POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

A Primal-Dual Approach for Online Learning in Constrained Tree-Form Sequential Decision Problems

Tesi di Laurea Magistrale in
Computer Science and Engineering - Ingegneria Informatica

Author: **Emanuele Pocelli**

Student ID: 251646
Advisor: Prof. Matteo Castiglioni
Co-advisors: Francesco Emanuele Stradi, Prof. Alberto Marchesi
Academic Year: 2024-25

Abstract

Learning in constrained environments and decision making under uncertainty play a central role in many modern applications, where actions must be taken in a safety critical environment. Constrained Markov Decision Processes (CMDPs) provide a general framework for modeling such problems. However, designing algorithms that simultaneously ensure strong performance and constraint satisfaction remains a significant challenge.
This work begins by presenting a comparative analysis of two established algorithms for CMDPs. The comparison highlights their theoretical guarantees, convergence behaviour and the practical trade-offs in realistic constrained scenarios. At the same time, it raises the question of whether similar results can be achieved in a more structured environment. Motivated by this challenge, the second part of this research extends the setting to account for sequential dynamics and partial observability.
Formally, the setting is called a Constrained Tree-Form Sequential Decision Making (CTFSDM) problem. There are existing algorithms that can achieve optimal performance, but they typically rely on linear programming formulations, which are inefficient. These limitations motivate the development of a novel algorithm that is computationally efficient and scalable, without requiring any prior knowledge or assumptions about the environment and maintaining strong theoretical guarantees.
More precisely, the proposed algorithm, called Constrained Primal-Dual Adaptive Follow the Regularized Leader (CPD-FTRL), is based on a primal-dual optimization scheme that leverages a state-of-the-art algorithm originally developed for unconstrained adversarial environments, namely the two-player zero-sum games, and adapts it to a constrained stochastic setting.
A theoretical analysis is provided, demonstrating that the proposed algorithm achieves optimal guarantees while remaining computationally efficient. Overall, this work contributes a theoretically grounded framework for constrained online decision-making in complex, partially observable settings.

Keywords: Artificial Intelligence, Online Learning, Game Theory, Constrained Optimization

Abstract in lingua italiana

L'apprendimento in ambienti vincolati e il processo decisionale in condizioni di incertezza svolgono un ruolo centrale in molte applicazioni moderne, dove le decisioni devono essere prese in contesti in cui la sicurezza è cruciale. I Processi Decisionali di Markov Vincolati (CMDP) offrono una struttura generale per modellare tali problemi. Tuttavia, sviluppare algoritmi che garantiscano prestazioni elevate e rispetto dei vincoli rimane una sfida.
Questo lavoro inizia presentando un'analisi comparativa di due algoritmi consolidati per i CMDP. Il confronto ne evidenzia le garanzie teoriche, la convergenza e i compromessi pratici in scenari vincolati realistici. Allo stesso tempo, solleva la questione se risultati simili possano essere ottenuti in un ambiente più strutturato. Motivata da questa sfida, la seconda parte di questa ricerca estende il contesto per tenere conto di una dinamica sequenziale e di osservabilità parziale.
Formalmente, il contesto è chiamato problema di Decision Making Sequenziale in Forma ad Albero Vincolato (CTFSDM). Esistono algoritmi in grado di raggiungere prestazioni ottimali, ma tipicamente si basano su formulazioni di programmazione lineare, che risultano inefficienti. Queste limitazioni motivano lo sviluppo di un nuovo algoritmo che sia computazionalmente efficiente e scalabile, senza richiedere alcuna conoscenza o assunzione sull'ambiente e mantenendo al contempo simili garanzie teoriche.
Più precisamente, l'algoritmo proposto, denominato Constrained Primal-Dual Adaptive Follow the Regularized Leader (CPD-FTRL), si basa su uno schema di ottimizzazione primale-duale che sfrutta un algoritmo allo stato dell'arte originariamente sviluppato per ambienti avversari non vincolati, in particolare i giochi a somma zero tra due giocatori, adattandolo ad un contesto stocastico vincolato.
Viene fornita un'analisi teorica che dimostra come l'algoritmo proposto raggiunga garanzie ottimali mantenendo al contempo efficienza computazionale. Nel complesso, questo lavoro contribuisce con un nuovo approccio teoricamente fondato per il processo decisionale sequenziale vincolato in contesti complessi e parzialmente osservabili.

Parole chiave: Intelligenza Artificiale, Apprendimento Sequenziale, Teoria dei Giochi, Ottimizzazione Vincolata

Contents

1 | Introduction

Algorithmic Game Theory and Online Learning are central to modern Artificial Intelligence development. In particular, these areas are closely related to Reinforcement Learning, which has recently attracted considerable attention following the success of large-scale learning systems such as OpenAI's ChatGPT and Google's Gemini. Indeed, learning-based agents have demonstrated remarkable capabilities, achieving superhuman performance in complex domains such as Go [12], Chess [47] and Dota 2 [8].

Beyond games, reinforcement learning methods have been studied in real-world applications such as robotics and autonomous driving. In these settings, the objective is to design agents capable of performing tasks traditionally carried out by humans, often in safety-critical environments [25, 26]. Such problems can be modeled by means of Constrained-Markov Decision Processes (CMDPs), where an agent seeks to maximise cumulative reward while satisfying additional constraints which may be related to safety, budget, or resource limitations. For example, in autonomous driving, safety constraints include the avoidance of collisions with other vehicles or pedestrians under all circumstances.

CMDPs assume perfect observability. In many realistic scenarios, however, an agent may operate under imperfect information and may not directly observe the full state of its environment. An autonomous driving agent, for instance, may be unable to determine whether another car is approaching from a blind turn, but still must act conservatively to account for this uncertainty.

Furthermore, decision making processes with imperfect information can be sequential, meaning that decisions are taken at different information states. Applications include online resource allocation and extensive-form games, where the agent does not directly observe the underlying state, but instead acts based on partial information accumulated over time, typically under a perfect recall assumption. This modeling provides a flexible representation and allows to capture both uncertainty and safety constraints.

The present work starts with an experimental comparison of two established constrained online learning algorithms under identical conditions. In particular, the Regularized Primal–Dual Algorithm with Optimistic Exploration introduced by Müller et al. [32] and the

Constrained Primal–Dual Policy Optimization method proposed by Stradi et al. [51] are evaluated and compared. The results obtained are consistent with theoretical guarantees and show that the latter has a stronger performance.
This initial study raises the question of whether similar results can be achieved in more structured environments.
Motivated by this challenge, a novel algorithm is proposed for a closely related setting, and a theoretical analysis is provided, offering guarantees on regret and cumulative constraint violation.
Therefore, the focus is shifted to Constrained Sequential Decision Making problems in an imperfect information environment, which is modeled as an Imperfect Information Game (IIG) [14]. Building upon the state-of-the-art Adapative Follow-the-Regularized-Leader (Adaptive-FTRL) algorithm introduced by Fiegel et al. [19], originally designed for computing Nash equilibria in Imperfect Information zero-sum games, this approach is extended to a constrained, single-agent environment through a primal-dual scheme. In such setting, a learner repeatedly interacts with an unknown environment, observes stochastic feedback and aims to maximize cumulative rewards while ensuring that cumulative constraint violations remain below a given threshold. The interaction takes place in an online fashion, which precludes prior knowledge of the environment, transition dynamics and reward functions.

The structure of the Thesis is the following:

- Chapter 2 introduces the fundamental background concepts necessary for the development of this research.
- Chapter 3 reviews the relevant literature in the field, highlighting existing methods, state-of-the-art results, and open challenges.
- Chapter 4 presents an experimental evaluation of two established algorithms under the same setting, detailing and comparing their performance.
- Chapter 5 introduces the algorithm that is the core contribution of this research, providing formal description, theoretical analysis and comparison with existing works in literature.
- Chapter 6 summarizes the main findings and outlines possible directions for future research.

2 | Preliminaries

This chapter introduces the fundamental concepts about *Game Theory* and *Online Learning* that are necessary for the development of this research.

2.1. Game Theory

Game Theory [17] is the study of mathematical models for strategic interaction among intelligent decision-makers. It has wide applications in fields such as social science, economics, evolutionary biology, logic and computer science [21, 36]. In the following sections, the main concepts of game theory are reviewed, from normal form games to Nash equilibrium. Ultimately, learning algorithms, which constitute the primary focus of this study, are presented.

2.1.1. Normal Form Game

Normal Form Game is a fundamental concept in Game Theory. It is the basic representation of an interaction between decision-makers who make a single move at the same time, and receive a payoff feedback. More formally:

Definition 1. ***Two-player Normal Form Game [45]****: a two-player Normal Form Game can be represented by a tuple* $< N, A, u >$ *where:*

- N *is a set of two players;*
- $A = A_1 \times A_2$*, where* A_1 *is a finite set of* n *actions available to player* 1*, and* A_2 *is a finite set of* m *actions available to player* 2*. Each vector* $e_1, e_2 \in A$ *is called an action profile;*
- $u = (u_1, u_2)$ *where* $u_1 : A \to \mathbb{R}$ *represents the utility function for player* 1*, and* $u_2 : A \to \mathbb{R}$ *represents the utlity function for player* 2*.*

Moreover, a two-player Normal Form Game is commonly represented using a 2-dimensional payoff matrix, where each row corresponds to a possible action for player 1, each column

to a possible action for player 2, and each cell to the resulting outcome. The cell lists the players' utilities for that outcome, with player 1's payoff given first.
A well-known example of Normal Form Game is the Prisoner Dilemma [39], whose payoff matrix is reported below:

	Cooperate	Defect
Cooperate	$(3, 3)$	$(0, 5)$
Defect	$(5, 0)$	$(1, 1)$

(2.1)

Having introduced normal-form games in their general formulation, it is useful to distinguish among different categories based on the structure of the utility functions. Among these, the class of *Constant Sum Games* is relevant for this research.

Definition 2. ***Constant Sum Game [45]****: a two-player normal-form game is constant-sum if there exists a constant c such that for each strategy profile* $(e1, e2) \in A1 \times A2$ *it is the case that* $u_1(e_1, e_2) + u_2(e_1, e_2) = c$.

The main setting of this study is a particular class of constant-sum game, namely the *Zero-Sum Games*, where $c = 0$. Such games model a situation of pure competition: since the sum of the agents' utilities is zero, each agent's goal of maximing its own utility is equivalent to minimizing the utility of its opponent. Formally, for any pair of action profiles (e_1, e_2):

$$u_1(e_1, e_2) = -u_2(e_1, e_2).$$

Finally, in a Normal Form Games, each decision-maker has a set of actions and chooses to play following a *strategy*. For instance, an agent may select a single action and always play it. This is referred to as a *pure* strategies. Another idea could be to randomize over the set of available actions according to a probability distribution. This type of strategy is called a *mixed* strategies. Formally:

Definition 3. ***Mixed Strategies [45]****: Let* $< N, A, u >$ *be a normal form game, and for any set S, let* $\Pi(S)$ *be the set of all probability distributions over S. Then, the set of mixed strategies for player* 1 *is* $\Delta_n = \Pi(A_1)$*, and the set of mixed strategies for player* 2 *is* $\Delta_m = \Pi(A_2)$*, where* Δ_d *is the d-dimensional simplex.*

In this context, each player's strategy can be represented as a vector. This notion of mixed strategy allows to define the utility that each player is expected to attain throughout a game in a compact form:

Definition 4. ***Utility****: given a two-player normal form game* $< N, A, u >$ *and mixed strategies* (x, y)*, player* 1*'s utility is defined as:*

$$u_1(x, y) := x^\top U_1 y,$$

where U_1 *is player* 1*'s payoff matrix. Similarly, player* 2*'s utility is defined as:*

$$u_2(x, y) := x^\top U_2 y.$$

2.1.2. Nash Equilibrium

In the previous section, Normal Form Games and Strategies were introduced. The next question concerns how to reason in such games. In Game Theory, this problem is addressed by identifying a subset of outcomes that are relevant, which can be referred to as solutions of the game.
In this setting, every agent wants to maximise its own utility. Hypothetically, if an agent knew in advance how the opponent is going to play, then its strategic problem would become trivial: it would reduce to a single agent environment where the agent simply aims to maximise its utility. Formally:

Definition 5. ***Best Response [45]****: player* 1*'s best response to the opponent strategy* y *is the strategy* $\arg\max_{x \in \Delta_n} x^\top U_1 y$ *Its value is* $f(y) := \max_{x \in \Delta_n} x^\top U_1 y$*. Similarly, Player* 2*'s best response to the opponent strategy* x *is the strategy* $\arg\max_{y \in \Delta_m} x^\top U_1 y$ *Its value is* $f(y) := \max_{y \in \Delta_m} x^\top U_1 y$*.*

However, an agent does not know which strategy its opponent plans to adopt. The strategic complexity of the game comes from this uncertainty, requiring both agents to observe and adapt their strategies. In general, an action that is optimal against one opponent strategy may be suboptimal against another.
The concept of best response leads to one of the central notions in non cooperative game theory: the *Nash equilibrium*.

Definition 6. ***Nash Equilibrium [33]****: given a two-player normal form game, a strategy profile* (x^*, y^*) *is a Nash equilibrium if* x^* *is the best response for player* 1*, and* y^* *is the best response for player* 2*.*

Intuitively, a Nash equilibrium is a stable strategy profile: any deviation would result in a worse or equal outcome than the current one. As a result, no agent would want to change their strategy.

The previosuly introduced Prisoner's dilemma is a particularly illustrative example. In matrix 2.1, it is possible to see that the game admits only one strict Nash equilibrium: $(Defect, Defect)$. Although both agents would receive higher utility by mutually cooperating, if one agent expects the other to cooperate, defecting becomes the action that maximises its own utility. When both agents follow this reasoning, the outcome converges to the equilibrium (Defect, Defect).
Finally, it is important to remark that:

Theorem 2.1. *Every game with a finite number of players and action profiles has at least one Nash equilibrium [33].*

2.2. Online Learning

Online Learning is a framework that provides algorithms and strategies for agents to adapt and learn strategic environments. Formally, an agent iteratively interacts with an environment over a sequence of rounds and receives feedback after each interaction [23]. The outcomes of the actions are unknown to the agent, and learning takes place in real time. The online fashion of the interaction precludes any prior knowledge about the environment. This setting is naturally analogous to a repeated game where agents select strategies and adapt based on past observations over a sequence of rounds.
A common example of an online learning problem is the multi-armed bandit model [48]. In this setting, the environment consists of a single state and a finite set of actions, referred to as arms. At each round, the agent selects an arm and receives the reward associated with that arm, and its objective is to maximise cumulative reward over time.
Another example are multi-state environments, which are a generalization of the single-state formulation that introduce environment dynamics. In this case, an agent interacts with an evolving environment and its actions influence future states. Additionally, the transition dynamics are unknown to the agent.
In general, there is a wide variety of settings that differ in the environment dynamics and the structure of utility functions. For instance, the feedback and the environment transitions may be *stochastic*, if they follow a fixed probability distribution, or *adversary*, if no assumptions can be made about them. Furthermore, the feedback available to the agent may differ: under *bandit* feedback, the agent only observes the feedback for the chosen action, while under *expert* feedback, the agent observes feedback for all available actions. The bandit setting is typically more challenging, since the agent must balance exploitation (selecting actions that yield high utility) and exploration (choosing less frequently played and potentially suboptimal actions in order to gather more information

about their outcomes).
The choice of environment dynamics and feedback type leads to more structured models. In this context, a central notion is the Markov Process (MDP) [40][56], which provides a standard framework for modeling agent learning in an environment.
More precisely, an MDP is defined by a tuple $< S, A, P, r, \gamma >, \mu$ where:

- S is the space of states;
- A is the space of actions;
- $P := X \times A \times X \to [0, 1]$ is a transition function;
- $r : X \times A \to \mathbb{R}$ is the reward function;
- γ is the discount factor;
- $\mu \in [0, 1]^{|S|}$ is the initial state distribution, where $\mu_i = \mathbb{P}(s_0 = i)$ and $\sum_{i \in S} \mu_i = 1$.

In online learning, the time horizon is typically finite and the discount factor γ is often set to 1. Under this framework, an agent interacts with an environment following a policy π, which maps each state to a probability distribution over actions:

$$\pi : S \times A \to [0, 1].$$

For clarity, a single round agent-environment interaction in an online learning problem is described below. Given an initial state s_0 and time horizon H, at each step, the agent observes the current state, selects an action according to its policy, receives a feedback and the environment transitions to a new state according to its transition function.

Algorithm 2.1 Online Agent-Environment interaction

1: Environment initializes state s_0
2: Agent chooses policy π
3: **for** $h = 0, ..., H - 1$ **do**
4: agent observes s_h
5: agent plays $a_h \sim \pi(\cdot \mid s_h)$
6: agent receives reward $r(s_h, a_h)$
7: environment evolves to state $s_{h+1} \sim P(\cdot \mid s_h, a_h)$
8: **end for**

While MDPs provide a useful framework for online sequential decision making, they assume full observability of the environment: states fully characterise the process. In the following sections, this framework is extended to settings with imperfect information.

2.3. Online Convex Optimisation

The next fundamental question concerns how learning can be carried out in online learning environments. Online Convex Optimization (OCO) [22] provides a general framework for modeling and solving optimization problems that arise in online learning scenarios. This section focuses specifically on the foundations of OCO, including the most commonly used algorithms.

2.3.1. Setting

The setting considered in OCO follows the Online Learning paradigm described in Section 2.2, with minor differences: now, an online agent iteratively interacts with an environment over a sequence of rounds, making a decision and incurring in a loss at each step. The objective is to minimize the total cumulative loss suffered over the rounds.
More specifically, the losses are chosen by an adversary (adversarial setting) and are assumed to be bounded. This assumption is necessary to ensure the problem remains meaningful. Without bounded losses, the adversary could decrease the scale of the loss after the first round preventing the agent from recovering. Similarly, the decision set must be bounded, otherwise the adversary could assign arbitrarily large losses to all the strategies chosen by the agent.
OCO models the agent's decision set as a convex set in Euclidean space denoted as $\mathcal{K} \subseteq \mathbb{R}^n$, while the losses are modeled as bounded convex functions over $\mathcal{K}$. The family of loss functions available to the adversary is denoted by $\mathcal{L}$
At iteration t, the agent chooses $x_t \in \mathcal{K}$. Then, a convex loss function $\ell_t : \mathcal{L} \times \mathcal{K} \to \mathbb{R}$ is drawn. The agent observes $\ell_t(x_t)$, which is the value of the cost function for the choice x_t.
In order to evaluate the performance of an agent in this scenario, it is useful to introduce the notion of *regret*. Let T denote the total number of rounds. Regret is the difference between the loss of the best strategy in hindsight and those of the strategies chosen by the agent at each round. Formally:

$$R_T := \sum_{t=1}^{T} \ell_t(x_t) - \min_{x \in \mathcal{K}} \sum_{t=1}^{T} \ell_t(x).$$

An agent is said to perform well if the regret grows sublinearly in T, which implies he improves his performance overtime. On the other hand, a linear regret shows the algorithm fails to learn. Formally:

Definition 7. ***No-Regret****: An algorithm is said to be no-regret if, for every adversary,* $\lim_{T\to\infty} \frac{R_T}{T} = 0$.

2.3.2. Follow the Leader

The most straightforward approach for an online learner is to always use the best strategy in hindsight (the highest value point in $\mathcal{K}$ according to the information collected up until that point). This idea is called Follow the Leader (FTL) [13]. At each step, the strategy is updated as follows:

$$x_{t+1} := arg\min_{x\in\mathcal{K}} \sum_{k=1}^{t} \ell_k(x).$$

This may work well in some circumstances, for instance in a stochastic setting. However, it doesn't perform well in an adversarial setting: in worst-case scenarios, regret grows linearly with the number of rounds. For example, given $\mathcal{K} = [0, 1]$, let $\ell_t(0) = 1$ if t is even or $\ell_t(0) = 0$ if t is odd, and $\ell_t(0) = 0$ if t is even or $\ell_t(0) = 1$ if t is odd. Then, the losses for each action alternate between 0 and 1 such that $\ell_1(0) = 0, \ell_1(1) = 1, \ell_2(0) = 1, \ell_2(1) = 0$, and so on. According to the Follow the Leader approach, the strategy will keep shifting between $x_t = 0$ and $x_t = 1$, but this results in a linear regret. The reason for this failure is the instability of the update. However, this problem can be mitigated by using a regularization term in the update rule to address the instability.

2.3.3. Follow the Regularized Leader

It is possible to stabilize the FTL strategy using a regularization function in the update rule. There are, however, some conditions that a suitable function must satisfy.

Definition 8. ***α-strongly convex function****: a differentiable function F is α-strongly convex with $\alpha > 0$ if $\forall x, y$ belonging to its domain:*

$$F(y) \geq F(x) + \nabla F(x)^\top (y - x) + \frac{\alpha}{2}||y - x||^2.$$

A suitable regularization function should be twice differentiable over $\mathcal{K}$ and α-strongly convex and smooth.
Follow the Regularized Leader (FTRL) [22] is an algorithm that extends FTL and features a regularization term in the update. As previously mentioned, the regularization function F is assumed to be α-strongly convex, smooth and twice differentiable, while η is the learning rate. For completeness, the agent-environment interaction is reported below:

Algorithm 2.2 Follow the Regularized Leader (FTRL)

1: Input: η 0, regularization function F, convex compact set $\mathcal{K}$;
2: $x_1 = arg\min_{x\in\mathcal{K}} F(x)$;
3: **for** $t = 1, ..., T$ **do**
4: Play x_t
5: Observe loss ℓ_t
6: Update

$$x_{t+1} := arg\min_{x\in\mathcal{K}} x^\top \left(\sum_{k=1}^{t} \nabla\ell_k(x_k)\right) + \frac{1}{\eta}F(x)$$

7: **end for**

The learning rate η controls the importance of the regularization term. Increasing η shrinks the regularization term, causing the algorithm to give more importance to new samples and updates are more aggressive. As a result, the algorithm requires less time to converge, but is less stable and may diverge. On the other hand, decreasing η causes the regularization term to have more importance, making the algorithm more conservative and stable, at the cost of a slower convergence.
Overall, the regularization term gives more stability and more flexibility, allowing strategy to not change "too quickly" [22]. As a result, the worst-case scenario regret guarantees are better than standard FTL. In particular:

Theorem 2.2. *[22] The FTRL Algorithm attains, for every $\widetilde{x} \in \mathcal{K}$ the following bound on the regret:*

$$R_T \leq 2\eta \sum_{t=1}^{T} ||\nabla\ell_t(x_t)||^2_{*_t} + \frac{F(\widetilde{x} - F(x_1))}{\eta}.$$

Additionally, if an upper bound on the local norms is known, i.e. $||\nabla\ell_t(x_t)||_{*_t} \leq G_R$ for all times t, then the regret can be further optimized over the choice of the learning rate η to obtain:

$$R_T \leq 2D_R G_R \sqrt{2T}$$

with D_R diameter of set $\mathcal{K}$.

2.3.4. Multiplicative Weights Update

The FTRL described above is a family of algorithms. Different regularization functions can be chosen, generating a wide variety of No-Regret algorithms. Multiplicative Weights Update (MWU) [22] belongs to the FTRL family and employs the negative entropy $F(\cdot)$

as its regularization function:

$$F(x) := \sum_{i=1}^{n} x(i) \ln x(i).$$

Thus, the FTRL update becomes:

$$x_{t+1} := arg \min_{x \in \mathcal{K}} x^\top \left(\sum_{k=1}^{t} \nabla \ell_k(x_k) \right) + \frac{1}{\eta} \sum_{i=1}^{n} x(i) \ln x(i).$$

In order to computer the minimum of the above function, the method of Lagrange multipliers can be used. Introducing a new parameter $\mu \in \mathbb{R}$, consider the following function:

$$g_\eta(x) := g(x) + \mu(a^\top x - b),$$

where $g(x)$ is the optimization problem objective. If a point x^* is the minimum of g, then there is at least one value of μ such that $\nabla g_\mu(x^*) = 0$. It is possible to find all x, μ such that $\nabla g_\mu(x) = 0$ while ignoring values of x such that $a^\top x \neq 0$. The constraint $x \in \mathcal{K}$ can be written as $\sum_{i=0}^{n} x(i) = 1$. Thus, it holds:

$$x^\top \left(\sum_{k=1}^{t} \nabla \ell_k(x_k) \right) + \frac{1}{\eta} \sum_{i=1}^{n} x(i) \ln x(i) + \mu(x^\top \mathbf{1} - 1).$$

Moreover, the partial derivative of the above function with respect to x_i is:

$$\left(\sum_{k=1}^{t} \nabla \ell_k(x_k) \right) + \frac{1}{\eta}(1 + \ln x(i)) + \mu.$$

Setting the gradient to zero yields the value:

$$x(i) = e^{-1-\eta\mu-\eta\left(\sum_{k=1}^{t} \nabla \ell_k(x_k)(i)\right)},$$

which gives the following closed form solution:

$$x_{t+1}(i) = \frac{e^{\eta \nabla \ell_k(x_k)(i)}}{\sum_{j=1}^{n} e^{-\eta \nabla \ell_k(x_k)(j)}}.$$

The final algorithm is reported below.

Algorithm 2.3 Multiplicative Weights Update (MWU)

1: Input: $\eta\, 0$, convex compact set $\mathcal{K}$;
2: $x_1 = [1/n, ..., 1/n]^\top$
3: **for** $t = 1, ..., T$ **do**
4: Play x_t
5: Observe loss ℓ_t
6: Update

$$x_{t+1}(i) = x_t(i)\frac{e^{\eta\nabla\ell_t(x_t)(i)}}{\sum_{j=1}^{n} x_t(j)e^{-\eta\nabla\ell_t(x_t)(j)}}$$

7: **end for**

This algorithm is computationally efficient, as the update has a closed form solution. Additionally, it guarantees similar Regret bounds when a proper learning rate η is chosen. The linear version of the algorithm [2] is also reported for completeness.

Algorithm 2.4 Multiplicative Weights Update (MWU)

1: Input: $\eta\, 0$, convex compact set $\mathcal{K}$;
2: $x_1 = [1/n, ..., 1/n]^\top$
3: **for** $t = 1, ..., T$ **do**
4: Play x_t
5: Observe loss ℓ_t
6: Update

$$x_{t+1}(i) = x_t(i)\frac{\eta\nabla\ell_t(x_t)(i)}{\sum_{j=1}^{n} x_t(j)(1-\eta\nabla\ell_t(x_t)(j))}$$

7: **end for**

2.3.5. Online Mirror Descent

Online Mirror Descent [22] is a family of algorithms that is closely related to FTRL. The main idea is to compute a decision using a gradient update and the previous decision. It can be seen as a generalization of Gradient Descent used in Deep Learning [43] and is an iterative algorithm in which the update is carried out in a dual space, where the duality is defined by the choice of regularization: the gradient of the regularization function defines a mapping from $\mathbb{R}^n$ onto itself, which is a vector field. The gradient updates are then carried out in this vector field. The *Bregman divergence* $B_F(x||y)$ [22] is used for this projection and is the difference between the value of the regularization function at x and the value of the first order Taylor approximation. Formally:

Definition 9. ***Bregman divergence*** *[22]: Bregman divergence with respect to a function F is defined as:*

$$B_F(x||y) := F(x) - F(y) - \nabla F(x)^\top (x - y).$$

Properties of the Bregman divergence include [22]:

- Strict convexity in the first argument x;
- Non negativity: $B_F(x||y) \geq 0 \quad \forall x, y$;
- $B_F(x||y) = 0$ iff $x = y$;
- Asymmetry;
- If F is α-strongly convex: $B_F(x||y) \geq \frac{\alpha}{2}||x - y||^2$.

The OMD algorithm can now be formally defined. There are 2 versions: one agile and one lazy [22]. In particular, the lazy version keeps track of a point in Euclidean space and projects onto the convex decision set $\mathcal{K}$ only at decision time. On the other hand, the agile version maintains a feasible point at all times. The complete pseudocode is reported below:

Algorithm 2.5 Online Mirror Descent (OMD)

1: Input: $\eta > 0$, convex compact set $\mathcal{K}$, regularization function $F(x)$;
2: Let y_1 be such that $\nabla F(y_1) = 0$ and $x_1 = arg\min_{x \in \mathcal{K}} B_F(x||y)$
3: **for** $t = 1, ..., T$ **do**
4: Play x_t
5: Observe loss ℓ_t
6: Let $\nabla_t = \nabla f_t(x_t)$
7: Update y_t according to the rule:

$$\text{[Lazy version]} \qquad \nabla F(y_{t+1}) = \nabla F(y_t) - \eta \nabla_t \tag{2.2}$$

$$\text{[Agile version]} \qquad \nabla F(y_{t+1}) = \nabla F(x_t) - \eta \nabla_t \tag{2.3}$$

8: Project according to B_F:

$$x_{t+1} = arg\min_{x \in \mathcal{K}} B_F(x||y_{t+1})$$

9: **end for**

It is worth noting that, for linear cost functions, the FTRL and the lazy-OMD algorithm

are equivalent.

Lemma 1. *[22] Let $\ell_1, ..., \ell_T$ be linear cost functions. The lazy OMD and FTRL algorithms produce identical predictions:*

$$arg\min_{x \in \mathcal{K}} B_F(x||y_t) = arg\min_{x \in \mathcal{K}} \left(\eta \sum_{s=1}^{t-1} \nabla_s^\top x + F(x) \right).$$

2.4. Tree-Form Sequential Decision Making problems

Having introduced the main aspects of Online Learning, from the general setting to the learning strategies, attention can now be turned to the specific setting that is the focus of this research. This section introduces more specialized models, namely *Tree-Form Sequential Decision Making* problems, which are the main topic of interest in this thesis.

2.4.1. Extensive-form Game

First of all, the notion of Normal Form Game is extended to the *extensive form* [17], which models a sequence of player interactions using a rooted game tree. In this setting, each node represents a decision point for one of the players, while edges from each node represent the actions that the corresponding player can take. The game starts from the root node, and the player corresponding to the root node takes the first decision. The idea is that choosing an action is equivalent to decide the next game state among the children of that node and that the game continues until a leaf node (also called a terminal node) is reached. Leaf nodes represent the possible outcomes of the game and each leaf node is associated with utility values for each player.
In addition to strategic players, an Extensive Form Game (EFG) may also include a fictitious player, called Chance (or Nature), who takes actions according to fixed probability distributions. This player does not have a utility and can be useful in real world problems to represent stochasticity. For example, in a card game, the act of drawing cards from a deck can be represented by having a Chance player that takes the action of dealing cards according to a uniform probability distribution.
In general, the extensive form of a game can capture both perfect and imperfect information games. Games like Tic-Tac-Toe or Chess are *full information*: each player can fully observe the opponent's moves before making a decision. Other games, such as card games, can be *imperfect information*: a player may not be sure about which node he is currently at when taking a decision. In the example of a card game, a player may be

uncertain, for instance, about which card the opponent has in its hand, or even about which action his opponent has played.
Practically, this uncertainty is represented using information sets: all nodes belonging to a player are partitioned into several information sets, and those belonging to the same information set are indistinguishable to the player. This also implies that nodes in the same information set must have the same actions available, otherwise the player could distinguish one node from the other. Furthermore, an information set consisting of a single node is called a singleton. If all information sets are singletons, the game is said to have perfect information.
Moreover, an important assumption in many extensive-form models is the perfect-recall, meaning that the players do not forget their own past actions and observations. More precisely, an EFG has perfect recall if, for any two nodes in the same information set, the paths from the root to these nodes is identical. This assumption ensures that player can take a decision based on all the information observed up until that point.
Formally, an Extensive-Form Game (EFG) [17] is defined by a tuple $< \mathcal{N}, \mathcal{A}, \mathcal{H}, \mathcal{Z}, \chi, \rho, \sigma, u >$ where:

- $\mathcal{N}$ is the set of players;
- $\mathcal{A}$ is the set of actions;
- $\mathcal{H}$ is the set of nonterminal nodes in the game trees. They are also known as histories, as each node corresponds to a history of actions taken by the players;
- $\chi : \mathcal{H} \to \{0,1\}^{|\mathcal{A}|}$ is a function that specifies the set of actions available at each node;
- $\rho : \mathcal{H} \to \mathcal{N}$ is a function that specifies the acting player at each node;
- $\mathcal{Z}$ is the set of terminal nodes;
- $\sigma : \mathcal{H} \times \mathcal{A} \to \mathcal{H} \cup \mathcal{Z}$ is the successor function that specifies the successor node when the acting player takes an action at a node;
- $u = (u_1, ..., u_N)$ where $u_i : \mathcal{Z} \to \mathbb{R}$ is the utility function of player i. $u_i(z)$ is used to denote the utility for player i if the game terminates at $\ddagger \in \mathcal{Z}$.

Note that it is possible to convert an EFG to an NFG. In the resulting NFG, a pure strategy for player i specifies an action for each information set. Furthermore, the payoff matrix can be constructed determining the terminal node reached following the combination of the players' pure strategies. However, such representation often has many redundancies and some information sets may not even be reachable due to a past action. The resulting

game is very large, and the number of rows and columns grows exponentially with the number of nodes in the game tree.

2.4.2. Tree-Form Sequential Decision Making problems

A Tree-Form Sequential Decision Making problem (TFSDM) [18] is a concept closely related to EFG. Specifically, it is a model structured as a tree, where each node represent either a decision point or an observation point. There is a clear analogy between a TFSDM problem and an EFG: a TFSDM can be seen as a single-player EFG, where an agent interacts with an environment. In this analogy, decision nodes correspond to the agent's choices, while observation nodes represent stochasticity and partial observability. A TFSDM problem is defined as a tuple $< \mathcal{J}, \mathcal{K}, \mathcal{Z}, S, A, \rho, u >$ where:

- $\mathcal{J}$ is the set of decision points;
- $\mathcal{K}$ is the set of observation points;
- $\mathcal{Z}$ is the set of terminal nodes;
- S is the set of signals, where S_k is the set of possible signals in node $k \in \mathcal{K}$;
- A is the set of actions, where A_j is the set of actions available in node $j \in \mathcal{J}$;
- $\rho : \mathcal{K} \to [0,1]^{|S|}$ is the transition function, where, given $k \in \mathcal{K}$ and $s \in S_k$, $\rho(k, s)$ is the probability of receiving signal s in node k;
- $u : \mathcal{Z} \to \mathbb{R}$ is the utility function, where, given $z \in \mathcal{Z}$, $u(z)$ is the utility of the terminal node z.

At each decision point $j \in \mathcal{J}$, the agent selects an action $a \in A_j$. Then, the environment evolves to the next observation point $k \in \mathcal{K}$, where a signal $s_k \in S_k$ is drawn according to $\rho(j, a) \in \mathcal{J} \cup \mathcal{K} \cup \mathcal{Z}$. This interaction continues sequentially until a terminal node $z \in \mathcal{Z}$ is reached and the agent receives $u(z)$.
The pair (j, a) where $j \in \mathcal{J}, a \in \mathcal{A}$ is called a *sequence*. The set of all sequences is defined as

$$\Sigma := \{(j, a) : j \in \mathcal{J}, a \in \mathcal{A}_j\}.$$

Given a node $v \in \mathcal{J} \cup \mathcal{K}$, p_v denotes its parent sequence, which is the last sequence encountered on the path from the root to v. If v is the root node or only observation points are encountered on the path, then $p_v = \varnothing$.
In a TFSDM problem, a strategy is represented as a distribution over the set of actions A_j at each decision point $j \in \mathcal{J}$. In particular, in this thesis, a strategy will be represented

using the *sequence-form representation*, that is, as a vector $\pi \in \mathbb{R}_{\geq 0}^{|\Sigma|}$ whose elements are indexed by Σ. The entry π_{ja} contains the product of the probabilities of all actions at all decision points on the path from the root of the process to action a at decision point j. In order for a sequence-form strategy to be valid, its entries must satisfy the following consistency constraints [27, 41]:

$$\sum_{a \in A_j} \pi_{ja} = \pi_{p_j} \quad \forall j \in \mathcal{J} \text{ s.t } p_j \neq \varnothing, \tag{2.4}$$

$$\sum_{a \in A_j} \pi_{ja} = 1 \quad \forall j \in \mathcal{J} \text{ s.t } p_j = \varnothing. \tag{2.5}$$

This means that the sum of the probabilities in a node $j \in \mathcal{J}$ over all actions $a \in A_j$ is equal to the probability of reaching the sequence p_j.

2.5. Learning in Constrained environments

In many real world scenarios, agents must operate under constraints in addition to optimizing performance. Such constraints may arise from safety requirements, limited resources or operational budgets.

2.5.1. Constrained environments

The environments introduced in the previous section can be extended to incorporate constraints. Common frameworks for constrained problems include TFSDMs and MDPs. In these cases, the agent receives not only a reward feedback, but also one or more costs, and its goal is to maximize the reward while keeping the total costs below a given threshold, which is usually represented as α and are fixed properties of the environment. As an example, the formal definition of a Constrained MDP (CMDP) is reported below. A CMDP can be defined as a tuple $< S, A, P, m, r, c, \alpha >$

- S is the space of states;
- A is the space of actions;
- $P := S \times A \times S \to [0, 1]$ is a transition function;
- m is the number of constraints;
- $r : S \times A \to \mathbb{R}$ is the reward function;
- $c : S \times A \to \mathbb{R}^m$ is the cost function;

- $\alpha \in \mathbb{R}^m$ is a thershold vector, where the α_i is related to the i-th constraint.

Let $V^\pi(r)$ and $V^\pi(c_i)$ denote the expected reward and cost for the i-th constraint, respectively, attained following a policy π in a CMDP. Given a generic function v defined over the space of states and actions, for an episode of length H:

$$V^\pi(v) := \mathbb{E}_\pi \left[\sum_{t=0}^{H-1} v(s_t, a_t) \right],$$

then, the objective is given as the following linear program:

$$\text{OPT}_{r,G} := \begin{cases} \max\limits_{\pi \in \Pi} & V^\pi(r) \\ \text{s.t.} & V^\pi(c_i) \leq \alpha_i \quad \forall i \in [m], \end{cases}$$

where $[m] := \{0, 1, ..., m-1\}$.

In such scenario, standard algorithms designed for unconstrained settings fail, because they only focus on reward maximization and therefore, they may converge to unsafe policies. For this reason, learning in constrained environments requires algorithms that take both optimization and feasibility in consideration.

2.5.2. Primal-dual approaches

There are different commonly used approaches to address this challenge.
A common and effective method for learning in constrained environments, which is also the one used by the algorithm presented in this research, is based on *primal-dual optimization*. In this scheme, the original constrained problem is relaxed into an unconstrained one by incorporating both rewards and constraints into an objective function and introducing *Lagrange multipliers* that control the penalty for violations.
Starting from the linear program 2.5.1 above, it is possible to define a *Lagrangian function* that combines both rewards and constraints.

Definition 10. ***Lagrangian function****: Given a CMDP with reward r and cost c, the Lagrangian function $\mathcal{L}_{r,c}$ is defined as follows for every policy π and vector of Lagrange multipliers $\lambda \in \mathbb{R}^m_{\geq 0}$:*

$$\mathcal{L}_{r,c} := V^\pi(r) - \sum_{i=0}^{m} \lambda_i \left(V^\pi(c_i) - \alpha_i \right).$$

In this formulation, the policy π is called the *primal variable*, while the vector of Lagrange multipliers λ is called the *dual variable*. A primal-dual algorithm iteratively updates both

variables at each step. This means that, the dual variable is updated dynamically to give more or less importance to constraint violation. In particular, when a policy satisfies a given constraint (i.e. $V^{\pi}(c_i) \leq \alpha_i$), the corresponding multiplier λ_i should be decreased, reducing the weight of that term. On the other hand, when a policy a constraint is violated, the corresponding multiplier should be increased, to penalize violations.
Furthermore, the learning procedure alternates between a primal and a dual step: In the primal step, the policy is updated in order to maximize the current Lagrangian objective, while in the dual step, the vector of Lagrange multipliers is updated based on current observed constraint violations. This approach is based on an artificial feedback signal which combines rewards and costs weighted by the current multipliers. In this way, the agent only sees one objective and maximizes over it, while the dual update ensures that constraints are also taken in consideration. Because of that, even when the reward and cost functions are stochastic, the resulting feedback is adversarial, since the multipliers are updated dynamically. As a result, the primal update must rely on algorithms designed for adversarial settings.

3 | Related works

This chapter presents the main results obtained on constrained environments in the literature, together with state-of-the-art algorithms, and commonly adopted strategies.

3.1. Constraint Violation

In the previous sections, the notion of regret was introduced as a standard performance metric in Online Learning. Regret quantifies the cumulative difference between the reward obtained by an agent and that of the optimal policy baseline. However, in a constrained environment, reward is not the only performance metric, but also constraints must be taken in consideration.
To address this problem, a similar concept, referred to as *Constraint Violation*, can be defined allowing to evaluate how much the constraints were violated by a learning agent by measuring the cumulative violation of the constraints.
In general, an algorithm that doesn't take constraints into consideration may achieve low regret for the rewards, while incurring large constraint violations. Such behavior indicates that the algorithm is learning an unsafe or unfeasible policy, which is unacceptable in most safety-critical applications. Therefore, both the notions of regret and constraint violation are used to evaluate performance in Online Learning in a Constrained environment.
In the case of CMDP, let Π be the set of all possible policies. It is assumed that there is at least one policy π° that satisfies the constraints: $V^{\pi^\circ}(c_i) \leq \alpha_i, \forall i \in [m]$. Among these policies, there is an optimal one, which is defined as the solution to the following optimization problem:

$$\mathrm{OPT}_{r,G} := \begin{cases} \max\limits_{\pi \in \Pi} & V^\pi(r) \\ \text{s.t.} & V^\pi(c_i) \leq \alpha_i \quad \forall i \in [m], \end{cases}$$

where c_i denotes the value of the i-th constraint and α_i its corresponding threshold.
This is the policy that yields the highest reward while satisfying the constraints, and is denoted as π^*

The notions of regret and constraint violation can now be defined. Following Efroni et al.'s formulation [15], *weak* and *strong* notions are distinguished:

Definition 11. ***Weak Regret and Constraint Violation****: given a CMDP, the weak regret R_T and the weak constraint violation V_T are defined as:*

$$R_T := \sum_{t=1}^{T} \left[V^{\pi^*}(r) - V^{\pi_t}(r) \right]$$

and:

$$V_T := \max_{i \in m} \sum_{t=1}^{T} \left[V^{\pi_t}(c_i) - \alpha_i \right],$$

where $[m] := \{0, 1, ..., m-1\}$,

Definition 12. ***Strong Regret and Constraint Violation****: given a CMDP, the strong regret R_T and the strong constraint violation V_T are defined as:*

$$R_T := \sum_{t=1}^{T} \left[V^{\pi^*}(r) - V^{\pi_t}(r) \right]^+$$

and:

$$V_T := \max_{i \in m} \sum_{t=1}^{T} \left[V^{\pi_t}(c_i) - \alpha_i \right]^+,$$

where $[\cdot]^+ = max(0, \cdot)$.

These are measures of the performance of an algorithm evaluated against the baseline π^*. The first definitions are weaker, as they allow for negative terms to cancel out positive ones. This means that the regret can be controlled using unsafe policies yielding large rewards, and, similarly, the violation can be controlled using policies that satisfy the constraints by a large margin. However, this is not acceptable for most real-world problems. For instance, in autonomous driving, it is not acceptable for an agent to be overly safe in order to compensate for previous crashes.
As a result, the notions of *strong* regret and *strong* violation are introduced, as they do not allow for cancellations. Moreover, a bound on them also implies a bound for their weak counterparts.

3.2. Online CMDPs

Online Learning has received significant attention in the last years [11, 30, 35, 49] and both stochastic [3, 16, 31] and adversarial settings [4, 20, 24, 34, 42, 53] have been extensively studied. In particular, in the context of CMDPs, three main approaches are adopted: Linear Programming [1, 5, 15, 52, 55], primal-dual ([38, 50, 54]) and dual algorithms [37].
Linear Programming (LP) methods maximise the rewards obtained while explicitly considering the constraints. At each round, a constrained optimization problem is solved to determine a new policy. While this approach has strong theoretical guarantees, it is also very expensive from a computational perspective due to the need to iteratively solving linear programs.
Primal-dual algorithms, on the other hand, as briefly introduced in the previous chapter (see Subsection 2.5.2), aim to maximize the rewards, while implicitly considering the constraints. They do so by introducing Lagrange multipliers and by converting the constrained optimization problem to a single objective maximization problem that incorporates both a reward term and a constraint penalization term. The learning process consists in two alternating steps of updating the policy (primal step) and the multipliers (dual step). In the primal step, the policy is updated to maximize the single objective function, while in the dual step, the weights are adapted to current observations to control the penalization terms. This approach is much more efficient and scales better than LP-based methods. However, strong regret and violation guarantees remain challenging.
Finally, dual algorithms focus on learning the optimal dual variable, while the policy at each round is obtained by solving a reward maximization problem under the current penalty parameters. Since the policy update is performed greedily with respect to the current dual variables, policies may change significantly across rounds. This can lead to instability and sensitivity to noise. This approach is, however, very efficient.
The works of Efroni et al. [15] study the exploration-exploitation trade-off in episodic CMDPs with unknown transitions. They were the first to introduce an algorithm with optimal $\widetilde{O}(\sqrt{T})$ strong regret and violation in general CMDPs. Moreover, the setting is a fully-stochastic episodic CMDP with unknown transitions and a loop-free structure, which means that the states are structured in layers, although it is possible to represent general episodic environments by state duplication.
Formally, the environment is defined by the tuple $< X, A, P, m, r, g, \alpha >$ where:

- S is the space of states;
- A is the space of actions;
- $P := S \times A \times S \to [0, 1]$ is the transition function;

- m is the number of constraints;
- $r : S \times A \to \mathbb{R}$ is the reward function;
- $c : S \times A \to \mathbb{R}^m$ is the cost function;
- $\alpha \in \mathbb{R}^m$ is a threshold vector, where the α_i is related to the i-th constraint.

In their study, Efroni et al. [15] propose two approaches. The first relies on linear programming and obtains *strong* regret and *strong* violation, but is computationally expensive. The second is based on a primal-dual scheme and only guarantees *weak* regret and *weak* violation. An open question left by this work is whether algorithms can achieve optimal guarantees for the strong metrics while being computationally efficient.
To this end, Müller et al. [32] recently proposed a primal-dual policy optimization algorithm achieving $R_T \leq \widetilde{O}(T^{0.93})$ *strong* regret and violation. Although the guarantees are sublinear, they are largely suboptimal.
Another approach is analysed by Stradi et al. [51], who introduced a learning algorithm based on a novel primal-dual scheme. The proposed approach is based on the state-of-the-art policy optimization algorithm designed for adversarial MDPs [29] as the primal regret minimizer, and uses an Upper Confidence Bound (UCB) method for the dual update. The resulting algorithm leverages a policy optimization approach, and is thus efficient. The *strong* regret and violation guarantees are sublinear and optimal: $R_T \leq \widetilde{O}(\sqrt{T})$.

3.3. Online Learning in Games and Sequential Decision Making problems

A more specific setting, which constitutes the main topic of this research and has recently received a lot of attention is the EFG. Offline algorithms have been developed capable of achieving superhuman performance in complex settings. Notable examples include two-player [9] and multi-player [10] poker, Go [46], Chess [47] and even complex games like Starcraft II [58] and Dota 2 [8].
While these approaches are designed for offline settings, recent research has focused on *online* learning in sequential decision-making environments, where an agent must adapt to unknown or evolving dynamics.
Fiegel et al. [19] introduced a novel algorithm capable of learning an optimal strategy in a two-player Imperfect Information Game, which can be seen as an extension of an EFG. They propose two variants: *Balanced FTRL* and *Adaptive FTRL*. Both are based on the Follow the Regularized Leader framework.
The first employs entropy-based regularization, using either Tsallis entropy [57] or Shan-

non entropy [44], and assumes knowledge of the tree structure. The latter uses adaptive parameters which allow the algorithm to be employed even without knowledge of the tree structure. Both achieve sublinear regret $R_T \leq \widetilde{O}(\sqrt{T})$ in their respective setting.
TFSDMs are another setting which share structural similarities with EFGs. In this case, a single agent interacts with a tree-form environment under potentially unknown transitions and receives a reward. Farina et al. [18] introduced this setting and developed an algorithm for adversarial rewards, achieving $O(\sqrt{T})$ regret.
When constraints are incorporated into TFSDMs, two main formulations arise: the *hard-threshold* and the *soft-threshold* problems. In the first, constraints must be satisfied at every round with probability at least $1 - \delta$, where $\delta \in (0, 1)$ is an algorithm parameter. When such requirement is satisfied, an algorithm is called δ-safe. In the latter, it is necessary that the cumulative constraint violation grows sublinearly. This corresponds to the violation notion discussed above in the CMDP setting. There is, however, a limitation in this context: it is impossible for an algorithm to be sublinear in regret while also being δ-safe without additional assumptions. This impossibility result motivates the study of the soft-treshold problem. Formally:

Theorem 3.1. *(Bernasconi et al.) [7] In general repeated Constrained Sequential Decision Making problems, if an algorithm is δ-safe for any $\delta \in (0, 1)$ given as input, then it incurs in a regret $R_T = \Omega(T)$ with probability at least $1 - \delta$.*

Nevertheless, sublinear regret can be achieved under additional assumptions: Bernasconi et al. [7] proposed a novel algorithm that achieves $\widetilde{O}(\sqrt{T})$ cumulative regret both for the soft-threshold problem and the hard-thershold one, assuming that the agent knows a strategy that is always *strictly* safe.

4 | Empirical Validation in CMDP

This chapter presents an empirical evaluation of two constrained online optimization algorithms: the Regularized Primal-Dual Algorithm with Optimistic Exploration proposed by Müller et al. [32], and the CPD-PO introduced by Stradi et al. [51]. The objective of this study is to compare their performance in terms of *strong* regret and constraint violation under identical conditions.

As discussed in Chapter 3, Müller et al. [32] were the first to establish sublinear *strong* regret and constraint violation guarantees within an efficient primal-dual framework. Their method achieves regret bounded by $R_T \leq \widetilde{\mathcal{O}}(T^{0.93})$, which, although represented an advancement at the time of publication, remains suboptimal.

The CPD-PO algorithm [51] addresses this limitation by closing the gap to optimal regret rates while maintaining computational efficiency through a novel primal-dual formulation.

4.1. Experimental Setting

The experimental setting follows the model introduced in Definition 3.2. A single agent interacts sequentially with a stochastic environment, whose state space is organized in a layered structure. At each decision step, the agent selects an action and receives a stochastic reward and constraint costs as feedback. Both state transitions and feedback signals distributions are unkown to the agent. The objective is to maximize cumulative reward while keeping costs below a given threshold.

Figure 4.1 illustrates a basic example of CMDP.

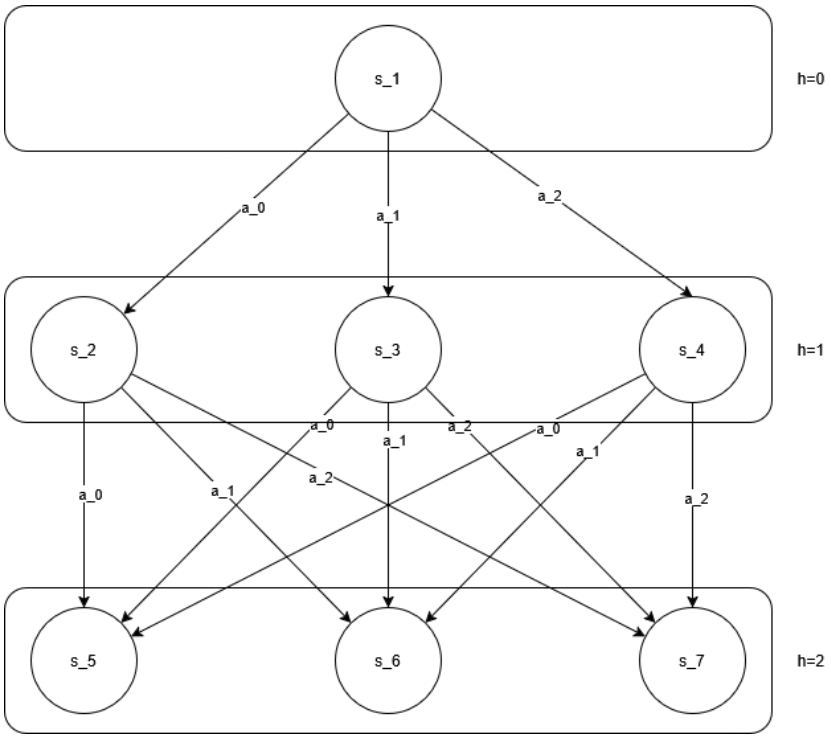

Figure 4.1: Example of CMDP with layered structure: starting from the first state, after any action taken, the environment transitions to a state in the next layer, and this process continues until the final layer is reached. Apart from the first layer, which consists of only one state, all subsequent layers contain the same number of states.

For completeness, the full agent-environment interaction is described:

Algorithm 4.1 Online Agent-Environment interaction in a CMDP

Environment initializes state x_0
Agent chooses policy π
for $h = 0, ..., H-1$ **do**
    agent observes x_h
    agent plays $a_h \sim \pi(\cdot \mid x_h)$
    agent receives reward and constraints $r(x_h, a_h), c_i(x_h, a_h),\ \forall i \in [m]$
    environment evolves to state $x_{h+1} \sim P(\cdot \mid x_h, a_h)$
end for

4.2. Regularized Primal-Dual Algorithm with Optimistic Exploration

The Regularized Primal-Dual Algorithm with Optimistic Exploration algorithm relies on empirical estimators combined with confidence bonuses, built to drive exploration, in order to construct optimistic models of the environment.

4.2.1. Estimators

Let $N_t(x,a)$ denote the number of visits to state-action pair $(x,a) \in X \times A$ up to time $t-1$. The estimators for rewards, constraints, and transitions are defined as:

$$\begin{aligned}
\hat{r}_t(x,a) &:= \frac{\sum_{\tau=1}^{t-1} r_\tau(x,a)\mathbb{1}_{\{x_\tau=x,a_\tau=a\}}}{\max(0,N_t(x,a))}, \\
\hat{g}_{i,t}(x,a) &:= \frac{\sum_{\tau=1}^{t-1} g_{i,\tau}(x,a)\mathbb{1}_{\{x_\tau=x,a_\tau=a\}}}{\max(0,N_t(x,a))}, \\
\hat{p}_t(x' \mid x,a) &:= \frac{\sum_{\tau=1}^{t-1} \mathbb{1}_{\{x_\tau=x,a_\tau=a,x_{\tau+1}=x'\}}}{\max(0,N_t(x,a))},
\end{aligned} \tag{4.1}$$

while the bonuses are defined as:

$$\begin{aligned}
b_t^r(x,a) &:= \mathcal{O}\left(\sqrt{\frac{\log(XAHmT\delta^{-1})}{max(0,N_t(x,a))}}\right), \\
b_t^p(x,a) &:= \mathcal{O}\left(H\sqrt{\frac{X+\log(XAHT\delta^{-1})}{max(0,N_t(x,a))}}\right), \\
b_t(x,a) &:= b_t^r(x,a) + b_t^p(x,a).
\end{aligned} \tag{4.2}$$

Then, the optimistic estimates are defined as:

$$\begin{aligned}
\overline{r}_t(x,a) &:= \hat{r}_t(x,a) + b_t(x,a), \\
\overline{g}_{i,t}(x,a) &:= \hat{g}_t(x,a) + b_t(x,a), \\
\overline{\psi}_t(x,a) &:= -\log(\pi_t(a \mid x)) + b_t^p(x,a)\log A, \\
\overline{p}_t(x,a) &:= \hat{p}_t(x,a),
\end{aligned} \tag{4.3}$$

where $\overline{\psi}$ is the entropy term and provides additional exploration.

4.2.2. Optimistic Value Functions

Given an estimated transition model $\overline{p}_t$ and a function v_t defined over state-action pairs, the expected value of policy π_t under the current optimistic model is defined as:

$$V^{\pi_t,\overline{p}_t}(v_t) := \mathbb{E}_{\pi_t,\overline{p}_t}\left[\sum_{t=0}^{H-1} v_t(s_t,a_t)\right].$$

Another value function is also computed, which corresponds to the Optimistic Lagrangian objective:

$$\overline{z}_t := V^{\pi,\overline{p}}(\overline{r}_t) + \lambda_t V^{\pi\overline{p}}(\overline{g}_t) + \tau V^{\pi\overline{p}}(\overline{\psi}_t).$$

At each round, the algorithm performs a truncated policy evaluation under the optimistic model. The quantities $\hat{Q}_t^{\overline{z}}(x,a)$ and $\hat{V}_t^{g_i}(x)$ denote the corresponding truncated action-value and value functions for the optimistic objective $\overline{z}$ and each constraint. These functions are computed by means of the Eval algorithm:

Algorithm 4.2 Eval (Truncated Policy Evaluation)

Input: $\pi_t, \lambda_t, \overline{r}_t, \overline{g}_t, \overline{p}_t, \overline{\psi}_t$

set $\hat{V}_t^{\overline{r}_t}(x) = \hat{V}_t^{\overline{g}_{i,t}}(x) = \hat{V}_t^{\overline{\psi}_t}(x) = 0, \quad \forall x \in X, i \in [m]$

for $h = H-1, ..., 0$ **do**

  for $(x,a) \in X \times A$ **do**

    Truncated DP step:

$$\hat{Q}_t^{\overline{r}_t}(x,a) := \min\left\{H-h+1, \quad \overline{r}_t(x,a) + < \overline{p}_t(\cdot \mid x,a), \hat{V}_t^{\overline{r}_t}(\cdot) >\right\}$$
$$\hat{Q}_t^{\overline{g}_{i,t}}(x,a) := \min\left\{H-h+1, \quad \overline{g}_{i,t}(x,a) + < \overline{p}_t(\cdot \mid x,a), \hat{V}_t^{\overline{g}_{i,t}}(\cdot) >\right\}, \quad \forall i \in [m]$$
$$\hat{Q}_t^{\overline{\psi}_t}(x,a) := \min\left\{\psi_r(x,a) + (H-h+1)\log A, \quad \overline{\psi}_t(x,a) + < \overline{p}_t(\cdot \mid x,a), \hat{V}_t^{\overline{\psi}_t}(\cdot) >\right\}$$

    Retrieve V-function:

$$\hat{V}_t^{\overline{r}_t}(x) := < \pi_t(\cdot \mid x), \hat{Q}_t^{\overline{r}_t}(x,\cdot) >$$
$$\hat{V}_t^{\overline{g}_{i,t}}(x) := < \pi_t(\cdot \mid x), \hat{Q}_t^{\overline{g}_{i,t}}(x,\cdot) >, \quad \forall i \in [m]$$
$$\hat{V}_t^{\overline{\psi}_t}(x) := < \pi_t(\cdot \mid x), \hat{Q}_t^{\overline{\psi}_t}(x,\cdot) >$$

  end for

end for

set $\hat{Q}_t^{\overline{z}_t}(\cdot) := \hat{Q}_t^{\overline{r}_t}(\cdot) + \sum_{i\in[m]}(\lambda_t \hat{Q}_t^{\overline{g}_{i,t}}(\cdot)) + \tau \hat{Q}_t^{\overline{\psi}_t}(\cdot)$

output $\hat{Q}_t^{\overline{z}_t}(\cdot)$ and $\hat{V}_t^{g_i}(s_0,0), \forall i \in [m]$

It is worth noting that these truncated estimates are all lower bounded by zero and upper bounded by the actual value functions under the estimated model. Moreover, they do not necessarily correspond to the true value function of a policy.

4.2.3. Dual Update

The Lagrange multipliers are updated via a mirror-descent step using the optimistic constraint value estimates and projected onto a convex set Λ:

$$\lambda_{t+1} = \text{proj}_{\Lambda} \left((1 - \eta\tau)\lambda_t - \eta(\hat{V}_t^{g_i} - \alpha_i), \quad \forall i \in [m] \right).$$

4.2.4. Primal Update

The policy is updated through an online mirror descent (OMD) step with entropic regularization:

$$\pi_{t+1(a|x)} \propto \pi_t(a \mid x) exp \left(\eta \hat{Q}_{t,\bar{z}_t}(x, a) \right).$$

4.2.5. Full Algorithm

The complete pseudocode of the Regularized Primal-Dual Algorithm with Optimistic Exploration is reported, followed by its theoretical guarantees:

Algorithm 4.3 Regularized Primal-Dual Algorithm with Optimistic Exploration

1: Input: $\Lambda = [0, \lambda_{max}]^m$, learning rate $\eta > 0$, regularization parameter $\tau > 0$, number of episodes T
2: set π_0 to the uniform policy ($\pi_0(a \mid x) = 1/A, \forall (x,a) \in |X \times A|$)
3: **for** $t = 0, ..., T-1$ **do**
4: update $\overline{r}_t, \overline{g}_t, \overline{p}_t, \overline{\psi}_t$ according to Equation 4.3
5: perform truncated policy evaluation:

$$\hat{Q}_{t,\bar{z}_t}(\cdot), \hat{V}_t := \text{Eval}(\pi_t, \lambda_t, \overline{r}_t, \overline{g}_t, \overline{p}_t, \overline{\psi}_t)$$

6: update policy for all $x, a \in |X \times A|$:

$$\pi_{t+1(a|x)} \propto \pi_t(a \mid x) exp\left(\eta \hat{Q}_{t,\bar{z}_t}(x,a)\right)$$

7: update dual variables:

$$\lambda_{t+1} = \text{proj}_\Lambda \left((1 - \eta\tau)\lambda_t - \eta(\hat{V}_t^{g_i} - \alpha_i), \quad \forall i \in [m]\right).$$

8: interact with environment following policy π_t and receiving feedback following Algorithm 4.1
9: update estimators $\overline{r}_t, \overline{g}_t, \overline{p}_t, \overline{\psi}_t$ according to Equation 4.1
10: **end for**

Moreover, the exact regret and violation bounds are given by the following theorems [32]:

Theorem 4.1. *Let $\tau = T^{-1/7}, \eta = (H^2 m)^{-1} \rho T^{-5/7}, \lambda_{max} = H\rho^{-1} T^{1/14}$. Then, with probability at least $1 - \delta$, Algorithm 4.3 obtains a regret of:*

$$R_t \leq \mathcal{O}\left(H^{9/2} m \rho^{-2} m^{1/2} + H^2 |X|^{1/2} |A|^{1/4}\right) T^{0.93}$$

with probability at least $1 - \mathcal{O}(\delta)$,

where ρ is a problem specific parameter related to the feasibility of the problem (see Equation 4.8).

Similarly, for the violations:

Theorem 4.2. *Let* $\tau = T^{-1/7}, \eta = (H^2 m)^{-1}\rho T^{-5/7}, \lambda_{max} = H\rho^{-1}T^{1/14}$. *Then, with probability at least* $1-\delta$, *Algorithm 4.3 obtains a violation of:*

$$V_t \leq \mathcal{O}\left(H^{9/2} m \rho^{-2} m^{1/2} + H^2 |X|^{1/2}|A|^{1/4} + H(1+\rho^{-1})\right) T^{0.93}$$

with probability at least $1-\mathcal{O}(\delta)$.

4.3. CPD-PO

The second algorithm considered in this comparison is CPD-PO, recently introduced by Stradi et al. [51]. This method closes the previous optimality gap and achieves the optimal regret rate $R_T \leq \widetilde{O}(\sqrt{T})$ while maintaining computational efficiency. As in the first approach, CPD-PO also follows a primal-dual scheme. Therefore, empirical estimators are constructed and subsequently used both in the dual update and to generate artificial feedback for the primal update.

4.3.1. Estimators

CPD-PO relies on empirical estimates of rewards, constraints, and transition dynamics. Let $N_t(x,a)$ denote the number of episodes up to time $t-1$ in which the state-action pair $(x,a) \in X \times A$ has been visited. The empirical reward and constraint estimators are defined as

$$\hat{r}_t(x,a) := \frac{\sum_{\tau=0}^{t-1} r_\tau(x,a)\mathbb{1}_\tau(x,a)}{max\{1, N_t(x,a)\}}, \tag{4.4}$$

$$\hat{g}_{i,t}(x,a) := \frac{\sum_{\tau=0}^{t-1} g_{i,\tau}(x,a)\mathbb{1}_\tau(x,a)}{max\{1, N_t(x,a)\}}. \tag{4.5}$$

To account for uncertainty, a UCB-style approach is used so that, with high probability, the estimation error does not exceed the prescribed bound. Such bounds are defined for rewards and constraints, respectively:

$$\phi_t(x,a) := \min\left\{1, \sqrt{\frac{4\ln(T|X||A|/\delta)}{max\{1, N_t(x,a)\}}}\right\}, \tag{4.6}$$

$$\xi(x,a) := \min\left\{1, \sqrt{\frac{4\ln(T|X||A|m/\delta)}{max\{1, N_t(x,a)\}}}\right\}. \tag{4.7}$$

Transition estimates are estimated in a similar way: let $M_t(x,a,x')$ be the total number of episodes in which the state-action pair $(x,a) \in X \times A$ is visited and the environment

evolves to the new state $x' \in X$. Then, the transitions dynamics can be estimated by means of:

$$\hat{P}_t(x' \mid x, a) := \frac{M_t(x, a, x')}{\max\{1, N_t(x, a)\}}.$$

For notational convenience, $\overline{r}_t := \hat{r}_t, \underline{g}_{i,t} := \hat{g}_{i,t}$ represent the compact notation for the optimistic values of the estimated rewards and constraints, respectively.
Finally, a problem-specific parameter $\rho \in [0, H]$ is introduced, which is related to the feasibility of the problem:

Finally, a problem-specific parameter $\rho \in [0, H]$ is defined. it is related to the feasibility of the problem:

$$\rho := \max_{\pi \in \Pi} \min_{i \in [m]} (\alpha_i - V^{\pi}(g_i)). \tag{4.8}$$

4.3.2. Dual Update

The dual update follows a novel scheme in which, rather than performing a mirror-descent update, the Lagrange multipliers are selected through a binary decision at each round. Specifically, the choice of the multiplier for each constraint depends on whether the current policy satisfies feasibility condition according to the optimistic estimates. This design aims to emphasize reward maximization when the current policy is estimated to be feasible, and to increase the weight placed on constraint satisfaction otherwise. The update is reported:

$$\lambda_{i,t} = arg\max_{\lambda \in \{0, H+1/\rho\}} \lambda \left(V^{\pi_t, \hat{P}_t}(\underline{g}_{i,t} - \alpha_i) \right), \quad \forall i \in [m].$$

4.3.3. Primal Update

The primal component of CPD-PO is based on the Policy Optimization with Dilated Bonuses (PO) framework introduced by Luo et al. [29], which is state-of-the-art for loss minimization in adversarial MDPs. The key innovation of their approach lies in the construction of *dilated bonuses*: they consist of a base bonus b_t defined for each state $x \in X$, and the actual dilated bonus is calculated recursively. They help propagate exploration incentives across future steps, with additional importance put on deeper nodes:

$$B_t(x, a) := b_t(x, a) + \left(1 + \frac{1}{H}\right) \mathbb{E}_{x' \sim \hat{P}_t(\cdot \mid x, a)} \mathbb{E}_{a' \sim \pi_t(\cdot \mid x')} \left[B_t(x', a') \right].$$

On the other hand, the policy update relies on a multiplicative weight scheme:

$$\pi_t(a \mid x) \propto \exp\left(-\eta \sum_{t=0}^{t-1} \left(\hat{Q}_\tau(x,a) - B_\tau(x,a)\right)\right),$$

where $\eta > 0$ is the learning rate, $\hat{Q}_\tau(x,a)$ denotes the cumulative estimated loss for the state-action pair (x,a) at round τ, and $B_\tau(x,a)$ is the corresponding dilated bonus. This update rule can be interpreted as an online mirror descent step in policy space, where actions incurring higher loss (after bonus correction) are exponentially down-weighted.

4.3.4. Full Algorithm

The complete algorithm can now be reported. At each round, the empirical estimators are first updated and then used to determine the dual variable. Subsequently, these estimators are employed to construct the artificial loss fed to the primal learner. Finally, the primal policy update is performed by PO. The full pseudocode of the algorithm is reported below:

Algorithm 4.4 Constrained Primal-Dual Policy Optimization (CPD-PO)

Input: confidence parameter $\delta \in [0,1]$, learning rate $\eta > 0$, number of episodes T
set π_0 to the first policy prescribed by PO-DB
Initialize all the estimators, counters and bounds
for $t = 0, ..., T-1$ **do**
  interact with the environment with policy π_t as in Algorithm 4.1
  observe $(x_h, a_h), r_t(x_h, a_h)$ and $g_{i,t}(x_h, a_h)$, for all $h \in [H]$ and $i \in [m]$ as feedback
  build estimators $\hat{r}_t, \hat{g}_{i,t}, \phi, \xi$ and $\hat{P}_t$ as prescribed in Section 4.3.1
  update dual variables:

$$\lambda_{i,t} = arg\max_{\lambda \in \{0, H+1/\rho\}} \lambda \left(V^{\pi_t, \hat{P}_t}(\underline{g}_{i,t} - \alpha_i) \right), \quad \forall i \in [m]$$

  build artificial loss for every pair (x_h, a_h) received as feedback:

$$\ell_t(x_h, a_h) \leftarrow \frac{(H+1)m}{\rho} - \left[\overline{r}_t(x_h, a_h) - \sum_{i \in [m]} \lambda_{i,t} \left(\underline{g}_{i,t}(x_h, a_h) - \frac{\alpha_i}{H} \right) \right]$$

  update policy $\pi_{t+1} \leftarrow$ PO-DB with artificial feedback $\{(x_h, a_h), \ell_t(x_h, a_h)\}$ for all $h \in [H]$
end for

Finally, the exact regret and violation bounds are given by the following theorems [51]:

Theorem 4.3. *Given any $\delta \in (0,1)$, Algorithm 4.4 attains:*

$$R_t \leq \widetilde{\mathcal{O}} \left(\frac{H^3 m}{\rho} |X| \sqrt{|A|T} + \frac{H^5 m}{\rho} \right),$$

with probability at least $1 - \mathcal{O}(\delta)$.

Similarly, for the guarantees:

Theorem 4.4. *Given any $\delta \in (0,1)$, Algorithm 4.4 attains:*

$$V_t \leq \widetilde{\mathcal{O}} \left(\frac{H^3 m}{\rho} |X| \sqrt{|A|T} + \frac{H^5 m}{\rho} \right),$$

with probability at least $1 - \mathcal{O}(\delta)$.

4.4. Results

As anticipated from the theoretical analysis, the first algorithm has weaker regret guarantees compared to CPD-PO. This is also confirmed empirically: across all experimental settings considered, CPD-PO consistently achieves lower cumulative regret.
Experiments were conducted on three environments and in all cases, rewards and constraint follow Bernoulli distributions. Furthermore, each state is designed to induce an explicit reward-constraint tradeoff: actions yielding higher expected rewards also incur higher expected constraint costs.

4.4.1. Setting I: Single-State Environment.

The first scenario consists of a single state with three available actions and one constraint. The actions are characterized by

$$(r, g) \in \{(0.25, 0.25), (0.5, 0.5), (0.75, 0.75)\},$$

with constraint threshold $\alpha = 0.5$. This configuration forces the agent to balance reward maximization against constraint feasibility, as the highest-reward action violates the constraint threshold in expectation.

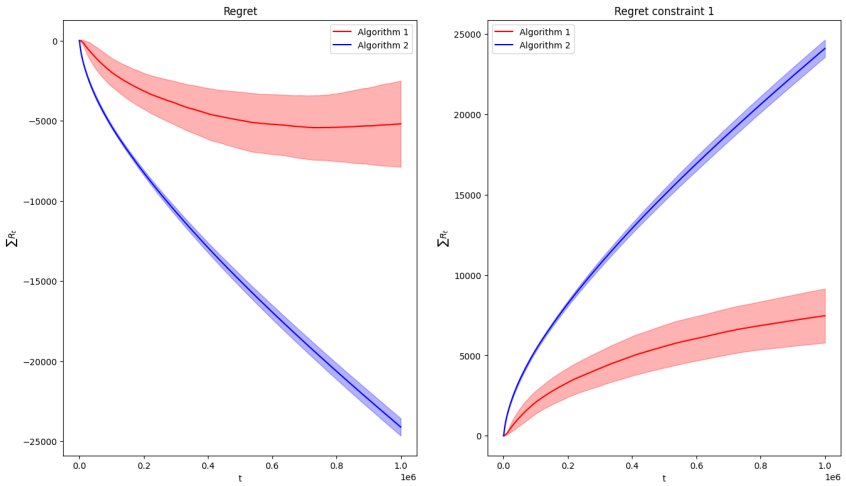

Figure 4.2: Regret and violations in single state environment[1].

[1]Note that, in the figures, the notion of *weak* regret and *weak* violations are used to better understand the behaviour of the algorithms.

4.4.2. Setting II: Three-Layer Environment.

The second scenario introduces a layered structure with three layers and three states per layer. Each state admits the same three actions as in Setting I, while the constraint threshold is set to $\alpha = 1.5$.

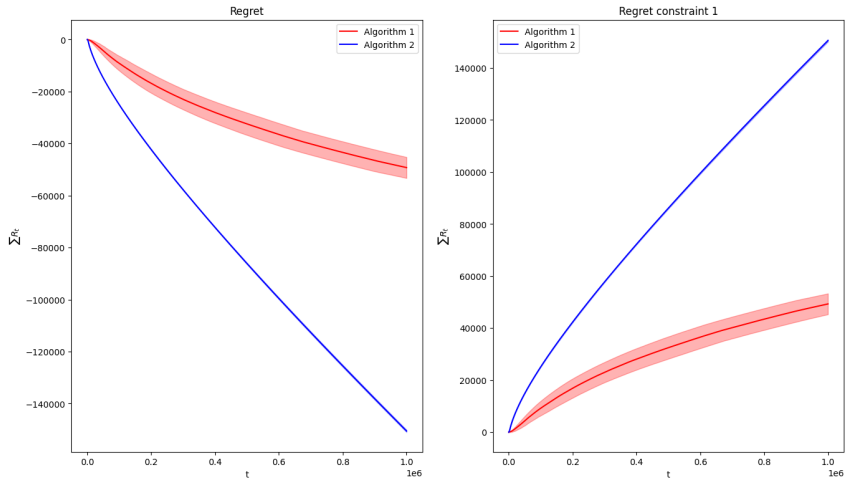

Figure 4.3: Regret and violations in environment with 3 layers and 3 states per layer.

4.4.3. Setting III: Five-Layer Environment.

The third scenario further increases the horizon and consists of five layers with three states per layer. The action structure remains unchanged across states and the constraint threshold is scaled to $\alpha = 2.5$.

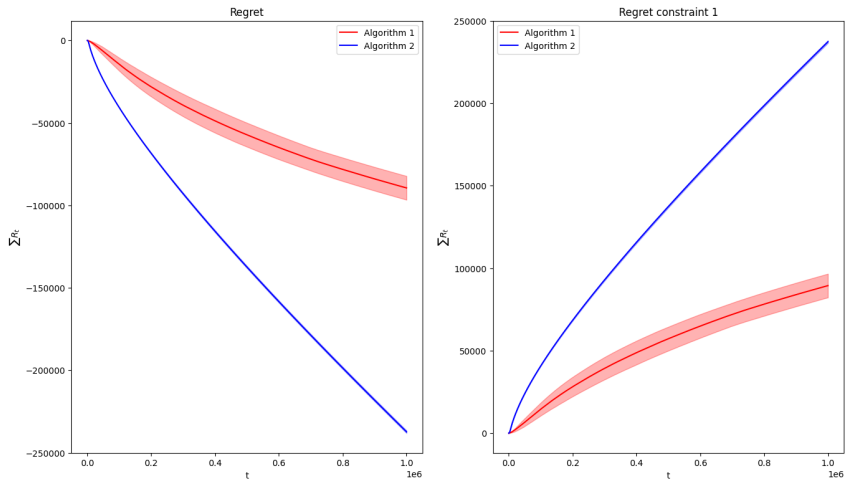

Figure 4.4: Regret and violations in environment with 5 layers and 3 states per layer.

4.4.4. Setting IV: Procedurally Generated Functions.

The last scenario considered consists in replicating the same conditions as in Muller et al. [32], where the means of reward and constraint distribution are sampled at random, so that there is a tradeoff.

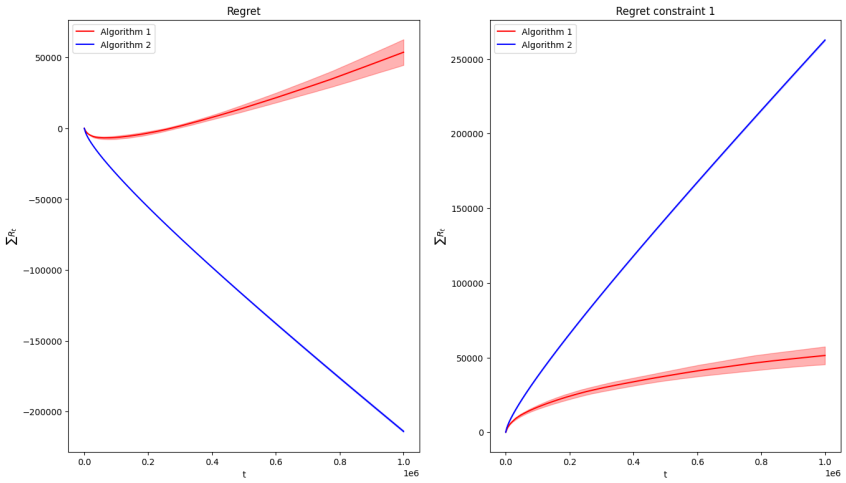

Figure 4.5: Regret and violations in procedurally generated environment with 3 layers and 3 states per layer.

In all the settings considered, the CPD-PO algorithm achieves better performance, in particular it exhibits much tighter constraint violations.

5 | The CPD-FTRL Algorithm

With all the fundamental concepts being introduced, the main contribution of this thesis can now be presented. In particular, this chapter presents the *Constrained Primal-Dual Adaptive Follow the Regularized Leader* (CPD-FTRL), a novel algorithm based on a primal-dual scheme similar to the one introduced by Stradi et al. [51], but designed for Constrained TFSDMs. Compared to the state-of-the-art literature for this setting, such approach achieves optimal performance efficiently and without additional assumptions.

5.1. Setting

This section introduces the general setting and assumptions on which the novel algorithm is based, as well as the relevant considerations useful to support the proofs and the results for this research.
Specifically, the setting taken into account is a Constrained Tree-Form Sequential Decision Making problem (CTFSDM) with stochastic rewards and constraints, and unknown transitions.
In such scenario, a single agent iteratively interacts with an environment taking sequential decisions and receives stochastic rewards and constraints. The goal of the agent is to maximise the cumulative rewards, while keeping constraints below a given threshold. Rewards and constraints are assumed to be bounded between 0 and 1.
This setting is similar to the one of Bernasconi et al. [7] in the soft-threshold version, but with an important distinction: in their case, the agent only receives a feedback in the terminal nodes, whereas here, the agent receives a feedback after each action.
More precisely, this setting is defined by a tuple $< S, X, A, \nu, m, r, g, \alpha >$, where:

- S set of states organized in a tree structure;
- X information set, partition of S. For a state $s \in S$, $x(s) \in X$ denotes an information set such that $s \in x(s)$;
- A_x set of actions available in $x \in X$;
- $\nu : S \to [0,1]^{|X|}$ probability distribution, with $\nu_s(x)$ denoting the probability of the

state s emitting signal x;

- $m \in \mathbb{N}_{>0}$ number of constraints;
- $r : S \times A \to \mathbb{R}$ is the reward function. $r(s, a)$ represents the reward taken playing action a in state s. $r \in [0, 1]^{|S \times A|}$ represents the corresponding reward vector;
- $g \in [0, 1]^{m|S \times A|}$ is the cost function. $g(s, a)$ represents the costs taken playing action a in state s. $G \in [0, 1]^{m|S \times A|}$ represents cost matrix, with $g_i(s, a)$ representing the cost incurred playing action a in state s for each constraint $i \in [m]$;
- $\alpha \in [0, H]^m$ threshold vector, the agent must keep the total costs for the i-th constraint below α_i, $i \in [m]$.

Figure 5.1 illustrates an example of a CTFSDM.

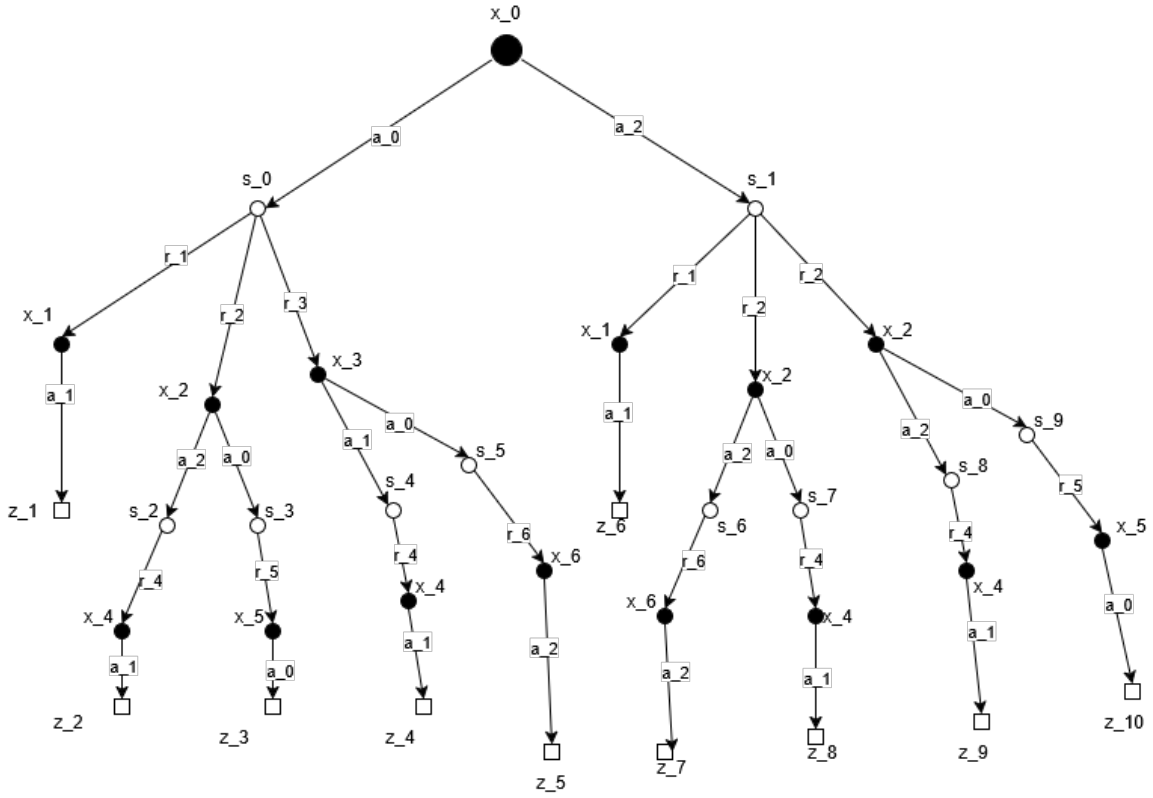

Figure 5.1: Example of CTFSDM: observation nodes (denoted by s) follow the action nodes (denoted by x): in each action node, the agent takes an action, the environment transitions to the next state that emits a signal, until a terminal node (denoted by z) is reached. It is worth noting that different states may emit the same signal: these states are indisinguishable to the agent and the available actions are identical.

The overall agent-environment interaction is detailed below:

Algorithm 5.1 Agent-Environment interaction

Input: Policy $\pi : X \times A \to [0, 1]$
Environment initializes state $s_0 \in S$
for $h = 0, ..., H - 1$ **do**
    environment draws signal $x_h \sim \nu_{s_h}$
    agent observes x_h
    agent plays action $a_h \sim \pi(\cdot \mid x_h)$
    agent receives reward $r(s_h, a_h)$
    agent incurs in cost $g_i(s_h, a_h)$ for each $i \in [m]$
    environment evolves to state s_{h+1}
end for

5.2. Optimization in Constrained TFSDMs

As discussed in the previous chapter (see Subsection 2.4.2), policies can be parametrized using the *sequence-form* representation under the setting just introduced. This representation allows to define the reward or cost achieved by a policy as a scalar product.
Formally, a sequence policy $\pi \in \Pi$ is a vector that assigns to each information set/action pair (x, a) its corresponding realization probability. Then, for any $x \in X$, let $\theta(x)$ be the probability of observing x due to environment transitions only. The expected utility that can be obtained by playing (x, a) can be written as:

$$r^*(x, a) := \theta(x)\mathbb{E}[r(x, a)]$$

and, similarly, the expected cost:

$$g_i^*(x, a) := \theta(x)\mathbb{E}[g_i(x, a)].$$

These probabilities are collected into a vector $\theta \in [0, 1]^{|X|}$ where each element θ_x corresponds to $\theta(x)$. Finally, let r^* and g^* be the vectors where each element $r^*_{x,a}$, $g^*_{i,x,a}$ corresponds to $r^*(x, a)$ and $g_i^*(x, a)$ respectively. Then, the expected reward received following a sequence π is given as the scalar product $\pi^\top r^*$. Similarly for the costs: $\pi^\top g_i^*$.
Let Π^* be the set of policies that are safe. Formally:

$$\Pi^* := \{\pi \in \Pi^* \mid \pi^\top g_i^* \leq \alpha_i, \forall i \in [m]\}.$$

It is assumed that there is always a strictly feasible policy π° such that:

$$\pi^{\circ\top} g_i^* < \alpha_i, \quad \forall i \in [m],$$

which means that Π^* is always non empty. The optimal policy π^* is then defined as:

$$\pi^* := \max_{\pi \in \Pi^*} \pi^\top r^*$$

or, similarly, by means of a linear program:

$$\mathrm{OPT}_{r,G} := \begin{cases} \max\limits_{\pi \in \Pi} & \pi^\top r^* \\ \text{s.t.} & \pi^\top g_i^* \leq \alpha_i \quad \forall i \in [m]. \end{cases}$$

The metrics used to evaluate the algorithm performance are *strong* Regret and *strong* Constraint Violation. The Regret measures the loss in total reward achieved by an agent relative to always playing the optimal strategy:

$$R_T := \sum_{t=1}^{T} \left[\pi^{*\top} r^* - \pi_t^\top r^* \right]^+.$$

Similarly, Violation evaluates how much the algorithm violates the constraints during the learning process:

$$V_T := \max_{i \in [m]} \sum_{t=1}^{T} \left[\pi^\top g_i^* - \alpha_i \right]^+.$$

Finally, in this setting, the Lagrangian function can be defined as follows:

Definition 13. *Given a CSDM characterised by a reward vector $r \in [0,1]^{|X \times A|}$ and cost matrix $G \in [0,1]^{i \times |X \times A|}$, the Lagrangian function $\mathcal{L}_{r,G}(\pi, \lambda) : \Pi \times \mathbb{R}^m_{\geq 0} \to \mathbb{R}$ is defined for every policy $\pi \in \Pi$ and lagrangian vector $\lambda \in \mathbb{R}^m_{\geq 0}$*

$$\mathcal{L}_{r,G}(\pi, \lambda) := \pi^\top r^* - \sum_{i \in [m]} \lambda_i (\pi^\top g_i^* - \alpha_i).$$

5.3. Main Components

Under the setting described, the main components of the novel algorithm are presented, namely the estimators and the primal algorithm.

5.3.1. Estimators

Since the algorithm follows a primal-dual approach, estimators are employed to construct an artificial feedback, which is used to feed the primal algorithm, and to update the dual variable.
More precisely, let $N_t(x,a)$ be the number of episodes in which each pair $(x,a) \in X \times A$ is visited until time $t-1 \in [T]$ and let $\mathbb{1}_\tau(x,a)$ be the indicator function equal to 1 when, at round τ the pair $(x,a) \in X \times A$. Then, the following estimators are defined:

- $\hat{r}_t(x,a) := \frac{\sum_{\tau=1}^{t-1} r_\tau(x,a)\mathbb{1}_\tau(x,a)}{max\{1,N_t(x,a)\}}$,
- $\hat{g}_{i,t}(x,a) := \frac{\sum_{\tau=1}^{t-1} g_{i,\tau}(x,a)\mathbb{1}_\tau(x,a)}{max\{1,N_t(x,a)\}}$.

These estimators are unbiased as they represent the empirical mean of the observed rewards and constraints for each state-action pair (x,a).
By applying Hoeffding's inequality, the following lemmas hold.

Lemma 2. *Given a confidence parameter $\delta \in (0,1)$, the following holds for every episode $t \in [T]$ and state-action pair $(x,a) \in X \times A$ with probability at least $1-\delta$:*

$$|\hat{r}_t(x,a) - r(x,a)| \leq \phi_t(x,a), \text{ with } \phi_t(x,a) := min\left\{1, \sqrt{\frac{\ln(T|X||A|)/\delta}{max(1,N_t(x,a)}}\right\}.$$

Lemma 3. *Given a confidence parameter $\delta \in (0,1)$, the following holds for every episode $t \in [T]$ and state-action pair $(x,a) \in X \times A$ with probability at least $1-\delta$:*

$$|\hat{g}_{i,t}(x,a) - g_i(x,a)| \leq \xi_t(x,a), \text{ with } \xi_t(x,a) := min\left\{1, \sqrt{\frac{\ln(T|X||A|m)/\delta}{max(1,N_t(x,a)}}\right\}.$$

The estimators $\hat{r}_t$, $\hat{g}_t$ are used to build the artificial reward which is then fed into the primal algorithm.
Furthermore, one last estimator needs to be defined for the dual update. Its main purpose is to evaluate the current policy and update λ accordingly.
Let $\mathbb{1}_\tau^{play}(x,a)$ be the indicator function equal to 1 when, at round τ, a sequence π_τ that leads to the action pair (x,a) is played, and $\mathbb{1}_\tau^{env}(x,a)$ the indicator function equal to 1 when, at round τ, following the agent policy, the environment actually leads to the pair (x,a). Finally, given $N_t^{play} := \sum_{\tau=0}^{t} \mathbb{1}_\tau^{play}(x,a)$, the following estimator is defined:

$$\widetilde{\hat{g}}_{i,t}(x,a) = \frac{\sum_{\tau=1}^{t-1} g_{i,\tau}(x,a)\mathbb{1}_\tau^{play}(x,a)\mathbb{1}_\tau^{env}(x,a)}{N_t^{play}(x,a)},$$

where $\widetilde{\hat{g}}_i$ is an estimator of g_i^* and implicitly contains the transitions of the environment. Moreover, similarly to Lemma 2 and 3 above:

Lemma 4. *Given a confidence parameter $\delta \in (0,1)$, the following holds for every episode $t \in [T]$ and state-action pair $(x,a) \in X \times A$ with probability at least $1-\delta$:*

$$|\widetilde{\hat{g}}_{t,i}(x,a) - g_i(x,a)| \leq \widetilde{\xi}_t(x,a), \text{ with } \widetilde{\xi}_t(x,a) := min\left\{1, \frac{\sqrt{\ln(T|X||A|m)/\delta}}{max(1, N_t^{play}(x,a)}\right\},$$

where $\hat{r}_t, \hat{g}_{i,t}, \widetilde{\hat{g}}_{i,t} \in [0,1]^{|X\times A|}$ denote the vectors whose components are the estimated rewards $\hat{r}_t(x,a)$ and constraints $\hat{g}_{i,t}(x,a)$, $\widetilde{\hat{g}}_{i,t}(x,a)$.
In the same way, $\phi_t, \xi_t, \widetilde{\xi}_t \in [0,1]^{|X\times A|}$ are the vectors whose components are the bounds $\phi_t(x,a)$, $\xi_t(x,a)$, $\widetilde{\xi}_t(x,a)$, and $\overline{r}_t := \hat{r}_t + \phi_t$, $\underline{r}_t := \hat{r}_t - \phi_t$, $\overline{g}_{i,t} := \hat{g}_{i,t} + \xi_t$, $\underline{g}_{i,t} := \hat{g}_{i,t} - \xi_t$, $\widetilde{\overline{g}}_{i,t} := \widetilde{\hat{g}}_{i,t} + \widetilde{\xi}_t$, $\widetilde{\underline{g}}_{i,t} := \widetilde{\hat{g}}_{i,t} - \widetilde{\xi}_t$ are the upper and lower bounds of the estimation, and are needed to be optimistic.
Finally, $\rho \in [0, L]$ denotes the largest margin on the constraints that can be achieved in the setting. It depends on the feasibility of the problem, i.e.:

$$\rho := \max_{\pi \in \Pi} \min_{i \in [m]} (\alpha_i - \pi^\top g_i^*).$$

5.3.2. Primal algorithm

Before presenting the complete algorithm, another preliminary step is necessary, namely the introduction of the primal component, which will later be combined with the previously defined estimators. In particular, Adaptive FTRL [19] is the primal algorithm employed in this work. Although originally designed for two-player Imperfect Information Games, its use can be extended to learning in adversarial TFSDMs, where a single agent interacts with an adversarial environment.
The authors Fiegel et al. [19] originally proposed two algorithms:

- **Balanced FTRL**, for settings where the structure of the tree is known and the number of actions is not fixed;
- **Adaptive FTRL**, for settings where the tree structure is unkown and the number of actions is fixed.

Both versions rely on a FTRL-like update, but the second scenario is the one related to this thesis, as it has less assumptions on the environment.

Balanced FTRL

The balanced version uses a *balanced transition* kernel p^*, following the idea of Bai et al. [6].
Formally, for an information set $x \in X$, define

$$A_x^\tau = \sum_{x'>x} A_{x'}$$

as the total number of actions that follow information set x in the whole sub-tree. Then, the balanced transition kernel is defined so that, at each step, the probability of transitioning to the next suitable information set x' is proportional to this value:

$$p_h^*(x^{h+1} \mid x^h, a^h) := \frac{A_\tau^{x_{h+1}}}{\sum_{x'^{h+1} \in X^{h+1}(x^h,a^h)} A_\tau^{x'^{h+1}}},$$

where $X^{h+1}(x^h, a^h)$ denotes all the possible information sets of depth $h+1$ that follow (x^h, a^h).
The balanced transition kernel is then used by the regularization function that ensures acceptable bounds against any adversarial transition.
Finally, the loss is computed by means of the following estimator:

$$\widetilde{\ell}_t^h(x^h, a^h) := \frac{\mathbb{1}_{\{x_t^h = x^h, a_t^h = a^h\}}}{\pi_t^{1:h}(x^h, a^h) + \gamma^h(x^h)}(1 - r_t^h),$$

where the parameter γ^h is an implicit exploration (IX) parameter, which renders the estimator biased, but also lowers its variance and promotes exploration.
The cumulative loss is given as $\widetilde{L}_t := \sum_{k=1}^t \widetilde{\ell}_k$.
The final policy update is as follows:

$$\pi_t = arg\min_{\pi \in \Pi} \langle \pi^{1:}, L_{t-1} \rangle + \sum_{h=1}^{H} \Psi^h(p^{*1:h} \cdot \pi^{1:h})/\eta^h,$$

where $\Psi(\cdot)$ is the regularization function, either Tsallis entropy [57] or Shannon entropy [44], p^* is the balanced transition kernel, η is the learning rate vector and the notation $v^{n:m}$ of a vector defines all the elements from position n to m.

Adaptive FTRL

Compared to the Balanced FTRL, its Adaptive counterpart has a slightly different setting, where no knowledge of the tree structure is assumed. In this case, the balanced transition

kernel p^* can not be computed, however it is possible to estimate the actual adversarial transitions.

$$\hat{p}_t^{1:h} := \mathbb{1}_{\{x_h = x_t^h, a^h = a_t^h\}} / \pi_t^{1:h}(x^h, a^h).$$

Additionally, the IX parameter becomes data dependant: early on, it allows for a more aggressive exploration, while later on it shrinks in the frequently visited nodes. To compute the IX parameter, the transition estimates are used :

$$\gamma_t^h(x^h, a^h) := \gamma / \left(1 + \widetilde{P}_{t-1}^{1:h}(x^h, a^h)\right),$$

where $\widetilde{P}_t := \sum_{k=1}^{t} \widetilde{p}_k$ is the cumulative estimated transition and $\gamma_t^h(x^h, a^h)$ the IX parameter that now also depends on the action a^h and time t.
The learning rate is also data dependant and is updated for each information set x depending on the visit counts:

$$\eta_t^h(x^h) := \min_{x'^{h'} \geq x^h} \eta / 1 + \widetilde{P}_{t-1}^{1:h'}(x'^{h'}).$$

The regularization function used is the weighted dilated Shannon divergence [28], which is defined between two policies $\pi_0, \pi_1 \in \Pi$ for a given vector of learning rates η_t:

$$D_{\eta_t}(\pi_0, \pi_1) := \sum_{h=1}^{H} \sum_{(x^h, a^h) \in A(X^h} \frac{\pi_1^{1:h}(x^h, a^h)}{\eta_t^h(x^h)} \log \frac{\pi_1^h(a^h \mid x^h)}{\pi_0^h(a^h \mid x^h)},$$

resulting in the following update:

$$\pi_t = arg \min_{\pi \in \Pi} \langle \pi^{1:}, L_{t-1} \rangle + D_{\eta_t}(\pi, \pi_0),$$

with $\pi_0 \in \Pi$ a base policy fixed over all iterations.
For clarity, the complete pseudocode is also reported [19]:

Algorithm 5.2 Adaptive FTRL

Input: learning rate η, IX base parameter γ, base policy π_0 (uniform by default)
for $t = 1, ..., T$ **do**
  for all depth h and $x_h \in X_h$ **do**
    $\eta_t^h(x_h) \leftarrow \min_{x'^{h'} \geq x^h} \eta / \left(1 + \widetilde{P}_{t-1}^{1:h'}(x'^{h'})\right)$
    for all $a^h \in A(x^h)$ **do**
      $\gamma_t^h(x^h, a^h) \leftarrow \gamma / \left(1 + \widetilde{P}_{t-1}^{1:h}(x^h, a^h)\right)$
    end for
  end for
  Compute update:

$$\pi_t = arg \min_{\pi \in \Pi} \langle \pi^{1:}, L_{t-1} \rangle + D_{\eta_t}(\pi, \pi_0)$$

  for $h = 1, ..., H$ **do**
    Observe information set x_t^h
    Execute $a_t^h \sim \pi(\cdot \mid x_t^h)$ and receive reward r_h^t
    $\widetilde{\ell}(x_t^h, a_t^h) \leftarrow {}^{(1-r_t^h)} / {}_{(\pi_t^{1:h}(x_t^h, a_t^h) + \gamma_h^t(x_t^h, a_t^h))}$
    $\widetilde{p}_t^{1:h}(x_t^h, a_t^h) \leftarrow {}^{1} / {}_{(\pi_t^{1:h}(x_t^h, a_t^h) + \gamma_t^h(x_t^h, a_t^h))}$
  end for
end for

The use of adaptive learning rates and IX parameters allows to naturally adjust to an unknown tree structure.
On the other hand, the algorithm is also computationally efficient. Each policy update has a $\mathcal{O}(HA)$ time complexity. Moreover, as $\widetilde{P}$ only increases on visited information states, the vectors η_t, γ_t can be updated in-place, with the same $\mathcal{O}(HA)$ time complexity. This results in an overall time complexity of $\mathcal{O}(THA)$.

5.4. CPD-FTRL implementation

With all the preliminary concepts introduced, this section presents the complete CPD-FTRL algorithm. In particular, the corresponding pseudocode is detailed below. The algorithm employs an instance of Adaptive FTRL, using the previously described estimators to construct an artificial loss that incorporates both rewards and constraints, which is then fed to the primal algorithm.
In this implementation, at every iteration, all the estimators and confidence bounds are updated. Then, the dual update is performed as a binary choice between two values,

0 and $^{H+1}/_{\rho}$, for each $i \in [m]$, in order to maximize the constraint penalization term $(\pi_t^\top \widetilde{\underline{g}}_{i,t} - \alpha_i), \forall i \in m$. On one hand, the value 0 is selected when the optimistic estimation of the i-th constraint is not violated by the current policy. On the other hand, if the optimistic estimation is not satisfied, then the value assigned is $^{H+1}/_{\rho}$. This quantity is chosen to be large enough to guarantee that any policy cannot gain more than $\mathrm{OPT}_{r,G}$ in the Lagrangian game with the true rewards and constraints. To perform the primal update, Adaptive FTRL is employed. Since it is tailored for adversarial, positive and bounded rewards, the artificial feedback is rescaled to be between $[0, 1]$.

Algorithm 5.3 Constrained Primal-Dual Follow the Regularized Leader (CPD-FTRL)

Input: Number of rounds $T \in \mathbb{N}_{>0}$, problem-specific parameter $\rho \in [0, H]$, confidence $\delta \in (0, 1)$
Initialise $\pi_0 \leftarrow$ uniform policy
Initialise all estimators, bounds and counters
for $t = 1, ..., T$ **do**
    Interact as in Algorithm 5.1 with $\pi = \pi_t$
    Observe $(x_t, a_t), r_t(x_t, a_t), g_{t,i}(x_t, a_t)$ for every $i \in [m]$ and $h \in \{0, ..., H-1\}$ as feedback
    Update estimators, bounds and counters $\hat{r}, \hat{g}, \widetilde{\hat{g}}, \phi, \xi, \widetilde{\xi}$ as prescribed in Section 5.3.1
    $\lambda_{t,i} \leftarrow \arg\max_{\lambda \in \{0, \frac{H+1}{\rho}\}} \lambda(\pi_t^\top \widetilde{\underline{g}}_{i,t} - \alpha_i) \qquad \forall i \in m$
    Build artificial reward for every pair (x_h, a_h) received as feedback:

$$\tilde{r}_t(x_k, a_k) \leftarrow \frac{\frac{(H+1)}{\rho} + \left[\overline{r}_t(x_k, a_k) - \sum_{i \in [m]} \lambda_{t,i}(\underline{g}_{t,i}(x_k, a_k) - \frac{\alpha_i}{H})\right]}{\frac{2(H+1)}{\rho} + 1}$$

    Update policy π_{t+1} employing **Balanced FTRL** with artificial rewards $\tilde{r}_t(x_h, a_h), \ \forall h \in \{0, ..., H-1\}$
end for

5.5. Theoretical Guarantees

The present section focuses on the theoretical guarantees attained by Algorithm 5.3. In particular, a high-level overview is provided, whereas an in-depth analysis along with proofs can be found in Section 5.6.

5.5.1. Results on the Lagrangian Formulation

Starting from the Lagrangian formulation, preliminary results regarding the primal-dual scheme employed by the algorithm are stated.
First of all, the dual variable decision space is well-defined. Indeed, given any policy $\pi_t \in \Pi$ selected by the primal algorithm, the dual decision space is sufficient to upper-bound the Lagrangian function with the constrained optimum.

Lemma 5. *Given a CSDM with reward vector $r \in [0,1]^{|X\times A|}$ and cost matrix $G \in [0,1]^{m|X\times A|}$, for every policy $\pi \in \Pi$ it holds:*

$$\pi^\top r^* - \max_{\lambda \in [0, H+1/\rho]^m} \sum_{i\in[m]} \lambda_i(\pi^\top g_i^* - \alpha_i) \leq OPT_{r,G}.$$

Lemma 5 states that it is not convenient for the primal algorithm to play non-safe policies in order to gain more rewards that one can possibly attain by means of safe policies.
It is important to note that Lemma 5 holds for an optimization problem where rewards, constraints and transitions are known a-priori. However, this is not the case in an online learning setting, where all the parameters are unknown and must be estimated in an online fashion.
If the algorithm were aware of the unknown distributions mentioned above, the dual update of the algorithm combined with Lemma 5 would lead to:

$$\mathcal{L}_{r,G}(\pi^*, \lambda_t) \geq \mathcal{L}_{r,G}(\pi_t, \lambda_t),$$

for all $t \in [T]$. Equation 5.5.1 would be crucial to prove strong regret guarantees, since it shows that any policy chosen by the algorithm cannot outperform π^*. However, it cannot be obtained without complete knowledge of the environment.
It is possible, however, to prove that Equation 5.5.1 holds up to sublinear terms which depend on the uncertainty of the environment estimation.

Lemma 6. *Algorithm 5.3 guarantees that, for every episode $t \in T$, with probability at least $1-\delta$:*

$$\mathcal{L}_{r,G}(\pi^*, \lambda_t) \geq \mathcal{L}_{r,G}(\pi_t, \lambda_t) - \frac{4Hm}{\rho}\pi_t^\top \widetilde{\xi}_t,$$

Lemma 6 shows that Equation 5.5.1 holds up to the term ${}^{4Hm}/_{\rho}\pi_t^\top \widetilde{\xi}_t$, which represents the uncertainty on the constraints and the environment transitions. This quantity decreases as the algorithm collects samples, thus it is sublinear in T over the episodes.

5.5.2. Primal Algorithm

The theoretical guarantees attained by the primal algorithm employed by Algorithm 5.3 are formulated in the following lemma:

Lemma 7. *The instance of Balanced-FTRL employed by Algorithm 5.3 guarantees that, given any $\delta \in (0,1)$ and for any policy $\pi' \in \Pi$, the primal regret $\mathcal{R}_T^{\mathcal{L}}(\pi)$ defined as:*

$$\mathcal{R}_T^{\mathcal{L}}(\pi) := \sum_{t=1}^{T} \left[\pi'^{\top} \theta \overline{r}_t - \sum_{i \in m} \lambda_{i,t} (\pi'^{\top} \theta \underline{g}_{i,t} - \alpha_i) \right] - \sum_{t=1}^{T} \left[\pi_t^{\top} \theta \overline{r}_t - \sum_{i \in m} \lambda_{i,t} (\pi_t^{\top} \theta \underline{g}_{i,t} - \alpha_i) \right]$$

is bounded as follows with probability at least $1 - \mathcal{O}(\delta)$

$$\mathcal{R}_T^{\mathcal{L}}(\pi) \leq \widetilde{\mathcal{O}} \left(\sqrt{\frac{H^2 m}{\rho} |A||X| T \log(1/(4\delta))} \right).$$

Lemma 7 is obtained by the regret guarantees of Adaptive FTRL [19]. Even though the artificial feedback is scaled between $[0,1]$, the combination of the rewards term and the constraint term shrink the reward feedback by a Hm/ρ factor.

5.5.3. Regret and Violation

At this stage, the final cumulative strong regret and strong violation guarantees attained by Algorithm 5.3 can be formulated. More precisely, the following Theorem 5.1 shows that the regret guarantees are of order $\widetilde{O}(\sqrt{T})$:

Theorem 5.1. *Given any $\delta \in (0,1)$, Algorithm 5.3 attains:*

$$R_t \leq \widetilde{\mathcal{O}} \left(\sqrt{\frac{H^2 m}{\rho} |A||X| T \log(1/(4\delta))} \right),$$

with probability at least $1 - \mathcal{O}(\delta)$.

Such Theorem is the result of a combination of Lemma 6 and Lemma 7 (see Section 5.6 complete proof).
Moreover, starting from the bound provided in Lemma 7, it is possible to recover the true Lagrangian function excluding sublinear terms. Through Lemma 6, the cumulative strong regret is recovered, using the regret with respect to the Lagrangian function. Finally, to get the final bound, it is sufficient to notice that the optimal solution $\pi^*$is safe and that, given the dual update performed by the Algorithm, the quantity $\sum_{i \in [m]} \lambda_{i,t} (\pi_t^{\top} g_i^* - \alpha_i)$ is

always non negative.
Analogous results are achieved for the cumulative strong violation.

Theorem 5.2. *Given any $\delta \in (0,1)$, Algorithm 5.3 attains:*

$$V_t \leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right),$$

with probability at least $1-\mathcal{O}(\delta)$.

These guarantees are similar to those of Bernasconi et al. [7].

To get the bound, it is possible to proceed similarly to Theorem 5.2. Employing Lemma 7, the following equality holds:

$$\begin{aligned}
&\sum_{i\in[m]} \lambda_{i,t}\left(\pi_t^\top \theta \widetilde{\underline{g}}_{i,t} - \alpha_i\right) \\
&= \frac{H}{H+1}\sum_{i\in[m]} \lambda_{i,t}\left(\pi_t^\top \theta \widetilde{\underline{g}}_{i,t} - \alpha_i\right) + \frac{1}{H+1}\sum_{i\in[m]} \lambda_{i,t}\left(\pi_t^\top \theta \widetilde{\underline{g}}_{i,t} - \alpha_i\right) \\
&= \frac{H}{H+1}\sum_{i\in[m]} \lambda_{i,t}\left(\pi_t^\top \theta \widetilde{\underline{g}}_{i,t} - \alpha_i\right) + \frac{1}{\rho}\sum_{i\in[m]}\left[\pi_t^\top \theta \widetilde{\underline{g}}_{i,t} - \alpha_i\right]^+, \qquad (5.1)
\end{aligned}$$

where equation 5.1 is obtained by definition of λ_t.
Given the $H/H+1$ factor, it is not possible to apply Lemma 6. However, a similar inequality can be proven:

Lemma 8. *Algorithm 5.3 guarantees that, for every episode $t \in T$, with probability at least $1-\delta$:*

$$\mathcal{L}_{r,G}(\pi^*, \lambda_t H/H+1) \geq \mathcal{L}_{r,G}(\pi_t, \lambda_t H/H+1) - \frac{2Hm}{\rho}\pi_t^\top \widetilde{\xi}_t.$$

Lemma 8 can be employed to sublinearly bound the following quantity:

$$\sum_{t=1}^{T} \mathcal{L}_{r,G}(\pi^*, \lambda_t H/H+1) - \sum_{t=1}^{T} \mathcal{L}_{r,G}(\pi^*, \lambda_t),$$

which in turn allows to sublinearly bound $1/\rho \sum_{t=1}^{T}\sum_{i\in[m]}\left[\pi_t^\top \theta g_i - \alpha_i\right]^+$, excluding the sublinear terms, and gives the desired result.

5.5.4. Comparison with Bernasconi et al.

Compared to the algorithm presented in this research, Bernasconi et al. [7] achieve similar sublinear $\widetilde{O}(\sqrt{T})$ Regret and Violation. With probability at least $1-\delta$, their guarantees are:

$$R_T \leq 4|X \times A|\sqrt{2T\log(2/\delta)},$$
$$V_T \leq 4|X \times A|\sqrt{2T\log(2/\delta)}.$$

However, they rely on a linear program formulation, which is computationally more expensive than the time complexity of Algorithm 5.3, especially as the tree size grows. On the other hand, their algorithm guarantees sublinear regret and violation in the hard threshold problem, which is not possible with a primal-dual approach.
Finally, it should be noted that, despite the difference in the feedback structure, with this setting providing it for every action step and theirs only in the terminal node, the first formulation can be adapted by accumulating feedback across all action steps and delivering it only at the terminal node.

5.6. Theoretical Analysis

Here the in-depth analysis and proofs of the guarantees attained by Algorithm 5.1 are reported.

5.6.1. High Confidence Bounds

Lemma 9. *Given a confidence parameter $\delta \in (0,1)$, with probability at least $1-\delta$, the following holds for a given $t \in [T]$, $(x,a) \in X \times A$, $i \in [m]$:*

$$\left|\hat{r}_t(x,a) - r(x,a)\right| \leq \iota_t(x,a),$$

and, similarly,

$$\left|\hat{g}_{i,t}(x,a) - g_i(x,a)\right| \leq \iota_t(x,a),$$

where

$$\iota_t(x,a) := \sqrt{\frac{\ln(2/\delta)}{2N_t(x,a)}}.$$

Proof. Following Hoeffding's Inequality, for any t, x, a, the following holds:

$$\mathbb{P}(|\hat{r} - r| \geq \iota) \leq 2e^{-2N\iota^2}.$$

Setting the right hand side to δ and solving for ι:

$$\iota = \sqrt{\frac{\ln(2/\delta)}{2N}},$$

which proves the result. □

Lemma 9 holds for a specific t, x, a, i. It is possible to provide high-confidence bounds for all t, x, a, i in a similar way.

Lemma 10. *Given a confidence parameter $\delta \in (0, 1)$, with probability at least $1 - \delta$, the following holds for all $t \in [T]$, $(x, a) \in X \times A$, $i \in [m]$:*

$$\left|\hat{r}_t(x, a) - r(x, a)\right| \leq \phi_t(x, a),$$

where

$$\phi_t(x, a) := \sqrt{\frac{\ln(T|X||A|)/\delta}{max(1, N_t(x, a))}}.$$

Proof. Starting from lemma 9, given a confidence parameter $\delta' \in (0, 1)$, the following holds for a given $t \in [T]$ and $(x, a) \in X \times A$:

$$\mathbb{P}\left[\left|\hat{r}_t(x, a) - r(x, a)\right| \leq \iota_t(x, a)\right] \geq 1 - \delta'.$$

To find the intersection between all the events:

$$\mathbb{P}\{\bigcap_{x,a,t}\left[\left|\hat{r}_t(x, a) - r(x, a)\right| \leq \iota_t(x, a)\right]\} \geq 1 - \delta',$$

which is equal to:

$$\begin{aligned}
&\mathbb{P}\{\bigcap_{x,a,t}\left[\left|\hat{r}_t(x, a) - i(x, a)\right| \leq \iota_t(x, a)\right]\} \\
&= 1 - \mathbb{P}\{\bigcup_{x,a,t}\left[\left|\hat{r}_t(x, a) - r(x, a)\right| \leq \iota_t(x, a)\right]^c\} \\
&= 1 - \mathbb{P}\{\bigcup_{x,a,t}\left[\left|\hat{r}_t(x, a) - r(x, a)\right| > \iota_t(x, a)\right]\} \\
&\geq 1 - \sum_{x,a,t}\mathbb{P}\left[\{\left|\hat{r}_t(x, a) - r(x, a)\right| > \iota_t(x, a)\}\right] \qquad (5.2) \\
&\geq 1 - |X||A|T\delta'.
\end{aligned}$$

Inequality 5.2 holds by union bound, since the events are statistically independent. Substituting $\delta = \frac{\delta'}{|X||A|T}$ gives $\phi_t(x,a)$. □

Lemma 11. *Given a confidence parameter $\delta \in (0,1)$, with probability at least $1-\delta$, the following holds for all $t \in [T]$, $(x,a) \in X \times A$, $i \in [m]$:*

$$\left|\hat{g}_{i,t}(x,a) - g(x,a)\right| \leq \xi_t(x,a),$$

where

$$\xi_t(x,a) := \sqrt{\frac{\ln(T|X||A|m)/\delta}{max(1, N_t(x,a))}}.$$

Lemma 12. *Given a confidence parameter $\delta \in (0,1)$, with probability at least $1-\delta$, the following holds for all $t \in [T]$, $(x,a) \in X \times A$, $i \in [m]$:*

$$\left|\widetilde{\hat{g}}_{i,t}(x,a) - g^*(x,a)\right| \leq \widetilde{\xi}_t(x,a),$$

where

$$\widetilde{\xi}_t(x,a) := \sqrt{\frac{\ln(T|X||A|m)/\delta}{max(1, \widetilde{N}_t(x,a))}}.$$

The proof for lemma 11 and 12 can be obtained in a similar way, also considering the number of constraints m.

5.6.2. Strong Duality

First, it is shown that, when Assumption 5.2 holds, in an optimal solution the vector of Lagrange multipliers is bounded.

Lemma 13. *Given a CSDM with reward vector $r \in [0,1]^{|X\times A|}$ and cost matrix $G \in [0,1]^{i\times|X\times A|}$, it holds:*

$$\min_{\lambda\in\mathbb{R}^m_+:\|\lambda\|_1\in[0,H/\rho]} \max_{\pi\in\Pi} \mathcal{L}_{r,G}(\pi,\lambda) = \max_{\pi\in\Pi} \min_{\lambda\in\mathbb{R}^m_+:\|\lambda\|_1\in[0,H/\rho]} \mathcal{L}_{r,G}(\pi,\lambda) = OPT_{r,G}.$$

The lemma shows that one can restrict the dual search in the region $\{\lambda \geq 0 : \|\lambda\|_1 < {}^H/_\rho\}$, as the optimum is in this region and here the min-max value attains the OPT value.

Proof. First, it is shown that:

$$\min_{\lambda \in \mathbb{R}^m_+ : \|\lambda\|_1 \in [0, H/\rho]} \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \lambda) = \min_{\lambda \in \mathbb{R}^m_{\geq 0}} \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \lambda).$$

Notice that, for every $\lambda \geq 0$ such that $\|\lambda\|_1 > H/\rho$:

$$\begin{aligned} \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \lambda) \geq \mathcal{L}_{r,G}(\pi^\circ, \lambda) &= \pi^{\circ\top} r^* - \sum_i \lambda_i (\pi^{\circ\top} g_i^* - \alpha_i) \\ &\geq \sum_i \lambda_i (\pi^{\circ\top} g_i^* - \alpha_i) \geq \|\lambda\|_1 \rho > H. \end{aligned}$$

In other words: for very large values of λ, the value of the constraint penalization term becomes larger than any feasible primal objective. So, minimizing over a 'too large' λ cannot produce a better value than minimizing over a reasonably bounded set. This is also expressed by the following inequality:

$$\min_{\lambda \in \mathbb{R}^m_+ : \|\lambda\|_1 \in [0, H/\rho]} \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \lambda) \leq \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \underline{0}) \leq H.$$

This implies:

$$\begin{aligned} \min_{\lambda \in \mathbb{R}^m_{\geq 0}} \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \lambda) &= \min \left\{ \min_{\lambda \in \mathbb{R}^m_+ : \|\lambda\|_1 \in [0, H/\rho]} \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \lambda), \min_{\lambda \in \mathbb{R}^m_+ : \|\lambda\|_1 > H/\rho} \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \lambda) \right\} \\ &= \min_{\lambda \in \mathbb{R}^m_+ : \|\lambda\|_1 \in [0, H/\rho]} \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \lambda). \end{aligned}$$

This also implies:

$$\begin{aligned} \mathrm{OPT}_{r,G} &= \max_{\pi \in \Pi} \min_{\lambda \in \mathbb{R}^m_{\geq 0}} \mathcal{L}_{r,G}(\pi, \lambda) \\ &\leq \max_{\pi \in \Pi} \min_{\lambda \in \mathbb{R}^m_+ : \|\lambda\|_1 \in [0, H/\rho]} \mathcal{L}_{r,G}(\pi, \lambda) \\ &\leq \min_{\lambda \in \mathbb{R}^m_+ : \|\lambda\|_1 \in [0, H/\rho]} \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \lambda) \\ &= \min_{\lambda \in \mathbb{R}^m_{\geq 0}} \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \lambda), \end{aligned} \tag{5.3}$$

and, by strong duality [1]:

$$\min_{\lambda \in \mathbb{R}^m_{\geq 0}} \max_{\pi \in \Pi} \mathcal{L}_{r,G}(\pi, \lambda) = \mathrm{OPT}_{r,G},$$

where the second inequality holds by max-min inequality [1]. □

Next, the above result is extended to show that, depending on whether a policy is safe or

not, there exist values of λ such that the optimal value of the Lagrangian coincides with the optimum of OPT.

Lemma 14. *For every reward vector $r \in [0,1]^{|X\times A|}$, cost matrix $G \in [0,1]^{m\times|X\times A|}$ and threshold vector $\alpha \in [0,H]^m$ and $\pi \in \Pi$ s.t. $\pi^\top g_i^* \leq \alpha_i \ \forall i \in [m]$, it holds:*

$$\mathcal{L}_{r,G}(\pi, \underline{0}) \leq OPT_{r,G}.$$

Proof. For $\lambda = \underline{0}$, the Lagrangian reduces to $\mathcal{L}_{r,G}(\pi, \underline{0}) = \pi^\top r^*$, which is simply the expected reward of policy π without accounting for any cost. By definition,

$$\mathrm{OPT}_{r,G} := \max_{\pi' \in \Pi} \left\{ \pi'^\top r^* \;\middle|\; \pi'^\top g_i^* \leq \alpha_i \ \ \forall i \in [m] \right\}$$

is the maximum achievable reward among all policies satisfying the constraints.

Since the policy π satisfies $\pi^\top g_i^* \leq \alpha_i$ for all $i \in [m]$, its expected reward cannot exceed the optimal value. □

Lemma 15. *Given a CSDM with reward vector $r \in [0,1]^{|X\times A|}$ and cost matrix $G \in [0,1]^{m\times|X\times A|}$, for every policy $\pi \in \Pi$ it holds:*

$$\pi^\top r^* - \max_{\lambda \in [0, {}^{H+1}/_{\rho}]^m} \sum_{i\in[m]} \lambda_i(\pi^\top g_i^* - \alpha_i) \leq OPT_{r,G}.$$

Proof. A policy π is either safe and satisfies all the constraints or unsafe and violates at least one of them.

1. If policy π is safe, then $arg\max_{\lambda\in[0,{}^{H+1}/_{\rho}]^m} \sum_{i\in[m]} \lambda_i(\pi^\top g_i^* - \alpha_i) = \underline{0}$. Employing lemma 14 proves the result.

2. If policy π is not safe,then it violates at least 1 costraint. Then, $arg\max_{\lambda\in[0,{}^{H+1}/_{\rho}]^m} \sum_{i\in[m]} \lambda_i(\pi^\top g_i^* - \alpha_i) = \overline{\lambda}$, where $\overline{\lambda}$ is a vector with element $\lambda_i = 0$ if the i-th constraint is satisfied, and $\lambda_i = {}^{H+1}/_{\rho}$ if the i-th constraint is violated. It

holds:

$$\begin{aligned}
\mathcal{L}_{r,G}(\pi', \overline{\lambda}) &= \pi'^{\top} r^* - \sum_{i\in[m]} \lambda_i (\pi'^{\top} g_i^* - \alpha_i) \\
&\leq \max_{\pi\in\Pi} \pi^{\top} r^* - \sum_{i\in[m]} \frac{H+1}{\rho} \left[\pi^{\top} g_i^* - \alpha_i\right]^+ \qquad (5.4) \\
&\leq \max_{\pi\in\Pi} \pi^{\top} r^* - \sum_{i\in[m]} \frac{H}{\rho} \left[\pi^{\top} g_i^* - \alpha_i\right]^+ \\
&\leq \max_{\pi\in\Pi} \min_{\|\lambda\|_1\in[0,H/\rho]} \pi^{\top} r^* - \sum_{i\in[m]} \lambda_i \left[\pi^{\top} g_i^* - \alpha_i\right]^+ \\
&\leq \min_{\|\lambda\|_1\in[0,H/\rho]} \max_{\pi\in\Pi} \pi^{\top} r^* - \sum_{i\in[m]} \lambda_i \left[\pi^{\top} g_i^* - \alpha_i\right]^+ \qquad (5.5) \\
&\leq \min_{\|\lambda\|_1\in[0,H/\rho]} \max_{\pi\in\Pi} \mathcal{L}_{r,G}(\pi, \lambda) \\
&= \mathrm{OPT}_{r,G}, \qquad (5.6)
\end{aligned}$$

where inequality 5.4 holds by definition of $\overline{\lambda}$, 5.5 holds by the *max − min inequality* [1] and 5.6 follows by Lemma 13.

□

Finally, it is shown that the sequence of Lagrangian vector λ_t selected by Algorithm 5.1 guarantees that the value of the Lagrangian by π_t is always upper bounded by the value of the Lagrangian attained by the best strategy π^* up to confidence bound terms.

Lemma 16. *Algorithm 5.1 guarantees that, for every episode $t \in T$, with probability at least $1 - \delta$:*

$$\mathcal{L}_{r,G}(\pi^*, \lambda_t) \geq \mathcal{L}_{r,G}(\pi_t, \lambda_t) - \frac{4Hm}{\rho} \pi_t^{\top} \widetilde{\xi}_t.$$

Proof. Notice that, when $\lambda_t = \underline{0}$, $\mathcal{L}_{r,G}(\pi^*, \lambda_t) = \mathrm{OPT}_{r,G}$ by definition. When λ_t is the $^{H+1}/_{\rho}$ vector, it holds that $\mathcal{L}_{r,G}(\pi^*, \lambda_t) \geq \mathrm{OPT}_{r,G}$, since π^* is safe. This is always true for any vector in $\{0, {}^{H+1}/_{\rho}\}^m$. Thus, $\mathcal{L}_{r,G}(\pi^*, \lambda_t) \geq \mathrm{OPT}_{r,G}$ since π^* is always true.

$$\begin{aligned}
\mathcal{L}_{r,G}(\pi_t, \lambda_t) &= \pi_t^\top r^* - \sum_{i \in m} \lambda_{t,i}(\pi_t^\top g_i^* - \alpha_i) \\
&\leq \pi_t^\top r^* - \sum_{i \in m} \lambda_{t,i}(\pi_t^\top \underline{\widetilde{g}}_{i,t} - \alpha_i) \qquad (5.7) \\
&\leq \pi_t^\top r^* - \sum_{i \in m} \lambda_{t,i}(\pi_t^\top (g_i^* - 2\widetilde{\xi}_t) - \alpha_i) \\
&\leq \pi_t^\top r^* - \sum_{i \in m} \lambda_{t,i}(\pi_t^\top g_i^* - \alpha_i) + \frac{4Lm}{\rho}\pi^\top \widetilde{\xi}_t \qquad (5.8) \\
&\leq \mathrm{OPT}_{r,G} + \frac{4Hm}{\rho}\pi^\top \widetilde{\xi}_t,
\end{aligned}$$

where inequality 5.7 holds with probability at least $1-\delta$ by lemma 11 and inequality 5.8 holds by definition of λ.
Thus it holds:

$$\mathcal{L}_{r,G}(\pi^*, \lambda_t) \geq \mathrm{OPT}_{r,G} \geq \mathcal{L}_{r,G}(\pi_t, \lambda_t) - \frac{4Hm}{\rho}\pi_t^\top \widetilde{\xi}_t.$$

□

Lemma 17. *Algorithm 5.1 guarantees that, for every episode $t \in T$, with probability at least $1-\delta$:*

$$\mathcal{L}_{r,G}(\pi^*, {}^{\lambda_t H}/_{H+1}) \geq \mathcal{L}_{r,G}(\pi_t, {}^{\lambda_t H}/_{H+1}) - \frac{2Hm}{\rho}\pi_t^\top \widetilde{\xi}_t.$$

Proof. Similarly to 16, notice that $\mathcal{L}_{r,G}(\pi^*, \lambda_t) \geq \mathrm{OPT}_{r,G}$.

$$\begin{aligned}
\mathcal{L}_{r,G}(\pi_t, {}^{\lambda_t H}/_{H+1}) &= \pi_t^\top r^* - \sum_{i \in m} \frac{\lambda_{t,i} H}{H+1}(\pi_t^\top g_i^* - \alpha_i) \\
&\leq \pi_t^\top r^* - \sum_{i \in m} \frac{\lambda_{t,i} H}{H+1}(\pi_t^\top \underline{\widetilde{g}}_{i,t} - \alpha_i) \qquad (5.9) \\
&\leq \pi_t^\top r^* - \sum_{i \in m} \frac{\lambda_{t,i} H}{H+1}(\pi_t^\top (g_i^* - 2\widetilde{\xi}_t) - \alpha_i) \\
&\leq \pi_t^\top r^* - \sum_{i \in m} \frac{\lambda_{t,i} H}{H+1}(\pi_t^\top g_i^* - \alpha_i) + 2\frac{H+1}{\rho} m \frac{H}{H+1}\pi^\top \widetilde{\xi}_t \qquad (5.10) \\
&\leq \mathrm{OPT}_{r,G} + \frac{2Hm}{\rho}\pi^\top \widetilde{\xi}_t,
\end{aligned}$$

where inequality 5.9 holds with probability at least $1-\delta$ by lemma 11 and inequality 5.10 holds by definition of λ.
Thus it holds:

$$\mathcal{L}_{r,G}(\pi^*, \lambda_t) \geq \mathrm{OPT}_{r,G} \geq \mathcal{L}_{r,G}(\pi_t, \lambda_t) - \frac{2Lm}{\rho}\pi_t^\top \widetilde{\xi}_t.$$

□

5.6.3. Regret and Violations

In this subsection, it is shown that the regret bound attained by CPD-FTRL on the Lagrangian loss built by Algorithm 5.1.

Lemma 18. *The instance of Balanced-FTRL employed by Algorithm 5.1 guarantees that, given any $\delta \in (0,1)$ and for any policy $\pi' \in \Pi$, the primal regret $\mathcal{R}_T^{\mathcal{L}}(\pi)$ defined as:*

$$\mathcal{R}_T^{\mathcal{L}}(\pi) := \sum_{t=1}^{T}\left[\pi'^{\top}\theta\overline{r}_t - \sum_{i\in m}\lambda_{i,t}(\pi'^{\top}\theta\underline{g}_{i,t} - \alpha_i)\right] - \sum_{t=1}^{T}\left[\pi_t^{\top}\theta\overline{r}_t - \sum_{i\in m}\lambda_{i,t}(\pi_t^{\top}\theta\underline{g}_{i,t} - \alpha_i)\right]$$

is bounded as follows with probability at least $1-\mathcal{O}(\delta)$

$$\mathcal{R}_T^{\mathcal{L}}(\pi) \leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right).$$

Proof. Even though the artificial loss does not take the environment transitions in consideration, Balanced-FTRL accounts for them. So, the estimated reward attained by Algorithm 5.1 can be written as: $\pi_t^{\top}\theta\overline{r}_t$. Similarly, the constraints incurred by the algorithm can be written as: $\pi_t^{\top}\theta\underline{g}_{i,t}$. Notice that:

$$\begin{aligned}\sum_{t=1}^{T}\frac{1}{\frac{2(H+1)m}{\rho}+1}\left\{\pi_t^{\top}\left(\frac{(H+1)m}{\rho}\right) - \sum_{t=1}^{T}\left[\pi_t^{\top}\theta\overline{r}_t - \sum_{i\in[m]}\lambda_{i,t}\pi_t^{\top}(\theta\underline{g}_{i,t} - \frac{\alpha_i}{H})\right]\right\}\\ -\sum_{t=1}^{T}\frac{1}{\frac{2(H+1)m}{\rho}+1}\left\{\pi^{\top}\left(\frac{(H+1)m}{\rho}\right) - \sum_{t=1}^{T}\left[\pi^{\top}\theta\overline{r}_t - \sum_{i\in[m]}\lambda_{i,t}\pi^{\top}(\theta\underline{g}_{i,t} - \frac{\alpha_i}{H})\right]\right\}\\ \leq \widetilde{\mathcal{O}}\left(\sqrt{H|A||X|T\log(1/(4\delta))}\right).\end{aligned}$$

This follows by the guarantees of Adaptive-FTRL, since the artificial rewards are scaled between $[0,1]$.
Then, multiplying both sides of the inequality it is possible to cancel out the normalization

term in the left hand side:

$$\sum_{t=1}^{T} \pi_t^\top \left(\frac{(H+1)m}{\rho}\right) - \sum_{t=1}^{T} \left[\pi_t^\top \theta \overline{r}_t - \sum_{i\in[m]} \lambda_{i,t} \pi_t^\top (\theta \underline{g}_{i,t} - \frac{\alpha_i}{H})\right]$$
$$- \sum_{t=1}^{T} \pi^\top \left(\frac{(H+1)m}{\rho}\right) - \sum_{t=1}^{T} \left[\pi^\top \theta \overline{r}_t - \sum_{i\in[m]} \lambda_{i,t} \pi^\top (\theta \underline{g}_{i,t} - \frac{\alpha_i}{H})\right]$$
$$\leq \left[\frac{2(H+1)m}{\rho} + 1\right] \widetilde{\mathcal{O}}\left(\sqrt{H|A||X|T\log(1/(4\delta))}\right)$$
$$= \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right).$$

Absorbing the constant terms into $\widetilde{\mathcal{O}}(\cdot)$ in the last step yields the stated bound. Then, notice that:

$$\sum_{t=1}^{T} \pi_t^\top \left(\frac{(H+1)m}{\rho}\right) - \sum_{t=1}^{T} \left[\pi_t^\top \theta \overline{r}_t - \sum_{i\in[m]} \lambda_{i,t} \pi_t^\top (\theta \underline{g}_{i,t} - \frac{\alpha_i}{H})\right]$$
$$- \sum_{t=1}^{T} \pi^\top \left(\frac{(H+1)m}{\rho}\right) + \sum_{t=1}^{T} \left[\pi^\top \theta \overline{r}_t - \sum_{i\in[m]} \lambda_{i,t} \pi^\top (\theta \underline{g}_{i,t} - \frac{\alpha_i}{H})\right]$$
$$= -\sum_{t=1}^{T} \left[\pi_t^\top \theta \overline{r}_t - \sum_{i\in[m]} \lambda_{i,t} \pi_t^\top (\theta \underline{g}_{i,t} - \frac{\alpha_i}{H})\right] + \sum_{t=1}^{T} \left[\pi^\top \theta \overline{r}_t - \sum_{i\in[m]} \lambda_{i,t} \pi^\top (\theta \underline{g}_{i,t} - \frac{\alpha_i}{H})\right] \tag{5.11}$$
$$= \sum_{t=1}^{T} \left[\pi^\top \theta \overline{r}_t - \sum_{i\in[m]} \lambda_{i,t} \pi^\top \theta \underline{g}_{i,t} - \alpha_i\right] - \sum_{t=1}^{T} \left[\pi_t^\top \theta \overline{r}_t - \sum_{i\in[m]} \lambda_{i,t} \pi_t^\top \theta \underline{g}_{i,t} - \alpha_i\right] \tag{5.12}$$
$$\leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right),$$

where equality 5.11 holds since the term $(H+1)m/\rho$ is constant for any state-action pair, and equality 5.12 holds since $\pi^\top \alpha_i / H = \alpha_i$ for any policy $\pi \in \Pi$ □

Then, the study of the concentration rate of the confidence intervals is shown.

Lemma 19. *With probability at least $1-\delta$ it holds:*

$$\sum_{t=1}^{T} \pi_t^\top \phi_t \leq 4\sqrt{H|X||A|T\ln\left(\frac{T|X||A|}{\delta}\right)} + H\sqrt{2T\ln\frac{1}{\delta}}.$$

Proof. First, notice that, for any policy $\pi \in \Pi$:

$$\pi^\top \phi_t \leq H.$$

Thus, it is possible to employ the Azuma inequality to bound the following Martingale difference sequence as:

$$\sum_{t=1}^{T} \pi_t^\top \phi_t - \sum_{t=1}^{T} \sum_{(x,a)} \phi_t(x,a) \mathbb{1}\{x,a\} \leq H\sqrt{2T \ln \frac{1}{\delta}},$$

which holds with probability at least $1-\delta$. Thus:

$$\begin{aligned}
\sum_{t=1}^{T} \pi_t^\top \phi_t &\leq \sum_{t=1}^{T} \sum_{(x,a)} \phi_t(x,a) \mathbb{1}\{x,a\} + H\sqrt{2T \ln \frac{1}{\delta}} \\
&= \sqrt{4 \ln \left(\frac{T|X||A|}{\delta}\right)} \sum_{t=1}^{T} \sum_{x,a} \sqrt{\frac{1}{\max\{1, N_t(x,a)\}}} \mathbb{1}_t\{x,a\} + H\sqrt{2T \ln \frac{1}{\delta}} \\
&\leq 2\sqrt{4 \ln \left(\frac{T|X||A|}{\delta}\right)} \sum_{x,a} \sqrt{N_T(x,a)} + H\sqrt{2T \ln \frac{1}{\delta}} \qquad (5.13) \\
&\leq 4\sqrt{H|X||A|T \ln \left(\frac{T|X||A|}{\delta}\right)} + H\sqrt{2T \ln \frac{1}{\delta}}, \qquad (5.14)
\end{aligned}$$

where Inequality 5.13 holds since $\Sigma_{t=1}^{T} \frac{1}{t} \leq 2\sqrt{T}$ and Inequality 5.14 holds from Cauchy-Schwarz inequality and from the fact that $\sqrt{\Sigma_{x,a} N_T(x,a)} \leq \sqrt{HT}$. □

Lemma 20. *With probability at least $1-\delta$ it holds:*

$$\sum_{t=1}^{T} \pi_t^\top \xi_t \leq 4\sqrt{H|X||A|T \ln \left(\frac{T|X||A|}{\delta}\right)} + H\sqrt{2T \ln \frac{1}{\delta}}.$$

Proof. The proof is similar to that of Lemma 19 replacing ϕ_t with ξ_t. □

Lemma 21. *With probability at least $1-\delta$ it holds:*

$$\sum_{t=1}^{T} \pi_t^\top \widetilde{\xi}_t \leq 4\sqrt{H|X||A|T \ln \left(\frac{T|X||A|}{\delta}\right)} + H\sqrt{2T \ln \frac{1}{\delta}}.$$

Proof. The proof is similar to that of Lemma 19 replacing ϕ_t with $\widetilde{\xi}_t$. □

Theorem 5.3. *Given any $\delta \in (0,1)$, Algorithm 5.1 attains:*

$$R_t \leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right)$$

with probability at least $1-\mathcal{O}(\delta)$

Proof. Employing lemma 16, it holds:

$$\mathcal{L}_{r,G}(\pi^*,\lambda_t) \geq \mathcal{L}_{r,G}(\pi_t,\lambda_t) - \frac{4Hm}{\rho}\pi_t^\top \widetilde{\xi}_t, \quad \forall t \in [T]$$

with probability at least $1-\delta$, where π^* is the optimal solution that corresponds to $\mathrm{OPT}_{r,G}$. Furthermore, an upper bound on

$$\sum_{t=1}^{T}\left(\mathcal{L}_{r,G}(\pi^*,\lambda_t) - \mathcal{L}_{r,G}(\pi_t,\lambda_t) + \frac{4Hm}{\rho}\pi_t^\top \widetilde{\xi}_t\right)$$

enforces the same bound on

$$\sum_{t=1}^{T}\left[\mathcal{L}_{r,G}(\pi^*,\lambda_t) - \mathcal{L}_{r,G}(\pi_t,\lambda_t) + \frac{4Hm}{\rho}\pi_t^\top \widetilde{\xi}_t\right]^+,$$

since the term in the summation is always non-negative.
Employing Lemma 18, it holds that, with probability at least $1-\mathcal{O}(\delta)$ and for any $\pi \in \Pi$:

$$\sum_{t=1}^{T}\left[\pi^{*\top}\theta\overline{r}_t - \sum_{i\in m}\lambda_{i,t}(\pi'^\top\theta\underline{g}_{i,t} - \alpha_i)\right] - \sum_{t=1}^{T}\left[\pi_t^\top\theta\overline{r}_t - \sum_{i\in m}\lambda_{i,t}(\pi_t^\top\theta\underline{g}_{i,t} - \alpha_i)\right] \leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right), \tag{5.15}$$

which is also valid for π^*. By Lemma 10, with probability at least $1-\delta$ it holds:

$$r^* = \theta r \preceq \theta\overline{r}_t.$$

Thus, for any $\pi \in \Pi$, $t \in T$, it is true that:

$$\pi^\top r^* \leq \pi^\top\theta\overline{r}_t.$$

So, it is possible to write Equation 5.15 as:

$$\sum_{t=1}^{T}\left[\pi^{*\top}r^{*}-\sum_{i\in m}\lambda_{i,t}(\pi'^{\top}\theta\underline{g}_{i,t}-\alpha_i)\right]-\sum_{t=1}^{T}\left[\pi_t^{\top}\theta\overline{r}_t-\sum_{i\in m}\lambda_{i,t}(\pi_t^{\top}\theta\underline{g}_{i,t}-\alpha_i)\right]\leq\widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2m}{\rho}|A||X|T\log(1/(4\delta))}\right).$$

Similarly, it also holds that $\theta\overline{r}_t\preceq\theta(r+2\phi_t)$. So, one can write:

$$\sum_{t=1}^{T}\left[\pi^{*\top}r^{*}-\sum_{i\in m}\lambda_{i,t}(\pi'^{\top}\theta\underline{g}_{i,t}-\alpha_i)\right]-\sum_{t=1}^{T}\left[\pi_t^{\top}r^{*}-\sum_{i\in m}\lambda_{i,t}(\pi_t^{\top}\theta\underline{g}_{i,t}-\alpha_i)\right]\leq\widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2m}{\rho}|A||X|T\log(1/(4\delta))}\right)+2\sum_{t=1}^{T}\pi_t^{\top}\theta\phi_t.$$

Applying Lemma 19 Union Bound yields:

$$\sum_{t=1}^{T}\left[\pi^{*\top}r^{*}-\sum_{i\in m}\lambda_{i,t}(\pi'^{\top}\theta\underline{g}_{i,t}-\alpha_i)\right]-\sum_{t=1}^{T}\left[\pi_t^{\top}r^{*}-\sum_{i\in m}\lambda_{i,t}(\pi_t^{\top}\theta\underline{g}_{i,t}-\alpha_i)\right]\leq\widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2m}{\rho}|A||X|T\log(1/(4\delta))}\right).$$

Similarly, note that $\theta\underline{g}_{i,t}\preceq g_i^{*}$ and $\theta(g_i-2\xi_t)\preceq\theta\underline{g}_{i,t}$. Thus:

$$\sum_{t=1}^{T}\left[\pi^{*\top}r^{*}-\sum_{i\in m}\lambda_{i,t}(\pi'^{\top}g_i^{*}-\alpha_i)\right]-\sum_{t=1}^{T}\left[\pi_t^{\top}r^{*}-\sum_{i\in m}\lambda_{i,t}(\pi_t^{\top}g_i^{*}-\alpha_i)\right]\leq\widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2m}{\rho}|A||X|T\log(1/(4\delta))}\right)+2m\sum_{t=1}^{T}\pi_t^{\top}\theta\xi_t.$$

Employing lemma 20 and Union Bound yields:

$$\sum_{t=1}^{T}\left[\pi^{*\top}r^{*}-\sum_{i\in m}\lambda_{i,t}(\pi'^{\top}g_i^{*}-\alpha_i)\right]-\sum_{t=1}^{T}\left[\pi_t^{\top}r^{*}-\sum_{i\in m}\lambda_{i,t}(\pi_t^{\top}g_i^{*}-\alpha_i)\right]\leq\widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2m}{\rho}|A||X|T\log(1/(4\delta))}\right).$$

Then, notice that, with probability at least $1-\delta$, for any $t \in T$:

$$\sum_{i\in[m]} \lambda_{i,t}(\pi_t^\top g_i^* - \alpha_i) \geq \sum_{i\in[m]} \lambda_{i,t}(\pi_t^\top \underline{\tilde{g}}_{i,t} - \alpha_i).$$

Since $\lambda_{i,t}(\pi_t^\top \underline{\tilde{g}} - \alpha_i)$ is the quantity observed by the algorithm and since λ is chosen according to:

$$\lambda_{t,i} \leftarrow \arg\max_{\lambda\in\{0,\frac{H+1}{\rho}\}} \lambda(\pi_t^\top \underline{\widetilde{g}}_{i,t} - \alpha_i) \qquad \forall i \in [m],$$

then this quantity is always non negative.

$$\sum_{i\in[m]} \lambda_{i,t}(\pi_t^\top \underline{\tilde{g}}_{i,t} - \alpha_i) \geq 0.$$

This implies that:

$$\sum_{i\in[m]} \lambda_{i,t}(\pi_t^\top g_i^* - \alpha_i) \geq 0.$$

It is now possible to bound the positive cumulative regret as follows:

$$\begin{aligned}
R_t &:= \sum_{t=1}^{T} \left[\pi^{*\top} r^* - \pi_t^\top r^*\right]^+ \\
&\leq \sum_{t=1}^{T} \left[\left(\pi^{*\top} r^* - \sum_{i\in[m]} \lambda_{i,t}(\pi^{*\top} g_i^* - \alpha_i)\right) - \left(\pi_t^\top r^* - \sum_{i\in[m]} \lambda_{i,t}(\pi_t^* g_i^* - \alpha_i)\right)\right]^+ \\
&\leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right).
\end{aligned}$$

□

Theorem 5.4. *Given any $\delta \in (0,1)$, Algorithm 5.1 attains:*

$$V_t \leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right)$$

with probability at least $1-\mathcal{O}(\delta)$.

Proof. First, notice that:

$$\begin{aligned}
&\sum_{i\in[m]} \lambda_{i,t}\left(\pi_t^\top \theta \widetilde{\underline{g}}_{i,t} - \alpha_i\right) \\
&= \frac{H}{H+1}\sum_{i\in[m]} \lambda_{i,t}\left(\pi_t^\top \theta \widetilde{\underline{g}}_{i,t} - \alpha_i\right) + \frac{1}{H+1}\sum_{i\in[m]} \lambda_{i,t}\left(\pi_t^\top \theta \widetilde{\underline{g}}_{i,t} - \alpha_i\right) \\
&= \frac{H}{H+1}\sum_{i\in[m]} \lambda_{i,t}\left(\pi_t^\top \theta \widetilde{\underline{g}}_{i,t} - \alpha_i\right) + \frac{1}{\rho}\sum_{i\in[m]} \left[\pi_t^\top \theta \widetilde{\underline{g}}_{i,t} - \alpha_i\right]^+, \qquad (5.16)
\end{aligned}$$

where equation 5.16 holds by definition of λ_t.

Following Lemma 18, with probability at least $1 - \mathcal{O}(\delta)$ it holds:

$$\begin{aligned}
\sum_{t=1}^{T}\left[\pi'^\top \theta \overline{r}_t - \sum_{i\in m}\lambda_{i,t}(\pi'^\top \theta \underline{g}_{i,t} - \alpha_i)\right] - \sum_{t=1}^{T}\left[\pi_t^\top \theta \overline{r}_t - \sum_{i\in m}\lambda_{i,t}(\pi_t^\top \theta \underline{g}_{i,t} - \alpha_i)\right] \\
\leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right).
\end{aligned}$$

It is possible to substitute the Lagrangian value of our algorithm with Equation 5.16:

$$\begin{aligned}
\sum_{t=1}^{T}\left[\pi'^\top \theta \overline{r}_t - \sum_{i\in m}\lambda_{i,t}(\pi'^\top \theta \underline{g}_{i,t} - \alpha_i)\right] - \left[\frac{H}{H+1}\sum_{i\in[m]} \lambda_{i,t}\left(\pi_t^\top \widetilde{\underline{g}}_{i,t} - \alpha_i\right) + \frac{1}{\rho}\sum_{i\in[m]} \left[\pi_t^\top \widetilde{\underline{g}}_{i,t} - \alpha_i\right]^+\right] \\
\leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right).
\end{aligned}$$

Notice that:

$$\begin{aligned}
&\frac{H}{H+1}\sum_{i\in[m]} \lambda_{i,t}(\pi_t^\top \widetilde{\underline{g}}_{i,t} - \alpha_i) \\
&\geq \frac{H}{H+1}\sum_{i\in[m]} \lambda_{i,t}(\pi_t^\top (g_i^* - 2\widetilde{\xi}_t) - \alpha_i) \\
&\geq \frac{H}{H+1}\sum_{i\in[m]} \lambda_{i,t}(\pi_t^\top g_i^* - \alpha_i) - \frac{2Hm}{\rho}\pi_t^\top \widetilde{\xi}_t.
\end{aligned}$$

Employing Lemma 21:

$$\sum_{t=1}^{T} \mathcal{L}_{r,G}(\pi^*, \lambda_t) - \sum_{t=1}^{T} \mathcal{L}_{r,G}(\pi_t, {}^{\lambda_t H}/_{H+1}) + \frac{1}{\rho} \sum_{t=1}^{T} \sum_{i \in [m]} \left[\pi_t^\top \widetilde{\underline{g}}_{i,t} - \alpha_i \right]^+ \leq \widetilde{\mathcal{O}} \left(\sqrt{\frac{H^2 m}{\rho} |A||X| T \log(1/(4\delta))} \right).$$

Similarly, observe that:

$$\begin{aligned} & \frac{1}{\rho} \sum_{i \in [m]} \left[\pi_t^\top \widetilde{\underline{g}}_{i,t} - \alpha_i \right]^+ \\ \geq & \frac{1}{\rho} \sum_{i \in [m]} \left[\pi_t^\top (g_i^* - 2\widetilde{\xi}_t) - \alpha_i \right]^+ \\ \geq & \frac{1}{\rho} \sum_{i \in [m]} \left[\pi_t^\top g_i^* - \alpha_i \right]^+ - \frac{2m}{\rho} \pi_t^\top \widetilde{\xi}_t. \end{aligned}$$

Thus, by Lemma 21:

$$\sum_{t=1}^{T} \mathcal{L}_{r,G}(\pi^*, \lambda_t) - \sum_{t=1}^{T} \mathcal{L}_{r,G}(\pi_t, {}^{\lambda_t H}/_{H+1}) + \frac{1}{\rho} \sum_{t=1}^{T} \sum_{i \in [m]} \left[\pi_t^\top g_i^* - \alpha_i \right]^+ \leq \widetilde{\mathcal{O}} \left(\sqrt{\frac{H^2 m}{\rho} |A||X| T \log(1/(4\delta))} \right),$$

which can be written as:

$$\frac{1}{\rho} \sum_{t=1}^{T} \sum_{i \in [m]} \left[\pi_t^\top g_i^* - \alpha_i \right]^+ \leq \widetilde{\mathcal{O}} \left(\sqrt{\frac{H^2 m}{\rho} |A||X| T \log(1/(4\delta))} \right) - \sum_{t=1}^{T} \mathcal{L}_{r,G}(\pi^*, \lambda_t) + \sum_{t=1}^{T} \mathcal{L}_{r,G}(\pi_t, {}^{\lambda_t H}/_{H+1}).$$

Employing Lemma 17, it holds that:

$$\mathcal{L}_{r,G}(\pi^*, {}^{\lambda_t H}/_{H+1}) - \mathcal{L}_{r,G}(\pi_t, {}^{\lambda_t H}/_{H+1}) \geq -\frac{2Hm}{\rho} \pi_t^\top \widetilde{\xi}_t.$$

Thus:

$$\begin{aligned}
&\frac{1}{\rho}\sum_{t=1}^{T}\sum_{i\in[m]}\left[\pi_t^\top g_i^* - \alpha_i\right]^+ \\
&\leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right) - \sum_{t=1}^{T}\mathcal{L}_{r,G}(\pi^*,\lambda_t) + \sum_{t=1}^{T}\mathcal{L}_{r,G}(\pi_t, {}^{\lambda_t H}/_{H+1}) \\
&\leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right) + \frac{2Hm}{\rho}\pi_t^\top\widetilde{\xi}_t \\
&\leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right), \qquad (5.17)
\end{aligned}$$

where Inequality 5.17 holds by Lemma 20. It is now possible to obtain the final bound:

$$\begin{aligned}
V_T &:= \max_{i\in m}\sum_{t=1}^{T}\left[\pi_t^\top g_i^* - \alpha_i\right]^+ \\
&\leq \sum_{t=1}^{T}\sum_{i\in[m]}\left[\pi_t^\top g_i^* - \alpha_i\right]^+ \\
&= \rho\cdot\frac{1}{\rho}\sum_{t=1}^{T}\sum_{i\in[m]}\left[\pi_t^\top g_i^* - \alpha_i\right]^+ \\
&\leq \widetilde{\mathcal{O}}\left(\sqrt{\frac{H^2 m}{\rho}|A||X|T\log(1/(4\delta))}\right)
\end{aligned}$$

with probability at least $1-\mathcal{O}(\delta)$ by Union Bound. □

6 | Conclusions and future developments

6.1. Conclusions

This Thesis investigates Online Learning in constrained environments through both empirical analysis and algorithmic development.
First, an experimental comparison of two established online learning algorithms under constrained settings is conducted, highlighting how guarantees translate into practice and confirming the theoretical results expected.
Building upon this, a novel algorithm for Constrained Tree-Form Sequential Decision-Making problem is introduced. The proposed method is based on state-of-the-art algorithm for learning in adversarial and uncostrained environments, and extends it with a primal-dual approach. The resulting algorithm achieved optimal regret and violation guarantees while improving computational efficiency with respect to existing methods without requiring prior knowledge or assumptions on the environment structure.

6.2. Future Developments

Several directions for future research can be taken starting from this work.
A first consideration concerns the empirical validation of the proposed algorithm. While the theoretical guarantees provide strong support for its performance, an experimental implementation would allow for a comprehensive understanding of its practical behavior. In particular, benchmarking the algorithm in structured environments, such as strategic card games or other sequential decision-making scenarios with partial feedback, would offer valuable information about its efficiency, robustness, and convergence properties.
A second promising direction involves the exploration of real-world applications. The proposed primal–dual framework is suitable to settings in which performance optimization must be balanced against operational constraints. Potential application domains include robotics, where safety and resource limitations are critical, safe navigation, online resource

allocation and management problems such as planning tasks under uncertainty.

Ringraziamenti

Un grazie di cuore a tutte le persone che mi sono state accanto durante questo percorso. In particolare alla mia famiglia, che da sempre mi sostiene e mi ama incondizionatamente: a mia madre, a mio padre, a mia sorella e a mio fratello. Infine, grazie ai miei cari amici, con cui condivido momenti di gioia e divertimento, in particolare a Valerio, Niccolò, Daniele, alla mia amica Nora e ai miei amici Fabrizio, Alex, Stefano e Andrea.

Bibliography

[1] E. Altman. *Constrained Markov decision processes*. Routledge, 2021.

[2] S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of computing*, 8(1):121–164, 2012.

[3] P. Auer, T. Jaksch, and R. Ortner. Near-optimal regret bounds for reinforcement learning. *Advances in neural information processing systems*, 21, 2008.

[4] F. Bacchiocchi, F. E. Stradi, M. Papini, A. M. Metelli, and N. Gatti. Online learning with off-policy feedback in adversarial mdps. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 3697–3705, 2024.

[5] F. Bacchiocchi, F. E. Stradi, M. Castiglioni, A. Marchesi, and N. Gatti. Markov persuasion processes: Learning to persuade from scratch. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=NWQ8KeoWje`.

[6] Y. Bai, C. Jin, S. Mei, and T. Yu. Near-optimal learning of extensive-form games with imperfect information. In *International Conference on Machine Learning*, pages 1337–1382. PMLR, 2022.

[7] M. Bernasconi, F. Cacciamani, M. Castiglioni, A. Marchesi, N. Gatti, and F. Trovò. Safe learning in tree-form sequential decision making: Handling hard and soft constraints. In *International Conference on Machine Learning*, pages 1854–1873. PMLR, 2022.

[8] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[9] N. Brown and T. Sandholm. Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018.

[10] N. Brown and T. Sandholm. Superhuman ai for multiplayer poker. *Science*, 365 (6456):885–890, 2019.

[11] N. Cesa-Bianchi and G. Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

[12] H. S. Chang, M. C. Fu, J. Hu, and S. I. Marcus. Google deepmind's alphago: operations research's unheralded role in the path-breaking achievement. *Or/Ms Today*, 43(5):24–30, 2016.

[13] S. De Rooij, T. Van Erven, P. D. Grünwald, and W. M. Koolen. Follow the leader if you can, hedge if you must. *The Journal of Machine Learning Research*, 15(1): 1281–1316, 2014.

[14] L. Doyen and J.-F. Raskin. Games with imperfect information: theory and algorithms. *Lectures in Game Theory for Computer Scientists*, 10, 2011.

[15] Y. Efroni, S. Mannor, and M. Pirotta. Exploration-exploitation in constrained mdps. *arXiv preprint arXiv:2003.02189*, 2020.

[16] E. Even-Dar, S. M. Kakade, and Y. Mansour. Online markov decision processes. *Mathematics of Operations Research*, 34(3):726–736, 2009.

[17] F. Fang, S. Liu, A. Basak, Q. Zhu, C. D. Kiekintveld, and C. A. Kamhoua. Introduction to game theory. *Game theory and machine learning for cyber security*, pages 21–46, 2021.

[18] G. Farina, R. Schmucker, and T. Sandholm. Bandit linear optimization for sequential decision making and extensive-form games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 5372–5380, 2021.

[19] C. Fiegel, P. Ménard, T. Kozuno, R. Munos, V. Perchet, and M. Valko. Adapting to game trees in zero-sum imperfect information games. In *International Conference on Machine Learning*, pages 10093–10135. PMLR, 2023.

[20] G. Genalti, F. E. Stradi, M. Castiglioni, A. Marchesi, and N. Gatti. Data-dependent regret bounds for constrained MABs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=vWeyFCtYxx`.

[21] P. Hammerstein and R. Selten. Game theory and evolutionary biology. *Handbook of game theory with economic applications*, 2:929–993, 1994.

[22] E. Hazan et al. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

[23] S. C. Hoi, D. Sahoo, J. Lu, and P. Zhao. Online learning: A comprehensive survey. *Neurocomputing*, 459:249–289, 2021.

[24] C. Jin, T. Jin, H. Luo, S. Sra, and T. Yu. Learning adversarial markov decision processes with bandit feedback and unknown transition. In *International Conference on Machine Learning*, pages 4860–4869. PMLR, 2020.

[25] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE transactions on intelligent transportation systems*, 23(6):4909–4926, 2021.

[26] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[27] D. Koller, N. Megiddo, and B. Von Stengel. Fast algorithms for finding randomized strategies in game trees. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 750–759, 1994.

[28] C. Kroer, K. Waugh, F. Kilinç-Karzan, and T. Sandholm. Faster first-order methods for extensive-form game solving. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, pages 817–834, 2015.

[29] H. Luo, C.-Y. Wei, and C.-W. Lee. Policy optimization in adversarial mdps: Improved exploration via dilated bonuses. *Advances in Neural Information Processing Systems*, 34:22931–22942, 2021.

[30] D. Maran, F. Bacchiocchi, F. E. Stradi, M. Castiglioni, N. Gatti, and M. Restelli. Bandits with ranking feedback. *Advances in Neural Information Processing Systems*, 37:80567–80608, 2024.

[31] D. Maran, P. Olivieri, F. E. Stradi, G. Urso, N. Gatti, and M. Restelli. Online markov decision processes configuration with continuous decision space. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 14315–14322, 2024.

[32] A. Müller, P. Alatur, V. Cevher, G. Ramponi, and N. He. Truly no-regret learning in constrained mdps. *arXiv preprint arXiv:2402.15776*, 2024.

[33] J. F. Nash Jr. Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, 36(1):48–49, 1950.

[34] G. Neu, A. Antos, A. György, and C. Szepesvári. Online markov decision processes under bandit feedback. *Advances in Neural Information Processing Systems*, 23, 2010.

[35] F. Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

[36] P. C. Ordeshook. Game theory and political theory. *Cambridge: Cambridge*, 1986.

[37] S. Paternain, L. Chamon, M. Calvo-Fullana, and A. Ribeiro. Constrained reinforcement learning has zero duality gap. *Advances in Neural Information Processing Systems*, 32, 2019.

[38] S. Paternain, M. Calvo-Fullana, L. F. Chamon, and A. Ribeiro. Safe policies for reinforcement learning via primal-dual methods. *IEEE Transactions on Automatic Control*, 68(3):1321–1336, 2022.

[39] W. Poundstone. *Prisoner's dilemma*. Anchor, 2011.

[40] M. L. Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

[41] I. Romanovskii. Reduction of a game with complete memory to a matrix game. *Soviet Mathematics*, 3:678–681, 1962.

[42] A. Rosenberg and Y. Mansour. Online convex optimization in adversarial markov decision processes. In *International Conference on Machine Learning*, pages 5478–5486. PMLR, 2019.

[43] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[44] C. E. Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[45] Y. Shoham and K. Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2008.

[46] 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.

[47] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[48] A. Slivkins et al. Introduction to multi-armed bandits. *Foundations and Trends® in Machine Learning*, 12(1-2):1–286, 2019.

[49] F. E. Stradi, F. Cipriani, L. Ciampiconi, M. Leonardi, A. Rozza, and N. Gatti. A primal-dual online learning approach for dynamic pricing of sequentially displayed complementary items under sale constraints. *arXiv preprint arXiv:2407.05793*, 2024.

[50] F. E. Stradi, J. Germano, G. Genalti, M. Castiglioni, A. Marchesi, and N. Gatti. Online learning in CMDPs: Handling stochastic and adversarial constraints. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 46692–46721. PMLR, 2024.

[51] F. E. Stradi, M. Castiglioni, A. Marchesi, and N. Gatti. Optimal strong regret and violation in constrained mdps via policy optimization. In *The Thirteenth International Conference on Learning Representations*, 2025.

[52] F. E. Stradi, M. Castiglioni, A. Marchesi, and N. Gatti. Learning adversarial mdps with stochastic hard constraints. In *Forty-second International Conference on Machine Learning*, 2025.

[53] F. E. Stradi, M. Castiglioni, A. Marchesi, N. Gatti, and C. Kroer. No-regret learning under adversarial resource constraints: A spending plan is all you need! In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=TpdrvezvEA`.

[54] F. E. Stradi, A. Lunghi, M. Castiglioni, A. Marchesi, and N. Gatti. Policy optimization for cmdps with bandit feedback: Learning stochastic and adversarial constraints. In *Forty-second International Conference on Machine Learning*, 2025.

[55] F. E. Stradi, A. Lunghi, M. Castiglioni, A. Marchesi, and N. Gatti. Taming adversarial constraints in CMDPs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=sT9nd1WQ76`.

[56] R. S. Sutton, A. G. Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[57] C. Tsallis. Possible generalization of boltzmann-gibbs statistics. *Journal of statistical physics*, 52(1):479–487, 1988.

[58] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

List of Figures