POLITECNICO DI MILANO
Corso di Laurea Magistrale in Computer Science and Engineering
Scuola di Ingegneria Industriale e dell'Informazione

Regret Minimization Algorithms for the Follower's Behavior Identification in Leadership Games

Relatore: Prof. Marcello Restelli
Correlatore: Ing. Francesco Trovò
Correlatore: Ing. Giuseppe De Nittis

Tesi di Laurea di:
Lorenzo Bisi
Matricola 836636

Anno Accademico 2016-2017

Vorrei essere libero, libero come un uomo.
Come l'uomo più evoluto
che si innalza con la propria intelligenza
e che sfida la natura
con la forza incontrastata della scienza
con addosso l'entusiasmo
di spaziare senza limiti nel cosmo
è convinto che la forza del pensiero
sia la sola libertà.

La libertà non è star sopra un albero
non è neanche un gesto o un'invenzione
la libertà non è uno spazio libero
libertà è partecipazione.

Gaber

Ringraziamenti

Non ci sono tanti modi di fare i ringraziamenti. Si può non ringraziare nessuno: in fondo questa laurea è mia, chi c'è da ringraziare se non me stesso?

Oppure si può ringraziare un gruppo di persone. In questo caso sarebbero da ringraziare coloro che mi hanno seguito in questo lavoro di tesi: Marcello, Francesco, Giuseppe e Nicola. Senza il loro supporto, la loro esperienza e la loro passione, non sarei riuscito a scrivere una pagina di questa tesi, figuriamoci arrivare ad una pubblicazione! In particolare Francesco: senza di lui, starei ancora vagando per i corridoi del Politecnico, come un agente in passeggiata aleatoria su un MDP sconosciuto. Tuttavia se si sceglie questa opzione, bisogna poi decidere chi escludere. Posso non ringraziare Luca e Alessandro? Certo, ogni tanto andavano foraggiati a marmellata e parmigiano, ma chi avrebbe rallegrato il mio soggiorno milanese altrimenti? E non si possono escludere dai ringraziamenti nemmeno Eric, Tommaso, Marco, Pietro, Andrea, Francesco, Germano e Riccardo, meglio conosciuti come "I Fauni". Come avrei potuto affrontare da solo gorilla, estintori, vuman e tutte le altre creature fantastiche che abbiamo incontrato in questa avventura politecnica?

In alternativa resta la terza opzione, cioè ringraziare tutti. Genitori, famiglia, amici, conoscenti e chi più ne ha più ne metta. I genitori, perchè sono i miei più grandi sostenitori. Mia sorella, perchè mi sopporta da quando è nata. I miei nonni, per la loro saggezza (non sempre eh!) e la mia famiglia per l'affetto incondizionato. E infine gli amici. E qui viene la parte difficile. Perchè non puoi mica scordarti qualcuno, altrimenti sono guai. Bisogna ringraziarli tutti, uno ad uno. Perchè, non prendiamoci in giro, se non ci fossero gli amici a ridere alle mie battute, chi riderebbe?

Ci sono quindi 3 modi di fare i ringraziamenti. Solo 3. Pensavo di più.

Sommario

In questo lavoro studiamo per la prima volta un *Leadership Game*, nel quale un agente, agendo da Leader, affronta un altro agente, che agisce da Follower, il cui comportamento non è conosciuto a priori dal leader, ma fa parte di un insieme di possibili profili comportamentali. La motivazione principale è che nelle applicazioni reali l'assunzione comunemente fatta in Teoria dei Giochi, ovvero che l'avversario sia completamente razionale, si verifica raramente ed ogni assunzione specifica, se sbagliata, può portare a una perdita significativa per il leader. La domanda che ci poniamo è *se* e *come* il leader possa apprendere il profilo comportamentale di un follower nei Leadership Games. Questo è per sua natura un problema di *online identification*: infatti il leader cerca di identificare il profilo comportamentale del follower per sfruttare al meglio la potenziale non-razionalità dell'avversario, minimizzando al contempo il regret dovuto all'iniziale mancanza di informazione. Proponiamo due algoritmi basati su due approcci differenti e forniamo una analisi del regret. Inoltre, valutiamo sperimentalmente lo *pseudo-regret* degli algoritmi in Leadership Game concreti, ispirati da contesti di sicurezza, mostrando che i nostri algoritmi superano drasticamente gli algoritmi disponibili nello stato dell'arte.

Abstract

We study for the first time, to the best of our knowledge, a *Leadership Game* in which one agent, acting as *leader*, faces another agent, acting as *follower*, whose behaviour is not known *a priori* by the leader, being one among a set of possible behavioural profiles. The main motivation is that in real-world applications the common game-theoretical assumption of perfect rationality is rarely met, and any specific assumption on bounded rationality models, if wrong, could lead to a significant loss for the leader. The question we pose is *whether* and *how* the leader can learn the behavioural profile of a follower in Leadership Games. This is a "natural" *online identification* problem: in fact, the leader aims at identifying the follower's behavioural profile to exploit at best the potential non-rationality of the opponent, while minimizing the regret due to the initial lack of information. We propose two algorithms based on different approaches and we provide a regret analysis. Furthermore, we experimentally evaluate the pseudo-regret of the algorithms in concrete Leadership Games inspired by security domains, showing that our algorithms dramatically outperform the online learning algorithms available in the state of the art.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context

The study of scenarios in which multiple strategic agents interact is a challenging problem that is central in Artificial Intelligence from many years. The modelling of these scenarios can be elegantly achieved by means of *non-cooperative game theory* tools [8], while the task of solving a game is in many cases an open problem, in which the most suitable techniques to adopt strictly depend on information available to the agents. Two extreme situations can be distinguished: when all the information about the game is common to the players (e.g., utility functions and rationality—either perfect or bounded), the problem is basically an *optimization problem*, solvable by means of techniques from *operations research* [25], conversely, when players have no information about the opponents, the problem is a *multi-learning problem*, and *learning* techniques are commonly employed [27]. Some attempts were also done to pair these two approaches, allowing agents to play at the equilibrium if the opponent is rational and to play off the equilibrium learning to exploit her at best otherwise [6].

Recently, there is an increasing interest in *leadership* games, where an agent—called *leader*—publicly commits to a strategy and subsequently another agent—called *follower*—observes the commitment and then takes her decision. Such paradigm has been successfully employed in a number of applications in the security domain [20, 26, 2], where a *defender* (acting as leader) must protect some targets in an environment from an *attacker* (acting as follower), who aims at compromising such targets without being

detected. The success of leadership games in real-world applications is due to a number of reasons: committing to a strategy is the best the leader can do, the equilibrium finding problem is conceptually simple since the follower can merely play her best response to the commitment of the leader without any strategic reasoning about the leader's behaviour, and the solution is unique except degeneracy. The crucial issue is that in real-world applications the follower may be not perfectly rational, not necessarily playing her best response to the leader's commitment. For instance, a terrorist could decide either to attack a target that is not patrolled, since she is sure to not be caught, or a target not so valuable itself, but that would cause a huge panic reaction in the population (e.g., this is what happened in November 2015 in Paris attacks at the Bataclan theatre). The same challenge may be faced by a company that aims at planning the production of a product and has to decide when and how it is convenient to enter the market when another company is already the leader in such market—this is the well-known *Stackelberg duopoly* [29]. Whenever the assumption of perfect rationality is not met, each agent may in principle exploit her opponent's strategy.

1.2 Problem

In our work we imagine a leader-follower scenario in which we are uncertain about the rationality of the *attacker*, but we are able to make some hyphothesis on it and consequently to reduce the possible attacker *profiles* to a finite set. As the game develops, we can observe how the attacker responds to our declared strategies in an *expert way*, i.e. we have complete information on her moves (but not on her strategies). Given the presence of uncertainties it is not possible to apply the conventional Game Theory solution concepts, instead our objective will be trying to *minimize the regret*, that means minimizing what we lose at each round for having chosen the *wrong* strategy. In a *Security Game* context we can imagine a situation in which we are trying to fight the crime in a city in which suddendly appears a new gang. We know the possible *behavioural profiles* of gangs but we do not know which one we are facing, and therefore we are not able to use our police *resources* properly. Generally speaking the described problem arises when, in a leader-follower scenario, there are not enough historical data, or expert knowledge to classify with certainty the attacker behavioral profile,

but we need a solution that exploits all the collected data in the best possible way.

The Security Game area has been the subject of increasing interest in these years, representing the most remarkable application of leaderdship games. Furthermore our group has experience in this area, which makes us able to make reasonable hypothesis on the profiles. Thus, this work, whose results are valid for every kind of leadership game anyhow, has a particular focus on this context. This choice also enables us to make more complex experiments and then interpret the results in a sensible way. In particular we will refer to the *gangs* setting each time we want to make a practical example of a theoretical concept. We define this problem as *Follower's Behaviour Identification in Security Games* (FBI-SG).

1.3 Original Contributions

Having defined a new problem, no dedicated solutions exist in literature, however general learning algorithms can be directly applied to it. Such techniques, that are based on the observed loss, result to be unable to correctly exploit the received feedback, because they do not take into account the defender commitment to a strategy. Thus, we introduce two novel approaches to deal with our problem, bridging together game-theoretical techniques and online learning tools. In the first approach, the leader has a *belief* about the follower and updates it during the game. We name the algorithm *Follow the Belief* (`FB`) and we provide a finite-time analysis showing that the regret of the algorithm is constant in the length of the time horizon. In the second approach, namely *Follow the Regret* (`FR`), the learning policy is driven directly by the estimated expected regret and is based on a backward induction procedure. Finally, we provide a thorough experimental evaluation in concrete leadership settings inspired to security domains, comparing our algorithms with the main algorithms available in the state of the art of the online learning field and showing that our approaches provide a remarkable improvement in terms of expected pseudo-regret minimization.

1.4 Structure

This work is organized as follows:

- Chapter 2 introduces the main works on machine learning applied in security games, and the main concepts and algorithms of prediction with expert advice;
- Chapter 3 exposes the faced problem in all its details, showing the main difficulties and challenges that issues;
- Chapter 4 explain the main issues in applying directly state-of-the-art online learning algorithms to the problem, and present our solutions to it.
- Chapter 5 illustrates and analyses the experimental phase of our work to evaluate our algorithms, pointing out the enhancements achieved respect to the state-of-the-art algorithms;
- Chapter 6 summarises the main challenges and results of this work, proposing possible extensions to the problem and to the solutions shown.

Chapter 2

Literature Review

In this chapter we explain the main concepts and tools that we use in the rest of the work. We introduce the concept of *Leadership Game* and the main techniques to which we confront, namely *Follow the Perturbet Leader*, *UCB1* and *Thompson Sampling*. Then, we shall also illustrate the main works in literature that are related to ours, highlighting differences and similarities.

2.1 Game Theory

Game Theory is the name given to the methodology of using mathematical tools to model and analyse situations of interactive decision making. These situations involve several decision makers, called *players*, with different goals, in which the decision of each affects the outcome for all the decision makers [14]. In principle, games can exhibit complex dynamics involving players who move in a certain order and can observe previous players' moves, or including random events that can alter the outcome of a certain strategy profile. However if the players move simultaeously and there are no random events, then, it is possible to represent a game in the so called *normal-form*:

Definition 1 (Normal-Form Game)**.** *A game in* normal-form *(or in* strategic-form*) is an ordered triple* $G = (N, (S_i)_{i \in N}, (u_i)_{i \in N})$ *in where:*

- $N = \{1, 2, \ldots, n\}$ *is a finite set of players.*

- *S_i is the set of strategies of player i, for every player $i \in N$.*
  We denote the set of all vectors strategies by $S = S_1 \times S_2 \times \ldots \times S_n$.
- *$u_i : S \to \mathbb{R}$ is a function, called* utility*, associating each vector of strategies $s = (s_i)_{i \in N}$ with the payoff $u_i(s)$ to player i, for every player $i \in N$.*

Games can model many different situations, where players' goals might be related to each other in a positive way, i.e., the players are cooperating, but also in a negative one, e.g., one player gets a utility opposite w.r.t. the one gained by her opponent. This latter case is modeled with *zero-sum* games.

Definition 2 (Zero-Sum Game)**.** *A two-player game is a* zero-sum game *if for each pair of strategies (s_1, s_2) one has:*

$$u_1(s_1, s_2) + u_2(s_1, s_2) = 0 \tag{2.1}$$

where s_1, s_2 are, respectively, the strategy of the first and second player. Many times the modeled situation involves more than one *interaction* between players.

Definition 3 (Repeated Game)**.** *A game is a* repeated game *if it consists in the repetition of another game, called its* base game*.*

We can have repeated games, where:

- the game lasts an infinite number of stages, and each player wants to maximize the time-discounted sum of her payoffs;
- the game lasts a finite number of stages T , and every player wants to maximize her average payoff;
- the game lasts an infinite number of stages, and every player wants to maximize the upper limit of her average payoffs.

2.1.1 Leadership Games

Leadership Games make their first appearence in a model described in Market Structure and Equilibrium by the German economist Heinrich Freiherr von Stackelberg [29]. The model presented in such work was a strategic game

in economics modeling a market situation in which the *leader* firm moved first while the *follower* observes the leader's choice and acts subsequently. This happens, for example, when a company aims at planning the production of a product and has to decide when and how it is convenient to enter the market when another company is already the leader in such market. In general a Leadership Game is now defined as a situation in which:

- the leader can *commit to a strategy*;
- the follower observes the leader's commitment and responds with another strategy, based on its *rationality level*.

If we assume that both players are perfectly rational, then we can solve this game adopting the *Leader-Follower equilibrium*.

Definition 4 (Leader-Follower Equilibrium)**.** *The strategy profile* (s_l, s_f^*) *is a* Leader-Follower equilibrium*, if* s_l *and* s_f^* *are solutions of the following problem:*

$$\begin{aligned} \arg\max_{s_l, s_f^*} u_l(s_l, s_f^*) \quad & s.t. \\ s_l \ &\in \ \Delta_l \\ s_f^* \ &\in \ \arg\max_{s_f} u_f(s_f, s_l) \quad s.t \\ & s_f \in \Delta_f \end{aligned}$$

where Δ_l, Δ_f are respectively the simplex where leader and follower strategies are defined, and u_f is the follower's utility function. In other words, the leader commits to the strategy that maximizes its own utility when the follower best-responds to it.

2.1.2 Security Games

The Leader-Follower model has been largely applied to tackle problems that arise when physical security of environments or infrastructures must be guaranteed. In this context the goal is to protect ports, airports, buses and trains, transportation or other infrastructure often with limited security resources to accomplish this goal. Due to the vastness of the environment, we cannot protect each area. Moreover, we can only deploy a scarce number of resources, being these expensive. Thus, we should carefully select

how to place them. Unfortunately, opponents can monitor our defenses and exploit any pattern in these selective deployments. A possible solution to this problem is a weighted randomization of the resources, assigning guards to targets based on targets' value. This prevents attackers to easily predict which targets are uncovered, however they are still able to conduct surveillance, learn the *weights* on the targets and act consequently. This scenario perfectly matches the Leader-Follower model, where the defender can commit to a strategy (the surveillance schedule) and the attacker must act according to it, as a follower. Solutions of this kind have been applied succefully to real-world problems, e.g., in [20] game theoretic techniques have been applied to ensure the security of the Los Angeles International Airport (LAX), in [26] the authors exploit the Stackelberg paradigm to study how to schedule undercover federal air marshals on domestic U.S. flights, while in [21] such paradigm is employed to allocate the Transportation Security Administration (TSA) scarce resources to provide protection within several U.S. airports. A higher degree of interaction among the agents is captured in [2], where an alarm system to detect potential attacks is introduced.

2.1.3 Issues

Even if Leader-Follower Equilibrium is perfectly suitable to solve this kind of problems, this solution needs to have all the parameters of the problem. In particular it is necessary to determine:

- the payoffs of the targets;
- the attacker's resources;
- the attacker's rationality degree.

Security Games represent adversarial interactions between players. However, in general, security games need not to be zero-sum. Some reasons could be that the opponent evaluates some targets as particularly important for her audience for their symbolic value, whereas they may not be of equal importance to the police. Or an opponent may not view even a failed attack as a negative outcome because of the publicity and fear it generates. Or the adversary may need to incur a significant cost in mounting a particular attack that may not be particularly important to the police. This problem

is sometimes faced with the Bayesian Stackelberg Games formalism. In this model, it is assumed that different types of attackers (which means different payoffs matrixes) appear to the defender according to a certain distribution. Determining the values of the targets is a job for domain experts, which is not easy and subject to errors.

Attacker's rationality is another issue: in the previous sections, we always assumed *perfectly rational* attackers, but in reality attackers are humans who do not usually reason in a rational fashion. Many human rationality models have been applied by psychologists to describe human behaviour about choices, though often these models are *parametric* or simply not suitable to describe all possible kinds of attackers.

In general, when the Security Game model is not completely specified, we have to overcome that *uncertainties*, developing new solutions that involve *learning* or *robust optimization* techniques. These and other issues that have been faced in literature, will be illustrated later in this chapter. However first we have to introduce the Online Learning framework that stands at the base of many of the proposed solutions and, as we shall see, has a strong connection to Game Theory itself.

2.2 Online Learning

A *sequential decision making* problem consists in taking a choice based on a past history of observations, having as a goal the maximizazion of a reward function (or the minization of a loss function, which is the convention we are going to maintain all over this work). Depending on which is the structure of the problem and our knowledge of it, we may solve the problem in an exact way (e.g. with the Bellman Equation). Unfortunately, in many cases we are not able to solve it exactly and we have to resort to find an approximated solution by means of some *Online Learning* technique. In this section we shall introduce two frameworks: Predictions with Expert Advice (for which it is used the notation of [4]) and Multi-Armed Bandit. These represent situations in which we are not able to specify the states of the system, and therefore we cannot use Reinforcement Learning tools. In practice, we implicitly assume that the system has only one state, and, as it happens in repeated games, we are asked, at each round, to choose among the same set of actions, based on what happened in the past.

2.2.1 Prediction with Expert Advice

Prediction with Expert Advice is based on the following protocol for sequential decisions: the decision maker is a forecaster whose goal is to predict an unknown sequence $y_1, y_2, \ldots$ of elements of an outcome space $\mathcal{Y}$. The forecaster's predictions $\hat{p_1}, \hat{p_2}, \ldots$ belong to a decision space $\mathcal{D}$ that we assume to be a convex subset of a vector space. In some special cases we take $\mathcal{D} = \mathcal{Y}$, but in general $\mathcal{D}$ may be different from $\mathcal{Y}$. The forecaster computes his predictions in a sequential fashion, and his predictive performance is compared to that of a set of reference forecasters that we call *experts*. More precisely at each time t:

- the forecaster has access to the set $\{f_{E,t} : E \in \mathcal{E}\}$ of expert predictions $f_{E,t} \in \mathcal{D}$, where $\mathcal{E}$ is a fixed set of indices for the experts;
- on the basis of the experts' predictions, the forecaster computes his own guess $\hat{p_t}$ for the next outcome y_t;
- after $\hat{p_t}$ is computed, the true outcome y_t is revealed.

The predictions of forecaster and experts are scored using a non-negative *loss function* $l : \mathcal{D} \times \mathcal{Y} \to \mathbb{R}$. This prediction protocol can be naturally viewed as a *repeated game* between *forecaster* that makes guesses $\hat{p_t}$, and *environment*, who chooses the expert advice $\{f_{E,t} : E \in \mathcal{E}\}$ and sets the true outcomes y_t, determining a certain loss for the first player.

The forecaster's goal is to keep as small as possible the *cumulative regret* with respect to each expert:

Definition 5 (Cumulative Regret)**.** *The* cumulative regret *(or simply* regret*) for expert E is equal to:*

$$R_{E,n} = \sum_{t=1}^{n} (l(\hat{p_t}, y_t) - l(f_{E,t}, y_t)) = \hat{L}_n - L_{E,n} \tag{2.2}$$

where $\hat{L}_n = \sum_{n_t=1}^{n} l(\hat{p_t}, y_t)$ denotes the forecaster's cumulative loss while $L_{E,n} = \sum_{t=1} l(f_{E,t}, y_t)$ represents the cumulative loss of expert E. Hence, $R_{E,n}$ is the difference between the forecaster's total loss and that of expert E after n prediction rounds. We also define the instantaneous regret with respect to expert E at time t as follows:

$$r_{E,t} = l(\hat{p_t}, y_t) - l(f_{E,t}, y_t). \tag{2.3}$$

Thus, $R_{E,n} = \sum_{t=1}^{n} r_{E,t}$. One may think of $r_{E,t}$ as the regret the forecaster feels of not having listened to the advice of expert E right after the t^{th} outcome y_t has been revealed. Throughout the rest of this work, we can assume that the number of experts is finite, $\mathcal{E} = 1, 2, \ldots, N$, and therefore use the index $i = 1, \ldots, N$ to refer to an expert.

The goal of the forecaster is make predictions in order to minimize the regret for the all sequences of outcomes. For example, the forecaster may want to have a vanishing per-round regret, or a *sub-linear regret*, that is, to achieve

$$\max_{i=1\ldots N} R_{i,n} = o(n) \quad \text{or, equivalently,} \quad \frac{1}{n}(\hat{L}_n - \min_{i=1,\ldots,N} L_{i,n}) \xrightarrow{n\to\infty} 0 \tag{2.4}$$

where the convergence is uniform over the choice of the outcome sequence and the choice of the expert advice. Conversely, if the regret has the same order of n (i.e it is a *linear regret*) the instantaneous regret is not reducing with time and the solution is not reaching the global minimum of the loss function.

Follow the Leader

The simplest forecasting strategy, namely *fictitious play*, consists in choosing, at time t, an expert that minimizes the cumulative loss over the past $t-1$ time instances. In other words, the forecaster always follows the expert that has had the smallest cumulative loss up to that time.

Algorithm 1 `FOLLOW THE LEADER`

1: **for all** $t \in \{1, \ldots, n\}$ **do**
2: Compute $E = \arg\min_{E_k \in \mathcal{E}} \sum_{i=1}^{t-1} l(f_{E_k,i}, y_i)$
3: Play expert $\hat{p}_{k_t} = f_{E,t}$
4: Observe y_t

Under some conditions (e.g. square loss function or convex losses with constant experts, see [4]), the regret of this algorithm grows as slowly as $O(\ln(n))$. However, in general, it can suffer in some case from linear regret. For example, consider $N = 2$ actions such that the sequence of losses $l(1, y_t)$ of the first is $(\frac{1}{2}, 0, 1, 0, 1, \ldots)$ while the values of $l(2, y_t)$ are $(\frac{1}{2}, 1, 0, 1, 0 \ldots)$. Then $L_{i,n}$ is about $\frac{n}{2}$ for both, but Follow the Leader suffers a loss of $O(n)$.

Follow the Perturbed Leader

As pointed out in [9], with a simple modification we can overcome the previous problem. In fact we can add a small random perturbation to the cumulative losses, and follow the *perturbed* leader. Formally, let $Z_1, Z_2 \ldots$ be independent, identically distributed random N-vectors with components $Z_{i,t} = (i = 1, \ldots, K)$, where $K = |\mathcal{E}|$. For simplicity, assume that Z_t has a uniform distribution. At time t, the *follow-the-perturbed-leader* forecaster selects an action:

$$I_t = \arg\min_{i=1,\ldots,n}(L_{i,t-1} + Z_{i,t}). \tag{2.5}$$

Algorithm 2 `FOLLOW THE PERTURBED LEADER`

1: **for all** $t \in \{1, \ldots, n\}$ **do**
2: **for all** $E_k \in \mathcal{E}$ **do**
3: Sample $Z_{k,t} \sim U(0, \sqrt{nK})$
4: Compute $E = \arg\min_{E_k \in \mathcal{E}} \sum_{i=1}^{t-1}[l(f_{E_k,i}, y_i)] + Z_{k,t}$
5: Play expert $\hat{p}_{k_t} = f_{E,t}$
6: Observe y_t

If $Z_{i,t}$ is uniformly distributed on $[0, \sqrt{nK}]$ then the actual regret, with probability at least $1 - \delta$, satisfies [4]:

$$R_n \leq 2\sqrt{nK} + \sqrt{\frac{n}{2}\ln(\frac{1}{\delta})}. \tag{2.6}$$

2.2.2 Multi-Armed Bandit (MAB)

The singular name of this problem derives from one of the examples used to describe it in an informal way, after the first formalization in [24]: a gambler has to play with some slot machines (which were once also called *one-armed-bandit*) and wants to minimize her cumulative loss (or maximize her cumulative reward). Therefore, based on the losses she observed playing different "arms", she has to decide which arm he will play next. More formally, in the multi-armed bandit setting we define a finite set of possible choices, $\mathcal{D}$, where the possible choices are called *arms*. At each turn, the player selects arm $\hat{p}_t$ from $\mathcal{D}$ and observes the associated loss l_t. The objective is to minimize the sum of the collected losses over the horizon H, i.e.

the number of rounds that have to be played after the beginning. The regret R after n rounds is defined as the expected difference between the sum of the collected losses and the loss sum associated with an optimal strategy :

$$R = \sum_{t=1}^{n} l_t - nl^* \tag{2.7}$$

where l^* is the minimal loss mean and l_t is the loss at time t. This problem can be also seen as a one-state *Markov Decision Process* or as a modified version of the Prediction with Expert Advice Problem, where the latter differs from the MAB on the received feedback. In fact, in the expert context, we are always informed of which expert (arm) is the best choice, while in the MAB setting we receive a loss, but we have no means of knowing what would have been the best arm to select at that round. In other words, we can call the expert feedback a *complete feedback* and the MAB one a *partial feedback*, because it gives us information only on the *pulled* arm, but not on the others. Unfortunately, all MAB algorithms are subjected to a lower bound[13]:

Theorem 2.2.1 (MAB Lower Bound)**.** *Given a MAB stochastic problem any algorithm satisfies:*

$$\lim_{t\to\infty} L_t \geq \log t \sum_{i|\Delta_i} \frac{\Delta_i}{KL(\mathcal{R}_i, \mathcal{R}^*)} \tag{2.8}$$

where $\mathcal{R}_i$ represents the distribution of the rewards of arm i, $\Delta_i = \mathbb{E}[\mathcal{R}^*] - \mathbb{E}[\mathcal{R}_i]$, and $KL(\mathcal{R}_i, \mathcal{R}^*)$ is the Kullback–Leibler *divergence between the two distributions $\mathcal{R}_i$ and $\mathcal{R}^*$.*

Upper Confidence Bound - UCB1

While in the expert setting we have no need of exploration, having a complete feedback, in MAB's one we can not resort to algorithm like *Follow the Leader*, because we are compelled to explore also the other choices. A way to do it is considering a *lower bound* $LB_{k,t}$ over the expected loss $L_{k,t}$ such that, with *high probability* $LB_{k,t} = \hat{L}_{k,t} + B_{k,t} \leq L_{k,t}$, where $B_{k,t}$ is a bound that depends on how much information we have on an arm, which is embodied by the number of times we have pulled arm k so far. Specifically, we want to have a large bound if we have chosen the arm a few times, and

Algorithm 3 `UCB1`

1: **for all** $t \in \{1, \dots, n\}$ **do**
2: **for all** $p_k \in \mathcal{D}$ **do**
3: Compute $\hat{L}_t = \sum_{i=1}^{t} l_{k,i} \mathbb{I}[p_k = p_{k_i}]$
4: Compute $B_{k,t}$
5: Play arm $\hat{p}_{k_t} = \arg\min_{p_k \in \mathcal{D}} LB_{k,t}$

a tight bound if the arm has been largely pulled. In order to set this bound we resort to a concentration inequality called Hoeffding Bound:

Definition 6 (Hoeffding Inequality Bound)**.** *Let $X_1, \dots, X_t$ be i.i.d random variables with support in $[0, 1]$ and identical mean $\mathbb{E}[X_i] = X$ and let $\overline{X} = \sum_{i=1}^{t} X_i$ be the sample mean. Then:*

$$\mathbb{P}(X < \overline{X} + u) \leq e^{(-2tu^2)}.$$

This inequality can be applied to each arm:

$$\mathbb{P}(L_{k,t} < \hat{L}_{k,t} + B_{k,t}) \leq e^{(-2tB_{k,t}^2)}. \tag{2.9}$$

Picking a probability p that the real value exceeds the bound $e^{(-2tu^2)} = p$ we can solve this equation to find $B_{k,t} = -\sqrt{\frac{\log(p)}{N_{t,k}}}$, where $N_{k,t}$ is the number of times we pulled arms k. UCB1 algorithm employs a typical *frequentist* approach to the problem, in fact it does not assume any underlying distribution for some arms' parameters. This algorithm has an upper bound:

Theorem 2.2.2 (UCB1 Upper Bound)**.** *At time T, the expected total regret of UCB1 algorithm applied to a stochastic MAB problem is:*

$$L_T \leq 8 \log T \sum_{i|\Delta_i > 0} \frac{1}{\Delta_i} + \left(1 + \frac{\pi^2}{3}\right) \sum_{i|\Delta_i > 0} \Delta_i. \tag{2.10}$$

Thompson Sampling

The other option w.r.t. a frequentist approach is a *Bayesian* one, adopted by Thompson Sampling. The algorithm assumes that rewards are Bernoulli variables and each arm has a different Beta distribution on the Bernoulli

Algorithm 4 THOMPSON SAMPLING

for all $p_k \in \mathcal{D}$ **do**
  Initialize $\alpha_k = 1$, $\beta_k = 1$
for all $t \in \{1, \dots, n\}$ **do**
  for all $p_k \in \mathcal{D}$ **do**
    Sample $\hat{l}_{k,t} = -\hat{r}_{k,t} \sim Beta(\alpha_k, \beta_k)$
  Play arm $\hat{p}_{k_t} = \arg\min_{p_k \in \mathcal{D}} \hat{l}_{k,t}$
  Observe l_t
  Sample $s \sim Bernoulli(p = 1 - \bar{l}_t)$
  if $s = 1$ **then**
    $\alpha_{k_t} = \alpha_{k_t} + 1$
  else
    $\beta_{k_t} = \beta_{k_t} + 1$

parameter. Every time we pull an arm, we observe the loss l_t, and, in case of success ($l_t = 0$) we update the α parameter of its Beta prior with $\alpha_{k,t+1} = \alpha_{k,t} + 1$, while in case of failure ($l_t = 1$) we update the parameter β with $\beta_{k,t+1} = \beta_{k,t} + 1$. Then, at each turn, we sample from each Beta prior, and choose the arm which has the maximum sampled probability of success. In principle, this algorithm should be used only with Bernoulli rewards (losses), but here we present an adjustment [1] that makes it work with every loss domain. We take the normalized observed loss $\bar{l}_t$ and then we interpret $1 - \bar{l}_1$ as the probability of success of a Bernoulli and after we have established this value we sampled from it, in order to classify the observation as a success or a failure. There exists an upper bound for this algorithm:

Theorem 2.2.3 (Thompson Sampling Upper Bound)**.** *At time T, the expected regret of Thompson Sampling applied to a stochastic MAB problem is:*

$$L_T \leq O\left(\sum_{i|\Delta_i>0} \frac{\Delta_i}{KL(\mathcal{R}_i, \mathcal{R}^*)}(\log T + \log\log T)\right). \tag{2.11}$$

2.3 Uncertainty in Security Games

Uncertainties can arise from multiple sources in Security Games. In this section we describe the main works in literature on this topic, comparing them to our work.

2.3.1 Bounded Rationality

First works in Security Games assumed a *completely rational* (sometimes called also "game-theorist") opponent. However, even if it can be supposed to be true for particularly critic situations, in most of the security games, the attackers will take decisions in a different way, a human way indeed, using essentially their feelings and intuitions to choose amongst the possible targets. Quantal choice [16] is one of the classical models used to explain the bounded rational behaviour. It has also been applied in games [17], bringing to the definition of a new kind of equilibrium, the Quantal Choice Equilibrium. This model has been used in security games in many ways, particularly in the *poaching* context. First it has been applied using the traditional *utility* function [32], with only one parameter λ used to measure the rationality. However, later works showed that human behaviour is better described by the use of a *Subjective Utility function* [18] that weights in a subjective way the coverage probability and the target values ($SUQR$). Successive refinements of the model took in consideration non-linear subjective function [11], taking inspiration from the Prospect Theory [28], and also adding other parameters to the SU, as animal density and distance. Another article on *green security games* also considered SUQR attackers that re-weighted the defender commitment using a linear combination of past commitments, with a *bounded memory*. In a paper on protecting natural resources by illegal extractors [22] attackers were modelled as *Fictitious Quantal Response* (FQR) agent, that means that these attackers assume the defender empirical distribution to be its strategy in the next round. Also attackers with a stocastic strategy, stationary or non-stationary, which are unaware of the defender commitment are taken into account in a work in the context of border patrolling [12] and in another one considering general solution to partial feedback Security Games [30]. The same works also consider *fictitious player* attackers: as for FQR, these attackers assume the defender to draw its move from a stochastic fixed distribution, which they

estimate with their observations. They respond to it by playing the pure strategy that maximizes their utility function.

2.3.2 Unknown Payoffs

Determining the exact values for the payoff matrix is a challenging task: estimates are usually obtained from risk analysis experts and historical data, but the degree of uncertainty is typically high. Consequently, researchers have introduced Bayesian frameworks [20] that capture uncertainty using a probability distribution over possible games. This framework assumes an underlying distribution of attackers with different payoff values for the targets. However, such a hypothesis may compromise the deployed solution if the supposed distribution is far from the real one. This issue could be solved by learning the distribution, which is, unfortunately, most of the time an impossible task. Another approach consists in bounding the payoffs into fixed intervals, and then using a robust optimization or a minimax regret algorithm [19]. Conversely, other works did not make any assumption on they payoffs, employing online learning techniques to determine them while playing. In [3] *membership-queries* to find the best response against an attacker without knowing the exact payoffs, leveraging on the presence of a membership oracle for the optimization region, which is represented by the response of the attacker to the commitment. In another article [30] a modified version of FPL algorithm is used to minimize the regret in a MAB way. Things can get also more complicated as we extend the domain of the security game: in a work on fighting illegal extraction of natural resources [22] the defender is supposed to be unaware of the values of a target until it has been attacked. This happens because the area that has to be monitored is so large, and forbidden to any exploitation, that only attackers can discover the value of its resources.

2.3.3 Feedback

Everytime it is necessary to resort to online learning in order to overcome uncertainties, learning performance strongly depends on the type of feedback received. In a security game context, a complete feedback (Section 2.2.2) for the defender consists in observing at each turn where the opponent attacked even if the defender was to able to catch it. This is the hypothesis made

in most of the papers in literature ([31], [11], [3], [19], [22],[7]), though in some cases it is necessary to assume the opposite, namely a partial feedback. For example poachers can catch their prey without leaving any track [23] or people can pass illegally the border without being noticed [12]. Most of the time attackers are modeled as *strategy-aware*. This means that they can see the defender's commitment to a strategy. However in some papers they only have a complete feedback ([22], [7]), which is usually used to estimate defender's next strategy (Fictitious Player, FQR).

2.3.4 Features of Security Game Models

Here it has been made the attempt to list some of the main features of security game models in literature, based on the possible uncertainties and other characteristics not mentioned before:

- **Attacker Rationality**: attacker can be completely rational (Stackelberg hypothesys), boundedly rational (SUQR, FQR, Fictitious Player) or she can have a stochastic behavior;
- **Payoff Knowledge**: attacker payoffs can be known, vary amongst the various type of attacker (Bayesian Security Games), or be completly unknwnow to the defender;
- **Feedback**: the defender is not always aware of all what happened in the last round, she can have a *partial* or a *complete* feedback. The attacker can be *strategy-aware* or have a complete feedback. No works in the literature consider attackers with a partial feedback;
- **Topology**: the game can have a topological structure, such that not all the actions are available to each position of players, or it can have no such a structure;
- **Coordination**: attackers are *coordinated* if they act as a unique entity (for example never attacking the same target at the same time). They have a *common knowledge* if they share collected information. When there are multiple defending resources, usually they have both these features, but in principle they could be, for example, not coordinated or temporary unable to share information;

- **States**: the system can be modeled with a single state or with multiple states.

Problem Classification

The problem considered in this work has *known payoffs*, *complete feedback* and a *complete coordination*. *No topology* is considered, and a *single state* is present. However, differently from the previous literature this work allows to mantain *unspecified* the attacker *rationality*. In fact our main goal is to discover if it is possible to learn the opponent's behavior from her moves, in connection to the defender's commitments, while minimizing the regret.

Chapter 3

Problem Formulation

Although our work can be employed in principle for any leadership scenario, here, for the sake of clarity, we focus on security domains, thus referring to the leader as *defender* and to the follower as *attacker*.

Let us consider a 2-player normal-form repeated game $\mathcal{G}_N$ defined over a finite number of rounds $N \in \mathbb{N}$, where a defender D and an attacker A play against each other in some environment with some valuable targets $\mathcal{M} = \{1, \dots, M\}$, characterized by values $\mathbf{v} = (v_1, \dots, v_M)^T, v_m \in (0, 1]$. The goal of the defender D is to protect such targets while the attacker A aims at compromising them. The space of actions of D and A is given by the set of targets such that D chooses the target to protect, while A chooses the target to attack. The course of the game is represented in Figure 3.1. Specifically, at each round $n \in \{1, \dots, N\}$, the defender D announces the strategy $\boldsymbol{\sigma}_{D,n} \in \Delta_M$, with Δ_M denoting the M-dimensional simplex, she commits to, while A observes such commitment. Then, they concurrently play their action over the target space. In particular, the defender plays actions $i_{D,n} \in \mathcal{M}$ according to $\boldsymbol{\sigma}_{D,n}$ while A, the follower, plays $i_{A,n} \in \mathcal{M}$ according to some attacker model $\boldsymbol{\sigma}_A(\boldsymbol{\sigma}_{D,n}) \in \Delta_M$. The game is zero-sum: if D and A choose the same target during some round, they both get a utility equal to 0, conversely if A attacks the i-th target while D decides to protect the j-th one, A gets v_i and D gets $-v_i$ due to the loss of the target. More concisely, the defender incurs in the *loss*:

$$l_n := v_{i_{A,n}} \mathbb{1}\{i_{A,n} \neq i_{D,n}\}, \tag{3.1}$$

for each $n \in \{1, \dots, N\}$
1. D publicly commits to a strategy $\boldsymbol{\sigma}_{D,n}$
2. A observes the strategy D committed to
3. D and A play $i_{D,n}$ and $i_{A,n}$, respectively
4. D incurs in loss l_n according to 3.1

Figure 3.1: Leader-follower interaction

not suffering from any loss if both players select the same target.[1] Hereafter, we assume that the defender is able to compute the best response strategy $\boldsymbol{\sigma}_D^*(A) \in \Delta_M$ if she is given the attacker model she is playing against. Similarly, we denote with $\boldsymbol{\sigma}_A^*(\boldsymbol{\sigma}_D) \in \Delta_M$ the best response A plays against strategy $\boldsymbol{\sigma}_D$ of D. According to such assumption, we can compute the expected loss of D against a generic attacker A as:

$$L(A) := \sum_{m \in \mathcal{M}} \sigma_A(\boldsymbol{\sigma}_D^*(A))_m \, v_m \, (1 - \sigma_D^*(A)_m), \tag{3.2}$$

where $\sigma_\cdot(\cdot)_m$ is the probability associated with target m by the strategy.

The problem we study in this work is defined as follows:

Definition 7. *The Follower's Behaviour Identification in Security Games (FBI-SG) problem is a tuple* $(\mathcal{G}_N, \mathcal{A}, A_{k^*})$*, where* $\mathcal{G}_N$ *is a 2-player normal-form repeated game as described above and* $\mathcal{A} = \{A_1, \dots, A_K\}$ *is a set of possible attacker behavioural profiles, with* $A_{k^*} \in \mathcal{A}$ *denoting the actual profile of the attacker in* $\mathcal{G}_N$*, unknown to the defender* D*.*

In this work, we cast the FBI-SG as a sequential decision learning problem, where, at each round n, the defender aims at selecting her best response to the attacker in order to identify the actual attacker profile $A_{k^*} \in \mathcal{A}$ while minimizing the loss suffered during the learning process.

Definition 8. *A* policy $\mathfrak{U}$ *is an algorithm able to provide at each round* n *a strategy profile* $\boldsymbol{\sigma}_{D,n}$ *for the defender* D*. Formally:*

$$\mathfrak{U}(h_n) := \boldsymbol{\sigma}_{D,n},$$

[1] Hereafter, we denote with $\mathbb{1}\{E\}$ the indicator function of a generic event E.

where h_n is the history collected so far, i.e., all the strategies declared by the defender $\{\boldsymbol{\sigma}_{D,1}, \dots, \boldsymbol{\sigma}_{D,n-1}\}$, the actions played by the two players $\{i_{D,1}, i_{A,1}, \dots, i_{D,n-1}, i_{A,n-1}\}$ in the past rounds and the corresponding losses $\{l_1, \dots, l_{n-1}\}$.

We evaluate the performance of a given policy $\mathfrak{U}$ over a finite-time horizon of N rounds by means of the expected cumulative *pseudo-regret*, defined as:

$$R_N(\mathfrak{U}) = \mathbb{E}\left[\sum_{n=1}^{N} l_n\right] - L^* N,$$

where $L^* := L(A_{k^*})$ is the expected loss incurred by the defender if she plays the best response to the actual attacker A_{k^*}, l_n is the loss incurred by using the policy $\mathfrak{U}$ at round n and the expectation $\mathbb{E}[\cdot]$ is taken w.r.t. the stochasticity of the attacker strategy, the defender policy and the policy $\mathfrak{U}$. The goal of a generic policy $\mathfrak{U}$ is to minimize the pseudo-regret $R_N(\mathfrak{U})$ incurred while learning the true attacker's profile.

3.1 Profiles Set Restriction Motivation

In the next section will follow the list and the description of the analysed attacker profiles. The necessity of restict the set $\mathcal{A}$ arises from the impossibility of obtaining a *sublinear expected pseudo-regret* without any assumption on the attacker, as we can easily see in the following counterexample.
Let's consider the most general attacker behaviour function:

$$f : \Delta_{M,D} \to \Delta_{M,A} \tag{3.3}$$

that given a defender strategy vector returns the attacker strategy vector, and define the expected loss function of f as:

$$l_f(\boldsymbol{\sigma}_D) := \sum_{m \in \mathcal{M}} f(\boldsymbol{\sigma}_D)_m \, v_m \, (1 - \sigma_{D,m}). \tag{3.4}$$

We are in general interested in having at least a *sublinear* cumulative regret (see Section 2.2.1) which means that we are asimptotically reaching the minimum of the expected loss function.

Let's now consider an FBI-SG problem in which all targets have the same value v and the attacker function is:

$$\hat{f}(\boldsymbol{\sigma}_D) = \begin{cases} \hat{\mathbf{i}}^* & \boldsymbol{\sigma}_D = \boldsymbol{\sigma}_D^* \\ \left(\frac{1}{M} \dots \frac{1}{M}\right) & \text{otherwise} \end{cases}, \tag{3.5}$$

where $\boldsymbol{\sigma}_D^* \neq \left(\frac{1}{M} \dots \frac{1}{M}\right)$ is a random strategy in $\Delta_{M,D}$, and $\hat{\mathbf{i}}^*$ is the unit M-dimension vector whose only not null component is $i^* = \max_{i \in M} \boldsymbol{\sigma}_{D,i}^*$. If we now try to compute l_f we obtain:

$$l_{\hat{f}}(\boldsymbol{\sigma}_D) = \begin{cases} v(1 - \sigma_{D,i^*}^*) & \boldsymbol{\sigma}_D = \boldsymbol{\sigma}_D^* \\ \frac{v}{M} & \text{otherwise} \end{cases}. \tag{3.6}$$

By construction $v(1 - \sigma_{D,i^*}) < \frac{v}{M}$ and therefore the function has only one global minimum in $\boldsymbol{\sigma}_D^*$. The only way to find the minimum of this *needle-in-the-haystack* function is to find $\boldsymbol{\sigma}_D^*$ randomly sampling the simplex. Since it is a point in an infinite set, the probability to find it is equal to zero and therefore we will have a *linear* regret.

3.2 Analysed Attacker Profiles

Since it is impossible to obtain general solutions to this problem, in order to face it we have to reduce the set of the considered profiles. In particular we can restrict it to:

- a finite set of profiles;
- a subfamily of profiles.

In this section, we describe the different attacker profiles we study in this work and formalize the definition of the attacker strategy $\boldsymbol{\sigma}_{A_{k^*}}(\cdot)$ for two sets of attackers, grouped depending on their ability to change their behavior w.r.t. the strategy D commits to. Specifically, on one side, we take into account stochastic attackers, which disregard the strategy of D, on the other, we focus on strategy-aware attackers, able to modify their strategies depending on the defender announced strategy $\boldsymbol{\sigma}_{D,n}$. In particular we will also indicate what is the *best response* that the defender can give to each one of these profiles.

3.2.1 Stochastic Attacker

The first class of attackers is the *Stochastic* (*Sto*) one, where the attacking player does not take into account the strategy $\boldsymbol{\sigma}_{D,n}$ announced by the defender D and thus has a fixed *Known* probability over time to attack the targets. This class of attackers models opponents focused on specific targets and whose preferences are not influenced by the defender behaviour. We can identify this attacker as the one with the least rationality, because it does not take into account neither the defender commitment neither the targets value. At round n, a stochastic attacker *Sto* plays according to the strategy:

$$\boldsymbol{\sigma}_{Sto}(\boldsymbol{\sigma}) = \mathbf{p}(Sto) \quad \forall \boldsymbol{\sigma} \in \Delta_M,$$

where $\mathbf{p}(Sto) \in \Delta_M$ is a probability distribution over the targets, which is known to D. In this case, the defender best response $\boldsymbol{\sigma}_D^*(\boldsymbol{\sigma}_{Sto})$ is defined as:

$$\sigma_D^*(Sto)_m = \begin{cases} 1 & \text{if } m = \arg\max_{i \in \mathcal{M}} \{v_i\, p(Sto)_i\} \\ 0 & \text{otherwise} \end{cases}.$$

Unknown Stochastic Attacker

Another possible profile family is a stochastic attacker with a probabiliy distribution that is *unknown* to the defender at the beginning of the game. In this case we can not have a closed-form best response as before, but we can *learn* the attacker empirical distribution as we play. During the learning process the best we can do is to use the *Follow-the-Leader* algorithm (Section 2.2.1) which selects the target that has the least expected loss based on the empirical distribution. What we are doing in practice is best responding to a model of attacker that uses empirical frequencies as estimators of the oppponent's strategy. These estimators are *consistent* and *unbiased*.

3.2.2 Strategy Aware Attackers

The second class of attackers we examine in this work consists of strategy aware attackers, corresponding to profiles able to modify their strategy depending on strategy of the defender D. In particular, we study:

- *Stackelberg* (*Sta*) attackers: they represents the attacker with the highest level (sometimes called *game-theorist* level) of rationality. This

kind of attacker plays the best possible move according to the declared defender strategy (see Section 2.1.1).

- Subjective Utility Quantal Response (SUQR) attackers: this attacker profile models a bounded rationality attacker which brings it to overweight (or underweight) the coverage probabilies and the targets values. This kind of model has shown to be effective for describing human behaviour (see Section 2.3.1).

Stackelberg Attacker

Given a strategy profile declaration $\boldsymbol{\sigma}_{D,n}$, a Stackelberg attacker Sta responds according to:

$$\boldsymbol{\sigma}_{Sta}(\boldsymbol{\sigma}) = \arg \max_{\boldsymbol{\sigma}' \in \Delta_M} \sum_{m \in \mathcal{M}} \sigma'_m \, v_m \, (1 - \sigma_m)$$

and defender best-responds to a Stackelberg attacker by committing to:

$$\boldsymbol{\sigma}^*_D(Sta) = \arg \min_{\boldsymbol{\sigma}' \in \Delta_M} \max_{\boldsymbol{\sigma} \in \Delta_M} \sum_{m \in \mathcal{M}} \sigma'_m \, v_m \, (1 - \sigma_m),$$

as reported in [5], where it is proved that, for 2-player zero-sum games, the optimal mixed strategy for the leader to commit to is equivalent to computing the minmax strategy, i.e., to minimize the maximum expected utility that the opponent can obtain.

SUQR Attacker

The SUQR attacker responds to the commitment $\boldsymbol{\sigma}_{D,n}$ as:

$$\sigma_{SUQR}(\boldsymbol{\sigma})_m = \frac{\exp\{-\alpha\sigma_m + \beta v_m\}}{\sum_{h=1}^{M} \exp\{-\alpha\sigma_h + \beta v_h\}},$$

where $\alpha \in \mathbb{R}^+$, $\beta \in \mathbb{R}$ are parameters known to the defender, characterizing the attacker and depending on the underlying application. In this case, we do not have a closed form for the best response, but we can compute the minmax solution to the problem following the steps taken in [32]. We will refer to $\boldsymbol{\sigma}^*_D(SUQR)$ as the best response to an attacker with a SUQR profile.

Unknown SUQR Attacker

As we have done for the Stochastic attacker, we will consider the profiles family of SUQR attackers with unknown parameters. In this case we have to use a different estimator, since we do not observe them directly that is the Maximum Likelihood Estimator (MLE). This estimator is consistent, but not always unbiased. The complexity of MLE grows linearly with the number of rounds, however we can also use an approximate minimization method (we minimize the negative log-likekihood, which is the same as maximizing the likelihood) as Stochastic Gradient Descent (SGD), which has a constant complexity instead. It can be shown that this likelihood function is a concave function, since the Hessian matrix is negative semi-definite, therefore the SGD converges to the optimum.

The best response to this kind of attacker is, as for the Stochastic one, the same of the *known* case, but using the estimated parameters.

3.3 An Example

Here it is introduced a real-life-inspired example scenario, which can be modeled using FBI-SG problem. We shall use this scenario in the following chapter's examples in order to better understand the analysed algorithms.

Example 1 (Example)**.** *In a small district, in the suburbs of a city, a gang started robbing local shops takings. Here are listed the four targeted shops with their average daily takings*[2]*:*

- *flower shop :* 300 €
- *drugstore:* 1500 €
- *cafè:* 500€
- *grocery:* 800€

There is only one patrol guarding the area, but the band plans carefully each attack, therefore if they see policemen guarding the chosen target, the attack is blocked, and they will not attack anymore until the next day.

[2]Targets' values have been inspired by data from *Agenzia delle Entrate*

Chapter 4

Proposed Solutions

Since the described problem is new, there are no dedicated solutions in literature. However we tried first to face the problem using general online learning techniques. As we shall see, traditional solutions fail in providing low regret, mainly because they can not take into account the *commitment* of the defender to a strategy, being *loss-based*. Our proposed techniques instead are able to exploit in an effective way all the possible information and are not deceived by the commitment.

4.1 Online Learning techniques

In order to properly apply these techniques we need first to specify what we will define as *arms* (or *experts*) and *feedback*. As we have seen in Section 3.2 for each profile we are able, at every round, to compute a best response strategy for the defender. Therefore we will take as arms the current *best responses to each profile*. For what it concerns the feedback, in our problem the defender knows at the end of each round which target has been attacked. This means that, based on what she played, the agent is able to compute the *incurred loss*. Furthermore she can also speculate about what she would have been the loss *having played another arm*, computing the *expected loss* for each profile's best response, given the attacker's move. Thus we define:

- *partial feedback*: the real loss received at t;
- *complete feedback*: the vector of the expected loss for each arm.

4.1.1 Follow the Perturbed Leader

Using the previous definitions, FPL algorithm (section 2.2.1) can be applied to the FBI-SG problem. This can be seen with our example setting:

Example 2 (FPL vs Stochastic attacker)**.** *Here we suppose to have observed the game defined in Example 1 for* 3 *time-steps and describe how to obtain the FPL strategy at* τ_4*. We are supposing a profile set* $\mathcal{A} = \{Sto_1, Sto_2\}$*, where the attackers have the following distributions:*

- $\boldsymbol{\sigma}_{Sto_1}$*:* $(0.3, 0.2, 0.1, 0.4)$
- $\boldsymbol{\sigma}_{Sto_2}$*:* $(0.5, 0.1, 0.3, 0.1)$

and the real attacker is Sto_1*. Having the distributions, then, it is possible to compute the best-responses:*

- $\boldsymbol{\sigma}^*_D(Sto_1) = (0, 0, 0, 1)$
- $\boldsymbol{\sigma}^*_D(Sto_2) = (1, 0, 0, 0)$

Table 4.1

	i	j	$f_{E_{Sto_1}}$	$f_{E_{Sto_2}}$
τ_1	t_4	t_1	t_4	t_1
τ_2	t_1	t_4	t_4	t_1
τ_3	t_4	t_4	t_4	t_1

Using the experts' predictions it is possible to compute $l(f_{E,t}, y_t)$ *for each time step:*

Table 4.2

	$l(f_{E_{Sto_1}}, y_\tau)$	$l(f_{E_{Sto_2}}, y_\tau)$	l_τ
τ_1	300	0	300
τ_2	0	800	800
τ_3	0	800	0

Therefore cumulative losses can be computed:

- $L_1 = \sum_{i=1}^{3} l_{Sto_1,i} = 300 + 0 + 0 = 300$
- $L_2 = \sum_{i=1}^{3} l_{Sto_2,i} = 0 + 800 + 800 = 1600$

A random perturbation sampled from $[0, \sqrt{6}]$ *is added to each* L*:*

$$L_1 = 300 + 2.4, \qquad L_2 = 1600 + 1.6$$

Thus the leader is Sto_1 *and the next defender strategies would be* $(0, 0, 0, 1)$*.*

However the previous example was not considering strategy-aware attackers, in the next example it is illustrated how the FPL algorithm can become problematic in this case:

Example 3 (FPL vs Stackelberg attacker)**.** *We now suppose a profile set* $\mathcal{A} = \{Sta, Sto\}$*, where* $\boldsymbol{\sigma}_{Sto} = (0.3, 0.2, 0.1, 0.4)$*, and the real attacker is* Sta*. The best-responses are:*

- $\boldsymbol{\sigma}_D^*(Sta) = (0, 0.65, 0, 0.35)$
- $\boldsymbol{\sigma}_D^*(Sto) = (0, 0, 0, 1)$

If at a certain time τ *the defender plays* $\boldsymbol{\sigma}_D^*(Sta)$*, attacker will respond with:*

$$\boldsymbol{\sigma}_{sta}(\boldsymbol{\sigma}_D^*(Sta)) = (0, 0, 0, 1)$$

With probability 0.65 *then:*

Table 4.3

	i	j	$f_{E_{Sta}}$	$f_{E_{Sto}}$	$l(f_{E_{Sta}}, y_\tau)$	$l(f_{E_{Sto_2}}, y_\tau)$	l_τ
τ	t_2	t_4	t_2	t_4	800	0	800

according to the algorithm, playing the right best response will cause to the Stackelberg expert a greater loss than playing the stochastic expert.

What we are missing is that we are not considering the defender's commitment to her strategy. Indeed, it is not possible for the defender to play a move without committing to a certain strategy, and this commitment will influence the attacker's move. We can express what we have understood with the intuition in a more formal way with the following theorem:

Theorem 1 (Expert Pseudo-regret upper bound)**.** *Let us consider an instance of the FBI-SG problem and apply the FPL algorithm, where each possible profile A_k is an expert and receives, at round n, an expert reward equal to minus the loss she would have incurred observing $i_{A_{k^*},n}$ by playing the best response to the attacker A_k. Then, there always exists an attacker set $\mathcal{A}$ s.t. the defender incurs in an expected pseudo-regret of:*

$$R_N(\mathfrak{U}) \propto \Delta L_k \, N.$$

Proof. Let us analyse the FBI-SG problem in which the attacker profile set is $\mathcal{A} = \{Sta, Sto\}$, the true attacker $A_{k^*} = Sta$ and we use the Follow the Leader algorithm [4]. Assume that the best response $\boldsymbol{\sigma}^*_D(Sto)$ to the stochastic attacker Sto corresponds to the pure strategy played by the Stackelberg attacker at the equilibrium, i.e, $\boldsymbol{\sigma}^*_{Sta}(\boldsymbol{\sigma}^*_D(Sta)) = \boldsymbol{\sigma}^*_D(Sto)$. Assume the chosen target by the two strategies has value $v_{\hat{m}}$ in target $\hat{m}$, maximum value $v_{\bar{m}}$ in target $\bar{m}$ and that the stochastic attacker has strategy $\mathbf{p}$ s.t.:

$$p_m = \begin{cases} \alpha & \text{if } m = \hat{m} \\ 1-\alpha & \text{if } m = \bar{m} \\ 0 & \text{otherwise} \end{cases},$$

where $\alpha = \frac{v_{\bar{m}} - L(Sta)}{v_{\bar{m}}}$ and $\alpha v_{\bar{m}} > (1-\alpha) v_{\bar{m}}$. In this case, the defender might commit to two different strategies:

- if the defender D declares its best response to the Stackerlberg attacker $\boldsymbol{\sigma}^*_D(Sta)$ for the turn, it would provide zero loss as feedback for the stochastic attacker expert and loss equal to $-L(Sta)$ to the Stackelberg one
- if the defender D selects the best response to the stochastic attacker $\boldsymbol{\sigma}^*_D(Sto)$, the defender would gain loss equal to $-(1-\alpha)v_{\bar{m}} = -L(Sta)$ for the stochastic attacker expert and $-L(Sta)$ for the Stackelberg one. Thus, in this case the two types would receive the same feedback.

Summarizing, we have that the Stackelberg attacker expert always incurs in a loss greater or equal to the one of the stochastic one, even if the real attacker is Stackelberg. Thus, with a probability grater than 0.5 we are incurring in a loss of ΔL_k for the entire horizon, with a total regret proportional to $\Delta L_k N$. Even by resorting to randomization, thus even adopting

the FPL we would have a probability of at least $0.5 - \varepsilon$ (being ε the probability with which the FPL chooses a suboptimal option) to select the wrong option, thus also the FPL algorithm would incur in a linear regret over the time horizon.

□

4.1.2 Multi-Armed Bandit Algorithms

As we have seen, expert algorithms have a disadvantage in complete feedback, because they use this information in the wrong way. Therefore we need to compare also with MAB algorithms, that use only the information of the played arm, which cannot be source of errors. The feedback of MAB algorithms is partial, therefore they do not risk to make the mistake of expert algorithms, i.e. assigning wrong loss to non-played arms. There are many MAB algorithms, but we will focus particularly on:

- UCB1
- Thompson Sampling

These are two of the most known algorithms in the field, and in particular they represent solutions for the MAB setting belonging to the Frequentist and Bayesian frameworks, respectively. We recall that for both the algorithms there exist upper bounds to the pseudo regret:

Theorem 2 (UCB1 Pseudo-regret upper bound)**.** *Let us consider an instance of the FBI-SG problem and apply the UCB1 algorithm, where each possible behavioural profile $A_k \in \mathcal{A}$ is an arm which receives reward $-l_n$ if played. Then, we incur in the following pseudo-regret:*

$$R_N(\mathfrak{U}) \leq 8 \sum_{k \neq k^*} \frac{\ln N}{(\Delta L_k)} + \left(1 + \frac{\pi^2}{3}\right) \sum_{k \neq k^*} \Delta L_k,$$

where $\Delta L_k = \sum_{m=1}^{M} \sigma_{A_{k^*}}(\boldsymbol{\sigma}_D^*(A_k))_m \, v_m \, (1 - \sigma_D^*(A_k)_m) - L^*$ is the expected regret of playing the best response to attacker A_k when the real attacker profile is A_{k^*}.

Theorem 4.1.1 (Thompson Sampling Upper Bound)**.** *At time T, the expected regret of Thompson Sampling applied to a stochastic MAB problem is:*

$$L_T \leq O\left(\sum_{k \neq k^*} \frac{\Delta L_k}{KL(\mathcal{R}_k, \mathcal{R}^*)}(\log T + \log\log T)\right). \tag{4.1}$$

Unfortunately, as seen in Section 2.2.2, when using MAB algorithms, we are subject to a *lower bound*:

Theorem 4.1.2 (MAB Lower Bound)**.** *Given a MAB stochastic problem any algorithm satisfies:*

$$\lim_{t \to \infty} L_t \geq \log t \sum_{k \neq k^*} \frac{\Delta L_k}{KL(\mathcal{R}_k, \mathcal{R}^*)}. \tag{4.2}$$

4.1.3 Considerations

By adopting these online learning techniques we get theoretical results that might be loose w.r.t. the problem lower bound. In particular, using expert techniques we sistematically learn in the wrong way against strategy-aware attackers, while, using MAB techniques, we protect ourselves from these kind of mistakes, but we suffer the partial use of information through a loose *lower bound*. The main problem with these approaches seems to be the fact that they take into account only the received loss and not the observations themselves, which results in losing a valuable part of the information. In the next section we will introduce two algorithms that will try to exploit as much as possible the received feedback to enhance the *belief* of the defender on a particular profile.

4.2 Belief-Based Algorithms

The concept of *belief* arises many times in the Artificial Intelligence context as a way to overcome *uncertainty*. In our case the belief will be the key to face the identification problem. The following algorithms will keep a belief vector for the attacker profiles, representing how much we are confident, at a specific round of the game, that one of the profile is the true one.

Recalling Section 3.1, at each interaction the defender commits to a strategy $\boldsymbol{\sigma}_D$, to which an attacker of profile A_k *would* respond with $\boldsymbol{\sigma}_{A_k}(\boldsymbol{\sigma}_D)$.

Denoting the move of the attacker with i, then we define the likelihood of target i with defender commitment to $\boldsymbol{\sigma}_D$ as:

$$B_k(i, \boldsymbol{\sigma}_D) := \sigma_{A_k}(\sigma_D)_i. \tag{4.3}$$

This is the *likelihood of the current action* if A_k was the true profile. Since we know the models of the possible profiles, if we have a *sequence of interactions*, defining $\mathbf{i}$ as the observations and $\mathbf{s}$ as the defender's strategies sequence played over time, then for each profile A_k, the *likelihood of the sequence* is:

$$\Lambda_k(\mathbf{i}, \mathbf{s}) := \prod_{j=1}^{|\mathbf{i}|} B_k(i_j, s_j). \tag{4.4}$$

When we are simply considering the real sequence of interactions until the n-th round made of $\mathbf{i}^{(n)} = \{i_{A_{k^*},1} \dots i_{A_{k^*},n}\}$ and $\mathbf{s}^{(n)} = \{\boldsymbol{\sigma}_{D,1} \dots \boldsymbol{\sigma}_{D,n}\}$ we will use the shortened notation $\Lambda_k^{(n)} := \Lambda_k(\mathbf{i}^{(n)}, \mathbf{s}^{(n)})$. Of course, for the unknown profiles we do not have the exact likelihood of their profile, but an estimate instead. Providing that the estimator is *consistent* then the likelihood is guaranteed to converge to the real one in the limit of the number of samples. Since at each round we compute new estimates for unknown profiles parameters, then we have also to recompute the likelihoods of all the previous actions to obtain the exact likelihood of the sequence, while for the known profiles it suffices to multiply the last likelihood in order to update the likelihood of the sequence. In the following pages, when we talk about *belief* of an attacker profile, we will always refer to Λ_k (or to some variations, as its logarithm, or a normalized version of it).

However the identification process alone, does not guarantee to accomplish our *regret minimization* goal. Indeed a series of strategies that speeds up the identification process can also cause a greater regret than a series of greedy strategies. This issue perfectly embodies the classic *exploration-exploitation tradeoff* problem of online learning seen in Section 2.2, which we have tried to tackle with the algorithms presented in the following pages.

4.2.1 Follow the Belief

The `FB` algorithm represents a greedy employment of the information contained in the belief vector. The idea is to take the profile that has the largest belief and best respond to it. Despite its semplicity, we can prove

Algorithm 5 FB

1: $\mathcal{P} = \mathcal{A}$
2: **for all** $A' \in \mathcal{P}$ **do**
3: Initialize $\Lambda_k^{(1)} = 1$
4: **for all** $n \in \{1, \dots, N\}$ **do**
5: Select $A_{k_n} = \arg\max_{A_k \in \mathcal{P}} \Lambda_k^{(n)}$
6: Play $\boldsymbol{\sigma}_D^*(A_{k_n})$
7: Observe attacker action $i_{A_{k^*},n}$
8: **for all** $A_k \in \mathcal{P}$ **do**
9: **if** $\sigma_{A_k}(\boldsymbol{\sigma}_D^*(A_{k_n}))_{i_{A_{k^*},n}} = 0$ **then**
10: $\mathcal{P} \leftarrow \mathcal{P} \setminus A_k$
11: **else**
12: Update $\Lambda_k^{(n+1)}$ with (4.4)

a strong theoretical bound for it that makes it overperform the traditional online learning algorithms. Moreover, it can be easily extended to support new attacker profiles, without much effort, as long as we are provided with a belief for each target given the commitment. Algorithm 5 presents the pseudocode of the FB algorithm. At the beginning, FB initializes a set of active attackers $\mathcal{P} = \mathcal{A}$ and a belief $\Lambda_k^{(1)} = 1$ for all the attacker profiles $A_k \in \mathcal{P}$. At each round n, the algorithm selects the attacker A_{k_n} for which the belief is the largest one (Line 5, where ties are broken *randomly*), best responds with the strategy $\boldsymbol{\sigma}_D^*(A_{k_n})$ and observes the action actually played by the attacker $i_{A_{k^*},n}$. After that, the belief is updated as in (4.4) at round n (Line 12) and if the realization $i_{A_{k^*},n}$ is not consistent for attacker A_k (zero likelihood), the profile A_k is removed from $\mathcal{P}$ (Line 10).

We can upper bound the regret of FB algorithm as stated by the following theorem:

Theorem 3 (FB pseudo-regret upper bound)**.** *Given an instance of the FBI-SG problem s.t.* $\Delta b_k > 0$*, with all* known *profiles in* $\mathcal{A}$*, for each* $A_k \in \mathcal{A}$ *and applying* FB*, the defender incurs in a pseudo-regret of:*

$$R_N(\mathfrak{U}) \leq \sum_{k=1}^{K} \frac{2(\lambda_k^2 + \lambda_{k^*}^2)\Delta L_k}{(\Delta b_k)^2},$$

where

$$\lambda_k := \max_{m\in\mathcal{M}} \max_{\boldsymbol{\sigma}\in\mathcal{S}} \ln(\sigma_{A_k}(\boldsymbol{\sigma})_m) - \min_{m\in\mathcal{M}} \min_{\boldsymbol{\sigma}\in\mathcal{S}} \ln(\sigma_{A_k}(\boldsymbol{\sigma})_m)\mathcal{I}\left\{\sigma_{A_k}(\boldsymbol{\sigma})_m \neq 0\right\} \tag{4.5}$$

is the range where the logarithm of the beliefs realizations lies (excluding realizations equal to zero, which end the exploration of a profile) and

$$\mathcal{S} := \cup_k \boldsymbol{\sigma}^*_D(A_k) \tag{4.6}$$

is the set of the available best responses to the attacker's profile.

Proof. Let us analyse the regret of the `FB` algorithm. We get some regret if the algorithm selects a strategy profile corresponding to a type different from the real one. Thus, the regret is upper bounded by :

$$\begin{aligned} R_N(\mathfrak{U}) &= \mathbb{E}\left[\sum_{n=1}^{N} l_n\right] - L^* N \\ &= \mathbb{E}\left[\sum_{n=1}^{N} l_n\right] - L^* N \\ &= \mathbb{E}\left[\sum_{n=1}^{N} l_n - L^*\right] = \sum_{k=1}^{K} \Delta L_k \mathbb{E}[T_k(N)], \end{aligned}$$

where we recall that:

- $T_k(N) = \sum_{n=1}^{N} \mathcal{I}\{A_{k_n} = A_k\}$ is the number of times we played the best response $\boldsymbol{\sigma}^*_D(A_k)$ to attacker A_k;
- $\Delta L_k = \sum_{m=1}^{M} \sigma_A(\boldsymbol{\sigma}^*_D(A_k))_m v_m(1-\sigma^*_D(A_k)_m) - L^*$ is the expected regret of playing the best response to attacker A_k when the real attacker is A.

Each round in which the algorithm selects a profile s.t. the best response is not equal to the one of A_{k^*} we are getting some regret.

Let us define variables $B_{k,n}$ and $B_{k^*,n}$ denoting the belief we have for the possible attacker A_k and of the real attacker A, respectively, of the action played by the real attacker A at turn n. Moreover, let $b_{kj,t} := \mathbb{E}_{\sigma^*_D(A_j)}[B_{k,t}]$ be the expected value of the belief we get for attacker A_k when we are best responding to A_j and the true type is $A_{k^*} \neq A_k$ at round t. Note that $b_{kj,t} < b_{k^*j,t}, \forall j$, since Δb_k is positive.

For each profile $A_k \neq A_{k^*}$, we have:

$$\mathbb{E}[T_k(N)] \leq \sum_{n=1}^{N} \mathbb{E}\left[\mathcal{I}\left\{\prod_{t=1}^{n} B_{k,t} \geq \prod_{t=1}^{n} B_{k^*,t}\right\}\right] \tag{4.7}$$

$$\leq \sum_{n=1}^{N} \mathbb{E}\left[\mathcal{I}\left\{\sum_{t=1}^{n} \ln(B_{k,t}) \geq \sum_{t=1}^{n} \ln(B_{k^*,t})\right\}\right] \tag{4.8}$$

$$= \sum_{n=1}^{N} \mathbb{P}\left(\frac{\sum_{t=1}^{n} \ln(B_{k,t})}{n} \geq \frac{\sum_{t=1}^{n} \ln(B_{k^*,t})}{n}\right) \tag{4.9}$$

$$= \sum_{n=1}^{N} \mathbb{P}\left(\frac{\sum_{t=1}^{n} \ln(B_{k,t})}{n} - \frac{\sum_{t=1}^{n} \ln(b_{kj_t,t})}{n} - \frac{\sum_{t=1}^{n} \ln(B_{k^*,t})}{n} + \right.$$
$$\left. + \frac{\sum_{t=1}^{n} \ln(b_{k^*j_t,t})}{n} \geq \underbrace{\left(\frac{\sum_{t=1}^{n} \ln(b_{k^*j_t,t})}{n} - \frac{\sum_{t=1}^{n} \ln(b_{kj_t,t})}{n}\right)}_{\geq \Delta b_k}\right) \tag{4.10}$$

$$\leq \sum_{n=1}^{N} \mathbb{P}\left(\frac{\sum_{t=1}^{n} \ln(B_{k,t})}{n} - \frac{\sum_{t=1}^{n} \ln(b_{kj_t,t})}{n} - \frac{\Delta b_k}{2} - \frac{\sum_{t=1}^{n} \ln(B_{k^*,t})}{n} + \right.$$
$$\left. + \frac{\sum_{t=1}^{n} \ln(b_{k^*j_t,t})}{n} - \frac{\Delta b_k}{2} \geq 0\right) \tag{4.11}$$

$$\leq \underbrace{\sum_{n=1}^{N} \mathbb{P}\left(\frac{\sum_{t=1}^{n} \ln(B_{k,t})}{n} \geq \frac{\sum_{t=1}^{n} \ln(b_{kj_t,t})}{n} + \frac{\Delta b_k}{2}\right)}_{R_1} +$$
$$+ \underbrace{\sum_{n=1}^{N} \mathbb{P}\left(\frac{\sum_{t=1}^{n} \ln(B_{k^*,t})}{n} \leq \frac{\sum_{t=1}^{n} \ln(b_{k^*j_t,t})}{n} - \frac{\Delta b_k}{2}\right)}_{R_2}, \tag{4.12}$$

where j_t is the index of the attacker A_{j_t} we selected at round t and we defined $\Delta b_k := \min_{j|A_j \in \mathcal{A}} \ln(b_{k^*j,t}) - \ln(b_{kj,t})$, i.e., the minimum w.r.t. the best response for the available attackers of the difference between the expected value of the loglikelihood of attacker A_{k^*} and A_k if the true profile is A_{k^*}. Equation (4.9) has been obtained from Equation (4.8) since $\mathbb{E}\left[\mathcal{I}\left\{\cdot\right\}\right] = \mathbb{P}\left(\cdot\right)$ while Equation (4.10) has been computed from Equation (4.9) adding $\left(\frac{\sum_{t=1}^{n} \ln(b_{k^*j_t,t})}{n} - \frac{\sum_{t=1}^{n} \ln(b_{kj_t,t})}{n}\right)$ to both l.h.s. and r.h.s. of the inequality. We

would like to point out that Δb_k does not depend on t since the distribution of $B_{k,t}$ and $B_{k^*,t}$ is the same over rounds.

Let us focus on R_1. We use the McDiarmid inequality (see Appendix A) to bound the probability that the empirical estimate of the loglikelihood expected value is higher than a certain upper bound as follows:

$$\begin{aligned} R_1 &= \sum_{n=1}^{N} \mathbb{P}\left(\frac{\sum_{t=1}^{n} \ln(B_{k,t})}{n} \geq \frac{\sum_{t=1}^{n} \ln(b_{kj_t,t})}{n} + \frac{\Delta b_k}{2} \right) \\ &\leq \sum_{n=1}^{\infty} \mathbb{P}\left(\frac{\sum_{t=1}^{n} \ln(B_{k,t})}{n} \geq \frac{\sum_{t=1}^{n} \ln(b_{kj_t,t})}{n} + \frac{\Delta b_k}{2} \right) \\ &\leq \sum_{n=1}^{\infty} \exp\left\{ -\frac{(\Delta b_k)^2 n}{2\lambda_k^2} \right\} \leq \frac{2\lambda_k^2}{(\Delta b_k)^2}, \end{aligned}$$

where we exploited $\sum_{x=1}^{\infty} e^{-\kappa x} \leq \frac{1}{\kappa}$ and we used the fact that $\mathbb{E}[B_{k,t}] = b_k \forall k, t$.

A similar reasoning can be applied to R_2 getting an upper bound of the following form:

$$R_2 \leq \frac{2\lambda_{k^*}^2}{(\Delta b_k)^2}.$$

The regret becomes:

$$\begin{aligned} R_N(\mathfrak{U}) = \sum_{i=1}^{K} \Delta L_k \mathbb{E}[T_k(N)] &\leq \sum_{i=1}^{K} \Delta L_k \left(\frac{2\lambda_k^2}{(\Delta b_k)^2} + \frac{2\lambda_{k^*}^2}{(\Delta b_k)^2} \right) \\ &\leq \sum_{i=1}^{K} \frac{2(\lambda_k^2 + \lambda_{k^*}^2)\Delta L_k}{(\Delta b_k)^2}, \end{aligned}$$

which concludes the proof.

□

In practice, this algorithm relies on the fact that the attacker's sequence of moves will always maximizes, in expectation, the likelihood Λ_{k*} of the true profile. In order to better understand the behaviour of `FB` algorithm, we apply it to the same situation described in Example 3:

Example 4 (FB vs Stackelberg attacker)**.** *As before we suppose a profile set* $M = \{Sta, Sto\}$*, where* $\boldsymbol{\sigma}_{Sto} = (0.3, 0.2, 0.1, 0.4)$*, and the real attacker is Sta. The best-responses are:*

- $\boldsymbol{\sigma}_D^*(Sta) = (0, 0.65, 0, 0.35)$
- $\boldsymbol{\sigma}_D^*(Sto) = (0, 0, 0, 1)$

If at a certain time τ the defender plays $\boldsymbol{\sigma}_D^*(Sta)$, attacker will respond with:

$$\boldsymbol{\sigma}_{sta}(\boldsymbol{\sigma}_D^*(Sta)) = (0, 0, 0, 1)$$

For each of the profiles, the defender has a belief on the current action, based on her commitment:

- $B_{Sta}(t_4, \sigma_D^*(Sta)) = 1$
- $B_{Sto}(t_4, \sigma_D^*(Sta)) = 1$

Therefore the defender updates the beliefs of the profiles:

- $\Lambda_{Sta}^{(\tau+1)} = \Lambda_{Sta}^{(\tau)} \cdot 1$
- $\Lambda_{Sto}^{(\tau+1)} = \Lambda_{Sto}^{(\tau)} \cdot 1$

In practice, if at a certain round the two beliefs are the same, then, while the defender plays this strategy, she can not gain information to help her choosing between the two profiles. However `FB` *algorithm* randomizes *strategies when they have the same belief. Thus if the algorithm, at a certain round τ', randomly chooses instead $\boldsymbol{\sigma}_D^*(Sto)$, then the attacker will respond with:*

$$\boldsymbol{\sigma}_{Sta}(\boldsymbol{\sigma}_D^*(Sto)) = (0, 1, 0, 0)$$

In other words, being a rational attacker, knowing that the defender guards only target t_4 (the grocery in our example), she will attack t_2 (the drugstore). Current action beliefs then become:

- $B_{Sta}(t_2, \sigma_D^*(Sto)) = 1$
- $B_{Sto}(t_2, \sigma_D^*(Sto)) = 0.2$

then, the defender updates the beliefs of the profiles:

- $\Lambda_{Sta}^{(\tau'+1)} = \Lambda_{Sta}^{(\tau')} \cdot 1$
- $\Lambda_{Sto}^{(\tau'+1)} = \Lambda_{Sto}^{(\tau')} \cdot 0.2$

Therefore, in this example, if the defender plays the wrong strategy once, then she will not play it anymore. In fact we are also able to compute the maximum attainable regret:

$$R_{max} = \Delta_{Sto} = 1500 - 521 = 979€$$

4.2.2 Follow the Regret

FB adopts the belief as discriminant factor to select the strategy profile to play in the next round. Conversely, in what follows, we provide an algorithm, FR, driven by a value iteration procedure that directly minimizes the expected regret over the remaining rounds $\{n+1,\dots,N\}$. In principle, one should perform the procedure until the last round N, but, for computational purposes, an approximate solution can be obtained by setting a maximum level of recursion $h_{\max}$ and carry on the optimization only on the rounds $\{n+1,\dots,\min\{n+h_{\max},N\}\}$.

Algorithm 6 FR($h_{\max}$)

1: **for all** $A_k \in \mathcal{A}$ **do**
2: Initialize $\Lambda_k^{(1)} = 1$
3: **for all** $n \in \{1,\dots,N\}$ **do**
4: $\hat{\mathbf{R}} = \texttt{RE}(1, \Lambda^{(n)}, (), (), \min(h_{\max}, N-n))$
5: Select A_{k_n} s.t. $k_n = \arg\min_t \hat{R}_t$
6: Play $\boldsymbol{\sigma}_D^*(A_{k_n})$
7: Observe attacker action $i_{A_{k^*},n}$
8: **for all** $A_k \in \mathcal{A}$ **do**
9: Update $\Lambda_k^{(n+1)}$ according to 4.4

The pseudo-code of the FR algorithm is presented in 6, which recursively exploits the subroutine 7. We define $\underline{\Lambda}$ as the normalized version of the belief vector Λ, in order to use it as a probability in expectation formulas we define $\underline{\Lambda}^{(n)}$ as:

$$\underline{\Lambda}_k^{(n)} = \frac{\Lambda_k^{(n)}}{\sum_{i\in\mathcal{A}} \Lambda_i^{(n)}}. \tag{4.13}$$

We have omitted the normalization steps in the pseudocode in order to leave the notation uncluttered. At first, the FR algorithm requires to initialize vectors $\Lambda^{(1)}$ and $\underline{\Lambda}^{(1)}$. At each round n, the algorithm computes the estimated expected regret vector $\hat{\mathbf{R}}$ suffered by D if she plays the best response $\boldsymbol{\sigma}_D^*(A_k)$ to A_k for each attacker profile $A_k \in \mathcal{A}$ (Line 4, Alg. 6), by recursively calling the *Regret Estimator* (RE) algorithm. This algorithm returns the vector $\hat{\mathbf{R}}$ of the estimated total regrets of choosing a specific attacker A_k for the next

turn, computed from the point of view of a player who is this situation:

- has observed a sequence $(\mathbf{j}^{(n)}, \mathbf{j}')$ of attacker's actions;
- has committed to the sequence $(\mathbf{s}^{(n)}, \mathbf{s}')$ of strategies;
- has sampled the sequence $(\mathbf{i}^{(n)}, \mathbf{i}')$ of actions.

Algorithm 7 $\texttt{RE}(h, \Lambda, \mathbf{i}', \mathbf{j}', \mathbf{s}', h_{\max})$

1: **for all** $A_k \in \mathcal{A}$ **do**
2: $\quad \mathbf{s}'' \leftarrow (\mathbf{s}', \boldsymbol{\sigma}_D^*(A_k))$
3: $\quad$ **for all** $(i, j) \in \mathcal{M}^2$ **do**
4: $\quad\quad \mathbf{i}'' \leftarrow (\mathbf{i}', i)$
5: $\quad\quad \mathbf{j}'' \leftarrow (\mathbf{j}', j)$
6: $\quad\quad$ **for all** $A_t \in \mathcal{A}$ **do**
7: $\quad\quad\quad \hat{\Lambda}_t \leftarrow \Lambda_t((\mathbf{i}^{(n)}, \mathbf{i}''), (\mathbf{s}^{(n)}, \mathbf{s}''))$ according to 4.4
8: $\quad\quad$ **if** $h < h_{\max}$ **then**
9: $\quad\quad\quad \tilde{\mathbf{R}} = \texttt{RE}(h+1, \hat{\Lambda}, \mathbf{i}'', \mathbf{j}'', \mathbf{s}'', h_{\max})$
10: $\quad\quad\quad r_{ij,k} \leftarrow \min_k \tilde{R}_k$
11: $\quad\quad$ **else**
12: $\quad\quad\quad$ Compute $r_{ij,k}$ according to 4.14
13: $\quad$ Compute $\hat{R}_k$ according to 4.15
14: Return $\hat{\mathbf{R}}$

For every possible attacker $A_k \in \mathcal{A}$ and for every pair of possible actions of the defender and the attacker $(i, j) \in \mathcal{M}^2$, we create a new belief vector $\hat{\Lambda}$ and update it according to the information the attacker played action j (Line 7, Alg. 7). After that, if the maximum recursion level has been reached, we compute $r_{ij,k}$, as follows:

$$r_{ij,k} = \sum_{\tau=0}^{h_{max}} v_{j'_\tau} \mathbb{1}\{i'_\tau \neq j'_\tau\} - h_{max} \cdot \sum_{t \in \{1,\dots,K\}} \underline{\hat{\Lambda}}_{\,t} \, L(A_t). \tag{4.14}$$

In practice, if $h = h_{max}$, $r_{ij,k}$ represents the expected pseudo-regret of the sequences $\mathbf{i}'$ and $\mathbf{j}'$ over the possible attacker's profiles. Otherwise we run another step of RE and we assign to $r_{ij,k}$ the minimum value among the ones

returned by this step. Then in order to obtain $\hat{\mathbf{R}}$ we compute (Line 13, Alg. 7):

$$\hat{R}_k := \sum_{i=1}^{M}\sum_{j=1}^{M} r_{ij,k} \underbrace{\sigma^*_D(A_k)_i}_{p_i} \cdot \underbrace{\sum_{A_{k'}\in\mathcal{A}} \underline{\Lambda}_{\ k'}\, \sigma_{A_{k'}}(\boldsymbol{\sigma}^*_D(A_k))_j}_{p_j}, \tag{4.15}$$

where the regret $r_{ij,k}$ is weighted with the probabilities that action i is selected by D and action j is selected by A.

The defender D plays, for the current round n, the best response to the attacker A_{k_n}, providing the minimum estimated expected regret $\hat{R}_{k_n}$ (Line 6, Alg. 6) and observing action $i_{A_{k^*},n}$ undertaken by the attacker A_{k^*}. Finally, the algorithm updates the beliefs (Line 9, Alg. 6) as usual.

4.2.3 Computational Complexity

We now analyse our algorithms from a computational perspective, providing the computational cost required for being executed. `FB` has complexity $O(KN)$, since it only performs an update of the belief for each of the K attacker profiles, repeating this operation for each one of the N rounds. Thus, it results being linear both in the number of profiles and the rounds the game is played. Conversely, `FR` requires much more computational time. In fact, for each attacker profile K, we consider M actions for both players and then update the expected regret over the K profiles current beliefs. This leads to a cost of $O(M^2K^2)$ for a single round of the game and an overall computational cost of $O(M^2K^2N)$ over the problem horizon N, in the case we set $h = 1$. If we want to employ the strategy that considers the regret from the current round n to the end of the horizon (i.e., $h = N - n$) to compute the estimated expected regret $\hat{R}_n(A_k)$ by means of a forward procedure, the computational cost required by `FR` is $O(M^{2(N-n)}K^{2(N-n)})$ for a single round. Thus, the final computational cost required by `FR` is $\sum_{n=1}^{N} O(M^{2(N-n)}K^{2(N-n)}) = O\left(\frac{(MK)^{2N}-1}{(MK)^2-1}\right) \approx O(M^{2N}K^{2N})$, which is exponential.

Chapter 5

Experiments

In this chapter we compare the proposed algorithms, `FB` and `FR`, with the state-of-the-art online learning approaches from the MAB (Section 2.2.2) and expert (Section 2.2.1) fields. In particular, we evaluate UCB1 and Thompson Sampling (both described in Section 2.2.2), from the MAB literature, and Follow the Perturbed Leader (FPL) algorithm (Section 2.2.1), from the expert literature. In particular, we want to verify how the algorithms behave in these situations:

- the attacker is randomly chosen from a give profile set A of *known* attackers;
- the attacker is fixed and we vary the profile set $\mathcal{A}$;
- the attacker is randomly chosen from a *family of profiles* (unknown profiles or stackelberg).

We use a time horizon of $N = 1000$ rounds, with a different amount of targets $M \in \{5, 10\}$ and we evaluate the performance in terms of expected pseudo-regret:

$$R_n(\mathfrak{U}) = \mathbb{E}\left[\sum_{h=1}^{n} l_h\right] - L^* n,$$

with $n \in \{1, \dots, N\}$, and computational time spent by the algorithms to execute a single run ($N = 1000$ rounds). We selected the parameters of the stochastic and SUQR attackers by following the same procedure in the various settings:

- The strategies of the stochastic behavioural profiles *Sto* are drawn from a Dirichlet distribution with $\boldsymbol{\theta} = \mathbf{1}_M$ (uniform distribution over Δ_M) and the target values $\mathbf{v}$ are uniformly sampled in $[0, 1]^M$;
- The parameters for the SUQR behavioural profiles are drawn from a uniform probability distribution over the intervals $\alpha \in [5, 15]$ and $\beta \in [0, 1]$, whose choice is motivated by the experimental results obtained by [18];

  Each experiment has been run on a *Intel(R) Xeon(R) CPU E5-4610* processor running at 2.30GHz. The code has been developed in Python with a broad use of the SciPy library [10].

5.1 Studying Attacker's Behavior

Table 5.1: Sets of attacker's profiles $\mathcal{A}$ used for the experiments and total number of attackers K. We report also the number of different stochastic, SUQR, and unknown stochastic behavioural profiles for each configuration. The configurations are ordered from the ones with smallest number of behavioural profiles ($K = 2$) to the largest one ($K = 11$).

	Sta	*Sto*	*SUQR*	K
C_1	1	1	-	2
C_2	1	-	1	2
C_3	1	1	1	3
C_4	1	5	-	6
C_5	1	-	5	6
C_6	1	5	5	11

The experimental setting is as follows:

- we use different profile configurations C_i, listed in 5.1;
- for each combination of behavioural profiles and targets size, 10 random configurations (i.e., target values $\mathbf{v}$ and attacker profile sets $\mathcal{A}$) are generated and the actual behavioural profile A_{k^*} is drawn from a uniform probability distribution over the given profiles set $\mathcal{A}$;

- for each configuration we run 100 independent experiments and we compute the average regret.

5.1.1 Regret Analysis

We report in Tables 5.2 and 5.3 the empiric pseudo-regret obtained in the experimental results. It can be observed that the algorithms we propose dramatically outperform the baselines provided by the state of the art. Furthermore, there is no strong statistical evidence that one algorithm between `FB` or `FR` outperforms the other. We recall that `FR` is more computationally demanding than `FB`, thus one might prefer `FB` for problems with many attacker behavioural profiles, since it has comparable performance w.r.t. `FR` and is computationally more efficient.

Table 5.2: Expected pseudo-regret $R_N(\mathfrak{U})$ over 1000 rounds with confidence intervals for configurations C_1, C_2, C_3.

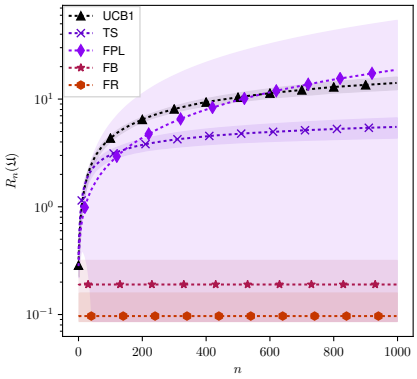
		C_1	C_2	C_3
$M = 5$	UCB1	14.12 ± 1.88	8.62 ± 3.73	23.92 ± 5.23
	FPL	18.71 ± 35.02	11.16 ± 5.98	38.5 ± 27.18
	`FB`	$\mathbf{0.19 \pm 0.13}$	$\mathbf{0.2 \pm 0.18}$	$\mathbf{0.5 \pm 0.24}$
	`FR`	$\mathbf{0.1 \pm 0.06}$	$\mathbf{0.27 \pm 0.36}$	$\mathbf{0.42 \pm 0.3}$
$M = 10$	UCB1	16.77 ± 1.2	5.24 ± 2.79	21.2 ± 3.76
	FPL	1.08 ± 0.2	5.97 ± 3.5	12.06 ± 4.31
	`FB`	$\mathbf{0.13 \pm 0.03}$	$\mathbf{0.1 \pm 0.02}$	$\mathbf{0.33 \pm 0.16}$
	`FR`	$\mathbf{0.06 \pm 0.05}$	$\mathbf{0.12 \pm 0.21}$	$\mathbf{0.21 \pm 0.12}$

In Figure 5.1 we show how the pseudo-regret $R_n(\mathfrak{U})$ evolves during the time horizon in C_1, C_2 and C_3. The plots are in a semilogarithmic scale for a better comprehension. In all the presented configurations there is statistical significance that the `FB` and `FR` algorithms outperform the baselines on average since the confidence intervals do not overlap after the first ≈ 50 rounds. As expected, configurations with more profiles are more difficult for all the algorithms, except for configuration C_5. It can be speculated that in a configuration with only strategy-aware profiles it is easier to identify the real attacker, due to the fact that defender's different commitments induce greater differences in strategy-aware likelihoods. Conversely, a larger

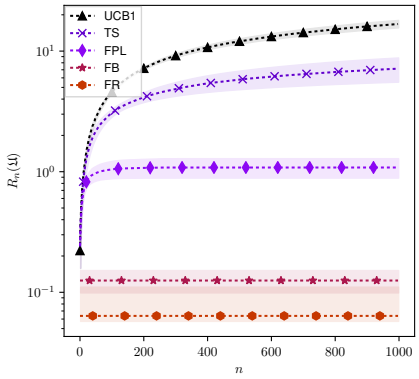
number of targets does not seem to impact on the performance of the algorithms: this means that the algorithms can be scaled on real-world many targets context without obtaining degrading performances. Notably, the FPL algorithm generally improves its performance when tested over larger target space $M = 10$. We think this could be induced by the experimental setting itself: indeed specific configurations in which the FPL gets linear regret (i.e., the ones considered in 1) are less likely to occur when we have a larger amount of targets.

Table 5.3: Expected pseudo-regret $R_N(\mathfrak{U})$ over 1000 rounds with confidence intervals for configurations C_4, C_5, C_6.

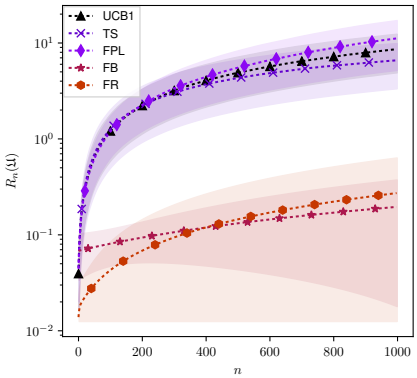
		C_4	C_5	C_6
$M = 5$	UCB1	45.75 ± 11.68	1.76 ± 0.41	75.82 ± 19.94
	FPL	49.8 ± 62.33	0.77 ± 0.12	68.88 ± 64.13
	FB	$\mathbf{0.48 \pm 0.2}$	$\mathbf{0.09 \pm 0.03}$	$\mathbf{0.67 \pm 0.2}$
	FR	$\mathbf{0.62 \pm 0.24}$	$\mathbf{0.07 \pm 0.04}$	$\mathbf{1.07 \pm 1.1}$
$M = 10$	UCB1	60.58 ± 8.89	4.24 ± 5.02	61.52 ± 22.48
	FPL	2.63 ± 0.99	3.24 ± 3.96	17.69 ± 16.03
	FB	$\mathbf{0.57 \pm 0.17}$	$\mathbf{0.05 \pm 0.01}$	$\mathbf{0.58 \pm 0.14}$
	FR	$\mathbf{0.43 \pm 0.19}$	$\mathbf{0.02 \pm 0.02}$	$\mathbf{0.6 \pm 0.43}$



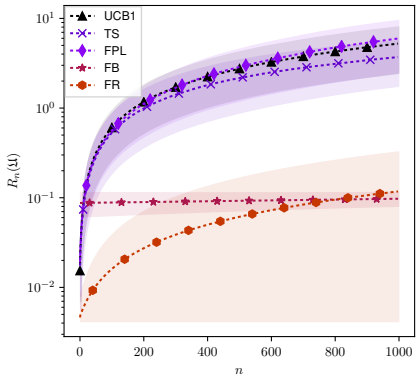
(a) Configuration C_1, $M = 5$.



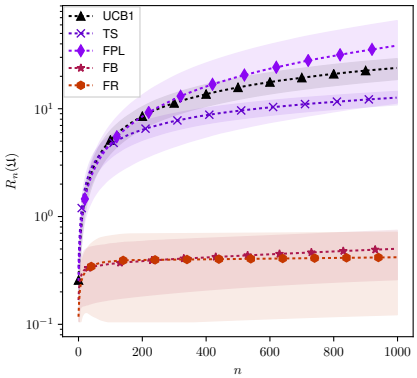
(b) Configuration C_1, $M = 10$.



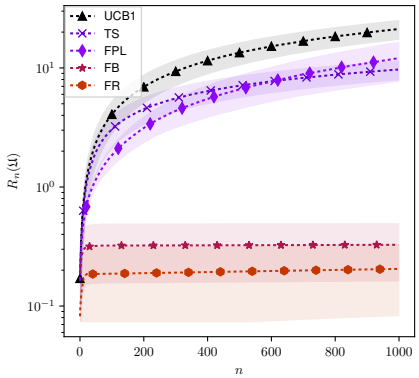
(c) Configuration C_2, $M = 5$.



(d) Configuration C_2, $M = 10$.

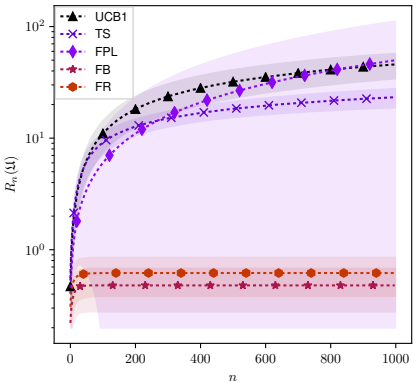


(e) Configuration C_3, $M = 5$.

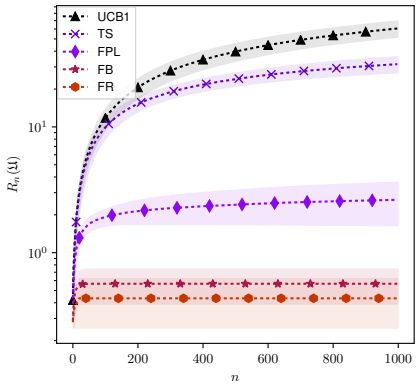


(f) Configuration C_3, $M = 10$.

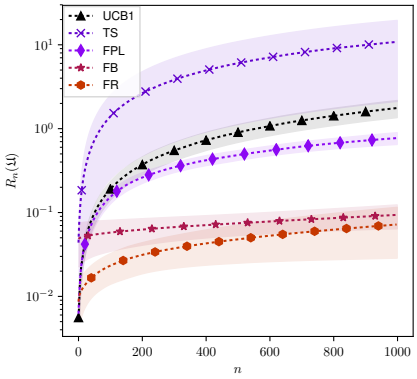
Figure 5.1: Expected pseudo-regret for the different configurations.



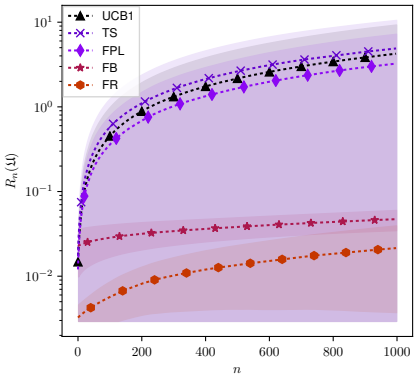
(a) Configuration C_4, $M = 5$.



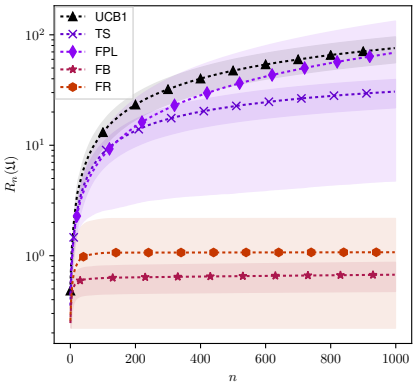
(b) Configuration C_4, $M = 10$.



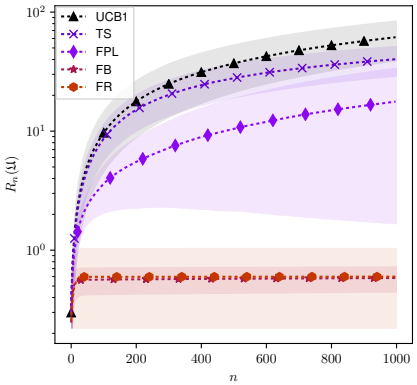
(c) Configuration C_5, $M = 5$.



(d) Configuration C_5, $M = 10$.



(e) Configuration C_6, $M = 5$.



(f) Configuration C_6, $M = 10$.

Figure 5.2: Expected pseudo-regret for the different configurations.

5.1.2 Time Performance

We analyse the computational effort required by our algorithms to solve instances over $N = 1000$ rounds. The computational times for the UCB1 and FPL algorithm are omitted since they are in line with the one of FB. The average computational times are reported in Table 5.4, where we group the results on the basis of the number of targets M. There are three observations we can make. First, as it can be immediately noticed, we could not report the values for $M \in \{20, 40\}$ for FR since the required computational cost is too high (≥ 3600 seconds). However, in real context, we may be asked to compute only one round of FR every day or week, depending on the context, therefore, in principle, we can use it also for more than 10 targets. We noticed that, as explained in Section 4.2.3, time needed to run even a single round increases exponentially with the number of targets, and therefore this algorithm presents limits in its applicability. Second, both FB and FR present the same trend w.r.t. the configurations: in fact, when the behavioral profile of the opponent can only be either *Sta* or *Sto*, both algorithms are twice more efficient than in cases in which SUQR adversaries are introduced. This is due to the fact that both *Sta* and SUQR models exploit the strategy the defender commits to, making more difficult to distinguish among them. Finally, as expected, we notice that FB is *always* faster than FR: in fact, while they are both polynomial in the actions available to the players, i.e.,

		C_1	C_2	C_3	C_4	C_5	C_6
$M = 5$	FB	5.9 ± 1.7	11.1 ± 2.2	11.7 ± 2.9	3.5 ± 1.0	23.7 ± 2.4	14.9 ± 4.3
	FR	77.0 ± 2.1	121.1 ± 3.2	170.4 ± 4.1	146.2 ± 4.7	651.7 ± 36.6	1029.2 ± 64.7
$M = 10$	FB	10.3 ± 2.6	21.9 ± 13.2	23.0 ± 17.9	7.1 ± 2.3	63.0 ± 7.4	47.22 ± 14.05
	FR	356.1 ± 14.3	678.5 ± 15.9	887.0 ± 11.1	960.4 ± 13.0	4402.5 ± 14.2	7526.5 ± 189.9
$M = 20$	FB	33.5 ± 3.0	222.2 ± 126.9	137.8 ± 77.6	33.7 ± 1.2	484.5 ± 107.7	226.8 ± 45.3
	FR	–	–	–	–	–	–
$M = 40$	FB	104.5 ± 7.1	2061.5 ± 837.2	1412.0 ± 812.1	128.9 ± 16.5	2347.9 ± 1223.2	1634.2 ± 487.6
	FR	–	–	–	–	–	–

Table 5.4: Computational time in seconds needed by FB and FR to solve an instance over $N = 1000$ rounds.

the number of targets, the former is linear while the latter quadratic (since we set $h_{\max} = 1$).

5.2 Analysis of the Impact of Prior Information

Table 5.5: Configurations are grouped by the chosen real attacker type.

	D	Sta	Sto	$SUQR$	U_{Sto}	U_{SUQR}
Sto	D_1	1	2	1	-	-
	D_2	1	-	-	1	1
	D_3	1	1	-	-	-
	D_4	1	-	-	1	-
$SUQR$	D_5	1	1	2	-	-
	D_6	1	-	-	1	1
	D_7	1	-	1	-	-
	D_8	1	-	-	-	1

The experimental setting is as follows:

- we use different profile configurations D_i, listed in, with $M = 5$ targets;
- for each combination of behavioural profiles and targets size, 10 random configurations (i.e., target values $\mathbf{v}$ and attacker profile sets $\mathcal{A}$) are generated and the actual behavioural profile of the type in the first column is randomly drawn;
- for each configuration we run 10 independent experiments and we compute the average regret.

where we make 10 repetitions of the same experiment instead of 100, since computing the likelihood of the sequence for unknown profiles is more computationally demanding, as explained in Section 4.2. The main motivation of this setting is to verify how the hypothesis made for the profiles setting influences the performance of the algorithms. In particular what we want to do is to confront these configurations pairwise (namely (D_1, D_2), (D_3, D_4), (D_5, D_6), (D_7, D_8)). For each of these pairs, in the first configuration the

true attacker profile is among the known profiles, while in the second she is among the unknown ones. For what it concerns the baselines, from Table 5.6, FPL, as expected, obtains a low regret in the configurations in which the attacker is a stochastic one. This happens because this kind of profile is not strategy-aware and therefore the algorithm does not suffer from the problem explained in Theorem 1. In fact when it has to face a strategy-aware attacker as SUQR, it could incurrs in a tremendous regret (D_6). In general we can notice that the second configuration obtains always a larger regret. In particular, for our solution there is statistical evidence of this trend. This means that in a real context using the external information we have on the attacker profile (e.g, from domain experts or from historical data) may significatively improve the results in terms of regret. The configurations with the least regret values result to be the ones in which the choice is between Stackelberg and a single other profile. In practice it seems that when the choice is only between a completely rational attacker and a boundedly rational attacker, then the problem becomes easier for our algorithms.

Table 5.6: Expected pseudo-regret $R_N(\mathfrak{U})$ over 1000 rounds with confidence intervals for configurations D_i, where the attacker is SUQR.

		D_1	D_2	D_3	D_4
$M = 5$	UCB1	34.65 ± 9.76	29.08 ± 4.60	14.92 ± 2.63	17.32 ± 2.78
	TS	14.68 ± 3.59	16.69 ± 6.35	8.35 ± 3.21	9.99 ± 2.76
	FPL	0.99 ± 0.28	3.10 ± 1.65	0.75 ± 0.20	2.43 ± 0.82
	FB	0.74 ± 0.48	2.23 ± 1.02	0.22 ± 0.14	1.63 ± 0.44
	FR	0.61 ± 0.18	2.56 ± 1.59	0.11 ± 0.09	2.63 ± 1.21

		D_5	D_6	D_7	D_8
$M = 5$	UCB1	16.62 ± 3.08	20.26 ± 4.06	7.22 ± 4.01	7.3 ± 4.01
	TS	10.28 ± 3.05	12.82 ± 5.04	6.23 ± 3.59	8.07 ± 4.93
	FPL	2.75 ± 2.02	92.39 ± 42.69	0.19 ± 0.02	0.34 ± 0.09
	FB	0.16 ± 0.06	2.63 ± 0.84	0.03 ± 0.02	0.12 ± 0.08
	FR	0.18 ± 0.11	3.51 ± 1.32	0.06 ± 0.06	0.11 ± 0.08

5.3 Introducing Unknown Attacker's Profiles

Table 5.7: Sets of attacker profiles $\mathcal{A}$ used for the experiments and total number of attacker K.

	Sta	Sto	$SUQR$	U_{Sto}	SU_{SUQR}	K
E_1	1	0	0	1	1	3
E_2	1	5	5	1	1	13

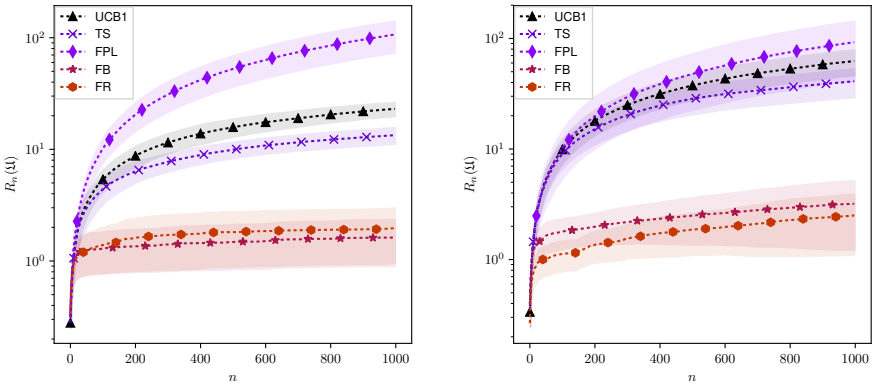
The experimental setting is as follows:

- we use different profile configurations E_i, listed in 5.7;
- for each combination of behavioural profiles and targets size, 10 random configurations (i.e., target values $\mathbf{v}$ and attacker profile sets $\mathcal{A}$) are generated and the actual behavioural profile A_{k^*} is drawn from a uniform probability distribution over the given profile set $\mathcal{A}$;
- for each configuration we run 10 independent experiments and we compute the average regret.

Configuration E_1 simulates a context in which the defender has *no prior information* about the attacker, so she may add to the profiles set only the unknown profiles and the Stackelberg one (for which no parameters are needed). Conversely configuration E_2 represents a situation in which the defender has some hypothesis on the nature of the attacker, but she can not exclude other behavioral profiles yet. Both these situations, which are likely to happen in real context, are not covered by our theoretical results (see Theorem3), hence need to be experimentally tested.

As we can se from Table 5.3, in both configurations our algorithms outperform the baselines. Configuration E_2 appears to be slightly more difficult, also for the two MAB algorithms. This induces us to think that including in the profile set a long list of known profiles, may damage the effectiveness of the algorithms, instead of improving it.

If we compare configurations from these setting to configurations of Section 5.1, we can see that the performance are in general worse. This is



(a) Configuration E_1, $M = 5$.

(b) Configuration E_2, $M = 5$.

Figure 5.3: Expected pseudo-regret for the different configurations.

clearly due to the fact that we do not known the exact best responses to the unknown profiles, but we have to learn them as the game develops. In practice we are learning at two levels: we are learning the parameters of the profiles and we are learning to identify the true attacker profiles.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this work, we study for the first time, a novel leadership game in which the leader plays against a follower whose behaviour is unknown, but it belongs to a set of known profiles. We tried to apply to it general online-learning algorithms, but they did not obtain satisfactory results. In fact, we proved that we incurrs in linear regret if we use the complete feedback with expert algorithms, while, using MAB algorithms we suffer the partial use of information through a loose *lower bound*, which forces at least a logarithmic regret. Such techniques, indeed, are based only on the observed loss and do not take into account the defender commitment to a strategy. Thus, we resort to a new approach that leverages on our knowledge of the possible attacker models (partial or complete). This consists in estimating the likelihoods of observed attacker actions in order to maintain a *belief* on each of the possible attacker profiles. We developed two algorithms based on this technique that exploit the aforementioned *identification* process in two different ways, in order to minimize the *regret*. They represent, in practice, two possible approaches to face the *exploration-exploitation dilemma* that arises in this problem. In the first one, named *Follow the Belief* (`FB`), the defender makes a greedy choice in terms of the belief, selecting the best response to the profile that maximizes it. We provide a finite-time analysis showing that

the regret of the algorithm is constant in the length of the time horizon, if all the profiles in the profile set are known. In the second approach, namely *Follow the Regret* (`FR`), the learning policy is driven directly by the estimated expected regret and is based on a backward induction procedure. Finally, we experimentally evaluate the performance of our algorithms in leadership settings inspired by concrete security domains, showing that our approaches provide a remarkable improvement in terms of empirical pseudo-regret minimization w.r.t. the main algorithms available in the state of the art of the online learning field.

6.2 Future work

There are many possible directions in which this first work on FBI-SG can be extended. First, we can try to extend the theoretical results of `FB` also for the unknown profiles or for other kinds of attacker not considered in this work as learner or adversarial ones. Another extension consists in generalizing the `FR` algorithm, using a policy-gradient algorithm to sample and evaluate random policies on the MDP generated by the game tree.

Another direction along which we can extend our work might be considering partial feedback instead of complete feedback: our algorithms are easily adaptable to this modification, but they need a proper theoretical analysis. We could also generalize the FBI-SG problem, allowing more resources for both players. In this case, we should adapt the current algorithms, comparing their performances w.r.t. the attackers'/defenders' resources, but also w.r.t. the level of coordination allowed for such resources.

Appendix A

McDiarmid Inequality

A *Doob martingale* is a generic construction that is always a martingale. Specifically, consider any set of random variables $\vec{X} = X_1, X_2, ..., X_n$ taking values in a set A for which we are interested in the function $f : A^n \to \mathbb{R}$ and define:

$$B_i = E_{X_{i+1}, X_{i+2}, ..., X_n}[f(\vec{X})|X_1, X_2, ...X_i]$$

where the above expectation is itself a random quantity since the expectation is only taken over $X_{i+1}, X_{i+2}, \ldots, X_n$ and $X_1, X_2, ...X_i$ are treated as random variables. It is possible to show that B_i is always a *martingale* regardless of the properties of X_i and the sequence B_i is the *Doob martingale* for f.

Suppose $X_1, \ldots, X_N$ are independent and assume that f satisfies for $1 \leq i \leq n$:

$$\sup_{x_1, x_2, ..., x_n, \hat{x}_i} |f(x_1, x_2, \ldots, x_n) - f(x_1, x_2, \ldots, x_{i-1}, \hat{x}_i, x_{i+1}, \ldots, x_n)| \leq c_i$$

In other words, replacing x_i by some other value changes the value of f by at most c_i. It follows that $|B_{i+1} - B_i| \leq c_i$ and therefore Azuma's inequality yields the following McDiarmid inequalities [15]: $\forall \epsilon > 0$

- $\Pr(f(X_1, X_2, \ldots, X_n) - E[f(X_1, X_2, \ldots, X_n)] \geq \epsilon) \leq \exp[-\frac{2\epsilon^2}{\sum_{i=1}^n c_i^2}]$
- $\Pr(E[f(X_1, X_2, \ldots, X_n)] - f(X_1, X_2, \ldots, X_n) \geq \epsilon) \leq \exp[-\frac{2\epsilon^2}{\sum_{i=1}^n c_i^2}]$

Bibliography

[1] Shipra Agrawal and Navin Goyal. Analysis of Thompson Sampling for the Multi-armed Bandit Problem. In *COLT*, pages 39–1, 2012.

[2] N. Basilico, G. De Nittis, and N. Gatti. Adversarial patrolling with spatially uncertain alarm signals. *ART INT*, 246:220–257, 2017.

[3] A. Blum, N. Haghtalab, and A. D. Procaccia. Learning to play Stackelberg security games. Technical report, Carnegie Mellon University, Computer Science Department, 2015.

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

[5] V. Conitzer and T. Sandholm. Computing the optimal strategy to commit to. In *EC*, pages 82–90, 2006.

[6] Vincent Conitzer and Tuomas Sandholm. Awesome: A general multi-agent learning algorithm that converges in self-play and learns a best response against stationary opponents. *MACH LEARN*, 67(1-2):23–43, 2007.

[7] F. Fang, P. Stone, and M. Tambe. When security games go green: designing defender strategies to prevent poaching and illegal fishing. In *IJCAI*, pages 2589–2595, 2015.

[8] Drew Fudenberg and Jean Tirole. *Game Theory*. MIT Press, 1991.

[9] James Hannan. Approximation to bayes risk in repeated play. *Contributions to the Theory of Games*, 3:97–139, 1957.

[10] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[11] Debarun Kar, Fei Fang, Francesco Delle Fave, Nicole Sintov, and Milind Tambe. A game of thrones: when human behavior models compete in repeated stackelberg security games. In *AAMAS*, pages 1381–1390. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

[12] Richard Klíma, Christopher Kiekintveld, and Viliam Lisỳ. Online learning methods for border patrol resource allocation. In *GameSec*, pages 340–349, 2014.

[13] Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

[14] M Maschler, Eilon Solan, and Shmuel Zamir. Game theory. translated from the hebrew by ziv hellman and edited by mike borns, 2013.

[15] Colin McDiarmid. On the method of bounded differences. *LOND MATH S*, 141(1):148–188, 1989.

[16] Daniel L McFadden. Quantal choice analaysis: A survey. In *Annals of Economic and Social Measurement, Volume 5, number 4*, pages 363–390. NBER, 1976.

[17] Richard D McKelvey and Thomas R Palfrey. Quantal response equilibria for normal form games. *Games and economic behavior*, 10(1):6–38, 1995.

[18] T. Nguyen, R. Yang, A. Azaria, S. Kraus, and M. Tambe. Analyzing the effectiveness of adversary modeling in security games. In *AAAI*, 2013.

[19] Thanh Hong Nguyen, Francesco Maria Delle Fave, Debarun Kar, Aravind S Lakshminarayanan, Amulya Yadav, Milind Tambe, Noa Agmon, Andrew J Plumptre, Margaret Driciru, Fred Wanyama, et al. Making the most of our regrets: Regret-based solutions to handle payoff uncertainty and elicitation in green security games. In *GameSec*, pages 170–191, 2015.

[20] J. Pita, M. Jain, C. Western, C. Portway, M. Tambe, F. Ordóñez, S. Kraus, and P. Paruchuri. Deployed ARMOR protection: The application of a game-theoretic model for security at the Los Angeles International Airport. In *AAMAS*, pages 125–132, 2008.

[21] James Pita, Milind Tambe, Chris Kiekintveld, Shane Cullen, and Erin Steigerwald. Guards: game theoretic security allocation on a national scale. In *AAMAS*, pages 37–44, 2011.

[22] Y. Qian, W. B. Haskell, A. X. Jiang, and M. Tambe. Online planning for optimal protector strategies in resource conservation games. In *AAMAS*, pages 733–740, 2014.

[23] Y. Qian, C. Zhang, B. Krishnamachari, and M. Tambe. Restless poachers: Handling exploration-exploitation tradeoffs in security domains. In *AAMAS*, pages 123–131, 2016.

[24] Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

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

[26] J. Tsai, S. Rathi, C. Kiekintveld, F. Ordóñez, and M. Tambe. IRIS-A tool for strategic security allocation in transportation networks. In *AAMAS*, pages 1327–1334, 2009.

[27] K. Tuyls and G. Weiss. Multiagent learning: Basics, challenges, and prospects. *AI MAG*, 33(3):41, 2012.

[28] Amos Tversky and Daniel Kahneman. Advances in prospect theory: Cumulative representation of uncertainty. *Journal of Risk and uncertainty*, 5(4):297–323, 1992.

[29] Heinrich Von Stackelberg. *Marktform und gleichgewicht*. J. Springer, 1934.

[30] H. Xu, L. Tran-Thanh, and N. R. Jennings. Playing repeated security games with no prior knowledge. In *AAMAS*, pages 104–112, 2016.

[31] R. Yang, B. Ford, M. Tambe, and A. Lemieux. Adaptive resource allocation for wildlife protection against illegal poachers. In *AAMAS*, pages 453–460, 2014.

[32] R. Yang, C. Kiekintveld, F. Ordóñez, M. Tambe, and R. John. Improving resource allocation strategy against human adversaries in security games. In *IJCAI*, pages 458–464, 2011.