complex environments. This can be applied in the real world by designing the most suitable tasks to train non-expert humans so that they can achieve the best zero-shot transfer performance under a limited training horizon and resources.

Chapter 4

Training Robuts Agent with Limited Resources

The use of reinforcement learning (RL) in training agents to possess general capabilities represents an emerging research area with potential applications for training human learners. A promising approach within this field is Unsupervised Environment Design (UED), which automatically generates a curriculum of training environments. This method optimizes the learning process by identifying environments that strike an ideal balance between challenge and the agent's or human learner's current capabilities. Agents trained in these environments are able to develop general capabilities, i.e., achieving good zero-shot transfer performance. However, existing UED approaches focus primarily on the random generation of environments for open-ended agent training. The notion of open-ended training requires training an agent across hundreds of thousands of randomly generated environments for hundreds of millions of time steps. This is impractical in real-world scenarios where resources are limited, such as the number of environments that can be generated.

In this chapter, we introduce a novel hierarchical MDP framework for environment design under resource constraints. It features an upper-level RL teacher agent tasked with generating suitable training environments for a lower-level

student agent. The RL teacher can leverage previously discovered environment structures and generate environments at the frontier of the student's capabilities by observing the student policy's representation. We incorporate a fairness reward to ensure the generated environments are suitable and equitable. Moreover, to reduce the time-consuming collection of experiences for the upper-level teacher, we utilize recent advances in generative modeling to synthesize a trajectory dataset to train the teacher agent. Our proposed method significantly reduces the resource-intensive interactions between agents and environments and empirical experiments across various domains demonstrate the effectiveness of our approach. Furthermore, our framework has the potential to benefit the training of non-professional humans by facilitating their learning process within suitably generated environments. This work is presented in:

- Li, Dexun, and Pradeep Varakantham. "A Hierarchical Approach to Environment Design with Generative Trajectory Modeling." arXiv e-prints (2023): arXiv-2310.

4.1 Introduction

The advances of reinforcement learning (RL [90]) have promoted research into the problem of training autonomous agents that are capable of accomplishing complex tasks. One interesting, yet underexplored, area is training agents to perform well in unseen environments, a concept referred to as zero-shot transfer performance. To this end, Unsupervised Environment Design (UED [22, 93]) has emerged as a promising paradigm to address this problem. The objective of UED is to automatically generate environments in a curriculum-based manner, and training agents in these sequentially generated environments can equip agents with general capabilities, enabling agents to learn robust and adaptive behaviors that can be transferred to new scenarios without explicit exposure during training.

Existing approaches in UED primarily focus on building an adaptive cur-

riculum for the environment generation process to train the generally capable agent. Dennis et al. [22] formalize the problem of finding adaptive curricula through a game involving an adversarial environment generator (teacher agent), an antagonist agent (expert agent), and the protagonist agent (student agent). The RL-based teacher is designed to generate environments that maximize regret, defined as the difference between the protagonist and antagonist agent's expected rewards. They show that these agents will reach a Nash Equilibrium where the student agent learns the minimax regret policy. However, since the teacher agent adapts solely based on the regret feedback, it is inherently difficult to adapt to student policy changes. Meanwhile, training such an RL-based teacher remains a challenge because of the high computational cost of training an expert antagonist agent for each environment.

In contrast, domain randomization [94] based approaches circumvent the overhead of developing an RL teacher by training the agent in randomly generated environments, resulting in good empirical performances. Building upon this, Jiang et al. [36] introduce an emergent curriculum by sampling randomly generated environments with high regret value [1] to train the agent. Parker-Holder et al. [70] then propose the adaptive curricula by manually designing a principled, regret-based curriculum, which involves generating random environments with increasing complexity. Li et al. [51] incorporate diversity measurement into the environment generation process to ensure that the agent is exposed to diverse environments. While these domain randomization-based algorithms have demonstrated good zero-shot transfer performance, they face limitations in efficiently exploring large environment design spaces and exploiting the inherent structure of previously discovered environments. Moreover, existing UED approaches typically rely on open-ended learning, necessitating a long training horizon, which is unrealistic in the real world due to resource constraints. Our goal is to develop a teacher policy capable of generating environments that are perfectly

[1] They approximate the regret value by the Generalized Advantage Estimate [81].

matched to the current skill levels of student agents, thereby allowing students to achieve optimal general capability within a strict budget for the number of environments generated and within a shorter training time horizon.

In this chapter, we address these challenges by introducing a novel, adaptive environment design framework. The core idea involves using a hierarchical Markov Decision Process (MDP) to simultaneously formulate the evolution of an upper-level MDP teacher agent, tasked with generating suitable environments to train the lower-level MDP student agent to achieve general capabilities. To accurately guide the generation of environments at the frontier of the student agent's current capabilities, we propose approximating the student agent's policy/capability by its performances across a set of diverse evaluation environments, which are used as the observations for the teacher agent. These transitions in the teacher's observations represent the trajectories of the student agent's capability after a complete training cycle in the generated environment. However, collecting experience for the upper-level teacher agent is slow and resource-intensive, since each upper-level MDP transition evolves a complete training cycle of the student agent on the generated environment. To accelerate the collection of upper-level MDP experiences, we utilize advances in diffusion models that can generate new data points capturing complex distribution properties, such as skewness and multi-modality, exhibited in the collected dataset [79]. Specifically, we employ diffusion probabilistic model [86, 31] to learn the evolution trajectory of student policy/capability and generate synthetic experiences to enhance the training efficiency of the teacher agent. Our method, called *S*ynthetically-enhanced *H*ierarchical *E*nvironment *D*esign (*SHED*), automatically generates increasingly complex environments suited to the current capabilities of student agents.

In summary, we make the following contributions:

- We develop a novel hierarchical MDP framework for UED that introduces a straightforward method to represent the current capability level of the student

agent.

- We introduce *SHED*, which utilizes diffusion-based techniques to generate synthetic experiences. This method can accelerate the training of the off-policy teacher agent.
- We demonstrate that our method outperforms existing UED approaches (i.e., achieving a better general capability under resource constraints) in different task domains.

4.2 Preliminaries

In this section, we provide an overview of two main research areas upon which our work is based.

4.2.1 Unsupervised Environment Design

The objective of UED is to generate a sequence of environments that effectively train the student agent to achieve a general capability. Dennis et al. [22] first model UED with an Underspecified Partially Observable Markov Decision Process (UPOMDP), which is a tuple

$$\mathcal{M} =< A, O, \Theta, S^{\mathcal{M}}, \mathcal{P}^{\mathcal{M}}, \mathcal{I}^{\mathcal{M}}, \mathcal{R}^{\mathcal{M}}, \gamma >$$

The UPOMDP has a set Θ representing the free parameters of the environments, which are determined by the teacher agent and can be distinct to generate the next new environment. Further, these parameters are incorporated into the environment-dependent transition function $\mathcal{P}^{\mathcal{M}} : S \times A \times \Theta \to S$. Here A represents the set of actions, S is the set of states. Similarly, $\mathcal{I}^{\mathcal{M}} : S \to O$ is the environment-dependent observation function, $\mathcal{R}^{\mathcal{M}}$ is the reward function, and γ is the discount factor. Specifically, given the environment parameters

$\vec{\theta} \in \Theta$, we denote the corresponding environment instance as $\mathcal{M}_{\vec{\theta}}$. The student policy π is trained to maximize the cumulative rewards $V^{\mathcal{M}_{\vec{\theta}}}(\pi) = \sum_{t=0}^{T} \gamma^t r_t$ in the given environment $\mathcal{M}_{\vec{\theta}}$ under a time horizon T, and r_t are the collected rewards in $\mathcal{M}_{\vec{\theta}}$. Existing works on UED consist of two main strands: the RL-based environment generation approach and the domain randomization-based environment generation approach.

The RL-based generation approach was first formalized by Dennis et al. [22] as a self-supervised RL paradigm for generating environments. This approach involves co-evolving an environment generator policy (teacher) with an agent policy π (student), where the teacher's role is to generate environment instances that best support the student agent's continual learning. The teacher is trained to produce challenging yet solvable environments that maximize the regret measure, which is defined as the performance difference between the current student agent and a well-trained expert agent π^* within the current environment.

$$Regret^{\mathcal{M}_{\vec{\theta}}}(\pi, \pi^*) = V^{\mathcal{M}_{\vec{\theta}}}(\pi^*) - V^{\mathcal{M}_{\vec{\theta}}}(\pi)$$

The domain randomization-based generation approach, on the other hand, involves randomly generating environments. Jiang et al. [36] propose to collect encountered environments with high learning potentials, which are approximated by the Generalized Advantage Estimation (GAE) [81], and then the student agent can selectively train in these environments, resulting in an emergent curriculum of increasing difficulty. Additionally, Parker-Holder et al. [70] adopt a different strategy by using predetermined starting points for the environment generation process and gradually increasing complexity. They manually divide the environment design space into different difficulty levels and employ human-defined edits to generate similar environments with high learning potentials. Their algorithm, ACCEL, is currently the state-of-the-art (SOTA) in the field, and we use an edited version of ACCEL as a baseline in our experiments.

4.2.2 Diffusion Probabilistic Models

Diffusion models [86, 31] are a specific type of generative model that learns the data distribution. Recent advances in diffusion-based models, including Langevin dynamics and score-based generative models, have shown promising results in various applications, such as time series forecasting [91], robust learning [68], anomaly detection [106] as well as synthesizing high-quality images from text descriptions [67, 79]. These models can be trained using standard optimization techniques, such as stochastic gradient descent, making them highly scalable and easy to implement.

In a diffusion probabilistic model, we assume a d-dimensional random variable $x_0 \in \mathbb{R}^d$ with an unknown distribution $q(x_0)$. Diffusion Probabilistic model involves two Markov chains: a predefined forward chain $q(x_k|x_{k-1})$ that perturbs data to noise, and a trainable reverse chain $p_\phi(x_{k-1}|x_k)$ that converts noise back to data. The forward chain is typically designed to transform any data distribution into a simple prior distribution (e.g., standard Gaussian) by considering perturb data with Gaussian noise of zero mean and a fixed variance schedule $\{\beta_k\}_{k=1}^K$ for K steps:

$$\begin{aligned} q(x_k|x_{k-1}) &= \mathcal{N}(x_k; \sqrt{1-\beta_k}x_{k-1}, \beta_t\mathbf{I}) \\ q(x_{1:K}|x_0) &= \Pi_{k=1}^K q(x_k|x_{k-1}), \end{aligned} \tag{4.1}$$

where $k \in \{1, \ldots, K\}$, and $0 < \beta_{1:K} < 1$ denote the noise scale scheduling. As $K \to \infty$, x_K will converge to isometric Gaussian noise: $x_K \to \mathcal{N}(0, \mathbf{I})$. According to the rule of the sum of normally distributed random variables, the choice of Gaussian noise provides a closed-form solution to generate arbitrary time-step x_k through:

$$x_k = \sqrt{\bar{\alpha}_k}x_0 + \sqrt{1-\bar{\alpha}_k}\epsilon, \quad \text{where} \quad \epsilon \sim \mathcal{N}(0, \mathbf{I}). \tag{4.2}$$

Here $\alpha_k = 1 - \beta_k$ and $\bar{\alpha}_k = \prod_{s=1}^k \alpha_s$. The reverse chain $p_\phi(x_{k-1}|x_k)$ reverses the forward process by learning transition kernels parameterized by deep neu-

ral networks. Specifically, considering the Markov chain parameterized by ϕ, denoising arbitrary Gaussian noise into clean data samples can be written as:

$$p_\phi(x_{k-1}|x_k) = \mathcal{N}(x_{k-1}; \mu_\phi(x_k, k), \Sigma_\phi(x_k, k)) \tag{4.3}$$

It uses the Gaussian form $p_\phi(x_{k-1}|x_k)$ because the reverse process has the identical function form as the forward process when β_t is small [86]. Ho et al. [31] consider the following parameterization of $p_\phi(x_{k-1}|x_k)$:

$$\begin{aligned} \mu_\phi(x_k, k) &= \frac{1}{\alpha_k}\left(x_k - \frac{\beta_k}{\sqrt{1-\alpha_k}}\epsilon_\phi(x_k, k)\right) \\ \Sigma_\phi(x_k, k) &= \tilde{\beta}_k^{1/2} \quad \text{where } \tilde{\beta}_k = \begin{cases} \frac{1-\alpha_{k-1}}{1-\alpha_k}\beta_k & k > 1 \\ \beta_1 & k = 1 \end{cases} \end{aligned} \tag{4.4}$$

where ϵ_ϕ is a trainable function to predict the noise vector ϵ from x_k. Ho et al. [31] show that training the reverse chain to maximize the log-likelihood $\int q(x_0) \log p_\phi(x_0) dx_0$ is equivalent to minimizing re-weighted evidence lower bound (ELBO) that fits the noise. They derive the final simplified optimization objective:

$$\mathcal{L}(\phi) = \mathbb{E}_{x_0,k,\epsilon}\left[\|\epsilon - \epsilon_\phi(\sqrt{\bar{\alpha}_k}x_0 + \sqrt{1-\bar{\alpha}_k}\epsilon, k)\|^2\right]. \tag{4.5}$$

Once the model is trained, new data points can be subsequently generated by first sampling a random vector from the prior distribution, followed by ancestral sampling through the reverse Markov chain in Equation 4.3.

4.3 Approach

In this section, we formally describe our method, *S*ynthetically-enhanced *H*ierarchical *E*nvironment *D*esign (*SHED*), which is a novel framework for UED under resource constraints. The *SHED* incorporates two key components that differentiate it from existing UED approaches:

- A hierarchical MDP framework to generate suitable environments,

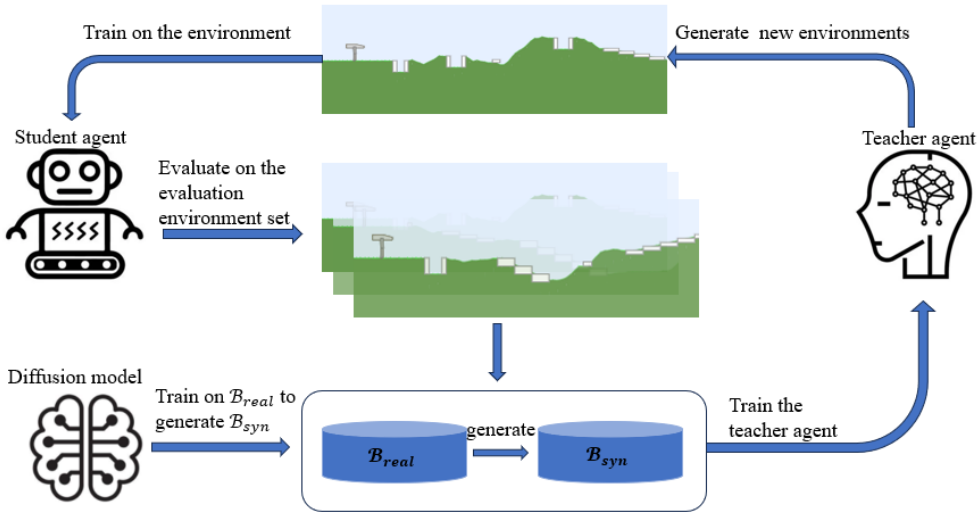


Figure 4.1: The overall framework of *SHED*.

- A generative model to generate the synthetic trajectories.

SHED uses a hierarchical MDP framework where an RL teacher leverages the observed student's policy representation to generate environments at the student's capabilities frontier. Such targeted environment generation process enhances the student's general capability by utilizing the underlying structure of previously discovered environments, rather than relying on the open-ended random generation. Besides, *SHED* leverages advances in generative models to generate synthetic trajectories that can be used to train the off-policy teacher agent, which significantly reduces the costly interactions between the agents and the environments. The overall framework is shown in Figure 4.1, and the pseudo-code is provided in Algorithm 6.

4.3.1 Hierarchical Environment Design

The objective is to generate a limited number of environments that are designed to enhance the general capability of the student agent. Inspired by the principles of PAIRED [22], we adopt an RL-based approach for the environment generation process. To better generate suitable environments tailored to the current student skill level, *SHED* uses the hierarchical MDP framework, consisting of an upper-

Algorithm 6: *SHED*

Input: real data ratio $\psi \in [0, 1]$, evaluate environment set $\theta^{\mathbf{eval}}$, reward function R;

Initialize: diffusion model D, teacher policy Λ, real and synthetic replay buffer $\mathcal{B}_{\mathbf{real}}, \mathcal{B}_{\mathbf{syn}} = \emptyset$;

for *episode* $ep = 1, \ldots, K$ **do**

  Initialize student policy π;

  Evaluate π on $\theta^{\mathbf{eval}}$ and get state $s^u = p(\pi)$;

  for *Budget* $t = 1, \ldots, T$ **do**

    generate $\vec{\theta} \sim \Lambda$, and create $\mathcal{M}_{\vec{\theta}}(\pi)$;

    generate $\vec{\theta} \sim \Lambda$, and create $\mathcal{M}_{\vec{\theta}}(\pi)$;

    train π on $\mathcal{M}_{\vec{\theta}}$ to maximize $V^{\vec{\theta}}(\pi)$;

    evaluate π on $\theta^{\mathbf{eval}}$ and get next state s';

    compute teacher's reward r_t according to R;

    add experience $(s_t^u, \vec{\theta}, r_t^u, s_t^{u,\prime})$ to $\mathcal{B}_{real}$;

    train D with samples from $\mathcal{B}_{\mathbf{real}}$;

    generate synthetic experiences from D and add them to $\mathcal{B}_{\mathbf{syn}}$;

    train Λ on samples from $\mathcal{B}_{\mathbf{real}} \bigcup \mathcal{B}_{\mathbf{syn}}$ mixed with ratio ψ;

    set $s = s'$;

Output: Λ, π, D

level RL teacher policy Λ and a lower-level student policy π. Specifically, the teacher policy, $\Lambda : \Pi \to \Theta$, maps from the space of all potential student policies Π to the space of environment parameters Θ. Existing RL-based methods (e.g., PARIED) rely solely on regret feedback and fail to effectively capture the nuances of the student policy. To address this challenge, *SHED* enhances understanding by encoding the student policy π into a vector that serves as the state abstraction for teacher Λ. Rather than compressing the knowledge in the student policy network, we approximate the embedding of the student policy π by assessing performance across a set of diverse evaluation environments. This performance vector, denoted as $p(\pi)$, gives us a practical estimate of the student's current general capabilities, enabling the teacher to customize the next training environments accordingly. In our hierarchical framework, the environment generation process is governed by discrete-time dynamics. We delve into the specifics below.

Upper-level teacher MDP. The upper-level teacher operates at a coarser layer

UED Approaches	Teacher Policy	Decision Rule
Domain Randomization [94]	$\Lambda(\pi) = U(\Theta)$	Randomly sample
PARIED [22]	$\Lambda(\pi) = \{\bar{\theta}_\pi : \frac{c_\pi}{v_\pi}, \tilde{D}_\pi : \text{otherwise}\}$	Minimax Regret
SHED (ours)	$\Lambda(\pi) = \arg\max_{\vec{\theta} \in \Theta} Q_\pi(s = \pi, a = \vec{\theta})$	Maximize cumulative reward

Table 4.1: The teacher policies corresponding to the three approaches for UED. $U(\Theta)$ is a uniform distribution over environment parameter space, $\tilde{D}_\pi$ is a baseline distribution, $\bar{\theta}_\pi$ is the trajectory which maximizes regret of π, and v_π is the value above the baseline distribution that π achieves on that trajectory, c_π is the negative of the worst-case regret of π. Details are described in PAIRED [22].

of student policy abstraction and generates environments to train the lower-level student agent. This process can be formally modeled as an MDP by the tuple $< S^u, A^u, P^u, R^u, \gamma^u >$:

- S^u represents the upper-level state space. Typically, $s^u = p(\pi) = [p_1, \ldots, p_m]$ denotes the student performance vector across m diverse evaluation environments. This vector serves as the representation of the student policy π and is observed by the teacher.
- A^u is the upper-level action space. The teacher observes the abstraction of the student policy, s^u and produces an upper-level action a^u which is the environment parameters $\vec{\theta}$. $\vec{\theta}\,(a^u)$ is then used to generate specific environment instances $\mathcal{M}_{\vec{\theta}}$. Thus the upper-level action space A^u is the environment parameter space Θ.
- P^u denotes the action-dependent transition dynamics of the upper-level state. The general capability of the student policy evolves due to training the student agent on the generated environments.
- R^u provides the upper-level reward to the teacher at the end of training the student on the generated environment. The design of R^u will be discussed in Section 4.3.3.

As shown in Figure 4.2, given the student policy π, the teacher Λ first observes the representation of the student policy, $s^u = [p_1, \ldots, p_m]$. Then teacher produces an upper-level action a^u which corresponds to the environment parameters. These environment parameters are subsequently used to generate specific environment

instances. The lower-level student policy π will be trained on the generated environments for C training steps. The upper-level teacher collects and stores the student policy evolution transition $(s^u, a^u, r^u, s^{u,\prime})$ every C times steps for off-policy training. The teacher agent is trained to maximize the cumulative reward giving the budget for the number of generated environments. The choice of the evaluation environments will be discussed in Section 4.3.3.

Lower-level student MDP. The generated environment is fully specified for the student, characterized by a Partially Observable Markov Decision Process (POMDP), which is defined by a tuple $\mathcal{M}_{\vec{\theta}} =< A, O, S^{\vec{\theta}}, \mathcal{P}^{\vec{\theta}}, \mathcal{I}^{\vec{\theta}}, \mathcal{R}^{\vec{\theta}}, \gamma >$, where A represents the set of actions, O is the set of observations, $S^{\vec{\theta}}$ is the set of states determined by the environment parameters $\vec{\theta}$, similarly, $\mathcal{P}^{\vec{\theta}}$ is the environment-dependent transition function, and $\mathcal{I}^{\vec{\theta}} : \vec{\theta} \rightarrow O$ is the environment-dependent observation function, $\mathcal{R}^{\vec{\theta}}$ is the reward function, and γ is the discount factor. At each time step t, the environment produces a state observation $s_t \in S^{\vec{\theta}}$, the student agent samples the action $a_t \sim A$ and interacts with environment $\vec{\theta}$. The environment yields a reward r_t according to the reward function $\mathcal{R}^{\vec{\theta}}$. The student agent is trained to maximize their cumulative reward $V^{\vec{\theta}}(\pi) = \sum_{t=0}^{C} \gamma^t r_t$ for the current environment under a finite time horizon C. The student agent will learn a good general capability from training on a sequence of generated environments.

The hierarchical framework enables the teacher agent to systematically measure and enhance the general capability of the student agent and to adapt the training process accordingly. However, it's worth noting that collecting student policy evolution trajectories $(s^u, a^u, r^u, s^{u,\prime})$ to train the teacher agent is notably slow and resource-intensive, since each transition in the upper-level teacher MDP encompasses a training horizon of C timesteps for the student in the generated environment. Thus, it is essential to reduce the need for costly collection of upper-level teacher experiences.

This hierarchical approach enables the teacher agent to systematically mea-

sure and enhance the performance of the student agent across various environments and to adapt the training process accordingly. However, it's worth noting that collecting student policy evolution trajectories $(s^u, a^u, r^u, s^{up,\prime})$ to train the teacher agent is notably slow and resource-intensive, since each transition in the upper-level teacher MDP encompasses a training horizon of C timesteps for the student in the generated environment. Figure 4.2 illustrates the environment generation process. After finishing training the student agent π on the previously generated environments, the teacher agent will first evaluate π across a set of diverse evaluation environments set to obtain the performance vector $p(\pi)$. This performance vector represents the approximation of the π and serves as the current state s^u for the teacher agent. Then the teacher will generate an action a^u, representing the parameters for the next environment in which the student agent will train for C time. Upon completion of these time steps, the teacher agent will evaluate the updated student agent π' in the same evaluation environment set to get the updated performance vector $p(\pi')$, which is the next state $s^{u,\prime}$. Across an entire episode, given the constraints of limited resources, the teacher agent can generate T different environments. This translates to the teacher agent gathering T experiences and the student agent accruing a total of $T \times C$ experiences in one episode. Overall, the collection of the teacher agent experience is much more computation-intensive and time-consuming than the collection of the student agent experience.

In the following section, we will formally introduce a generative model designed to ease the collection of upper-level MDP experience. This will allow us to upsample the teacher-agent experiences to alleviate such problems and train our teacher policy more efficiently.

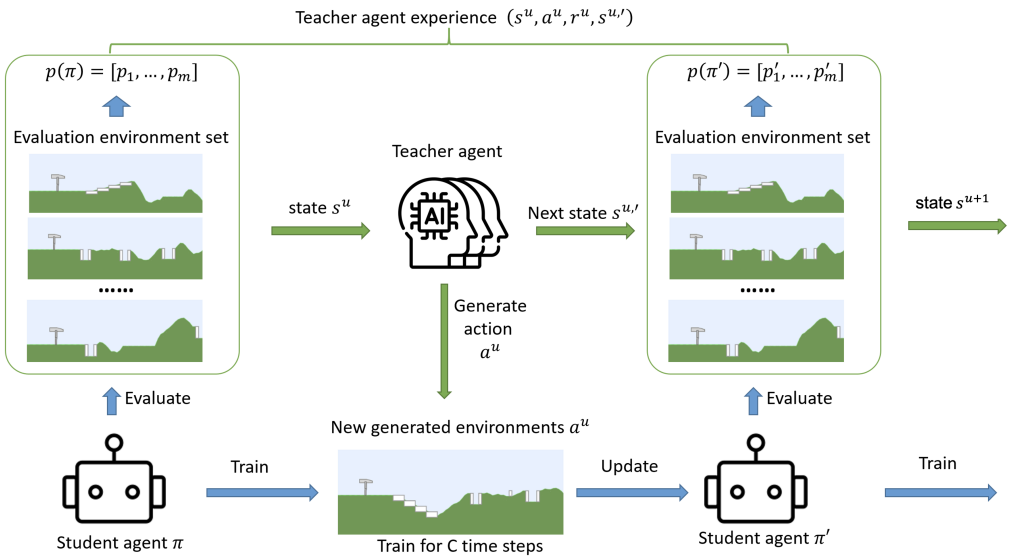


Figure 4.2: The overall framework of *SHED*.

4.3.2 Generative Trajectory Modeling

Here, we describe how to leverage the diffusion model to learn the conditional data distribution in the collected experiences $\tau = \{(s_t^u, a_t^u, r_t^u, s_t^{up,\prime})\}$ simultaneously during training the teacher-student framework. Later we can use the trainable reverse chain in the diffusion model to generate the synthetic trajectories that can be used to help train the teacher agent, resulting in reducing the resource-intensive and time-consuming collection of upper-level teacher experiences. Note that the diffusion model does not require external data collection, as its training data is entirely derived from the trajectories of student agent policies during training in environments generated by different teachers. Therefore, the diffusion model can be considered a by-product of training the general capable student agents. We deal with two different types of timesteps in this section: one for the diffusion process and the other for the upper-level teacher agent, respectively. We use subscripts $k \in 1, \ldots, K$ to represent diffusion timesteps and subscripts $t \in 1, \ldots, T$ to represent trajectory timesteps in the teacher's experience.

In the image domain, the diffusion process is implemented across all pixel values of the image. In our setting, we diffuse over the next state $s^{u,\prime}$ conditioned

the given state s^u and action a^u. We construct our generative model according to the conditional diffusion process:

$$q(s_k^{u,\prime}|s_{k-1}^{u,\prime}), \quad p_\phi(s_{k-1}^{u,\prime}|s_k^{u,\prime}, s^u, a^u)$$

As usual, $q(s_k^{u,\prime}|s_{k-1}^{u,\prime})$ is the predefined forward noising process while $p_\phi(s_{k-1}^{u,\prime}|s_k^{u,\prime}, s^u, a^u)$ is the trainable reverse denoising process. We begin by randomly sampling the collected experiences $\tau = \{(s_t^u, a_t^u, r_t^u, s_t^{up,\prime})\}$ from the real experience buffer $\mathcal{B}_{real}$. Giving the observed state s^u and action a^u, we use the reverse process p_ϕ to represent the generation of the next state $s^{u,\prime}$:

$$p_\phi(s_{0:K}^{u,\prime}|s^u, a^u) = \mathcal{N}(s_K^{u,\prime}; 0, \mathbf{I}) \prod_{k=1}^{K} p_\phi(s_{k-1}^{u,\prime}|s_k^{u,\prime}, s^u, a^u)$$

At the end of the reverse chain, the sample $s_0^{u,\prime}$, is the generated next state $s^{u,\prime}$. Similar to Ho et al. [31], we parameterize $p_\phi(s'_{k-1}|s'_k, s^u, a^u)$ as a noise prediction model with the covariance matrix fixed as $\Sigma_\phi(s_k^{u,\prime}, s^u, a^u, k) = \beta_i \mathbf{I}$, and the mean is

$$\mu_\phi(s_i^{u,\prime}, s^u, a^u, k) = \frac{1}{\sqrt{\alpha_k}} \left(s_k^{u,\prime} - \frac{\beta_k}{\sqrt{1-\bar{\alpha}_k}} \epsilon_\phi(s_k^{u,\prime}, s^u, a^u, k) \right)$$

$\epsilon_\phi(s_k^{u,\prime}, s^u, a^u, k)$ is the trainable denoising function, which aims to estimate the noise ϵ in the noisy input $s_k^{u,\prime}$ at step k.

Training objective. We employ a similar simplified objective to train the conditional ϵ- model:

$$\mathcal{L}(\phi) = \mathbb{E}_{(s^u, a^u, s^{u,\prime}) \sim \tau, k \sim \mathcal{U}, \epsilon \sim \mathcal{N}(0,\mathbf{I})} \left[\|\epsilon - \epsilon_\phi(s_k^{u,\prime}, s^u, a^u, k)\|^2 \right] \tag{4.6}$$

Where $s_k^{u,\prime} = \sqrt{\bar{\alpha}_k} s^{u,\prime} + \sqrt{1-\bar{\alpha}_k}\epsilon$. The intuition for the loss function $\mathcal{L}(\phi)$ is to predict the noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ at the denoising step k, and the diffusion model is essentially learning the student policy involution trajectories collected

in the real experience buffer $\mathcal{B}_{reals}$. Note that the reverse process necessitates a substantial number of steps K [86]. Recent research by Xiao et al. [107] has demonstrated that enabling denoising with large steps can reduce the total number of denoising steps K. To expedite the relatively slow reverse sampling process (as it requires computing ϵ_ϕ networks K times), we use a small value of K. Similar to Wang et al. [99], while simultaneously setting $\beta_{\min} = 0.1$ and $\beta_{\max} = 10.0$, we define:

$$\beta_k = 1 - \exp\left(\beta_{\min} \times \frac{1}{K} - 0.5(\beta_{\max} - \beta_{\min})\frac{2k-1}{K^2}\right)$$

This noise schedule is derived from the variance-preserving Stochastic Differential Equation by Song et al. [87].

Generate synthetic trajectories. Once the diffusion model has been trained, it can be used to generate synthetic experience data by starting with a draw from the prior $s_K^{u,\prime} \sim \mathcal{N}(0, \mathbf{I})$ and successively generating denoised next state, conditioned on the given s^u and a^u through the reverse chain p_ϕ. Note that the giving condition action a can either be randomly sampled from the action space or use another diffusion model to learn the action distribution giving the initial state s^u. This new diffusion model is essentially a behavior-cloning model that aims to learn the teacher policy $\Lambda(a^u|s^u)$. This process is similar to the work of Wang et al. [99]. We discuss this process in detail in the appendix. In this work, we randomly sample a^u as it is straightforward and can also increase the diversity in the generated synthetic experience to help train a more robust teacher agent.

After obtaining the generated next state $s^{u,\prime}$ conditioned on s^u, a^u, we compute reward r^u using teacher's reward function $R(s^u, a^u, s^{u,\prime})$. The specifics of how the reward function is chosen are explained in the following section.

4.3.3 Rewards and Choice of evaluate environments

Selection of evaluation environments. Our upper-level teacher policy generates environments tailored specifically for the lower-level student policy, aligning with the most suitable environments to improve the general capability of the lower-level student policy. Thus it is important to select a set of diverse suitable evaluation environments as the performance vector reflects the student agent's general capabilities and serves as an approximation of the policy's embedding. Fontaine and Nikolaidis [26] propose the use of quality diversity (QD) optimization to collect high-quality environments that exhibit diversity for the agent behaviors. Similarly, Bhatt et al. [8] introduce a QD-based algorithm for dynamically designing such evaluation environments based on the current agent's behavior. However, it's worth noting that this QD-based approach can be tedious and time-consuming, and the collected evaluation environments heavily rely on the given agent policy.

Given these considerations, it is natural to take advantage of the domain randomization algorithm, as it has demonstrated compelling results in generating diverse environments and training generally capable agents. In our approach, we first discretize the environment parameters into different ranges, then randomly sample from these ranges, and combine these parameters to generate evaluation environments. This method can generate environments that may induce a diverse performance for the same policy, and it shows promising empirical results in the final experiments.

We assume that there exists a finite evaluation environment set that can capture the student's general capabilities and the performance vector $[p_1, \ldots, p_m]$ is a good representation of the student policy. If this is true, we then can construct a finite set of environments, and the student performances in those environments can represent the performances in all potential environments generated within the certain environment parameters open interval combinations, and the set of

those open intervals combinations cover the environment parameter space Θ.

Reward design. We define the reward function for the upper-level teacher policy as a parameterized function based on the improvement in student performance in the evaluation environments after training in the generated environment:

$$R(s^u, a^u, s^{u,\prime}) = \sum_{i=1}^{m} (p_i' - p_i)$$

This reward function gives positive rewards to the upper-level teacher for taking action to create the right environment to improve the overall performance of students across diverse environments. However, it may encourage the teacher to obtain higher rewards by sacrificing student performance in one subset of evaluation environments to improve student performance in another subset, which conflicts with our objective to develop a student agent with general capabilities. Therefore, we need to consider fairness in the reward function to ensure that the generated environment can improve student's general capabilities. Similar to [23], we build our fairness metric on top of the change in student's performance in each evaluation environment, denoted as $\omega_i = p_i' - p_i$, and we have $\bar{\omega} = \frac{1}{m}\sum_{i=1}^{m} \omega_i$. We then measure the fairness of the teacher's action using the coefficient of variation of student performances:

$$cv(s^u, a^u, s^{u,\prime}) = \sqrt{\frac{1}{m-1}\sum_{i} \frac{(\omega_i - \bar{\omega})^2}{\bar{\omega}^2}} \tag{4.7}$$

A teacher is considered to be fair if and only if the cv is smaller. As a result, our reward function is:

$$R(s^u, a^u, s^{u,\prime}) = \sum_{i=1}^{m} (p_i' - p_i) - \eta \cdot cv(s^u, a^u, s^{u,\prime}) \tag{4.8}$$

Here η is the coefficient that balances the weight of fairness in the reward function (We set a small value to η). This reward function motivates the teacher to generate training environments that can improve student's general capability.

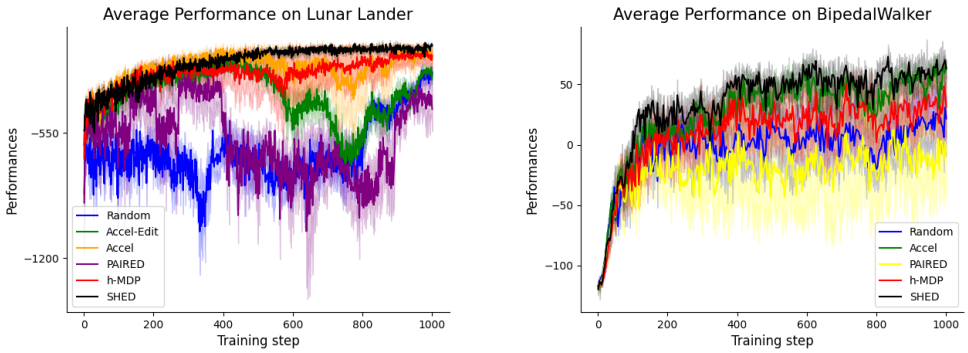


Figure 4.3: *Left*: The average zero-shot transfer performances on the test environments in the Lunar lander environment (mean and standard error). *Right*: The average zero-shot transfer performances on the test environments in the BipedalWalker (mean and standard error).

4.4 Experiments

In this section, we conduct experiments to compare *SHED* to other leading approaches on three domains: Lunar Lander, maze and a modified BipedalWalker environment. Experimental details and hyperparameters can be found in the Appendix. Specifically, our primary comparisons involve *SHED* and *h-MDP* (our proposed hierarchical approach without diffusion model aiding in training) against four baselines: domain randomization [94], ACCEL, [70], Edited ACCEL(with slight modifications that it does not revisit the previously generated environments), PAIRED [22]. In all cases, we train a student agent via Proximal Policy Optimization (PPO [82], and train the teacher agent via Deterministic policy gradient algorithms(DDPG [84]), because DDPG is an off-policy algorithm and can learn from both real experiences and the synthetic experiences.

Setup. For each domain, we construct a set of evaluation environments and a set of test environments. The vector of student performances in the evaluation environments is used as the approximation of the student policy (as the observation to teacher agent), and the performances in the test environments are used to represent the student's zero-shot transfer performances (general capabilities). Note that in order to obtain a fair comparison of zero-shot transfer performance, the evaluation environments and test environments do not share

the same environment and they are not present during training.

Lunar Lander. This is a classic rocket trajectory optimization problem. In this domain, student agents are tasked with controlling a lander's engine to safely land the vehicle. Before the start of each episode, teacher algorithms determine the environment parameters that are used to generate environments in a given play-through, which includes gravity, wind power, and turbulence power. These parameters directly alter the difficulty of landing the vehicle safely. The state is an 8-dimensional vector, which includes the coordinates of the lander, its linear velocities, its angle, its angular velocity, and two booleans that represent whether each leg is in contact with the ground or not.

We train the student agent for 1e6 environment time steps and periodically test the agent in test environments. The parameters for the test environments are randomly generated and fixed during training. We report the experiment results on the left side of Figure 4.3. As we can see, student agents trained under *SHED* consistently outperform other baselines and have minimal variance in transfer performance. During training, the baselines, except h-MDP, show a performance dip in the middle. This phenomenon could potentially be attributed to the inherent challenge of designing the appropriate environment instance in the large environment parameter space. This further demonstrates the effectiveness of our hierarchical design (*SHED* and h-MDP), which can successfully create environments that are appropriate to the current skill level of the students.

Bipedalwalker. We also evaluate *SHED* in the modified BipedalWalker from Parker-Holder et al. [70]. In this domain, the student agent is required to control a bipedal vehicle and navigate across the terrain, and the student receives a 24-dimensional proprioceptive state with respect to its lidar sensors, angles, and contacts. The teacher is tasked to select eight variables (including ground roughness, the number of stairs steps, min/max range of pit gap width,

min/max range of stump height, and min/max range of stair height) to generate the corresponding terrain.

We use similar experiment settings in prior UED works, we train all the algorithms for 1e7 environment time steps, and then evaluate their generalization ability on ten distinct test environments in Bipedal-Walker domain. The parameters for the test environments are randomly generated and fixed during training. As shown in Figure 4.3, our proposed method *SHED* surpasses all other baselines and achieves performance levels nearly on par with the SOTA (ACCEL). Meanwhile, SHED maintains a slight edge in terms of stability and overall performance and PAIRED suffers from a considerable degree of variance in its performance.

Partially observable Maze. Here we study navigation tasks, where an agent must explore to find a goal while navigating around obstacles. The environment is partially observable, and the agent's field of view is limited to a 3×3 grid area. Unlike the previously mentioned domains, maze environments are non-parametric and cannot be directly represented by compact parameter vectors due to their high complexity. To solve this challenge, we propose a novel method to generate maze by leveraging advances in large language models (e.g., ChatGPT). Specifically, we implement a retrieval-augmented generation (RAG) process to optimize the ChatGPT's output such that it can generate desired maze environments. This process ensures that large language models reference authoritative knowledge bases to generate feasible mazes. To simplify the teacher's action space, we extracted several key factors that constitute the teacher's action space (environmental parameters) for maze generation. Details on maze generation are provided in Appendix 6.7.3, and prompt are included in Appendix 6.7.4.

The average zero-shot transfer performances are reported in Figure 4.4. Notably, *SHED* demonstrates the highest performance, consistently improving and achieving the highest cumulative rewards. The performance of h-MDP steadily improves but does not reach the highest levels, which further highlights the advantages of incorporating the generated synthetic datasets to train an effective RL teacher agent. Meanwhile, Accel-Edit and Accel show higher variances in performance, indicating that random teachers are less stable in finding a suitable environment to train student agents.

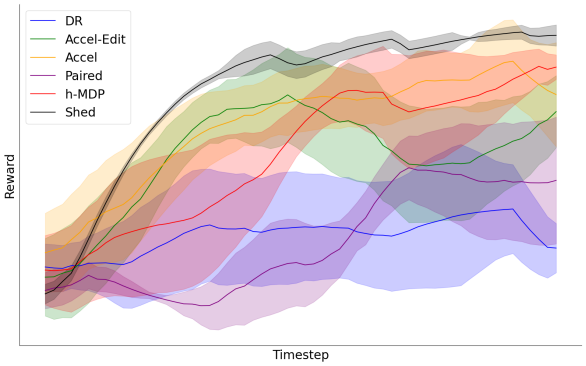


Figure 4.4: Average zero-shot transfer performance on the test environments in the maze environments.

Ablation and additional Experiments In Appendix 6.6, we evaluate the ability of the diffusion model to generate the synthetic student policy involution trajectories. We further provide ablation studies to assess the impact of different design choices in Appendix 6.8.1. Additionally, in Appendix 6.8.2, we conduct experiments to show how the algorithm performs under different settings, including scenarios with a larger budget constraint on the number of generated environments or a larger weight assigned to CV fairness rewards. Notably, all results consistently demonstrate the effectiveness of our approach.

4.5 Conclusion

In this chapter, we introduce an adaptive approach for efficiently training a generally capable agent under resource constraints. Our approach is general, utilizing an upper-level MDP teacher agent that can guide the training of the

lower-level MDP student agent. The hierarchical framework can incorporate techniques from existing UED works, such as prioritized level replay (revisiting environments with high learning potential). Furthermore, we have described a method to assist the experience collection for the teacher when it is trained in an off-policy manner. Our experiment demonstrates that our method outperforms existing UED methods, highlighting its effectiveness as a curriculum-based learning approach within the UED framework. We believe that this adaptive curriculum environment design can help train non-specialists by designing the most appropriate environment for them to acquire different level of skills quickly.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this dissertation, we focused on sequential decision learning for social good and fairness in both RMAB and RL settings:

- In the first part of the dissertation, we focus on Restless Multi-Armed Bandits, which is an apt model to represent decision learning problems in public health interventions (e.g., tuberculosis, maternal, and child care), anti-poaching planning, sensor monitoring, personalized recommendations and many more. Existing research in RMAB has contributed mechanisms and theoretical results to a wide variety of settings, where the focus is on maximizing expected value. In Chapter 2, we are interested in ensuring that RMAB decision-making is also fair to different arms while maximizing expected value. In the context of public health settings, this would ensure that different people and/or communities are fairly represented while making public health intervention decisions. To achieve this goal, we formally define two different types of fairness constraints in RMAB: deterministic fairness constraints and probabilistic fairness constraints, and then we provide planning and learning methods to solve RMAB in a fair manner. We demonstrate key theoretical properties of fair RMAB and

experimentally demonstrate that our proposed methods handle fairness constraints without sacrificing significantly on solution quality.

- In the second part of the dissertation, we focus on reinforcement learning settings. We first consider the problem of influence maximization on unknown social networks. This has found application in HIV prevention, substance abuse prevention, micro-finance adoption, etc. In these applications, the goal is to identify the set of peer leaders in a real-world physical social network who can disseminate information to a large group of people. Unlike online social networks, real-world networks are not completely known, and collecting information about network is costly as it involves surveying multiple people. Therefore, in these applications, it is important to efficiently discover a subset of the network within a limited budget such that selecting peer leaders from this subgraph can help in maximizing the influence of the complete network. We develop an RL-based approach to automatically discover network information by using learned representations of nodes and graphs that encode important structural properties of the network. Our proposed approach is sample efficient to reduce costly interactions between RL agents and the environment.

- Lastly, The advances of RL have also promoted research into the problem of training autonomous agents and humans to execute complex tasks proficiently. One interesting and yet underexplored area is enabling agents and humans to perform well in unseen environments, a concept referred to as zero-shot transfer performance. A promising approach involves generating a curriculum of training environments, enabling agents to develop general capabilities through training in these environments. To this end, we proposed SHED, a novel hierarchical MDP framework for environment design under resource constraints. Crucially, SHED leverages the recent advances in generative modeling to reduce the resource-intensive

interactions between agents and environments, and incorporates a fairness reward to ensure the teacher agent generates suitable environments. Our work has the potential to be a valuable tool for developing safer and more reliable reinforcement learning agents, which can significantly enhance the training of non-expert humans and facilitate their adaptation to different real-world situations.

5.2 Future work

This dissertation has presented comprehensive discussions of sequential decision learning for social good, particularly under the challenge of resource constraints in real-world scenarios. Such constraints necessitate the incorporation of fairness considerations and the development of sample-efficient learning algorithms. This also opens up several important extensions as well as further open-ended research programs. Specifically, we highlight the following directions:

Considering the interdependence in the RMAB settings: In Chapter 2, we explore the RMAB, which allows non-active bandits to undergo the Markovian state transition. In RMAB, there are N independent arms, each of which evolves according to an associated Markov Decision Process (MDP). Existing works often assume that each arm evolves independently, neglecting the prevalent interdependencies among arms in many real-world scenarios. For example, when using RAMB in the education domain, i.e., developing personalized and adaptive educational tools that cater to a student's individual learning progress, RMAB would be an ideal model that features a teacher agent selecting a subset of arms (concepts) to teach in each round. However, the inherent assumption in RMAB that arms are independent of each other proves unrealistic in practical educational settings. As a motivating example, consider a student engaging with a math question focused on determining the area of a triangle. Success in this particular

question may necessitate tapping into the student's knowledge of basic algebra, arithmetic, and geometry. Intuitively, practicing this question should contribute to enhancing the student's proficiency in all three areas. However, models that overlook such interdependencies are likely to fall short in predicting knowledge levels across all three areas, especially if operating under the assumption that a single exercise exclusively impacts knowledge in a single area. In response to this challenge, a future direction involves developing an interdependency-aware RMAB model for education settings. Additionally, designing a sample-efficient offline RL algorithm capable of learning from historical datasets will be another key focus.

Considering multiple queries at each timestep for Influence Maximization Problems: In Chapter 3, we focus on only one query at each timestep during the network discovery process in the Influence Maximization problems. In the future, the work can be extended to provide more general approaches that allow for multiple queries at each timestep. This expansion is particularly relevant in the context of collecting physical social networks, where querying multiple individuals simultaneously can be more time-efficient. Additionally, in the future, it is reasonable to consider uncertainties during the network discovery process. For instance, challenges arise when selected individuals fail to respond to queries, or their feedback does not provide detailed information about all their neighbors. To overcome this, algorithms for predicting potential missing connections between nodes or individuals should be developed. Furthermore, we consider fairness constraints in the RMAB problem and in training robust agents, it is important to note that ensuring equitable selection of nodes through fairness constraints, such as gender balance, is crucial in influence maximization problems. Taking these multifaceted considerations into account could substantially improve both the acceptance and the effectiveness of the algorithm.

Applying environment design algorithms in human subject experiments: In Chapter 4 we concentrate on developing an algorithm for training generally capable agents through environment design. To further advance the practical utility and impact of our method, future work will explore its application in human subject experiments. Specifically, we propose a two-stage process when employing our algorithm to assist the training of the non-expert. During the first stage, we initialize the RL teacher agent to collect the necessary experience data on student-environment interactions. This RL teacher is trained to generate suitable environments that support the student's learning process. In the second stage, the trained RL teacher agent is deployed to assist in real human learning scenarios. The RL teacher assesses the human students' current capability level by observing their interactions with the environments and then suggests the next set of environments for further training. This framework has the potential to be applied in various fields to assist in training non-specialists. Notably, it could revolutionize skill acquisition in areas such as healthcare, where tailored and effective training is crucial.

Bibliography

[1] S. Agrawal and N. Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on learning theory*, pages 39–1. JMLR Workshop and Conference Proceedings, 2012.

[2] N. Akbarzadeh and A. Mahajan. Restless bandits with controlled restarts: Indexability and computation of whittle index. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 7294–7300. IEEE, 2019.

[3] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[4] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba. Hindsight experience replay. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5048–5058. Curran Associates, Inc., 2017.

[5] K. Asadi and M. L. Littman. An alternative softmax operator for reinforcement learning. In *International Conference on Machine Learning*, pages 243–252. PMLR, 2017.

[6] K. Avrachenkov and V. S. Borkar. Whittle index based q-learning for restless bandits with average reward. *arXiv preprint arXiv:2004.14427*, 2020.

[7] A. Banerjee, A. G. Chandrasekhar, E. Duflo, and M. O. Jackson. The diffusion of microfinance. *Science*, 341(6144), 2013.

[8] V. Bhatt, B. Tjanaka, M. Fontaine, and S. Nikolaidis. Deep surrogate assisted generation of environments. *Advances in Neural Information Processing Systems*, 35:37762–37777, 2022.

[9] B. Bhattacharya. Restless bandits visiting villages: A preliminary study on distributing public health services. In *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, pages 1–8, 2018.

[10] A. Biswas, G. Aggarwal, P. Varakantham, and M. Tambe. Learn to intervene: An adaptive learning policy for restless bandits in application to preventive healthcare. *arXiv preprint arXiv:2105.07965*, 2021.

[11] J. N. Brownstein, F. M. Chowdhury, S. L. Norris, T. Horsley, L. Jack Jr, X. Zhang, and D. Satterfield. Effectiveness of community health workers in the care of people with hypertension. *American journal of preventive medicine*, 32(5):435–447, 2007.

[12] J. Bruce, M. Dennis, A. Edwards, J. Parker-Holder, Y. Shi, E. Hughes, M. Lai, A. Mavalankar, R. Steigerwald, C. Apps, et al. Genie: Generative interactive environments. *arXiv preprint arXiv:2402.15391*, 2024.

[13] C. Budak, D. Agrawal, and A. El Abbadi. Limiting the spread of misinformation in social networks. In *Proceedings of the 20th international conference on World wide web*, pages 665–674, 2011.

[14] Y. Burda, H. Edwards, D. Pathak, A. Storkey, T. Darrell, and A. A. Efros. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018.

[15] A. H. Chang, A. Polesky, and G. Bhatia. House calls by community health workers and public health nurses to improve adherence to isoniazid monotherapy for latent tuberculosis infection: a retrospective study. *BMC public health*, 13(1): 1–7, 2013.

[16] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1029–1038, 2010.

[17] Y. Chen, A. Cuellar, H. Luo, J. Modi, H. Nemlekar, and S. Nikolaidis. Fair contextual multi-armed bandits: Theory and experiments. In *Conference on Uncertainty in Artificial Intelligence*, pages 181–190. PMLR, 2020.

[18] H. Claure, Y. Chen, J. Modi, M. Jung, and S. Nikolaidis. Multi-armed bandits with fairness constraints for distributing resources to human teammates. In *2020 15th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 299–308. IEEE, 2020.

[19] C. Colas, P. Fournier, M. Chetouani, O. Sigaud, and P.-Y. Oudeyer. Curious: intrinsically motivated modular multi-goal reinforcement learning. In *International conference on machine learning*, pages 1331–1340. PMLR, 2019.

[20] G. Cornuejols, M. Fisher, and G. L. Nemhauser. On the uncapacitated location problem. In *Annals of Discrete Mathematics*, volume 1, pages 163–177. Elsevier, 1977.

[21] S. Davis, B. Abbasi, S. Shah, S. Telfer, and M. Begon. Spatial analyses of wildlife contact networks. *Journal of the Royal Society Interface*, 12(102):20141004, 2015.

[22] M. Dennis, N. Jaques, E. Vinitsky, A. Bayen, S. Russell, A. Critch, and S. Levine. Emergent complexity and zero-shot transfer via unsupervised environment design. *Advances in neural information processing systems*, 33:13049–13061, 2020.

[23] S. Elmalaki. Fair-iot: Fairness-aware human-in-the-loop reinforcement learning for harnessing human variability in personalized iot. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, pages 119–132, 2021.

[24] M. Fang, T. Zhou, Y. Du, L. Han, and Z. Zhang. Curriculum-guided hindsight experience replay. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 12623–12634. Curran Associates, Inc., 2019.

[25] N. Ferdosian, M. Othman, B. M. Ali, and K. Y. Lun. Fair-qos broker algorithm for overload-state downlink resource scheduling in lte networks. *IEEE Systems Journal*, 12(4):3238–3249, 2017.

[26] M. Fontaine and S. Nikolaidis. Differentiable quality diversity. *Advances in Neural Information Processing Systems*, 34:10040–10052, 2021.

[27] J. Fu, Y. Nazarathy, S. Moka, and P. G. Taylor. Towards q-learning the whittle index for restless bandits. In *2019 Australian & New Zealand Control Conference (ANZCC)*, pages 249–254. IEEE, 2019.

[28] A. Gopalan and S. Mannor. Thompson sampling for learning parameterized markov decision processes. In *Conference on Learning Theory*, pages 861–898. PMLR, 2015.

[29] C. Herlihy, A. Prins, A. Srinivasan, and J. Dickerson. Planning to fairly allocate: Probabilistic fairness in the restless bandit setting. *arXiv preprint arXiv:2106.07677*, 2021.

[30] M. W. Hirsch. Convergent activation dynamics in continuous time networks. *Neural networks*, 2(5):331–349, 1989.

[31] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[32] W. Hu and P. Frazier. An asymptotically optimal index policy for finite-horizon restless bandits. *arXiv preprint arXiv:1707.00205*, 2017.

[33] T. Jaakkola, M. I. Jordan, and S. P. Singh. On the convergence of stochastic iterative dynamic programming algorithms. *Neural computation*, 6(6):1185–1201, 1994.

[34] S. Jabbari, M. Joseph, M. Kearns, J. Morgenstern, and A. Roth. Fairness in reinforcement learning. In *International conference on machine learning*, pages 1617–1626. PMLR, 2017.

[35] P. Jacko. Restless bandits approach to the job scheduling problem and its extensions. *Modern trends in controlled stochastic processes: theory and applications*, pages 248–267, 2010.

[36] M. Jiang, E. Grefenstette, and T. Rocktäschel. Prioritized level replay. In *International Conference on Machine Learning*, pages 4940–4950. PMLR, 2021.

[37] M. Joseph, M. Kearns, J. Morgenstern, and A. Roth. Fairness in learning: Classic and contextual bandits. *arXiv preprint arXiv:1605.07139*, 2016.

[38] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

[39] H. Kamarthi, P. Vijayan, B. Wilder, B. Ravindran, and M. Tambe. Influence maximization in unknown social networks: Learning policies for effective graph sampling. *arXiv preprint arXiv:1907.11625*, 2019.

[40] Y. Kang, V. V. Prabhu, A. M. Sawyer, and P. M. Griffin. Markov models for treatment adherence in obstructive sleep apnea. *Age*, 49:11–6, 2013.

[41] M. N. Katehakis and A. F. Veinott Jr. The multi-armed bandit problem: decomposition and computation. *Mathematics of Operations Research*, 12(2):262–268, 1987.

[42] C. J. Kelly, A. Karthikesalingam, M. Suleyman, G. Corrado, and D. King. Key challenges for delivering clinical impact with artificial intelligence. *BMC medicine*, 17(1):1–9, 2019.

[43] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 137-146, 07 2003.

[44] J. A. Killian, A. Biswas, S. Shah, and M. Tambe. Q-learning lagrange policies for multi-action restless bandits. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 871–881, 2021.

[45] T. Kozuno, E. Uchibe, and K. Doya. Theoretical analysis of efficiency and robustness of softmax and gap-increasing operators in reinforcement learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2995–3003. PMLR, 2019.

[46] E. Lee, M. S. Lavieri, and M. Volk. Optimal screening for hepatocellular carcinoma: A restless bandit model. *Manufacturing & Service Operations Management*, 21(1):198–212, 2019.

[47] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[48] D. Li and P. Varakantham. Efficient resource allocation with fairness constraints in restless multi-armed bandits. In *The 38th Conference on Uncertainty in Artificial Intelligence*, 2022.

[49] D. Li, M. Lowalekar, and P. Varakantham. Claim: Curriculum learning policy for influence maximization in unknown social networks. In *Uncertainty in Artificial Intelligence*, pages 1455–1465. PMLR, 2021.

[50] F. Li, J. Liu, and B. Ji. Combinatorial sleeping bandits with fairness constraints. *IEEE Transactions on Network Science and Engineering*, 7(3):1799–1813, 2019.

[51] W. Li, P. Varakantham, and D. Li. Effective diversity in unsupervised environment design. *arXiv preprint arXiv:2301.08025*, 2023.

[52] H. Lin, J. Bilmes, and S. Xie. Graph-based submodular selection for extractive summarization. In *2009 IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 381–386. IEEE, 2009.

[53] K. Liu and Q. Zhao. Indexability of restless bandit problems and optimality of whittle index for dynamic multichannel access. *IEEE Transactions on Information Theory*, 56(11):5547–5567, 2010.

[54] M. Lowalekar, P. Varakantham, and A. Kumar. Robust influence maximization. *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems*, 2016.

[55] B. Löwe, J. Unützer, C. M. Callahan, A. J. Perkins, and K. Kroenke. Monitoring depression treatment outcomes with the patient health questionnaire-9. *Medical care*, pages 1194–1201, 2004.

[56] A. Mate, J. A. Killian, H. Xu, A. Perrault, and M. Tambe. Collapsing bandits and their application to public health interventions. *arXiv preprint arXiv:2007.04432*, 2020.

[57] A. Mate, A. Biswas, C. Siebenbrunner, and M. Tambe. Efficient algorithms for finite horizon and streaming restless multi-armed bandit problems. *arXiv preprint arXiv:2103.04730*, 2021.

[58] A. Mate, A. Perrault, and M. Tambe. Risk-aware interventions in public health: Planning with restless multi-armed bandits. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, pages 880–888, 2021.

[59] R. Meshram, A. Gopalan, and D. Manjunath. Optimal recommendation to users that react: Online learning for a class of pomdps. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 7210–7215. IEEE, 2016.

[60] R. Meshram, A. Gopalan, and D. Manjunath. A hidden markov restless multi-armed bandit model for playout recommendation systems. In *International Conference on Communication Systems and Networks*, pages 335–362. Springer, 2017.

[61] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[62] C. Mundorf, A. Shankar, T. Moran, S. Heller, A. Hassan, E. Harville, and M. Lichtveld. Reducing the risk of postpartum depression in a low-income community through a community health worker intervention. *Maternal and child health journal*, 22(4):520–528, 2018.

[63] I. Nahum-Shani, M. Qian, D. Almirall, W. E. Pelham, B. Gnagy, G. A. Fabiano, J. G. Waxmonsky, J. Yu, and S. A. Murphy. Q-learning: a data analysis method for constructing adaptive interventions. *Psychological methods*, 17(4):478, 2012.

[64] A. V. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine. Visual reinforcement learning with imagined goals. *Advances in Neural Information Processing Systems*, 31:9191–9200, 2018.

[65] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1): 265–294, 1978.

[66] P. M. Newman, M. F. Franke, J. Arrieta, H. Carrasco, P. Elliott, H. Flores, A. Friedman, S. Graham, L. Martinez, L. Palazuelos, et al. Community health workers improve disease control and medication adherence among patients with

diabetes and/or hypertension in chiapas, mexico: an observational stepped-wedge study. *BMJ global health*, 3(1):e000566, 2018.

[67] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[68] W. Nie, B. Guo, Y. Huang, C. Xiao, A. Vahdat, and A. Anandkumar. Diffusion models for adversarial purification. *arXiv preprint arXiv:2205.07460*, 2022.

[69] J. R. Ong'ang'o, C. Mwachari, H. Kipruto, and S. Karanja. The effects on tuberculosis treatment adherence from utilising community health workers: a comparison of selected rural and urban settings in kenya. *PLoS One*, 9(2):e88937, 2014.

[70] J. Parker-Holder, M. Jiang, M. Dennis, M. Samvelyan, J. Foerster, E. Grefenstette, and T. Rocktäschel. Evolving curricula with regret-based environment design. *arXiv preprint arXiv:2203.01302*, 2022.

[71] V. Patil, G. Ghalme, V. Nair, and Y. Narahari. Achieving fairness in the stochastic multi-armed bandit problem. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):5379–5386, Apr. 2020. doi: 10.1609/aaai.v34i04.5986.

[72] V. Patil, G. Ghalme, V. Nair, and Y. Narahari. Achieving fairness in the stochastic multi-armed bandit problem. *Journal of Machine Learning Research*, 22(174): 1–31, 2021.

[73] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '14*, 2014.

[74] Y. Qian, C. Zhang, B. Krishnamachari, and M. Tambe. Restless poachers: Handling exploration-exploitation tradeoffs in security domains. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 123–131, 2016.

[75] V. Raghunathan, V. Borkar, M. Cao, and P. R. Kumar. Index policies for real-time multicast scheduling for wireless broadcast systems. In *IEEE INFOCOM 2008-The 27th Conference on Computer Communications*, pages 1570–1578. IEEE, 2008.

[76] A. Rajkomar, M. Hardt, M. D. Howell, G. Corrado, and M. H. Chin. Ensuring fairness in machine learning to advance health equity. *Annals of internal medicine*, 169(12):866–872, 2018.

[77] P. Reverdy and N. E. Leonard. Parameter estimation in softmax decision-making models with linear objective functions. *IEEE Transactions on Automation Science and Engineering*, 13(1):54–67, 2015.

[78] R. A. Rossi and N. K. Ahmed. Networkrepository: A graph data repository with visual interactive analytics. In *29th AAAI Conference on Artificial Intelligence, Austin, Texas, USA*, pages 25–30, 2015.

[79] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

[80] T. Schaul, D. Horgan, K. Gregor, and D. Silver. Universal value function approximators. In *International conference on machine learning*, pages 1312–1320. PMLR, 2015.

[81] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[82] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[83] S.-P. Sheng, M. Liu, and R. Saigal. Data-driven channel modeling using spectrum measurement. *IEEE Transactions on Mobile Computing*, 14(9):1794–1805, 2014.

[84] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. Pmlr, 2014.

[85] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587): 484–489, 2016.

[86] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[87] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[88] Z. Song, R. Parr, and L. Carin. Revisiting the softmax bellman operator: New benefits and new perspective. In *International conference on machine learning*, pages 5916–5925. PMLR, 2019.

[89] S. Sukhbaatar, Z. Lin, I. Kostrikov, G. Synnaeve, A. Szlam, and R. Fergus. Intrinsic motivation and automatic curricula via asymmetric self-play. *arXiv preprint arXiv:1703.05407*, 2017.

[90] R. S. Sutton, A. G. Barto, et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

[91] Y. Tashiro, J. Song, Y. Song, and S. Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34:24804–24816, 2021.

[92] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

[93] S. Tio and P. Varakantham. Transferable curricula through difficulty conditioned generators. *arXiv preprint arXiv:2306.13028*, 2023.

[94] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[95] T. W. Valente and P. Pumpuang. Identifying opinion leaders to promote behavior change. *Health education & behavior*, 34(6):881–896, 2007.

[96] P. Varakantham, S.-F. Cheng, G. Gordon, and A. Ahmed. Decision support for agent populations in uncertain and congested environments. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[97] C. Wang, W. Chen, and Y. Wang. Scalable influence maximization for independent cascade model in large-scale social networks. *Data Mining and Knowledge Discovery*, 25(3):545–576, 2012.

[98] S. Wang, L. Huang, and J. Lui. Restless-ucb, an efficient and low-complexity algorithm for online restless bandits. *Advances in Neural Information Processing Systems*, 33:11878–11889, 2020.

[99] Z. Wang, J. J. Hunt, and M. Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=AHvFDPi-FA`.

[100] C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992.

[101] T. E. Weaver and R. R. Grunstein. Adherence to continuous positive airway pressure therapy: the challenge to effective treatment. *Proceedings of the American Thoracic Society*, 5(2):173–178, 2008.

[102] R. R. Weber and G. Weiss. On an index policy for restless bandits. *Journal of applied probability*, pages 637–648, 1990.

[103] P. Whittle. Restless bandits: Activity allocation in a changing world. *Journal of applied probability*, pages 287–298, 1988.

[104] B. Wilder, N. Immorlica, E. Rice, and M. Tambe. Maximizing influence in an unknown social network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4743–4750. AAAI Press, 2018.

[105] B. Wilder, L. Onasch-Vera, J. Hudson, J. Luna, N. Wilson, R. Petering, D. Woo, M. Tambe, and E. Rice. End-to-end influence maximization in the field. In E. André, S. Koenig, M. Dastani, and G. Sukthankar, editors, *AAMAS*, pages 1414–1422. International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, 2018.

[106] J. Wyatt, A. Leach, S. M. Schmon, and C. G. Willcocks. Anoddpm: Anomaly detection with denoising diffusion probabilistic models using simplex noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 650–656, 2022.

[107] Z. Xiao, K. Kreis, and A. Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. *arXiv preprint arXiv:2112.07804*, 2021.

[108] G. Xiong, J. Li, and R. Singh. Reinforcement learning augmented asymptotically optimal index policy for finite-horizon restless bandits. 2022.

[109] A. Yadav, H. Chan, A. X. Jiang, H. Xu, E. Rice, and M. Tambe. Using social networks to aid homeless shelters: Dynamic influence maximization under uncertainty. In *AAMAS*, volume 16, pages 740–748, 2016.

[110] R. Ying, J. You, C. Morris, X. Ren, W. L. Hamilton, and J. Leskovec. Hierarchical graph representation learning with differentiable pooling, 2019.

[111] B. D. Ziebart. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University, 2010.

Chapter 6

Appendix

6.1 Appendix for Chapter 2.3

6.1.1 Proof for Boundary Lemma

Lemma 3 *For the finite time horizon T, $V_{\lambda_T,T}(\omega_1) - V_{\lambda_T,T}(\omega_2)$ is bounded, where we have*

$$(\omega_1-\omega_2)\sum_{t=0}^{T-1}\beta^t(P^a_{1,1}-P^a_{0,1})^t \le V_{\lambda_T,T}(\omega_1) - V_{\lambda_T,T}(\omega_2) \le (\omega_1-\omega_2)\sum_{t=0}^{T-1}\beta^t \tag{6.1}$$

Proof. We prove the lower bound by induction, and the upper bound can be proven similarly.

When $T=1$, we start from the definition of the value function $V_{\lambda_T,T}(\omega)$ to have

- passive actions:

$$\begin{aligned} V_{\lambda_1,1}(\omega_1, a=0) - V_{\lambda_1,1}(\omega_2, a=0) &= \lambda_1+\omega_1-\lambda_1-\omega_2 \\ &= \omega_1-\omega_2 \end{aligned} \tag{6.2}$$

- active actions:

$$V_{\lambda_1,1}(\omega_1, a=1) - V_{\lambda_1,1}(\omega_2, a=1) = \omega_1-\omega_2 \tag{6.3}$$

We get $V_{\lambda_1,1}(\omega_1) - V_{\lambda_1,1}(\omega_2) = \omega_1-\omega_2$. Now we assume $V_{\lambda_T,T}(\omega_1) - V_{\lambda_T,T}(\omega_2) \ge (\omega_1-\omega_2)\sum_{t=0}^{T-1}\beta^t(P^a_{1,1}-P^a_{0,1})^t$ hold for $\forall T>1$, then for time horizon $T+1$, we have

- passive actions:

$$\begin{aligned} &V_{\lambda_{T+1},T+1}(\omega_1, a=0) - V_{\lambda_{T+1},T+1}(\omega_2, a=0) \\ =&(\lambda_1+\omega_1+\beta V_{\lambda_T,T}(\omega_1(1))) - (\lambda_1+\omega_2+\beta V_{\lambda_T,T}(\omega_2(1))) \\ =&\omega_1-\omega_2+\beta\left(V_{\lambda_T,T}(\omega_1(1)) - V_{\lambda_T,T}(\omega_2(1))\right) \\ \ge&\omega_1-\omega_2+\beta(\omega_1-\omega_2)\sum_{t=0}^{T-1}\beta^t(P^a_{1,1}-P^a_{0,1})^t \\ \ge&\omega_1-\omega_2+(\omega_1-\omega_2)\sum_{t=1}^{T}\beta^t(P^a_{1,1}-P^a_{0,1})^t \text{ Line } * \\ =&(\omega_1-\omega_2)\sum_{t=0}^{T}\beta^t(P^a_{1,1}-P^a_{0,1})^t \end{aligned} \tag{6.4}$$

Line $*$ is because $0 \leq P^a_{1,1} - P^a_{0,1} < 1$.

- active actions:

$$
\begin{aligned}
& V_{\lambda_{T+1},T+1}(\omega_1, a=1) - V_{\lambda_{T+1},T+1}(\omega_2, a=1) \\
=& \left(\omega_1 + \beta(\omega_1 V_{\lambda_T,T}(P^a_{1,1}) + (1-\omega_1)V_{\lambda_T,T}(P^a_{0,1}))\right) \\
& - \left(\omega_2 + \beta(\omega_2 V_{\lambda_T,T}(P^a_{1,1}) + (1-\omega_2)V_{\lambda_T,T}(P^a_{0,1}))\right) \\
=& \omega_1 - \omega_2 + \beta\left((\omega_1 - \omega_2)(V_{\lambda_T,T}(P^a_{1,1}) - V_{\lambda_T,T}(P^a_{0,1}))\right) \\
=& \omega_1 - \omega_2 + \beta(V_{\lambda_T,T}(\omega_1(1)) - V_{\lambda_T,T}(\omega_2(1))) \\
\geq& \omega_1 - \omega_2 + \beta(\omega_1 - \omega_2)\sum_{t=0}^{T-1}\beta^t(P^a_{1,1} - P^a_{0,1})^t \\
\geq& \omega_1 - \omega_2 + (\omega_1 - \omega_2)\sum_{t=1}^{T}\beta^t(P^a_{1,1} - P^a_{0,1})^t \text{ Line } * \\
=& (\omega_1 - \omega_2)\sum_{t=0}^{T}\beta^t(P^a_{1,1} - P^a_{0,1})^t
\end{aligned}
\tag{6.5}
$$

Line $*$ is because $0 \leq P^a_{1,1} - P^a_{0,1} < 1$. Thus we have $V_{\lambda_T,T}(\omega_1) - V_{\lambda_T,T}(\omega_2) \geq (\omega_1 - \omega_2)\sum_{t=0}^{T-1}\beta^t(P^a_{1,1} - P^a_{0,1})^t$. Similarly, we can prove the lower bound. □

Lemma 4 *For the infinite residual time horizon $T \to \infty$, $V_{\lambda_T,T}(\omega_1) - V_{\lambda_T,T}(\omega_2)$is bounded. Specifically, we have*

$$
\frac{\omega_1 - \omega_2}{1 - \beta(P^a_{1,1} - P^a_{0,1})} \leq V_{\lambda_T,T}(\omega_1) - V_{\lambda_T,T}(\omega_2) \leq \frac{\omega_1 - \omega_2}{1-\beta} \tag{6.6}
$$

Proof. This can be viewed as a special case of the finite residual time horizon setting where $T \to \infty$. Thus we can easily derive the lower and upper bound according to the formula for the geometric series:

$$
\lim_{T\to\infty}(\omega_1 - \omega_2)\sum_{t=0}^{T-1}\beta^t(P^a_{1,1} - P^a_{0,1})^t = \frac{\omega_1 - \omega_2}{1 - \beta(P^a_{1,1} - P^a_{0,1})}
$$

and

$$
\lim_{T\to\infty}(\omega_1 - \omega_2)\sum_{t=0}^{T-1}\beta^t = \frac{\omega_1 - \omega_2}{1-\beta}
$$

□

Consider the single-armed bandit process with subsidy λ under the infinite time horizon $T \to \infty$, we have:

$$
V_{\lambda,\infty}(\omega) = \max\begin{cases} \lambda + \omega + \beta V_{\lambda,\infty}(\tau^1(\omega)) & \text{passive} \\ \omega + \beta\left(\omega V_{\lambda,\infty}(P^a_{1,1}) + (1-\omega)V_{\lambda,\infty}(P^a_{0,1})\right) & \text{active} \end{cases} \tag{6.7}
$$

and we can get

$$
\frac{\partial V_{\lambda,\infty}(\omega)}{\partial \omega} = \begin{cases} 1 + \beta\frac{\partial V_{\lambda,\infty}(\tau^1(\omega))}{\partial \tau^1(\omega)}\frac{\partial \tau^1(\omega)}{\partial \omega} & \text{passive} \\ 1 + \beta\left(V_{\lambda,\infty}(P^a_{1,1}) - V_{\lambda,\infty}(P^a_{0,1})\right) & \text{active} \end{cases} \tag{6.8}
$$

Similarly, for the finite residual time horizon T we have:

$$\frac{\partial V_{\lambda,T}(\omega)}{\partial \omega} = \begin{cases} 1+\beta \frac{\partial V_{\lambda,T-1}(\tau^1(\omega))}{\partial \tau^1(\omega)} \frac{\partial \tau^1(\omega)}{\partial \omega} & \text{passive} \\ 1+\beta \left(V_{\lambda,T-1}(P^a_{1,1}) - V_{\lambda,T-1}(P^a_{0,1}) \right) & \text{active} \end{cases} \tag{6.9}$$

Note that for any belief state ω, $\tau^1(\omega)$ is the 1-step belief state update of ω when the passive arm is unobserved for another 1 consecutive slot. According to the Eq. 2.1, we have $\tau^1(\omega) = \omega P^p_{1,1} + (1-\omega)P^p_{0,1}$, thus

$$0 < \frac{\partial \tau^1(\omega)}{\partial \omega} = (P^p_{1,1} - P^p_{0,1}) < 1 \tag{6.10}$$

Lemma 5 *For the finite residual time horizon T, we have* $\frac{\partial V_{\lambda_T,T}(\omega)}{\partial \omega} \geq \min\{1 + \beta(P^p_{1,1} - P^p_{0,1}) \sum_{t=0}^{T-2} [\beta^t (P^a_{1,1} - P^a_{0,1})^t], 1 + \beta(P^a_{1,1} - P^a_{0,1}) \sum_{t=0}^{T-2} [\beta^t (P^a_{1,1} - P^a_{0,1})^t]\}$

Proof. According to Eq. 6.9, for the passive action, we have:

$$\begin{aligned} & 1+\beta \frac{\partial V_{\lambda,T-1}(\tau^1(\omega))}{\partial \tau^1(\omega)} \frac{\partial \tau^1(\omega)}{\partial \omega} \\ =& 1+\beta \lim_{\delta \to 0} \frac{V_{\lambda,T-1}(\tau^1(\omega)+\delta) - V_{\lambda,T-1}(\tau^1(\omega))}{\delta} (P^p_{1,1} - P^p_{0,1}) \end{aligned} \tag{6.11}$$

According to Lemma 3, let $\omega_1 = \tau^1(\omega) + \delta$ and $\omega_2 = \tau^1(\omega)$, then we have $V_{\lambda,T-1}(\tau^1(\omega) + \delta) - V_{\lambda,T-1}(\tau^1(\omega)) \geq (\omega_1 - \omega_2) \sum_{t=0}^{T-1} \beta^t (P^a_{1,1} - P^a_{0,1})^t = \delta \sum_{t=0}^{T-1} \beta^t (P^a_{1,1} - P^a_{0,1})^t$. Thus Eq. 6.11 becomes:

$$\begin{aligned} & 1+\beta \frac{\partial V_{\lambda,T-1}(\tau^1(\omega))}{\partial \tau^1(\omega)} \frac{\partial \tau^1(\omega)}{\partial \omega} \\ \geq& 1+\beta (P^p_{1,1} - P^p_{0,1}) \sum_{t=0}^{T-2} [\beta^t (P^a_{1,1} - P^a_{0,1})^t] \end{aligned} \tag{6.12}$$

Similarly, for the active action, according to lower bound in Lemma 3, we have:

$$\begin{aligned} & 1+\beta \left(V_{\lambda,T-1}(P^a_{1,1}) - V_{\lambda,T-1}(P^a_{0,1}) \right) \\ \geq& 1+\beta (P^a_{1,1} - P^a_{0,1}) \sum_{t=0}^{T-2} [\beta^t (P^a_{1,1} - P^a_{0,1})^t] \end{aligned} \tag{6.13}$$

Therefore, we have $\frac{\partial V_{\lambda_T,T}(\omega)}{\partial \omega} \geq \min\{(P^p_{1,1} - P^p_{0,1}), (P^a_{1,1} - P^a_{0,1})\} \cdot \beta \cdot \sum_{t=0}^{T-2} [\beta^t (P^a_{1,1} - P^a_{0,1})^t] + 1$

□

Lemma 6 *For the infinite residual time horizon $T \to \infty$, we have* $\frac{\partial V_{\lambda_T,T}(\omega)}{\partial \omega} \geq \min\{1 + \frac{\beta(P^p_{1,1} - P^p_{0,1})}{1-(\beta(P^a_{1,1} - P_{0,1})}, \frac{1}{1-\beta(P^a_{1,1} - P^a_{0,1})}\}$

Proof. The proof is similar to the proof for Lemma 5 of the finite setting. We can get the result with assuming $T \to \infty$. □

Lemma 7 *For the finite residual time horizon T, we have* $\frac{\partial V_{\lambda_T,T}(\omega)}{\partial \omega} \leq \min\{1 + (P^p_{1,1} - P^p_{0,1}) \sum_{t=1}^{T-1} \beta^t, 1 + (P^a_{1,1} - P^a_{0,1}) \sum_{t=1}^{T-1} \beta^t\}$

Proof. The proof is similar to the proof of Lemma 5. According to Eq. 6.9, we have:

- passive actions:

$$\begin{aligned} &1+\beta\frac{\partial V_{\lambda,T-1}(\tau^1(\omega))}{\partial\tau^1(\omega)}\frac{\partial\tau^1(\omega)}{\partial\omega} \\ =&1+\beta\lim_{\delta\to 0}\frac{V_{\lambda,T-1}(\tau^1(\omega)+\delta)-V_{\lambda,T-1}(\tau^1(\omega))}{\delta}(P^p_{1,1}-P^p_{0,1}) \end{aligned} \tag{6.14}$$

  According to Lemma 3, let $\omega_1 = \tau^1(\omega)+\delta$ and $\omega_2 = \tau^1(\omega)$, then we have $V_{\lambda,T-1}(\tau^1(\omega)+\delta) - V_{\lambda,T-1}(\tau^1(\omega)) \leq (\omega_1-\omega_2)\sum_{t=0}^{T-1}\beta^t = \delta\sum_{t=0}^{T-1}\beta^t$. Thus Eq. 6.14 becomes:

$$\begin{aligned} &1+\beta\frac{\partial V_{\lambda,T-1}(\tau^1(\omega))}{\partial\tau^1(\omega)}\frac{\partial\tau^1(\omega)}{\partial\omega} \\ \leq&1+\beta(P^p_{1,1}-P^p_{0,1})\sum_{t=0}^{T-2}[\beta^t] \end{aligned} \tag{6.15}$$

- active actions, similarly, according to upper bound in Lemma 3, we have:

$$\begin{aligned} &1+\beta\left(V_{\lambda,T-1}(P^a_{1,1})-V_{\lambda,T-1}(P^a_{0,1})\right) \\ \leq&1+\beta(P^a_{1,1}-P^a_{0,1})\sum_{t=0}^{T-2}[\beta^t] \end{aligned} \tag{6.16}$$

Therefore, we have $\frac{\partial V_{\lambda_T,T}(\omega)}{\partial\omega} \leq \min\{1+(P^p_{1,1}-P^p_{0,1})\sum_{t=1}^{T-1}\beta^t, 1+(P^a_{1,1}-P^a_{0,1})\sum_{t=1}^{T-1}\beta^t\}$

□

Lemma 8 *For the infinite residual time horizon $T\to\infty$, we have $\frac{\partial V_{\lambda_T,T}(\omega)}{\partial\omega} \leq \min\{1+\frac{\beta(P^p_{1,1}-P^p_{0,1})}{1-\beta}, 1+\frac{\beta(P^a_{1,1}-P^a_{0,1})}{1-\beta}\}$*

Proof. The proof is similar to the proof for Lemma 7 of the finite setting. We can get the result with assuming $T\to\infty$. □

6.1.2 Condition for the optimality of Algorithm 1 under infinite horizon

We now give the proof for the Theorem 1.

According to the Eq. 2.1, we can compute the belief gap between ω and 1-time step belief update $\tau^1(\omega)$:

$$\Delta\omega = \tau^1(\omega)-\omega = (P^p_{1,1}-P^p_{0,1}-1)\omega + P^p_{0,1} \tag{6.17}$$

Remark that we could get a strict condition that depends only on arm A. This is because change from policy π^* to π will only lead to a decrease in the value function for other arms as the optimal actions determined by the Whittle index algorithm will be influenced, henceforth the value will be decreased. Consider the single-arm A, as we discussed earlier, the belief state update process is either monotonically increasing or monotonically decreasing.

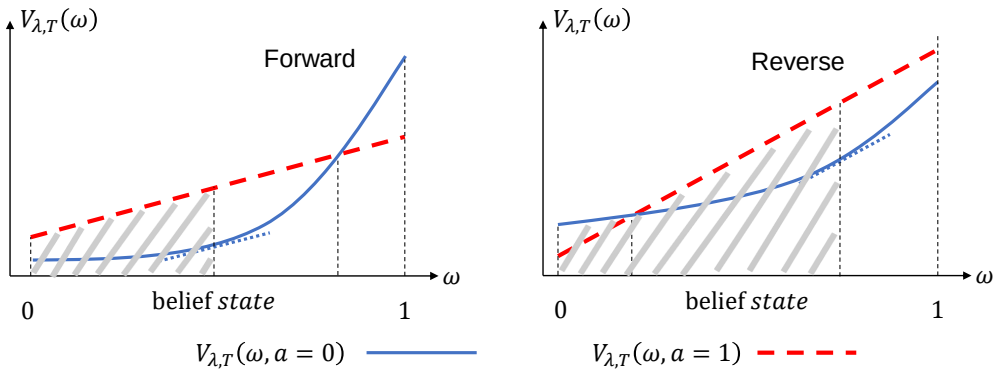


Figure 6.1: The forward and reverse policy

Case 1 : The belief state monotonically increases as the time passed. Formally, this can be expressed as $\frac{\partial \omega_t}{\partial t} > 0$, or $\Delta\omega > 0$. We now derive the condition for the optimality of our algorithm for the case 1 under finite time horizon T. Consider first (any) period of length L, an arm i has not been activated for the past $(L-1)$ time slots. Thus it needs to be pulled at time step $t = L$ according to our algorithm. Assume that the residual time horizon is h at time step $t = L$, where we have $h + L = T$. We move the active action at time step $t = L$ to one slot earlier, then at time step $t = L - 1$, the residual time horizon is $h + 1$, and assume that belief state is ω at time step $t = L - 1$. We here discuss the finite horizon case because the infinite horizon could be viewed as a special case of the finite horizon setting as $T \to \infty (h \to \infty)$.

Because the belief state will increase as the time passed, thus we define the value gap $\Delta V_{\lambda,h}(\omega) = V_{\lambda,h}(\omega, a = 1) - V_{\lambda,h}(\omega, a = 0)$ will move from left to the right as the residual time horizon decrease. For the single-arm process, if we can show that the gap difference $\Delta V_{\lambda,h}(\omega)$ increases from left to the right (i.e.,$\Delta V_{\lambda,h}(\omega)$ increases as belief state increases), then this implies that moving the active action that ensuring the fairness at time step $t = L$ to one step earlier (i.e., from right to left) will result in a smaller gap $\Delta V_{\lambda,h}(\omega)$. Thus it is optimal to keep the active action at the end of the period to ensure the fairness constraint. This requires that

$$\frac{\partial V_{\lambda,h}(\omega, a=0)}{\partial \omega} \leq \frac{\partial V_{\lambda,h}(\omega, a=1)}{\partial \omega} \tag{6.18}$$

According to the expression for λ, we have

$$\frac{\partial V_{\lambda,h}(\omega)}{\partial \omega} = \begin{cases} 1 + \beta \frac{\partial V_{\lambda,h-1}(\tau^1(\omega))}{\partial \tau^1(\omega)} \frac{\partial \tau^1(\omega)}{\partial \omega} & \text{passive} \\ 1 + \beta \left(V_{\lambda,h-1}(P^a_{1,1}) - V_{\lambda,h-1}(P^a_{0,1}) \right) & \text{active} \end{cases} \tag{6.19}$$

As shown in the gray area of the left Fig. 6.1, at time step $t = L - 1$, we derive the technical condition for the optimality of our algorithm in the gray area under the infinite

residual time horizon:

$$
\begin{aligned}
&(P^p_{1,1} - P^p_{0,1})\left(1 + \frac{\beta\Delta_3}{1-\beta}\right)\left(1 - \beta(P^a_{1,1} - P^a_{0,1})\right) \leq (P^a_{1,1} - P^a_{0,1}) \\
\rightarrow &(P^p_{1,1} - P^p_{0,1})\left(1 + \frac{\beta\Delta_3}{1-\beta}\right) \leq \frac{P^a_{1,1} - P^a_{0,1}}{1 - \beta(P^a_{1,1} - P^a_{0,1})} \text{ Line 1} \\
\rightarrow &(P^p_{1,1} - P^p_{0,1})\left(1 + \frac{\beta\Delta_3}{1-\beta}\right) \leq V_{\lambda,h-1}(P^a_{1,1}) - V_{\lambda,h-1}(P^a_{0,1}) \text{ Line 2} \\
\rightarrow &(P^p_{1,1} - P^p_{0,1})\frac{\partial V_{\lambda,h-1}(\tau^1(\omega))}{\partial \tau^1(\omega)} \leq V_{\lambda,h-1}(P^a_{1,1}) - V_{\lambda,h-1}(P^a_{0,1}) \text{ Line 3} \\
\rightarrow &\frac{\partial \tau^1(\omega)}{\partial \omega}\frac{\partial V_{\lambda,h-1}(\tau^1(\omega))}{\partial \tau^1(\omega)} \leq \left(V_{\lambda,h-1}(P^a_{1,1}) - V_{\lambda,h-1}(P^a_{0,1})\right) \text{ Line 4} \\
\rightarrow &1 + \beta\frac{\partial V_{\lambda,h-1}(\tau^1(\omega))}{\partial \tau^1(\omega)}\frac{\partial \tau^1(\omega)}{\partial \omega} \leq 1 + \beta\left(V_{\lambda,h-1}(P^a_{1,1}) - V_{\lambda,h-1}(P^a_{0,1})\right) \text{ Line 5} \\
\rightarrow &\frac{\partial V_{\lambda,h}(\omega, a=0)}{\partial \omega} \leq \frac{\partial V_{\lambda,h}(\omega, a=1)}{\partial \omega} \text{ Line 6}
\end{aligned}
\tag{6.20}
$$

Line 1 is obtained via mathematical transformation. Line 2 is obtained from the lower bound in Lemma 3. Line 3 is obtained from the Lemma 8 when assuming $h \to \infty$. Line 4 is obtained from Eq. 6.10. Line 5 is obtained from the mathematical transformation. Line 6 is obtained from the Eq. 6.19. And $\Delta_3 = \min\{(P^p_{1,1} - P^p_{0,1}), (P^a_{1,1} - P^a_{0,1})\}$

Similarly, we can derive the technical condition for the finite residual time horizon, which is

$$
(P^p_{1,1} - P^p_{0,1})\left(\Delta_4\beta\textstyle\sum_{t=0}^{h-2}[\beta^t] + 1\right) \leq (P^a_{1,1} - P^a_{0,1})\textstyle\sum_{t=0}^{h-2}[\beta^t(P^a_{1,1} - P^a_{0,1})^t] \tag{6.21}
$$

where $\Delta_4 = \min\{(P^p_{1,1} - P^p_{0,1}), (P^a_{1,1} - P^a_{0,1})\}$.

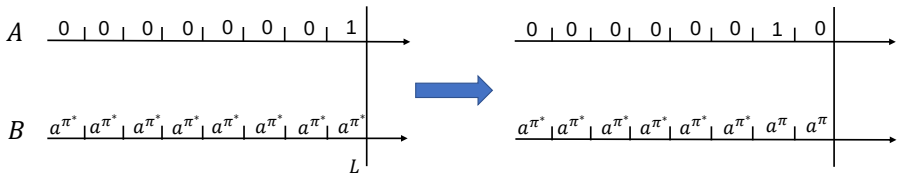


Figure 6.2: Proof of Theorem 1.

Case 2 : The belief state monotonically decreases as the time passed. Formally, this can be expressed as $\frac{\partial \omega_t}{\partial t} < 0$, or $\Delta\omega < 0$. Similarly, we can derive the condition for the optimality of our algorithm for the case 2 under finite time horizon T. Because the belief state will decrease as the time passed, thus we define the value gap $\Delta V_{\lambda,h}(\omega) = V_{\lambda,h}(\omega, a=1) - V_{\lambda,h}(\omega, a=0)$ will move from right to the left as the residual time horizon decrease. For the single-arm process, if we can show that the gap difference $\Delta V_{\lambda,h}(\omega)$ decreases from right to the left (i.e., as belief state decrease, $\Delta V_{\lambda,h}(\omega)$ decreases), then this implies that moving the active action that ensuring the fairness at time step $t = L$ to one step earlier will result in a larger gap $\Delta V_{\lambda,h}(\omega)$. Thus it is optimal to keep the active action at the end of the period, i.e., at time step $t = L$. this requires that

$$
\frac{\partial V_{\lambda,h}(\omega, a=0)}{\partial \omega} \geq \frac{\partial V_{\lambda,h}(\omega, a=1)}{\partial \omega} \tag{6.22}
$$

As shown in the gray area of the left Fig. 6.1, at time step $t = L - 1$, we derive the technical condition for the optimality of our algorithm in the gray area under the infinite residual time horizon:

$$
\begin{aligned}
&(P^p_{1,1} - P^p_{0,1})(1-\beta)\Delta_1 \geq (P^a_{1,1} - P^a_{0,1})\left(1 - \beta(P^a_{1,1} - P^a_{0,1})\right) \\
\rightarrow &\frac{(P^p_{1,1} - P^p_{0,1})\Delta_1}{1 - \beta(P^a_{1,1} - P^a_{0,1})} \geq \frac{P^a_{1,1} - P^a_{0,1}}{1-\beta} \text{ Line 1} \\
\rightarrow &(P^p_{1,1} - P^p_{0,1})\Delta_1 \lim_{h\to\infty} \sum_{t=0}^{h-2} \beta^t (P^a_{1,1} - P^a_{0,1})^t \geq \frac{P^a_{1,1} - P^a_{0,1}}{1-\beta} \text{ Line 2} \\
\rightarrow &(P^p_{1,1} - P^p_{0,1})\Delta_1 \lim_{h\to\infty} \sum_{t=0}^{h-2} \beta^t (P^a_{1,1} - P^a_{0,1})^t \geq V_{\lambda,h-1}(P^a_{1,1}) - V_{\lambda,h-1}(P^a_{0,1}) \text{ Line 3} \\
\rightarrow &(P^p_{1,1} - P^p_{0,1})\frac{\partial V_{\lambda,h-1}(\tau^1(\omega))}{\partial \tau^1(\omega)} \geq V_{\lambda,h-1}(P^a_{1,1}) - V_{\lambda,h-1}(P^a_{0,1}) \text{ Line 4} \\
\rightarrow &\frac{\partial \tau^1(\omega)}{\partial \omega}\frac{\partial V_{\lambda,h-1}(\tau^1(\omega))}{\partial \tau^1(\omega)} \geq \left(V_{\lambda,h-1}(P^a_{1,1}) - V_{\lambda,h-1}(P^a_{0,1})\right) \text{ Line 5} \\
\rightarrow &1 + \beta\frac{\partial V_{\lambda,h-1}(\tau^1(\omega))}{\partial \tau^1(\omega)}\frac{\partial \tau^1(\omega)}{\partial \omega} \geq 1 + \beta\left(V_{\lambda,h-1}(P^a_{1,1}) - V_{\lambda,h-1}(P^a_{0,1})\right) \text{ Line 6} \\
\rightarrow &\frac{\partial V_{\lambda,h}(\omega, a=0)}{\partial \omega} \geq \frac{\partial V_{\lambda,h}(\omega, a=1)}{\partial \omega} \text{ Line 7}
\end{aligned}
\tag{6.23}
$$

Line 1 is obtained via mathematical transformation. Line 2 is obtained from the formula for the geometric series as $\beta(P^a_{1,1} - P^a_{0,1}) < 1$. Line 3 is obtained from the upper bound in Lemma 3. Line 4 is obtained from the Lemma 5 when assuming $h \to \infty$. Line 5 is obtained from Eq. 6.10. Line 6 is obtained from the mathematical transformation. Line 7 is obtained from the Eq. 6.19.

Similarly, we can derive the technical condition for the finite residual time horizon, which is

$$
(P^p_{1,1} - P^p_{0,1})\left(\Delta_2\beta\textstyle\sum_{t=0}^{h-2}[\beta^t(P^a_{1,1} - P^a_{0,1})^t] + 1\right) \geq (P^a_{1,1} - P^a_{0,1})\textstyle\sum_{t=0}^{h-2}\beta^t \tag{6.24}
$$

where $\Delta_2 = \min\{(P^p_{1,1} - P^p_{0,1}), (P^a_{1,1} - P^a_{0,1})\}$.

When the belief state is in the white area of the passive set. Then we need to consider arm A and other arms in the active set. We give detailed discussion in Appendix.

6.1.3 Proof of Theorem 2

Proof. We provide our proof which is based on the work by Biswas et al. [10]. Let set ϕ^* to be the set of actions containing the k arms with the highest-ranking values of $Q_i(s, a = 1, l) - Q_i(s, a = 0, l)$, and any k arms that aren't among the top k are included in the set ϕ'. Let $\phi^{-,*}$ and $\phi^{-,\prime}$ denote the set that includes all of the arms except those in set ϕ^* and ϕ', respectively. We add the subscript i here in order to avoid ambiguity in the Q-values of distinct arms i at a given state. We could have:

$$
\begin{aligned}
&\sum_{i^*\in\phi^*}[Q^*_{i^*}(s_{i^*}, a_{i^*} = 1, l_{i^*}) - Q^*_{i^*}(s_{i^*}, a_{i^*} = 0, l_{i^*})] \geq \\
&\qquad \sum_{j\in\phi'}\left[Q^*_j(s_j, a_j = 1, l_j) - Q^*_j(s_j, a_j = 0, l_j)\right]
\end{aligned}
\tag{6.25}
$$

$$\sum_{i^* \in \phi^*} Q^*_{i^*}(s_{i^*}, a_{i^*} = 1, l_{i^*}) + \sum_{j \in \phi'} Q^*_j(s_j, a_j = 0, l_j) \geq \sum_{j \in \phi'} Q^*_j(s_j, a_j = 1, l_j) + \sum_{i^* \in \phi^*} Q^*_{i^*}(s_{i^*}, a_{i^*} = 0, l_{i^*}) \tag{6.26}$$

Adding $\sum_{i \notin \phi^* \& i \notin \phi'} Q^*_i(s_i, a_i = 0, l_i)$ on both sides,

$$\sum_{i^* \in \phi^*} Q^*_{i^*}(s_{i^*}, a_{i^*} = 1, l_{i^*}) + \sum_{j \in \phi^{-,*}} Q^*_j(s_j, a_j = 0, l_j) \geq \sum_{i \in \phi'} Q^*_i(s_i, a_i = 1, l_i) + \sum_{j \in \phi^{-,\prime}} Q^*_j(s_j, a_j = 0, l_j) \tag{6.27}$$

Thus from Equation 6.27, we can see that taking intervention action in the action set As can be seen from Equation 6.27, adopting intervention action for the arms in the set ϕ^* would maximizes $\left\{\sum_{i=1}^{N} Q^*_i(s, a, l)\right\}$. □

6.1.4 Proof of Theorem 3

Proof. The key to the convergence is contingent on a particular sequence of episodes observed in the real process [100]. The first condition is easy to be satisfied as to the presence of the fairness constraint. It is a reasonable assumption under the ϵ-greedy action selection mechanism, that any state-action pair can be visited an unlimited number of times as $T \to \infty$. The second condition has been well-studied in [30, 100, 33], and it guarantees that when the condition is met, the Q-value converges to the optimal $Q^*(s, a, l)$. As a result, $Q_i(s, a = 1, l) - Q_i(s, a = 0, l)$ converges to $Q^*_i(s, a = 1, l) - Q^*_i(s, a = 0, l)$. Also, $Q^*_i(s, a = 1, l) - Q^*_i(s, a = 0, l)$ is the calculated Q-Learning based Whittle index, and choosing top-ranked arms based on these values would lead to an optimal solution.

□

6.1.5 Additional Results

Fairness Constraints Strength In this part, we provide the average reward results for different fairness constraints, and see how they influence the overall performance. The strength of fairness restrictions is represented by the combination of L and η. For instance, η is a parameter to determine the lowest bound of the number of times an arm should be activated in a decision period of length L. Smaller L, on the other hand, indicates that a strict fairness constraint should be addressed in a shorter time length. For ease of explanation, we fix the value of $\eta = 2$, which means that an arm will be activated twice in any given time steps of length L. We can change the value of L to measure the fairness constraint level. We investigate three different categories of fairness constraint strength as follows,

- **Strong level**: The strong fairness constraints impose a strict restrictions on the action. Here we assume that the strong fairness constraints L satisfy $\frac{k \times L}{N} = 1.3$, this can translate to at most 30% arms can be engaged twice when before all arms have been pulled previously.
- **Medium level**: We define the medium fairness constraints by solving: $\frac{k \times L}{N} = 2$.

- **Low level**: The low strength of fairness constraints can be interpreted as a low fairness restriction on the distribution of the resources, i.e., we have $\frac{k \times L}{N} = 3$, which means all arms will receive the health intervention before each arm has been activated three times on average.

We provide the average reward results in Figure 6.3. Again, the left graph shows the performance of Whittle index approach with fairness constraint when the transition model is known, the middle graph presents the result of the Thompson sampling-based approach for Whittle index calculation, the the right graph shows the result for the Q-Learning based Whittle index approach. Our proposed approach can handle fairness constraints at different strength level without sacrificing significantly on the solution quality.

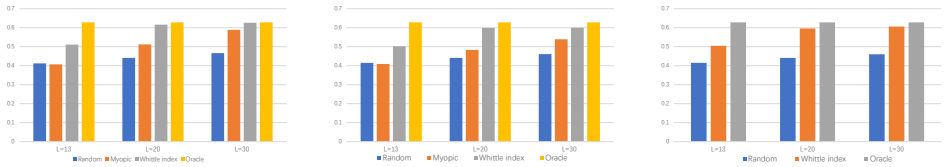


Figure 6.3: The average reward of each arm over the time length $T = 1000$ with a small penalty for the violation of the fairness constraint.

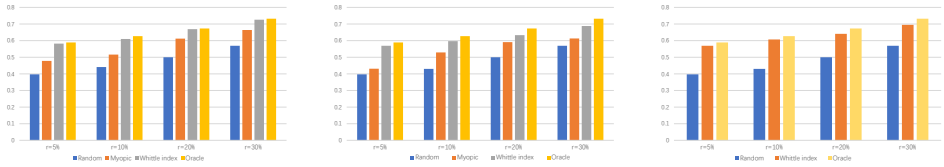


Figure 6.4: The average reward of each arm over the time length $T = 1000$ with a small penalty for the violation of the fairness constraint.

Intervention Level. In Figure 6.4, we present the performance results for various resource levels where the fairness constraint L is fixed, and we ensure that $k \times L < N$. Here, we let $L = 30$, and $N = 100$, and we're looking at the performance of the intervention ratio where $\frac{k}{N} = 5\%, 10\%, 20\%, 30\%$ respectively. We can see that our proposed approach to solve the fairness constraint can consistently outperform the Random and Myopic baselines regardless of the intervention strength while does not have significant differences when compared to the optimal value without taking fairness constraints into account.

6.2 Appendix for Chapter 2.4

6.2.1 More Details about *SoftFair*

We can also rewrite the update equation for the state-action value function:

$$Q_i^{t,ep}(s,a) = \begin{cases} R(s,a) + \sum\limits_{s_i^{t+1}} \Pr(s_i^{t+1}|s,a)(\gamma \sum\limits_{a'} \Pr(a'|\mathbf{s}^{t+1}) Q_i^{t+1,ep-1}(s_i^{t+1},a') & \text{if } (s,a) = (s_i^t, a_i^t) \\ Q_i^{t,ep-1}(s,a) & \text{otherwise} \end{cases} \tag{6.28}$$

The probability of choosing an arm is the *softmax* function on λ. We can write down the probability of $\pi(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{i\}})$, where i is the selected arms when $k = 1$. More specifically, we have $\pi(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{i\}}) = softmax_c(c \cdot \lambda_i)$, and note that $\Pr(a_i = 1|\mathbf{s}) = softmax_c(c \cdot \lambda_i)$ denote that probability that arm i is in the set of selected arms when $k = 1$.

When $k \neq 1$, let the ϕ denote the set of selected arms, and $\mathbb{I}_{\{\phi\}}$ denote the action to select arms in set ϕ while keeping other arms passive. We have $\pi(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi\}}) = \Pi_{i \in \phi} \Pr(a_i = 1|\mathbf{s})$, where $\Pr(a_i = 1|\mathbf{s})$ can be obtained through the brute-force permutation iteration over $\pi(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{i\}}) = softmax_c(c \cdot \lambda_i)$.

We rewrite our definition of fairness and the Proposition 1 here again.

Definition 4 *(Fairness) Equivalently, a stochastic policy, π is fair if for any time step $t \in [T]$, any joint state $\mathbf{s}$ and any two arms i, j, where $i \neq j$, The following two statements are equal:*

$$\begin{aligned} &\pi_t(\mathbf{s}, \mathbf{a}) \geq \pi_t(\mathbf{s}, \mathbf{a}') \text{ if and only if } Q^*(\mathbf{s}, \mathbf{a}) \geq Q^*(\mathbf{s}, \mathbf{a}') \\ \lambda_i \geq \lambda_j \text{ if and only if } &\sum_{\phi_i : \phi_i \in \Phi_i} \pi_t(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi_i\}}) \geq \sum_{\phi_j : \phi_j \in \Phi_j} \pi_t(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi_j\}}) \end{aligned} \tag{6.29}$$

Here $\Phi_i = \{\phi_i\}$ and $\Phi_j = \{\phi_j\}$ denote any set include arm i as the selected set.

Proposition 2 *Fairness of a stochastic policy defined in Equation 2.16 can also be stated in terms of arm selection as follows:*

$$Pr(a_i^t = 1) \geq Pr(a_j^t = 1) \text{ only if } \lambda_i^t \geq \lambda_j^t \tag{6.30}$$

A proof to show this two statements are equivalent is provided in next section (see Section 6.2.2). ***In summary, the goal of a solution approach is to generate a stochastic policy that never prefers one action over another if the cumulative long-term reward of selecting the latter one is higher.***

6.2.2 Proof of Proposition 2

We prove the two statements that are equal:

$$\begin{aligned} &\pi_t(\mathbf{s}, \mathbf{a}) \geq \pi_t(\mathbf{s}, \mathbf{a}') \text{ if and only if } Q^*(\mathbf{s}, \mathbf{a}) \geq Q^*(\mathbf{s}, \mathbf{a}') \\ &\lambda_i \geq \lambda_j \text{ if and only if } \sum_{\phi_i : \phi_i \in \Phi_i} \pi_t(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi_i\}}) \geq \sum_{\phi_j : \phi_j \in \Phi_j} \pi_t(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi_j\}}) \end{aligned} \tag{6.31}$$

Proof. For any set of selected arms, let ϕ_i denote a set that always include arm i, and ϕ_j denote a set that always include arm j, and $\phi_{i,j}$ denote a set that always include arm i and arm j. And Φ_i denote the set of ϕ_i, formally, we have $\Phi_i = \{\phi_i\}$ and $\Phi_j = \{\phi_j\}$ and $\Phi_{i,j} = \{\phi_{i,j}\}$. Similarly, let $\phi_{i,-j}$ denote the a that always include arm i and but not include arm j, and $\phi_{j,i}$ denote a set that always include arm j and but not include arm i. Thus we have:

$$\begin{aligned} \phi_i &= \phi_{i,-j} + \phi_{i,j} \text{ and } \phi_j = \phi_{j,-i} + \phi_{j,i} \\ \Phi_i &= \Phi_{j,-i} + \Phi_{i,j} \text{ and } \Phi_j = \Phi_{j,-i} + \Phi_{j,i} \end{aligned} \tag{6.32}$$

We divide this into two terms:

- 1st Term: $\Phi_{i,-j}$ and $\Phi_{j,-i}$. We can instead of consider a subset of $k-1$ selected arms which does not include arm i and arm j, we denote a subset of arms as $\phi'_{-i,-j} \in \Phi'_{-i,-j}$. Thus $\phi_{i,-j}$ and $\phi_{j,-i}$ can be writing as:

$$\begin{aligned} \phi_{i,-j} &= \{i\} + \phi'_{-i,-j} \text{ and } \phi_{j,-i} = \{j\} + \phi'_{-i,-j} \\ \Phi_{i,-j} &= \sum_{\phi'_{-i,-j} \in \Phi'_{-i,-j}} \left[\{i\} + \phi'_{-i,-j}\right] \text{ and } \Phi_{j,-i} = \sum_{\phi'_{-i,-j} \in \Phi'_{-i,-j}} \left[\{j\} + \phi'_{-i,-j}\right] \end{aligned} \tag{6.33}$$

  In this case, if $\lambda_i \geq \lambda_j$, we add $\sum_{h \in \phi'_{-i,-j}} \lambda_h$ on both side, which we get:

$$\begin{aligned} \lambda_i &\geq \lambda_j \\ \lambda_i + \sum_{h \in \phi'_{-i,-j}} \lambda_h &\geq \lambda_j + \sum_{h \in \phi'_{-i,-j}} \lambda_h \\ \sum_{i \in \phi_{i,-j}} \lambda_i &\geq \sum_{j \in \phi_{j,-i}} \lambda_j \end{aligned} \tag{6.34}$$

  According to Theorem 5 and Theorem 6, we can conclude that $\pi_t(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi_{i,-j}\}}) \geq \pi_t(\mathbf{s}, \mathbf{a}' = \mathbb{I}_{\{\phi_{j,-i}\}})$. This is equal to the first statement that $\pi_t(\mathbf{s}, \mathbf{a}) \geq \pi_t(\mathbf{s}, \mathbf{a}')$ if and only if $Q^*(\mathbf{s}, \mathbf{a}) \geq Q^*(\mathbf{s}, \mathbf{a}')$.

  Furthermore, because for $\forall \phi'_{-i,-j} \in \Phi'_{-i,-j}$, we have $\lambda_i + \sum_{h \in \phi'_{-i,-j}} \lambda_h \geq \lambda_j + \sum_{h \in \phi'_{-i,-j}} \lambda_h$, and by summation over second line of Equation 6.34 like Equation 6.33, we have:

$$\begin{aligned} \lambda_i + \sum_{h \in \phi'_{-i,-j}} \lambda_h &\geq \lambda_j + \sum_{h \in \phi'_{-i,-j}} \lambda_h \\ \sum_{\phi'_{-i,-j} \in \Phi'_{-i,-j}} \left\{ \lambda_i + \sum_{h \in \phi_{-i,-j}} \lambda_h \right\} &\geq \sum_{\phi'_{-i,-j} \in \Phi'_{-i,-j}} \left\{ \lambda_j + \sum_{h \in \phi'_{-i,-j}} \lambda_h \right\} \\ \sum_{\phi_{i,-j} \in \Phi_{i,-j}} \left\{ \sum_{i \in \phi_{i,-j}} \lambda_i \right\} &\geq \sum_{\phi_{j,-i} \in \Phi_{j,-i}} \left\{ \sum_{j \in \phi_{j,-i}} \lambda_j \right\} \text{ (according to Eq. 6.33)} \\ \rightarrow \sum_{\phi_{i,-j} \in \Phi_{i,-j}} \left\{ \pi_t(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi_{i,-j}\}}) \right\} &\geq \sum_{\phi_{j,-i} \in \Phi_{j,-i}} \left\{ \pi_t(\mathbf{s}, \mathbf{a}' = \mathbb{I}_{\{\phi_{j,-i}\}}) \right\} \end{aligned} \tag{6.35}$$

- 2nd Term: $\Phi_{i,j}$ and $\Phi_{j,i}$. Because we can easily see that $\phi_{i,j} = \phi_{j,i}$ and $\Phi_{i,j} = \Phi_{j,i}$, thus we have:

$$\sum_{\phi_{i,j} \in \Phi_{i,j}} \pi_t(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi_{i,j}\}}) = \sum_{\phi_{j,i} \in \Phi_{j,i}} \pi_t(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi_{j,i}\}}). \tag{6.36}$$

Overall, according to Equation 6.32, by summing this two terms (Equation 6.35 and Equation 6.36), we have $\lambda_i \geq \lambda_j$ if and only if $\sum_{\phi_i : \phi_i \in \Phi_i} \pi_t(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi_i\}}) \geq \sum_{\phi_j : \phi_j \in \Phi_j} \pi_t(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi_j\}})$. And this is equal to the first statement that $\pi_t(\mathbf{s}, \mathbf{a}) \geq \pi_t(\mathbf{s}, \mathbf{a}')$ if and only if $Q^*(\mathbf{s}, \mathbf{a}) \geq Q^*(\mathbf{s}, \mathbf{a}')$.

□

6.2.3 Proofs for Chapter 2.4.3

We begin with a lemma and then prove Theorem 4 based on the lemma.

Lemma 9 *Consider the single arm i with a finite horizon T, let $V^t_{m,i}(s_i)$ denote the value function start from time step $t \in [T]$ under the state s_i, we can have $V^t_{m,i,}(s_i) > V^{t+1}_{m,i}(s_i) \geq 0$, for $\forall s_i \in \{0,1\}$.*

Proof of Lemma 9 *Proof.* We drop the subscript i, i.e., $V^t(s) = V^t_i(s_i)$. This is easy to prove. For state $s \in \{0,1\}$, we can always find a algorithm that ensures $V^t_m(s) > V^{t+1}_m(s)$. For example, we assume the optimal algorithm for the state s start from the time step $t+1$ is π, we can always find a algorithm π': keep the same actions as the algorithm π until reach the last time step $t = T$ as $V^t_m(s)$ will has one more time slot compared to $V^{t+1}_m(s)$, and then we pick the action for the last time step T according to the observed state s'. Since the reward is either 0 or 1, thus $V^T(s) \geq 0$,so we can have

$$V^t_{m;\pi'}(s) = V^{t+1}_{m;\pi}(s) + \gamma^{T-t} V^T(s') \geq V^{t+1}_{m;\pi}(s). \tag{6.37}$$

□

For the ease of explanation, in the remaining appendix section, we use $P^a_{s,s'} = 1$ and $P^a_{s,s'} = 0$ to represent the probability from state s to s' under action $a = 1$ and $a = 0$, i.e., $P(s, a = 1, s')$ and $P(s, a = 0, s')$, respectively.

Proof of Theorem 4 *Proof.* Consider the discount reward criterion with discount factor of $\gamma \in [0,1]$ (where $\gamma = 1$ corresponds to the average criterion). Again, we drop the subscript i and let: $Q^t_i(s_i, a_i) = Q^t(s,a)$. Because the state $s \in \{0,1\}$ is fully observable, We can easily calculate m^T, where it needs to satisfy $Q^T(s, a = 0) = Q^T(s, a = 1)$, i.e., $m^T + P^0_{s,1} = P^1_{s,1}$, thus $m^T = P^1_{s,1} - P^0_{s,1}$. Similarly, m^{T-1} can be solved by assuming equation $Q^{T-1}(s, a = 0) = Q^{T-1}(s, a = 1)$:

$$\begin{aligned} P^0_{s,1} + m^{T-1} + \gamma(P^0_{s,1} V^T_{m^T}(1) + P^0_{s,0} V^T_{m^T}(0)) = \\ P^1_{s,1} + \gamma(P^1_{s,1} V^T_{m^T}(1) + P^1_{s,0} V^T_{m^T}(0)) \\ \rightarrow m^{T-1} = (P^1_{s,1} - P^0_{s,1}) + \gamma(V^T_{m^T}(1)(P^1_{s,1} - P^0_{s,1}) + V^T_{m^T}(0)(P^1_{s,0} - P^0_{s,0})) \end{aligned} \tag{6.38}$$

Because $P^1_{1,1} - P^0_{1,1} > 0$ and $P^1_{0,1} - P^0_{0,1} > 0$ from the structural constraint we mentioned before and $V^T_{m^T}(s) \geq 0$ according to Lemma 9, we have $m^{T-1} > m^T = P^1_{s,1} - P^0_{s,1}$. Now we show $m^t > m^{t+1}$. Because the state is fully observable, we first get the close form of m^t.

- `Case 1`: The state $s = 0$,

$$\begin{aligned} P^0_{0,1} + m^t + \gamma(P^0_{0,0} V^{t+1}_{m^{t+1}}(0) + P^0_{0,1} V^{t+1}_{m^{t+1}}(1)) \\ = P^1_{0,1} + \gamma(P^1_{0,0} V^{t+1}_{m^{t+1}}(0) + P^1_{0,1} V^{t+1}_{m^{t+1}}(1)) \\ \rightarrow m^t = (P^1_{0,1} - P^0_{0,1}) + \gamma(V^{t+1}_{m^{t+1}}(0)(P^1_{0,0} - P^0_{0,0}) + V^{t+1}_{m^{t+1}}(1)(P^1_{0,1} - P^0_{0,1})). \end{aligned} \tag{6.39}$$

  Similarly, we have

$$m^{t+1} = (P^1_{0,1} - P^0_{0,1}) + \gamma(V^{t+2}_{m^{t+2}}(0)(P^1_{0,0} - P^0_{0,0}) + V^{t+2}_{m^{t+2}}(1)(P^1_{0,1} - P^0_{0,1})). \tag{6.40}$$

  Thus $m^t - m^{t+1} = \gamma((V^{t+1}_{m^{t+1}}(0) - V^{t+2}_{m^{t+2}}(0))(P^1_{0,0} - P^0_{0,0}) + (V^{t+1}_{m^{t+1}}(1) - V^{t+2}_{m^{t+2}}(1))(P^1_{0,1} - P^0_{0,1}))$

Intuitively, we can have $V^{t+1}_{m^{t+1}}(0) > V^{t+2}_{m^{t+2}}(0)$ (see Lemma 9), and $V^{t+1}_{m^{t+1}}(1) > V^{t+2}_{m^{t+2}}(1)$, Hence, we can get $m^t > m^{t+1}$

- `Case 2`: For state $s = 1$, similarly, we can get $m^t - m^{t+1} = \gamma((V^{t+1}_{m^{t+1}}(0) - V^{t+2}_{m^{t+2}}(0))(P^1_{1,0} - P^0_{1,0}) + (V^{t+1}_{m^{t+1}}(1) - V^{t+2}_{m^{t+2}}(1))(P^1_{1,1} - P^0_{1,1}))$

Thus $\forall t < T : m_t > m_{t+1} \geq m_T = P^1_{s,1} - P^0_{s,1}$.

□

Proof of Theorem 5 *Proof.* According to the Equation 2.18, we have

$$\pi_i(s_i, a_i) = e^{Q_i(s_i,a_i)-V_i(s_i)} = \frac{e^{Q_i(s_i,a)}}{e^{V_i(s_i)}} \tag{6.41}$$

By replacing this into Equation 2.19, we can get

$$\begin{aligned}
\lambda_i &= \log \pi_i(s_i, a_i = 1) - \log \pi_i(s_i, a_i = 0) \\
&= \log\left(\frac{e^{Q_i(s_i,a_i=1)}}{e^{V_i(s_i)}}\right) - \log\left(\frac{e^{Q_i(s_i,a_i=0)}}{e^{V_i(s_i)}}\right) \\
&= \log\left(\frac{e^{Q_i(s_i,a_i=1)}}{e^{Q_i(s_i,a_i=0)}}\right) \\
&= Q_i(s_i, a_i = 1) - Q_i(s_i, a_i = 0).
\end{aligned} \tag{6.42}$$

Because as c approaches infinity, our algorithm becomes deterministically selecting the arm with the highest value of λ. Let set ϕ^* to be the set of actions containing the k arms with the highest-ranking of λ value, and any k arms that aren't among the top k are included in the set ϕ'. Let $\phi^{-,*}$ and $\phi^{-,\prime}$ denote the set that includes all of the arms except those in set ϕ^* and ϕ', respectively. Thus the first action vector can be represented as $\mathbf{a} = \mathbb{I}_{\{\phi\}}$, and the latter action vector is $\mathbf{a}' = \mathbb{I}_{\{\phi'\}}$. We could have:

$$\begin{aligned}
\sum_{i\in\phi^*} \lambda_i &\geq \sum_{j\in\phi'} \lambda_j \\
\sum_{i\in\phi^*} [Q(s_i, a_i = 1) - Q(s_i, a_i = 0)] &\geq \sum_{j\in\phi'} [Q(s_j, a_i = 1) - Q(s_j, a_i = 0)] \\
\sum_{i\in\phi^*} Q(s_i, a_i = 1) + \sum_{j\in\phi'} Q(s_j, a_i = 0) &\geq \sum_{j\in\phi'} Q(s_j, a_i = 1) + \sum_{i\in\phi^*} Q(s_i, a_i = 0)
\end{aligned} \tag{6.43}$$

Adding $\sum_{z\notin\phi^*\wedge z\notin\phi'} Q(s_z, a_i = 0)$ on both sides, we can have,

$$\begin{aligned}
\sum_{i\in\phi^*} Q(s_i, a_i = 1) + \sum_{j\in\phi^{-,*}} Q(s_j, a_i = 0) &\geq \sum_{i\in\phi'} Q(s_i, a_i = 1) + \sum_{j\in\phi^{-,\prime}} Q(s_j, a_i = 0) \\
\rightarrow Q(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi\}}) &\geq Q(\mathbf{s}, \mathbf{a}' = \mathbb{I}_{\{\phi'\}})
\end{aligned} \tag{6.44}$$

Thus from this we can see that selecting action in the action set ϕ^* according to the λ value can maximize the cumulative long-term reward, which lead to the optimal state-action value function $Q^*(\mathbf{s}, \mathbf{a})$ as well as the optimal value function, $V^*(\mathbf{s})$ □

Proof of Theorem 6 *Proof.* According to the Equation. 2.19, the probability of choosing an arm is the *softmax* function on λ, which can guarantee that the higher the value of λ, the higher the probability of selecting that arm. We can write down the probability of $\pi(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi\}})$, where ϕ is the set of selected arms when $k \neq 1$. More specifically, $\pi(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi\}}) = \Pi_{i \in \phi} \Pr(a_i = 1|\mathbf{s})$. Note that $\Pr(a_i = 1|\mathbf{s}) = softmax_c(c \cdot \lambda_i)$ if $k = 1$. Intuitively, when $k \neq 1$, $\Pr(a_i = 1|\mathbf{s})$ can be obtained through the brute-force permutation iteration over $\pi(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{i\}}) = softmax_c(c \cdot \lambda_i)$. It is easy to conclude that when $k \neq 1$, the higher the value of λ_i, the higher probability $\Pr(a_i = 1|\mathbf{s})$. Formally, this is equivalent to:

$$\pi(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi\}}) \geq \pi(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi'\}}) \text{ if and only if } \sum_{i \in \phi} \lambda_i \geq \sum_{j \in \phi'} \lambda_j. \tag{6.45}$$

Similar to the proof of Theorem 6.2.3, we can have:

$$\sum_{i \in \phi} \lambda_i \geq \sum_{j \in \phi'} \lambda_j \text{ if and only if } Q(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi\}}) \geq Q(\mathbf{s}, \mathbf{a}' = \mathbb{I}_{\{\phi'\}}). \tag{6.46}$$

Thus we have $\pi(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi\}}) \geq \pi(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi'\}})$ if $Q(\mathbf{s}, \mathbf{a} = \mathbb{I}_{\{\phi\}}) \geq Q(\mathbf{s}, \mathbf{a}' = \mathbb{I}_{\{\phi'\}})$. Our *SoftFair* is fair under our proposed fairness constraint.

The trade-off is governed by c, where a larger c means *SoftFair* tends to choose arms with higher value, while a small c means *SoftFair* tends to ensure fairness among arms. More specifically, we have shown that selecting the top k arms according to the λ value at each time step t when c approaches infinity is equivalent to maximizing the cumulative long-term reward (Theorem 5). When c is close to 0, the difference between $c \cdot \lambda$ is small and *SoftFair* tends to uniformly sample k arms. Therefore *SoftFair* remains fair under our proposed fairness constraints, and the trade-off between fairness and optimal value is controlled by the multiplier parameter c. □

6.2.4 Proofs for Chapter 2.4.3

Proof of Lemma 1 *Proof.* The upper bound can be obtained by showing that $\forall(\mathbf{s}, \mathbf{a})$, state-action value at the $ep-$th iteration are bounded. More specifically,

$$Q^{ep}(\mathbf{s}, \mathbf{a}) = \sum_{i=1}^{n} Q_i^{ep}(s_i, a_i) \leq n \cdot \sup_i Q_i^{ep}(s_i, a_i) = n \sum_{t=0}^{T} \gamma^t R_{max} \tag{6.47}$$

We can prove this Equation 6.47 through induction as follows, When $t = 1$, we start from the definition of our *SoftFair* in Equation 2.21 to have

$$\begin{aligned}
Q^1(\mathbf{s}, \mathbf{a}) &= \Psi_{soft} Q^0(\mathbf{s}, \mathbf{a}) \\
&= \sum_{\mathbf{a}} \Pr(\mathbf{a}|\mathbf{s}) \sum_{\mathbf{s}'_{t+1}} \Pr(\mathbf{s}'_{t+1}|\mathbf{s}, \mathbf{a})(R(\mathbf{s}, \mathbf{a}) + \gamma V^0(\mathbf{s}')) \\
&= \sum_{\mathbf{a}} \Pr(\mathbf{a}|\mathbf{s}) \sum_{\mathbf{s}'_{t+1}} \Pr(\mathbf{s}'_{t+1}|\mathbf{s}, \mathbf{a})(R(\mathbf{s}, \mathbf{a}) + \gamma \max_{\mathbf{a}'} Q^0(\mathbf{s}', \mathbf{a}')) \\
&\leq R_{max} + \sum_{\mathbf{s}'_{t+1}} \Pr(\mathbf{s}'_{t+1}|\mathbf{s}, \mathbf{a})(\gamma \max_{\mathbf{a}'} Q^0(\mathbf{s}', \mathbf{a}')) \\
&\leq R_{max} + \gamma \sum_{\mathbf{s}'_{t+1}} \Pr(\mathbf{s}'_{t+1}|\mathbf{s}, \mathbf{a}) R_{max} \\
&= (1 + \gamma) R_{max}
\end{aligned} \tag{6.48}$$

We then assume that $ep = K$, where $K > 1$, $Q^K(\mathbf{s}, \mathbf{a}) \leq \sum_{t=0}^{K} \gamma^t R_{max}$ holds, then we have

$$\begin{aligned}
Q^{K+1}(\mathbf{s}, \mathbf{a}) &= \Psi_{soft} Q^K(\mathbf{s}, \mathbf{a}) \\
&= \sum_{\mathbf{a}} \Pr(\mathbf{a}|\mathbf{s}) \sum_{\mathbf{s}'_{t+1}} \Pr(\mathbf{s}'_{t+1}|\mathbf{s}, \mathbf{a})(R(\mathbf{s}, \mathbf{a}) + \gamma V^K(\mathbf{s}')) \\
&\leq \sum_{\mathbf{a}} \Pr(\mathbf{a}|\mathbf{s}) \sum_{\mathbf{s}'_{t+1}} \Pr(\mathbf{s}'_{t+1}|\mathbf{s}, \mathbf{a})(R_{max} + \gamma \max_{\mathbf{a}'} Q^K(\mathbf{s}', \mathbf{a}')) \\
&\leq R_{max} + \gamma \sum_{\mathbf{s}'_{t+1}} \Pr(\mathbf{s}'_{t+1}|\mathbf{s}, \mathbf{a})(\max_{\mathbf{a}'} Q^K(\mathbf{s}', \mathbf{a}')) \\
&\leq R_{max} + \gamma \sum_{\mathbf{s}'_{t+1}} \Pr(\mathbf{s}'_{t+1}|\mathbf{s}, \mathbf{a}) \sum_{t=0}^{K} \gamma^t R_{max} \\
&= \sum_{t=0}^{K+1} \gamma^t R_{max}
\end{aligned} \tag{6.49}$$

$R(\mathbf{s}, \mathbf{a}) \in [R_{min}, R_{max}]$ where $R_{max} = n$ as we have n arms, and $R_{min} = 0$. Thus the upper bound is $\frac{n}{1-\gamma}$. Similarly, we can prove the lower bound is 0. we can conclude that the state-action value function is bounded within $[0, \frac{n}{1-\gamma}]$. □

Proof of Lemma 2 *Proof.* let $\mathbf{a}_{\{i\}}$ denote the action to select the arm i and remain other arms in passive, which is the shorthand for $\mathbf{a} = \mathbb{I}_{\{i\}}$. We first sort $Q(\mathbf{s}, \mathbf{a}_{\{i\}})$ in the ascending order according to the λ value. Assume we get $Q(\mathbf{s}, \mathbf{a}'_{\{1\}'}) \geq \cdots \geq Q(\mathbf{s}, \mathbf{a}'_{\{n\}'})$ after sorting, and corresponding $\Pr(\cdot|\mathbf{s})$ becomes $[\Pr(\mathbf{a} = \mathbb{I}_{\{1\}'}|\mathbf{s}), \ldots, \Pr(\mathbf{a} = \mathbb{I}_{\{n\}'}|\mathbf{s})]^\top$. According to Equation 2.19, when $k = 1$, we have $\Pr(a_i = 1|\mathbf{s}) = \textit{softmax}_c(c \cdot \lambda_i)$

Then $\forall Q$ and $\forall s$, we can get

$$\begin{aligned}
&\max_{\mathbf{a}} Q(\mathbf{s}, \mathbf{a}) - (\Pr(\cdot|\mathbf{s}))^\top Q(\mathbf{s}, \cdot) = \max_{\mathbf{a}} Q(\mathbf{s}, \mathbf{a}) - \textit{softmax}_c^\top(c \cdot \lambda_{i'}))Q(\mathbf{s}, \cdot) \\
=&Q(\mathbf{s}, \mathbf{a}'_{\{1\}'})) - \frac{\sum_{i=1}^{n} \exp[c \cdot \lambda_{i'}] \cdot Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})}{\sum_{i=1}^{n} \exp[c \cdot \lambda_{i'}]} \\
=&\frac{\sum_{i=1}^{n} \exp[c \cdot \lambda_{i'}] \cdot [Q(\mathbf{s}, \mathbf{a}'_{\{1\}'}) - Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})]}{\sum_{i=1}^{n} \exp[c \cdot \lambda_{i'}]}
\end{aligned} \tag{6.50}$$

According to Equation 2.18, we can get $\lambda_{i'} = Q(s_{i'}, a_{i'} = 1) - Q(s_{i'}, a_{i'} = 0)$. Let $K = \exp[\sum_{j=1}^{n} Q(s_j, a_j = 0)]$, and we have

$$K \cdot \exp[c \cdot \lambda_{i'}] = \exp[Q(s_{i'}, a_{i'} = 1) + \sum_{j \in \phi^{-.i'}} Q(s_j, a_j = 0)] = \exp[Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})]. \tag{6.51}$$

Here $\phi^{-.i'}$ denote the set that include all of the arms except arm i'. Thus by applying Equation 6.51 to Equation 6.50, we have

$$\begin{aligned}
&\frac{\sum_{i=1}^{n} \exp[c \cdot \lambda_{i'}] \cdot [Q(\mathbf{s}, \mathbf{a}'_{\{1\}'}) - Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})]}{\sum_{i=1}^{n} \exp[c \cdot \lambda_{i'}]} \\
=&\frac{K \cdot \sum_{i=1}^{n} \exp[c \cdot \lambda_{i'}] \cdot [Q(\mathbf{s}, \mathbf{a}'_{\{1\}'}) - Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})]}{K \cdot \sum_{i=1}^{n} \exp[c \cdot \lambda_{i'}]} \\
=&\frac{\sum_{i=1}^{n} \exp[c \cdot Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})] \cdot [Q(\mathbf{s}, \mathbf{a}'_{\{1\}'}) - Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})]}{\sum_{i=1}^{n} \exp[c \cdot Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})]}
\end{aligned} \tag{6.52}$$

Let $\delta_{\{i\}'}(\mathbf{s}) = Q(\mathbf{s}, \mathbf{a}'_{\{1\}'}) - Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})$, and we have $\delta_{\{i\}'}(\mathbf{s}) \geq 0$, and $\delta_{\{1\}'}(\mathbf{s}) = 0$. Using Equation 6.52, we have

$$\begin{aligned}
&\frac{\sum_{i=1}^{n} \exp[c \cdot Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})] \cdot [Q(\mathbf{s}, \mathbf{a}'_{\{1\}'}) - Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})]}{\sum_{i=1}^{n} \exp[c \cdot Q(\mathbf{s}, \mathbf{a}'_{\{i\}'})]} \\
=&\frac{\sum_{i=1}^{n} \exp\left[c \cdot \left(Q(\mathbf{s}, \mathbf{a}'_{\{1\}'}) - \delta_{\{i\}'}(\mathbf{s})\right)\right] \cdot \delta_{\{i\}'}(\mathbf{s})}{\sum_{i=1}^{n} \exp\left[c \cdot \left(Q(\mathbf{s}, \mathbf{a}'_{\{1\}'}) - \delta_{\{i\}'}(\mathbf{s})\right)\right]} \\
=&\frac{\sum_{i=1}^{n} \exp[-c \cdot \delta_{\{i\}'}(\mathbf{s})] \cdot \delta_{\{i\}'}(\mathbf{s})}{\sum_{i=1}^{n} \exp[-c \cdot \delta_{\{i\}'}(\mathbf{s})]} \\
=&\frac{\sum_{i=2}^{n} \exp[-c \cdot \delta_{\{i\}'}(\mathbf{s})] \cdot \delta_{\{i\}'}(\mathbf{s})}{1 + \sum_{i=2}^{n} \exp[-c \cdot \delta_{\{i\}'}(\mathbf{s})]}
\end{aligned} \tag{6.53}$$

Now from equation 6.53, we can derive the upper bound. We follow the work in [88], we take advantage of the fact that for any two non-negative sequences $\{x_i\}$ and $\{y_i\}$,

$$\frac{\sum_i x_i}{1 + \sum_i y_i} \leq \sum_i \frac{x_i}{1 + y_i} \tag{6.54}$$

Apply Equation 6.54 to Equation 6.53

$$\begin{aligned}
\frac{\sum_{i=2}^{n} \exp[-c \cdot \delta_{\{i\}'}(\mathbf{s})] \cdot \delta_{\{i\}'}(\mathbf{s})}{1 + \sum_{i=2}^{n} \exp[-c \cdot \delta_{\{i\}'}(\mathbf{s})]} &\leq \sum_{i=2}^{n} \frac{\exp[-c \cdot \delta_{\{i\}'}(\mathbf{s})] \cdot \delta_{\{i\}'}(\mathbf{s})}{1 + \exp[-c \cdot \delta_{\{i\}'}(\mathbf{s})]} \\
&= \sum_{i=2}^{n} \frac{\delta_{\{i\}'}(\mathbf{s})}{1 + \exp[c \cdot \delta_{\{i\}'}(\mathbf{s})]}
\end{aligned} \tag{6.55}$$

Intuitively, we have $0 \leq \delta_{\{i\}'}(\mathbf{s}) \leq 1$, thus through using Taylor series, we can rewrite Equation 6.2.4 as

$$\begin{aligned}
&\sum_{i=2}^{n} \frac{\delta_{\{i\}'}(\mathbf{s})}{1 + \exp[c \cdot \delta_{\{i\}'}(\mathbf{s})]} \\
=&\sum_{i=2}^{n} \frac{\delta_{\{i\}'}(\mathbf{s})}{1 + 1 + c \cdot \delta_{\{i\}'}(\mathbf{s}) + 0.5c^2 \cdot \delta_{\{i\}'}(\mathbf{s})^2 + \ldots} \\
\leq&\sum_{i=2}^{n} \frac{1}{2 + c} = \frac{n-1}{2 + c}
\end{aligned} \tag{6.56}$$

We can also get the lower bound as

$$\begin{aligned}
&\frac{\sum_{i=2}^{n} \exp[-c \cdot \delta_{\{i\}'}(\mathbf{s})] \cdot \delta_{\{i\}'}(\mathbf{s})}{1 + \sum_{i=2}^{n} \exp[-c \cdot \delta_{\{\}'}(\mathbf{s})]} \geq \frac{\sum_{i=2}^{n} \exp[-c \cdot \delta_{\{i\}'}(\mathbf{s})] \cdot \delta_{\{i\}'}(\mathbf{s})}{n} \\
=& \frac{\sum_{i=2}^{n} \delta_{\{i\}'}(\mathbf{s})}{n \exp[c \cdot \delta_{\{i\}'}(\mathbf{s})]} \geq \frac{\sum_{i=2}^{n} \delta_{\{i\}'}(\mathbf{s})}{n \exp[c \cdot \delta(\mathbf{s})]} \geq \frac{\delta(\mathbf{s})}{n \exp[c \cdot \delta(\mathbf{s})]}
\end{aligned} \tag{6.57}$$

□

Proof of Theorem 7 *Proof.* The proof is similar to the work by Song et al. [88] but modified for our RMAB setting.

Upper bound : We derive the upper bound through induction. We start from the definition for Ψ and Ψ_{soft} in Equation 2.26 and Equation 2.25. When $ep = 1$, we have

$$\begin{aligned}
&\Psi Q^0(\mathbf{s},\mathbf{a}) - \Psi_{soft} Q^0(\mathbf{s},\mathbf{a}) \\
=&\gamma \sum_{\mathbf{s}'} \Pr(\mathbf{s}'|\mathbf{s},\mathbf{a})[\max_{\mathbf{a}'} Q^0(\mathbf{s}',\mathbf{a}') - Pr(\mathbf{a}'|s')Q^0(\mathbf{s}',\mathbf{a}')] \\
\geq& 0
\end{aligned} \tag{6.58}$$

Assume $\Psi Q^K(\mathbf{s},\mathbf{a}) - \Psi_{soft} Q^K(\mathbf{s},\mathbf{a}) \geq 0$ holds for $ep = K$ where $K > 1$. When $ep = K + 1$, we have

$$\begin{aligned}
&\Psi Q^{K+1}(\mathbf{s},\mathbf{a}) - \Psi_{soft} Q^{K+1}(\mathbf{s},\mathbf{a}) \\
=&\Psi \Psi^K Q^0(\mathbf{s},\mathbf{a}) - \Psi_{soft} \Psi^K_{soft} Q^0(\mathbf{s},\mathbf{a}) \\
\geq&\Psi \Psi^K_{soft} Q^0(\mathbf{s},\mathbf{a}) - \Psi_{soft} \Psi^K_{soft} Q^0(\mathbf{s},\mathbf{a}) \\
\geq& 0
\end{aligned} \tag{6.59}$$

Since $\lim_{ep\to\infty} \Psi^K Q^0(\mathbf{s},\mathbf{a}) = Q^*(\mathbf{s},\mathbf{a})$, where $Q^*(\mathbf{s},\mathbf{a})$ is the optimal state action value, and thus $\lim_{ep\to\infty} \Psi^K V^0(\mathbf{s}) = V^*(\mathbf{s})$, where $V^*(\mathbf{s})$ is the optimal value function. Therefore, we have $\limsup_{ep\to\infty} \Psi^K Q^0(\mathbf{s},\mathbf{a}) \leq Q^*(\mathbf{s},\mathbf{a})$ and $\limsup_{ep\to\infty} \Psi^K V^0(\mathbf{s}) \leq V^*(\mathbf{s})$.

Lower bound : Similarly, we derive the lower bound through induction from the definition for Ψ and Ψ_{soft} in Equation 2.26 and Equation 2.25. When $ep = 1$, we have

$$\begin{aligned}
&\Psi Q^0(\mathbf{s},\mathbf{a}) - \Psi_{soft} Q^0(\mathbf{s},\mathbf{a}) \\
=&\gamma \sum_{\mathbf{s}'} \Pr(\mathbf{s}'|\mathbf{s},\mathbf{a})[\max_{\mathbf{a}'} Q^0(\mathbf{s}',\mathbf{a}') - Pr(\mathbf{a}'|s')Q^0(\mathbf{s}',\mathbf{a}')] \\
\leq&\gamma \sum_{\mathbf{s}'} \Pr(\mathbf{s}'|\mathbf{s},\mathbf{a})\eta = \gamma\eta = \gamma \frac{n-1}{2+c}
\end{aligned} \tag{6.60}$$

where η is the upper bound of $\max_{\mathbf{a}'} Q^0(\mathbf{s}',\mathbf{a}') - Pr(\mathbf{a}'|\mathbf{s}')Q^0(\mathbf{s}',\mathbf{a}')$, can be written as $\eta = \sup_{\mathbf{s}'}[\max_{\mathbf{a}'} Q^0(\mathbf{s}',\mathbf{a}') - Pr(\mathbf{a}'|\mathbf{s}')Q^0(\mathbf{s}',\mathbf{a}')]$. According to Lemma 2, we have $\eta = \frac{n-1}{2+c}$. Then we assume $\Psi Q^K(\mathbf{s},\mathbf{a}) - \Psi_{soft} Q^K(\mathbf{s},\mathbf{a}) \leq \sum_{i=1}^K \gamma^i \frac{n-1}{2+c}$ holds for $ep = K$ where $K > 1$. When $ep = K + 1$, we have

$$\begin{aligned}
&\Psi Q^{K+1}(\mathbf{s},\mathbf{a}) - \Psi_{soft} Q^{K+1}(\mathbf{s},\mathbf{a}) \\
=&\Psi \Psi^K Q^0(\mathbf{s},\mathbf{a}) - \Psi_{soft} \Psi^K_{soft} Q^0(\mathbf{s},\mathbf{a}) \\
\leq&\Psi \left(\Psi^K_{soft} Q^0(\mathbf{s},\mathbf{a}) + \sum_{i=1}^K \gamma^i \frac{n-1}{2+c} \right) - \Psi_{soft} \Psi^K_{soft} Q^0(\mathbf{s},\mathbf{a}) \\
=& \sum_{i=2}^{K+1} \gamma^i \frac{n-1}{2+c} + (\Psi - \Psi_{soft}) \Psi^K_{soft} Q^0(\mathbf{s},\mathbf{a}) \\
\leq& \sum_{i=2}^{K+1} \gamma^i \frac{n-1}{2+c} + \gamma^i \frac{n-1}{2+c} \\
=& \sum_{i=1}^{K+1} \gamma^i \frac{n-1}{2+c}
\end{aligned} \tag{6.61}$$

Thus we have

$$\lim_{K\to\infty} \Psi Q^K(\mathbf{s},\mathbf{a}) - \Psi_{soft} Q^K(\mathbf{s},\mathbf{a}) \leq \lim_{K\to\infty} \sum_{i=1}^{K} \gamma^i \frac{n-1}{2+c} = \frac{n-1}{(2+c)(1-\gamma)} \quad (6.62)$$

□

6.2.5 Datasets

Realistic dataset : Obstructive sleep apnea (OSA) is a common disorder, and CPAP is considered the gold standard of treatment, effectively addressing a range of adverse outcomes. However, effectiveness is often limited by non-adherence. CPAP adherence varies substantially across settings; 29% to 83% of patients enrolled in the study reported using CPAP recommended hours per night [101].

Our CPAP dataset is provided by Kang et al. [40], they define states as the CPAP usage levels (0: did not adhere, 1: adhere), and estimate transition probabilities between CPAP usage levels to build a Markov chain. They divide patients into two groups where patients in the first cluster exhibit 'adherent' behavior, they are most likely to transition to and remain in a good CPAP usage state, while the 'non-adherent' patient type we consider in the second cluster shows a weak trend to transition to good CPAP usage state.

We consider an intervention effect broadly characterizing supportive interventions such as text message reminders, telemonitoring and telephone support, which is associated with a random increase from 5-50% to good CPAP use status per night for both two groups. We add a small noise to each transition matrix so that the dynamics of each individual arm in same group is different. The initial state vector $\mathbf{s}$ is randomly assigned.

For the ease of explanation, we use $P^a_{s,s'} = 1$ and $P^a_{s,s'} = 0$ to represent the probability from state s to s' under action $a = 1$ and $a = 0$, i.e., $P(s, a = 1, s')$ and $P(s, a = 0, s')$, respectively.

Synthetic dataset : For the synthetic dataset, it is natural to require strict positive transition matrix entries, and in order to mimic the real-world setting, following [56], we impose four structural constraints: (i) $P^0_{0,1} < P^0_{1,1}$, (ii) $P^1_{0,1} < P^1_{1,1}$, (iii) $P^0_{0,1} < P^1_{0,1}$ and (iv) $P^0_{1,1} < P^1_{1,1}$. Those constraints imply that arms are more likely to stay in a good state when there is positive intervention involved (active action) compared to no intervention (passive action).

6.3 Appendix for Chapter 4

6.4 Theorem

Theorem 8 *There exists a finite evaluation environment set that can capture the student's general capabilities and the performance vector $[p_1, \ldots, p_m]$ is a good representation of the student policy.*

To prove this, we first provide the following Assumption:

Assumption 1 *Let $p(\pi, \vec{\theta})$ denote the performance of student policy π in an environment $\vec{\theta}$. For $\forall i$-th dimension of the environment parameters, denoted as θ_i, when changing the θ_i to θ'_i to get a new environment $\vec{\theta'}$ while keeping other environment parameters fixed, there $\exists \delta_i > 0$, if $|\theta'_i - \theta_i| \leq \delta_i$, we have $|p(\pi, \vec{\theta'}) - p(\pi, \vec{\theta})| \leq \epsilon_i$, where $\epsilon_i \to 0$.*

If this is true, we then can construct a finite set of environments, and the student performances in those environments can represent the performances in all potential environments generated within the certain environment parameters open interval combinations, and the set of those open intervals combinations cover the environment parameter space Θ.

We begin from the simplest case where we only consider using one environment parameter to generate environments, denoted as θ_i. We can construct a finite environment parameter set for environment parameters, which is $\{\theta_i^{min} + 1/2 * \delta_i, \theta_i^{min} + 3/2 * \delta_i, \theta_i^{min} + 7/2 * \delta_i, \ldots, \theta_i^{max} - \delta_i/2\}$. Assume the set size is L_i. We let the set $\{\vec{\theta_i}\}_{i=1}^{L_i}$ denote the corresponding generated environments. This is served as the **representative environment set**. Then the student performances in those environments are denoted as $\{p(\pi, \vec{\theta_i})\}_{i=1}^{L_i}$, which we call it as **representative performance vector set**. We can divide the space for θ_i into a finite set of open intervals with size L_i, which is $\{[\theta_i^{min}, \theta_i^{min} + 3/2 * \delta_i), (\theta_i^{min} + 1/2 * \delta_i, \theta_i^{min} + 5/2\delta_i), (\theta_i^{min} + 5/2 * \delta_i, \theta_i^{min} + 9/2 * \delta_i), \ldots, (\theta_i^{max} - 3/2 * \delta_i, \theta_i^{max}]\}$, which we call it as **representative parameter interval set**, also denoted as $\{(\theta_i - \delta, \theta_i + \delta)\}_{i=1}^{L_i}$. For any environment generated in those intervals, denoted as $\vec{\theta_i'}$, the performance $p(\pi, \vec{\theta_i'})$ can always be represented by the $p(\pi, \vec{\theta_i})$ which is in the same interval, as $|p(\pi, \vec{\theta_i'}) - p(\pi, \vec{\theta_i})| \leq \epsilon_i$, where $\epsilon_i \to 0$. In such cases, the finite set of environmental parameter intervals $\{\theta_i^{min} + 1/2 * \delta_i, \theta_i^{min} + 3/2 * \delta_i, \theta_i^{min} + 7/2 * \delta_i, \ldots, \theta_i^{max} - \delta_i/2\}$ fully covers the entire parameter space Θ. We can find a representative environment set $\{\vec{\theta_i}\}_{i=1}^{L_i}$ that is capable of approximating the performance of the student policy within the open parameter intervals combination. This set effectively characterizes the general performance capabilities of the student policy π.

Then we extend to two environment parameter design space cases. Let's assume that the environment is generated by two-dimension environment parameters. Then, for each environment parameter, $\theta_i \in \{\theta_1, \theta_2\}$. We can find the same open interval set for each parameter. Specifically, for each θ_i, there exists a δ_i, such that if $|\theta_i' - \theta_i| \leq \delta_i$, we have $|p(\pi, \vec{\theta'}) - p(\pi, \vec{\theta})| \leq \epsilon_i$, where $\epsilon_i \to 0$. Hence, we let $\delta = \min\{\delta_1, \delta_2\}$ and $\epsilon = \epsilon_1 + \epsilon_2$. Thus the new **representative environment set** is the set that includes the any combination of $\{[\theta_1, \theta_2]\}$ where $\theta_1 \in \{\vec{\theta_i}\}_{i=1}^{L_1}$ and $\theta_2 \in \{\vec{\theta_j}\}_{j=1}^{L_2}$. We can get the **representative performance vector set** as $\{p(\pi, [\vec{\theta_i}, \vec{\theta_j}])\}_{i \in [1, L_1], j \in [1, L_2]}$. We then can construct the **representative parameter interval set** as $\{[(\theta_i - \delta, \theta_i + \delta), (\theta_j - \delta, \theta_j + \delta)]\}_{i \in [1, L_1], j \in [1, L_j]}$. As a result, for any new environments $[\vec{\theta_i'}, \vec{\theta_j'}]$, we can find the representative environment whose environment parameters are in the same parameter interval $[\vec{\theta_i}, \vec{\theta_j}]$, such that their performance difference is smaller than $\epsilon = \epsilon_1 + \epsilon_2$ for all $\forall i \in [1, L_1], \forall j \in [1, L_2]$:

$$
\begin{aligned}
|p(\pi, [\vec{\theta_i'}, \vec{\theta_j'}]) - p(\pi, [\vec{\theta_i}, \vec{\theta_j}])| &= |p(\pi, [\vec{\theta_i'}, \vec{\theta_j'}]) - p(\pi, [\vec{\theta_i'}, \vec{\theta_j}]) + p(\pi, [\vec{\theta_i'}, \vec{\theta_j}]) - p(\pi, [\vec{\theta_i}, \vec{\theta_j}])| \\
&\leq |p(\pi, [\vec{\theta_i'}, \vec{\theta_j'}]) - p(\pi, [\vec{\theta_i'}, \vec{\theta_j}])| + |p(\pi, [\vec{\theta_i'}, \vec{\theta_j}]) - p(\pi, [\vec{\theta_i}, \vec{\theta_j}])| \\
&\leq \delta_j + \delta_i \\
&= \delta
\end{aligned}
\tag{6.63}
$$

In such cases, the finite set of environmental parameter intervals $\{[(\theta_i - \delta, \theta_i + \delta), (\theta_j - \delta, \theta_j + \delta)]\}_{i \in [1, L_1], j \in [1, L_j]}$ fully covers the entire parameter space Θ. We can find a representative environment set $\{\vec{\theta_i}\}_{i=1}^{L_i}$ that is capable of approximating the performance of the student policy within the open parameter intervals combination. This set effectively characterizes the general performance capabilities of the student policy π.

Similarly, we can show this still holds when the environment is constructed by a larger dimension environment parameters, where we set $\delta = \min\{\delta_i\}$, and $\epsilon = \sum_i \epsilon_i$, and we have $\delta > 0$, $\epsilon \to 0$. The overall logic is that we can find a finite set, which is called **representative environment set**, and we can use performances in this set to represent any performances in the environments generated in the **representative parameter interval set**, which is called **representative performance vector set**. Finally, we can show that **representative parameter interval set** fully covers the environment parameter space. Thus there exists a finite evaluation environment set that can capture the student's general capabilities and the performance vector, called **representative performance vector set**, $[p_1, \ldots, p_m]$ is a good representation of the student policy.

6.5 Details about the Generative model

6.5.1 Generative model to generate synthetic next state

Here, we describe how to leverage the diffusion model to learn the conditional data distribution in the collected experiences $\tau = \{(s_t^u, a_t^u, r_t^u, s_t^{u,\prime})\}$. Later we can use the trainable reverse chain in the diffusion model to generate the synthetic trajectories that can be used to help train the teacher agent, resulting in reducing the resource-intensive and time-consuming collection of upper-level teacher experiences. We deal with two different types of timesteps in this section: one for the diffusion process and the other for the upper-level teacher agent, respectively. We use subscripts $k \in 1, \ldots, K$ to represent diffusion timesteps and subscripts $t \in 1, \ldots, T$ to represent trajectory timesteps in the teacher's experience.

In the image domain, the diffusion process is implemented across all pixel values of the image. In our setting, we diffuse over the next state $s^{u,\prime}$ conditioned the given state s^u and action a^u. We construct our generative model according to the conditional diffusion process:

$$q(s_k^{u,\prime}|s_{k-1}^{u,\prime}), \quad p_\phi(s_{k-1}^{u,\prime}|s_k^{u,\prime}, s^u, a^u)$$

As usual, $q(s_k^{u,\prime}|s_{k-1}^{u,\prime})$ is the predefined forward noising process while $p_\phi(s_{k-1}^{u,\prime}|s_k^{u,\prime}, s^u, a^u)$ is the trainable reverse denoising process. We begin by randomly sampling the collected experiences $\tau = \{(s_t^u, a_t^u, r_t^u, s_t^{u,\prime})\}$ from the real experience buffer $\mathcal{B}_{real}$.

We drop the superscript u here for ease of explanation. Giving the observed state s and action a, we use the reverse process p_ϕ to represent the generation of the next state s':

$$p_\phi(s'_{0:K}|s, a) = \mathcal{N}(s'_K; 0, \mathbf{I}) \prod_{k=1}^{K} p_\phi(s'_{k-1}|s'_k, s, a) \tag{6.64}$$

At the end of the reverse chain, the sample s'_0, is the generated next state s'. As shown in Section 4.2.2, $p_\phi(s'_{k-1}|, s'_k, s, a)$ could be modeled as a Gaussian distribution $\mathcal{N}(s'_{k-1}; \mu_\theta(s'_k, s, a, k), \Sigma_\theta(s'_k, s, a, k))$. Similar to Ho et al. [31], we parameterize $p_\phi(s'_{k-1}|s'_k, s, a)$ as a noise prediction model with the covariance matrix fixed as

$$\Sigma_\theta(s'_k, s, a, k) = \beta_i \mathbf{I}$$

and mean is

$$\mu_\theta(s'_i, s, a, k) = \frac{1}{\sqrt{\alpha_k}} \left(s'_k - \frac{\beta_k}{\sqrt{1 - \bar{\alpha}_k}} \epsilon_\theta(s'_k, s, a, k) \right)$$

Where $\epsilon_\theta(s'_k, s, a, k)$ is the trainable denoising function, which aims to estimate the noise ϵ in the noisy input s'_k at step k. Specifically, giving the sampled experience (s, a, s'), we begin by sampling $s'_K \sim \mathcal{N}(0, \mathbf{I})$ and then proceed with the reverse diffusion chain $p_\phi(s'_{k-1}|, s'_k, s, a)$ for $k = K, \ldots, 1$. The detailed expression for s'_{k-1} is as follows:

$$\frac{s'_k}{\sqrt{\alpha_k}} - \frac{\beta_k}{\sqrt{\alpha_k(1-\bar{\alpha}_k)}}\epsilon_\theta(s'_k, s, a, k) + \sqrt{\beta_k}\epsilon, \tag{6.65}$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. Note that $\epsilon = 0$ when $k = 1$.

Training objective. We employ a similar simplified objective, as proposed by Ho et al. [31] to train the conditional ϵ- model through the following process:

$$\mathcal{L}(\theta) = \mathbb{E}_{(s,a,s')\sim\tau, k\sim\mathcal{U}, \epsilon\sim\mathcal{N}(0,\mathbf{I})}\left[\|\epsilon - \epsilon_\phi(s'_k, s, a, k)\|^2\right] \tag{6.66}$$

Where $s'_k = \sqrt{\bar{\alpha}_k}s' + \sqrt{1-\bar{\alpha}_k}\epsilon$. $\mathcal{U}$ represents a uniform distribution over the discrete set $\{1, \ldots, K\}$. The intuition for the loss function $\mathcal{L}(\theta)$ tries to predict the noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ at the denoising step k, and the diffusion model is essentially learning the student policy involution trajectories collected in the real experience buffer $\mathcal{B}_{reals}$. Note that the reverse process necessitates a substantial number of steps K, as the Gaussian assumption holds true primarily under the condition of the infinitesimally limit of small denoising steps [86]. Recent research by Xiao et al. [107] has demonstrated that enabling denoising with large steps can reduce the total number of denoising steps K. To expedite the relatively slow reverse sampling process outlined in Equation 4.3.2 (as it requires computing ϵ_ϕ networks K times), we use a small value of K, while simultaneously setting $\beta_{\min} = 0.1$ and $\beta_{\max} = 10.0$. Similar to Wang et al. [99], we define:

$$\begin{aligned}\beta_k &= 1 - \alpha_k \\ &= 1 - \exp\left(\beta_{\min} \times \frac{1}{K} - 0.5(\beta_{\max} - \beta_{\min})\frac{2k-1}{K^2}\right)\end{aligned}$$

This noise schedule is derived from the variance-preserving Stochastic Differential Equation by Song et al. [87].

Generate synthetic trajectories. Once the diffusion model has been trained, it can be used to generate synthetic experience data by starting with a draw from the prior $s'_K \sim \mathcal{N}(0, \mathbf{I})$ and successively generating denoised next state, conditioned on the given s and a through the reverse chain p_ϕ in Equation 4.3.2. Note that the giving condition action a can either be randomly sampled from the action space (which is also the environment parameter space) or use another diffusion model to learn the action distribution giving the initial state s. In such case, this new diffusion model is essentially a behavior-cloning model that aims to learn the teacher policy $\Lambda(a|s)$. This process is similar to the work of Wang et al. [99]. We discuss this process in detail in the appendix. In this paper, we randomly sample a as it is straightforward and can also increase the diversity in the generated synthetic experience to help train a more robust teacher agent.

6.5.2 Generative model to generate synthetic action

Once the diffusion model has been trained, it can be used to generate synthetic experience data by starting with a draw from the prior $s'_K \sim \mathcal{N}(0, \mathbf{I})$ and successively generating denoised next state, conditioned on the given s and a through the reverse chain p_ϕ in

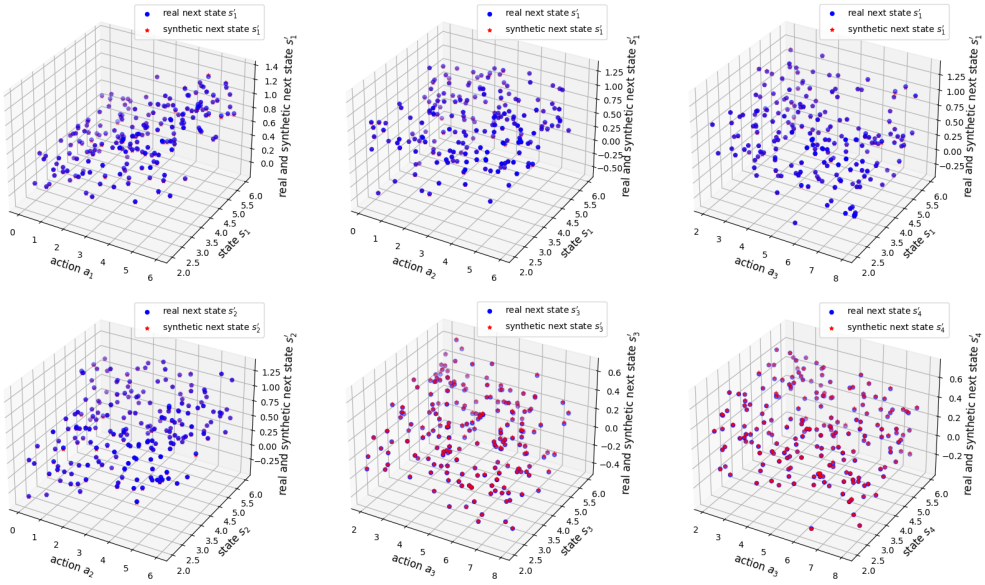


Figure 6.5: The distribution of the real s' and the synthetic s' conditioned on (s, a).

Equation 4.3.2. Note that the giving condition action a can either be randomly sampled from the action space (which is also the environment parameter space) or we can train another diffusion model to learn the action distribution giving the initial state s, and then use the trained new diffusion model to sample the action a giving the state s. This process is similar to the work of Wang et al. [99].

In particular, We construct another conditional diffusion model as:

$$q(a_k|a_{k-1}), \quad p_\phi(a_{k-1}|a_k, s)$$

As usual, $q(a_k|a_{k-1})$ is the predefined forward noising process while $p_\phi(a_{k-1}|a_k, s)$ is the trainable reverse denoising process. we represent the action generation process via the reverse chain of the conditional diffusion model as

$$p_\phi(a_{0:K}|s) = \mathcal{N}(a_K; 0, \mathbf{I}) \prod_{k=1}^{K} p_\phi(a_{k-1}|a_k, s) \tag{6.67}$$

At the end of the reverse chain, the sample a_0, is the generated action a for the giving state s. Similarly, we parameterize $p_\phi(a_{k-1}|a_k, s)$ as a noise prediction model with the covariance matrix fixed as

$$\Sigma_\theta(a_k, s, k) = \beta_i \mathbf{I}$$

and mean is

$$\mu_\theta(a_i, s, k) = \frac{1}{\sqrt{\alpha_k}} \left(a_k - \frac{\beta_k}{\sqrt{1-\bar{\alpha}_k}} \epsilon_\theta(a_k, s, k) \right)$$

Similarly, the simplified loss function is

$$\mathcal{L}^a(\theta) = \mathbb{E}_{(s,a)\sim\tau, k\sim\mathcal{U}, \epsilon\sim\mathcal{N}(0,\mathbf{I})} \left[\|\epsilon - \epsilon_\phi(a_k, s, k)\|^2 \right] \tag{6.68}$$

Where $a_k = \sqrt{\bar{\alpha}_k} a + \sqrt{1-\bar{\alpha}_k}\epsilon$. $\mathcal{U}$ represents a uniform distribution over the discrete set $\{1, \ldots, K\}$. The intuition for the loss function $\mathcal{L}^a(\theta)$ tries to predict the noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ at the denoising step k, and the diffusion model is essentially a

behavior cloning model to learn the student policy collected in the real experience buffer $\mathcal{B}_{reals}$.

Once this new diffusion model is trained, the generation of the synthetic experience can be formulated as:

- we first randomly sample the state from the collected real trajectories $s \sim \tau$;
- we use the new diffusion model discussed above to mimic the teacher's policy to generate the actions a;
- giving the state s and action a, we use the first diffusion model presented in the main paper to generate the next state s';
- we compute the reward r according to the reward function, and add the final generated synthetic experience (s, a, r, s') to the synthetic experience buffer $\mathcal{B}_{syn}$ to help train the teacher agent.

6.6 Empirical analysis of generative model

6.6.1 Ability to generate good synthetic trajectories

We begin by investigating *SHED*'s ability to assist in collecting experiences for the upper-level MDP teacher. This involves the necessity for *SHED* to prove its ability to accurately generate synthetic experiences for teacher agents. To check the quality of these generated synthetic experiences, we employ a diffusion model to simulate some data for validation (even though Diffusion models have demonstrated remarkable success across vision and NLP tasks).

We design the following experiment: given the teacher's observed state $s^u = [p_1, p_2, p_3, p_4, p_5]$, where p_i denotes the student performance on i-th evaluation environment. and given the teacher's action $a^u = [a_1, a_2, a_3]$, which is the environment parameters and are used to generate corresponding environment instances. We use a neural network $f(s^u, a^u)$ to mimic the involution trajectories of the student policy π. That is, with the input of the state s^u and action a^u into the neural network, it outputs the next observed state $s^{u,\prime} = [p'_1, p'_2, p'_3, p'_4, p'_5]$, indicating the updated student performance vector on the evaluation environments after training in the environment generated by a^u. In particular, we add a noise ε into $s^{u,\prime}$ to represent the uncertainty in the transition. We first train our diffusion model on the real dataset $(s^u, a^u, s^{u,\prime})$ generated by neural network $f(s^u, a^u)$. We then set a fixed (s^u, a^u) pair and input them into $f(s^u, a^u)$ to generate 200 samples of real $s^{u,\prime}$. The trained diffusion model is then used to generate 200 synthetic $s^{u,\prime}$ conditioned on the fixed (s^u, a^u) pair.

The results are presented in Figure 6.6, we can see that the generative model can effectively capture the distribution of real experience even if there is a large uncertainty in the transition, indicated by the value of ε. This provides evidence that the diffusion model can generate useful experiences conditioned on (s^u, a^u). It is important to note that the marginal distribution derived from the reverse diffusion chain provides an implicit, expressive distribution, such distribution has the capability to capture complex distribution properties, including skewness and multi-modality.

6.6.2 addition experiments on diffusion model

We further provide more results to show the ability of our generative model to generate synthetic trajectories where the noise is extremely small. In such cases, the actual next

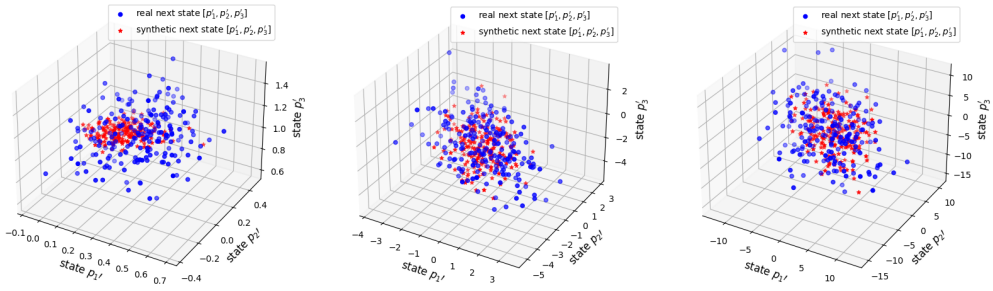


Figure 6.6: The distribution of the real $[s'_1, s'_2, s'_3]$(red) and the synthetic $[s'_1, s'_2, s'_3]$(blue) giving the fixed (s^u, a^u). Specifically, the noise ε in $f(s^u, a^u)$ is (i).*left* figure: $\varepsilon = \epsilon$, (ii).*middle* figure: $\varepsilon = 3 * \epsilon$, (iii).*right* figure: $\varepsilon = 10 * \epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$.

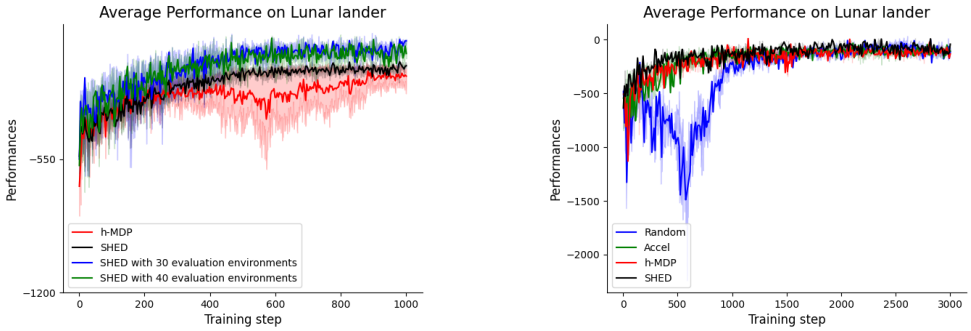


Figure 6.7: *Left*: The ablation study in the Lunar lander environment which investigates the effect of the size of the evaluation environment set. We provide the average zero-shot transfer performances on the test environments (mean and standard error). *Right*: Zero-shot transfer performance on the test environments under a longer time horizon in Lunar lander environments(mean and standard error).

state s' will converge to a certain value, and the synthetic next state $s^{syn,\prime}$ generated by the diffusion model should also be very close to that value, then the diffusion model has the ability to sample the next state $s_0^{syn,\prime}$ which can accurately represent the next state. We present the results in Figure 6.5. Specifically, this figure shows when the noise is very small in the actual next state, which is $0.05 * \epsilon$, and $\epsilon \sim \mathcal{N}(0, 1)$. Giving any condition (s, a) pair, we selectively report on (s_i, a_i), where x-axis is the a_i value, and y-axis is the s_i value. The student policy with initial performance vector s is trained on the environments generated by the teacher's action a. We report the new performance s'_i of student policy on i-th environments after training in the z-axis. In particular, if two points s'_i and $s_i^{syn,\prime}$ are close, it indicates that the diffusion model can successfully generate the actual next state. As we can see, when the noise is extremely small, our diffusion model can accurately predict the next state of s'_i giving any condition (s, a) pair.

6.7 Additional Experiment Details

6.7.1 Hyperparameters

We set the learning rate $1e-3$ for actor, and $3e-3$ for critic, we set gamma $\gamma = 0.999$, $\lambda = 0.95$, and set coefficient for the entropy bonus (to encourage exploration) as 0.01. For each environment, we conduct 50 PPO updates for the student agent, and We can train on up to 50 environments, including replay. For our diffusion model, the diffusion discount is 0.99, and batch size is 64, τ is 0.005, learning rate is $3e-4$. The synthetic buffer size is 1000, and the ratio is 0.25.

6.7.2 Experiments Compute Resources

All the models were trained on a single NVIDIA GeForce RTX 3090 GPU and 16 CPUs.

6.7.3 Maze document

Here we provide the document shows the instruction to generate feasible maze environments.

```
There are several factors that can affect the difficulty
of a maze. Here are some key factors to consider:
1. Maze Size: Larger mazes generally increase the complexity
and difficulty as the agent has more states to explore.
Typically, the maze size should be larger than 4x4 and
smaller than 15*15.
- If the size is 7*7 or smaller, the maze size is considered
easy.
- If the size is larger than 7*7 but smaller than 10*10,
the maze size is considered medium.
- If the maze size is larger than 10x10 but smaller than
15*15, the maze size is considered hard.
2. Maze Structure: The complexity of the paths, including
the number of twists, turns, and dead-ends, can significantly
impact navigation strategies. The presence of narrow
corridors versus wide-open spaces also plays a role.
- If there are fewer than 2 turns in the feasible path
from the start position  to the end position, the maze
structure is considered easy.
- If there are more than 2 turns but fewer than 4 turns
in the path from the start position to the end position,
the maze structure is considered medium.
- If there are 4 or more turns in the path from the start
position to the end position, the maze structure is considered
hard.
3. Goal Location: The distance from the starting position to
the end position also affects difficulty.
- If the path from the start position to the end position
requires fewer than 5 steps, the goal location is
considered easy.
- If the path from the start position to the end position
```

```
requires 5 to 10steps, the goal location is considered medium.
- If the path from the start position to the end position
requires more than 10 steps, the goal location is
considered hard.
4. Start Location: The starting position can also affect the
difficulty of the maze. The starting position is
categorized into five levels:
- If the start position is close to 1, it means it should be
located as close to the top left of the maze.
- If the start position is close to 2, it means it should be
located as close to the top right of the maze.
- If the start position is close to 3, it means it should be
located as close to the bottom left of the maze.
- If the start position is close to 4, it means it should be
located as close to the bottom right of the maze.
- If the start position is close to 5, it means it should be
located as close to the center of the maze.
Please note that the generated maze uses -1 to represent
blocks, 0 to represent the feasible path, 1 to represent
the start position, and 2 to represent the end position.
Must ensure that there is a feasible path in the generated
maze! A feasible path means that 1 and 2 are connected
directly through 0s, or 1 and 2 are connected directly.
For example: Feasible Maze:
Maze = [
  [0, -1, -1, 2],
  [1, -1, 0, 0],
  [0, -1, 0, -1],
  [0, 0, 0, -1],
]
Non-Feasible Mazes:
Maze = [
  [0, -1, -1, 2],
  [1, -1, 0, 0],
  [0, -1, -1, 0],
  [0, 0, 0, -1],
]
Or
Maze = [
  [1, -1],
  [-1, 2]
]
These second example does not have any feasible path.
```

6.7.4 Prompt for RAG

We provide our prompt for the Retrieval Augmented Generation as follows:

```
Please refer to the document, and generate a maze with a
```

```
feasible path. The difficulty level for the maze size is
{maze_size_level}, and the difficulty level for the maze
structure is {maze_structure_level}, the difficulty level
for the goal location is {goal_location_level}, the difficulty
level for the start location is {start_position_level}.
```

6.8 Additional experiments

6.8.1 Additional experiments about ablation studies

We also provide ablation analysis to evaluate the impact of different design choices in Lunar lander domain, including (a) a larger evaluation environment set; (b) a bigger budget for constraint on the number of generated environments (which incurs a longer training time horizon). The results are reported in Figure 6.7.

We explore the impact of introducing the diffusion model in collecting synthetic teacher's experience and varying the size of the evaluation environment set. Specifically, as we can see from the right side of Figure 6.7, the *SHED* consistently outperforms h-MDP, indicating the effectiveness of introducing the generative model to help train the upper-level teacher policy. Furthermore, we find that when increasing the size of the evaluation environment set, we can have a better result in the student transfer performances. The intuition is that a larger evaluation environment set, encompassing a more diverse range of environments, provides a better approximation of the student policy according to the Theorem 8. However, the reason why *SHED* with 30 evaluation environments slightly outperforms *SHED* with 40 evaluation environments is perhaps attributed to the increase in the dimension of the student performance vector, which amplifies the challenge of training an effective diffusion model with a limited dataset.

We conduct experiments in Lunar lander under a longer time horizon. The results are provided on the right side of Figure 6.7. As we can see, our proposed algorithm *SHED* can efficiently train the student agent to achieve the general capability in a shorter time horizon, This observation indicates that our proposed environment generation process can better generate the suitable environments for the current student policy, thereby enhancing its general capability, especially when there is a constraint on the number of generated environments.

6.8.2 Additional experiments on Lunar lander

we also conduct experiments to show how the algorithm performs under different settings, such as a larger weight of cv fairness rewards ($\eta = 10$). The results are provided in Figure 6.8. We noticed an interesting finding: when fairness reward has a high weightage, our algorithm tends to generate environments at the onset that lead to a rapid decline and subsequent improvement in student performance across all test environments. This is done to avoid acquiring a substantial negative fairness reward and thereby maximize the teacher's cumulative reward. Notably, the student's final performance still surpasses other baselines at the end of training.

We further show in detail how the performance of different methods changes in each testing environment during training (see Figure 6.9 and Figure 6.10).

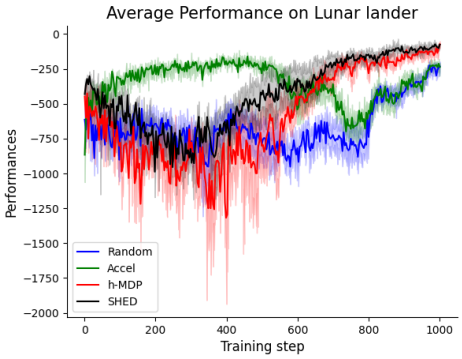


Figure 6.8: Zero-shot transfer performance on the test environments with a larger cv value coefficient in Lunar lander environments.

6.8.3 Additional experiments on Maze

We selectively report some results of zero-shot transfer performances in maze environments. The results are provided in Figure

6.9 Discussion

6.9.1 Limitations

The limitation of this work comes from the UED framework, as UED is limited to the use of parameterized environments. This results in our experimental domain being relatively simple. However, our work proposes a new hierarchical structure, and our policy representation is not only of great help for UED, but also has certain inspirations for hierarchical RL. Additionally, in the world model of UED (Genie [12]), the environment generator (teacher) focuses on creating video games, a domain that is compatible with our proposed application of upsampling the teacher agent's experience using a diffusion model (since the state is image-based).

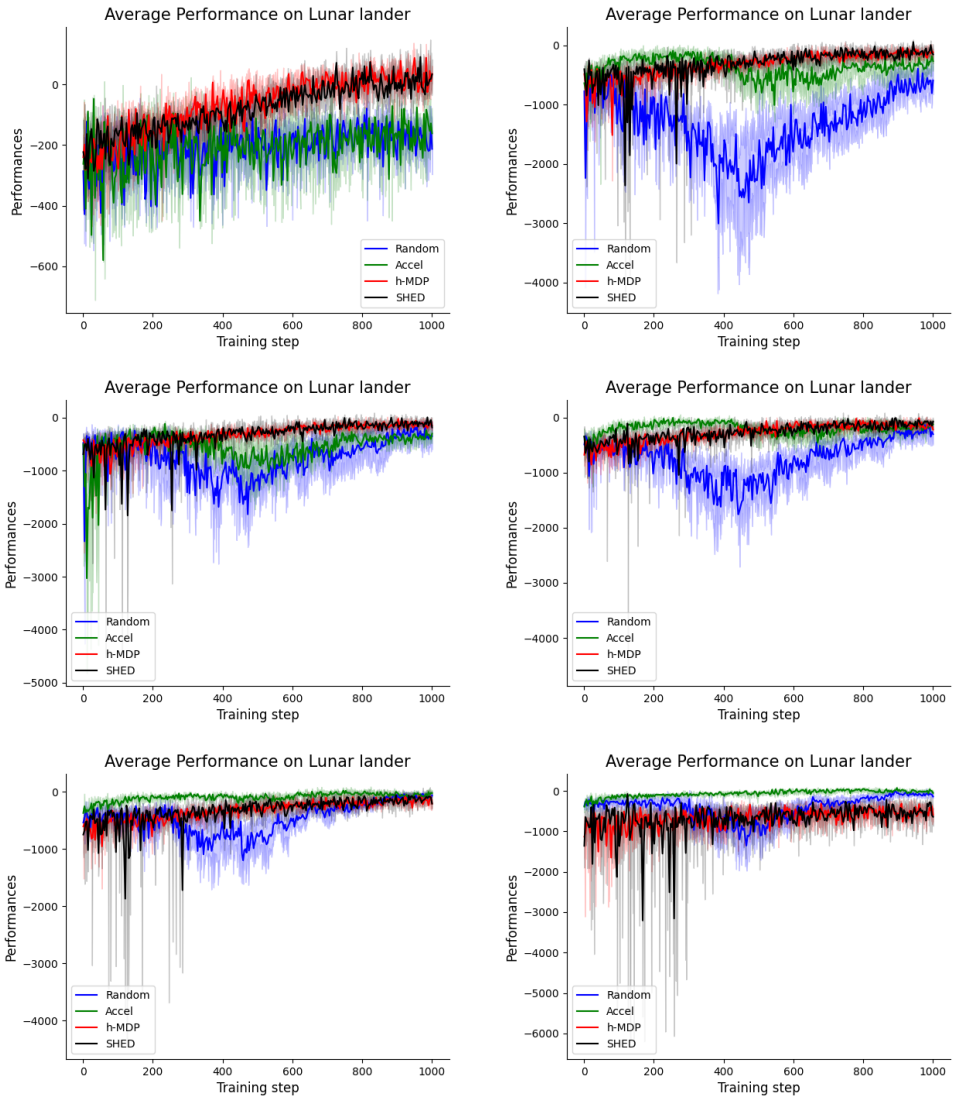


Figure 6.9: Detail how the performance of different methods changes in each testing environment during training (mean and error)

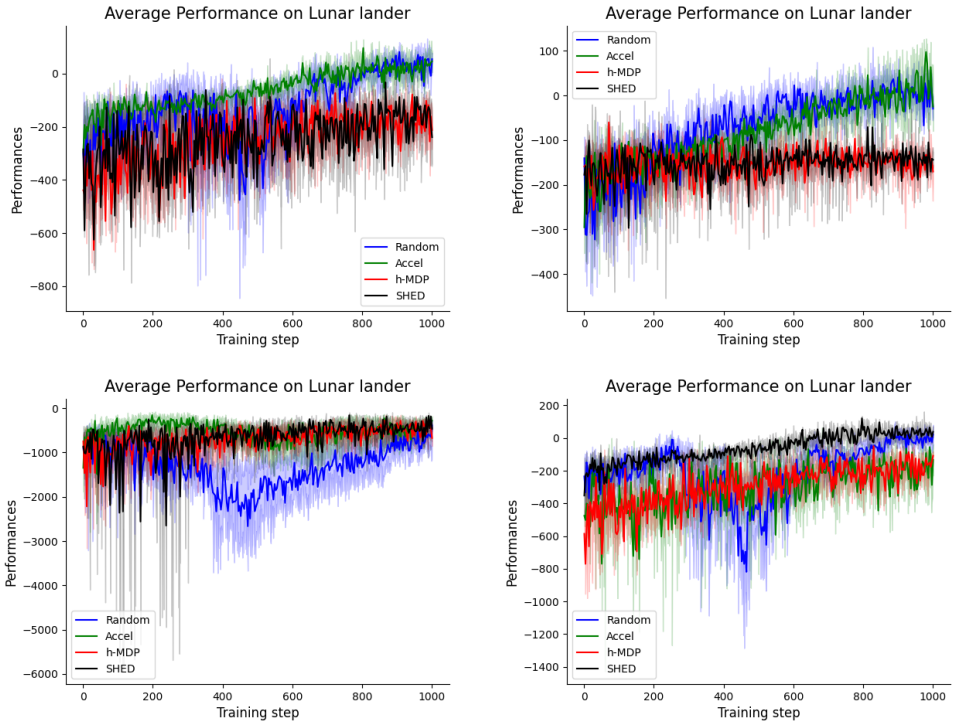
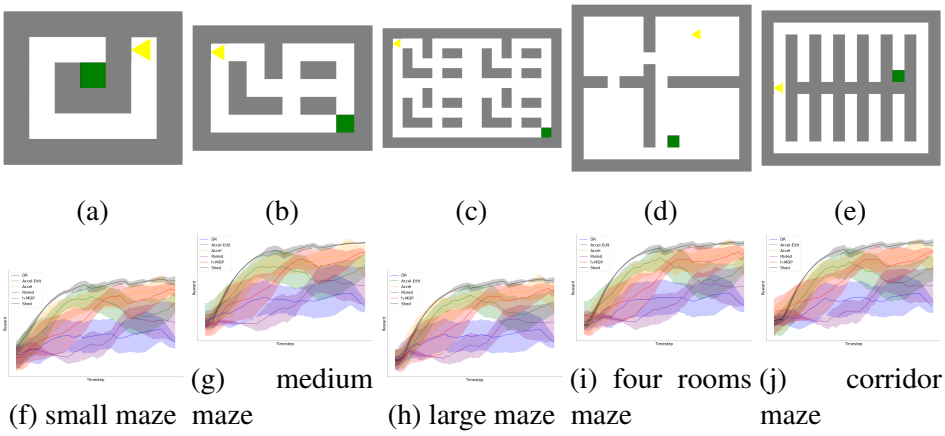


Figure 6.10: Detail how the performance of different methods changes in each testing environment during training (mean and error)



(a) (b) (c) (d) (e)

(f) small maze (g) medium maze (h) large maze (i) four rooms (j) corridor maze

Figure 6.11: Zero-shot transfer performance on test environments in maze environments