Exploiting Opponents under Utility Constraints in Sequential Games

Martino Bernasconi-de-Luca*  
Politecnico di Milano  
martino.bernasconideluca@polimi.it

Federico Cacciamani*  
Politecnico di Milano  
federico.cacciamani@polimi.it

Simone Fioravanti  
Gran Sasso Science Institute  
simone.fioravanti@gssi.it

Nicola Gatti  
Politecnico di Milano  
nicola.gatti@polimi.it

Alberto Marchesi  
Politecnico di Milano  
alberto.marchesi@polimi.it

Francesco Trovò  
Politecnico di Milano  
francesco1.trovo@polimi.it

Abstract

Recently, game-playing agents based on AI techniques have demonstrated superhuman performance in several sequential games, such as chess, Go, and poker. Surprisingly, the multi-agent learning techniques that allowed to reach these achievements do not take into account the actual behavior of the human player, potentially leading to an impressive gap in performances. In this paper, we address the problem of designing artificial agents that learn how to effectively exploit unknown human opponents while playing repeatedly against them in an online fashion. We study the case in which the agent’s strategy during each repetition of the game is subject to constraints ensuring that the human’s expected utility is within some lower and upper thresholds. Our framework encompasses several real-world problems, such as human engagement in repeated game playing and human education by means of serious games. As a first result, we formalize a set of linear inequalities encoding the conditions that the agent’s strategy must satisfy at each iteration in order to do not violate the given bounds for the human’s expected utility. Then, we use such formulation in an upper confidence bound algorithm, and we prove that the resulting procedure suffers from sublinear regret and guarantees that the constraints are satisfied with high probability at each iteration. Finally, we empirically evaluate the convergence of our algorithm on standard testbeds of sequential games.

1 Introduction

Algorithmic game theory and machine learning have recently contributed to groundbreaking achievements in artificial intelligence, leading to the deployment of artificial agents that defeated top human professionals in several recreational games. See, for example, the well-known milestones achieved in chess [Campbell et al., 2002], Go [Silver et al., 2016], and poker [Brown and Sandholm, 2018, 2019]. Surprisingly, multi-agent learning techniques that have been recently employed in such settings learn how to defeat humans without taking into account their actual behavior. Indeed, they learn strategies by simulating millions of plays in a self-play approach, without including any human player in the learning process. A direct effect of this methodology is that the resulting artificial agents do not

*Equal contribution.
We show that such region is characterized by a set of linear constraints defined over the sequence-form whose cardinality is linear in the size of the game tree. In particular, these constraints describe a which the human learns over time, the resulting model still presents several technical challenges that

Another application scenario for our framework is that of serious games whose purpose is to educate humans by asking them to perform tasks engagingly. If some tasks are

Another application scenario for our framework is that of serious games whose purpose is to educate humans by asking them to perform tasks engagingly. If some tasks are excessively easy, the human will give up the training since he/she is sure that he/she can solve all remaining tasks. Serious games are used in many different fields, such as, e.g., military [DeFalco et al., 2018], transportation [Rossetti et al., 2013], urban planning [Poplin, 2011], and healthcare [Wang et al., 2016], and can be modeled as general-sum games between a human learner and a computer teacher [Mayer, 2012].

Original contributions. We study two-player sequential (i.e., extensive-form) games in which an artificial agent repeatedly plays against an unknown human opponent. We assume that the human has a fixed stochastic behavior and he/she does not learn over time. We do not make any structural assumption on the human’s strategy, which requires the agent learning a probability distribution for each decision point of the human. While this is a crucial first step towards a more complex setting in which the human learns over time, the resulting model still presents several technical challenges that are worth to be investigated. First, we show how to derive, after each game repetition, a confidence region for the human’s strategy such that his/her actual strategy stays within it with high probability. We show that such region is characterized by a set of linear constraints defined over the sequence-form strategy space. Notice that, during each game repetition, the agent only observes a partial sample of the human’s strategy, made by the human’s actions on the path in the game tree followed during play. By exploiting strong duality and the specific structure of the confidence region, we show that the thresholds constraints on the human’s expected utility can be formulated as a set of linear inequalities, whose cardinality is linear in the size of the game tree. In particular, these constraints describe a subspace of the agent’s sequence-form strategy space such that, for every possible human’s strategy in the confidence region, the human’s expected utility is within the given thresholds. We also derive a linear program with a linear number of constraints and variables to find the best agent’s strategy satisfying the constraints. Then, we design an upper confidence bound algorithm, called COX-UCB, and we prove that it suffers from a sublinear regret and guarantees that the aforementioned constraints are satisfied with high probability at every iteration. Finally, we empirically evaluate the convergence of our algorithm on standard testbeds and show that our bounds are asymptotically tight. All the omitted proofs are provided in Appendices A, B, and C.

Related works. Our work is mainly related to opponent modeling, whose primary goal is to build models describing the behavior of one or multiple opponents from past interactions. Many methods are known in the literature. Specifically, Mealing and Shapiro [2017] and Foerster et al. [2018] propose a method to infer the parameters of the opponents’ policies from the data collected during
past interactions. Instead, several other works propose methods that use beliefs over a finite set of opponents’ types distinguishing for their behavior: Albrecht and Ramamoorthy [2014] treat types as black-box mappings, Albrecht and Stone [2017] infer parameters for the types, Barrett and Stone [2015] use deep learning to explicitly learn models, while He and Boyd-Graber [2016] do the same implicitly. These methods usually require huge amount of training data to be precise and adaptable to new opponents. In a recent work, Wu et al. [2021] propose a learning-to-exploit deep framework for implicit opponent modeling and an adversarial training procedure to automatically generate opponents so as to reduce the data needed for training. An approach that has more common ground with ours is the one adopted by Ganzfried and Sandholm [2011], who propose a game-theoretic approach to develop a deviation-based best-response algorithm. In particular, the work builds an opponent model based on the deviations between the opponent’s strategy and a precomputed approximate equilibrium, and, then, computes a best response in real-time. The only works providing theoretical guarantees are Ganzfried and Sandholm [2015] Ganzfried and Sun [2018], which deal with safe opponent exploitation, i.e., guaranteeing a certain agent’s payoff in expectation, given any opponent’s strategy. Differently, in our setting, the goal is to guarantee a certain human’s expected utility.

2 Preliminaries

In this section, we review the basic concepts and definitions related to sequential games that we need in the rest of this work (see the book by Shoham and Leyton-Brown [2008] for more details).

Extensive-form games. We focus on two-player extensive-form games (EFGs) with imperfect information in which an artificial agent faces a human opponent. We denote by $i$ the agent player, while $j$ is the human. Then, the set of players is $P' \cup \{c\}$, where we let $P := \{i, j\}$ and $c$ denotes a chance player that selects actions according to fixed known probabilities, representing exogenous stochasticity. An EFG is usually defined by means of a game tree, where $H$ is the set of nodes of the tree and $Z \subseteq H$ is the subset of terminal nodes, which are the leaves of the game tree. A node $h \in H$ is identified by the ordered list of actions encountered on the path from the root of the game tree to the node. Given a non-terminal node $h \in H \setminus Z$, we let $P(h) \in P \cup \{c\}$ be the unique player who acts at $h$ and $A(h)$ be the set of actions he/she has available. We let $u_i, u_j : Z \to \mathbb{R}$ be the payoff functions of players $i$ and $j$, respectively. Moreover, we denote by $p_c : H \to [0, 1]$ the function assigning each node $h \in H$ to the product of probabilities of chance moves on the path from the root of the game tree to $h$. Imperfect information is encoded by using information sets (infosets for short). A player $i$’s infoset $I$ groups nodes belonging to player $i$ that are indistinguishable for him/her, that is, $I \subseteq H \setminus Z$ is such that $P(h) = P(k) = i$ and $A(h) = A(k)$ for any pair of nodes $h, k \in I$. We let $\mathcal{I}$ be the set of all player $i$’s infosets, which define a partition of the set of player $i$’s non-terminal nodes $\{h \in H \setminus Z | P(h) = i\}$. Moreover, with a slight abuse of notation, we let $A(I)$ be the set of actions available at all the nodes in infoset $I \in \mathcal{I}$. Analogously, we define $\mathcal{J}$ as the set of player $j$’s infosets, while, for any infoset $J \in \mathcal{J}$, we let $A(J)$ be the set of actions available at nodes in $J$. We focus on games with perfect recall in which infosets are such that no player forgets information once acquired.

The sequence form of EFGs. The sequence form is a compact way of representing EFGs with perfect recall [Von Stengel 1996 Koller et al. 1996], where the pure strategies of a player—specifying an action at each infoset of that player—are replaced by the concept of sequence. Any node $h \in H$ defines a sequence $\sigma_i(h)$ of player $i$, which is identified by the ordered list of player $i$’s actions on the path from the root of the game tree to $h$. In perfect-recall EFGs, all the nodes belonging to an infoset $I \in \mathcal{I}$ of player $i$ define the same player $i$’s sequence, which, by overloading notation, we denote by $\sigma_i(I)$. Sequence $\sigma_i(I)$ can be extended by appending any action $a \in A(I)$ available at $I$ at its end, obtaining another valid player $i$’s sequence that we denote as $\sigma_i(I)a$. Then, the set of player $i$’s sequences is $\Sigma_i := \{\sigma_i(I)a | I \in \mathcal{I}, a \in A(I)\} \cup \{\emptyset\}$, where $\emptyset$ is the empty sequence (defined by all the nodes such that player $i$ never plays before them in the game tree). Analogously, we define $\Sigma_j := \{\sigma_j(J)a | J \in \mathcal{J}, a \in A(J)\} \cup \{\emptyset\}$ as the set of all player $j$’s sequences. Mixed strategies in the sequence form are specified by defining the realization probability of each sequence. A sequence-form strategy of player $i$ is denoted by a vector $x \in [0, 1]^{2|\Sigma_i|}$, with $x[\sigma_i]$ being the realization probability of sequence $\sigma_i \in \Sigma_i$.

\[\text{To be well defined, a sequence-form strategy must}\]
We let \( \rho \) be the probability of reaching \( y \) given two sequence-form strategies \( x \) and \( y \), which is defined by linear constraints analogous to Constraints (1). For the ease of presentation, we let \( X := \{ x \in [0, 1]^{[\Sigma_i]} \mid F_i x = f_i \} \) and \( Y := \{ y \in [0, 1]^{[\Sigma_j]} \mid F_j y = f_j \} \) be the sets of all sequence-form strategies of players \( i \) and \( j \), respectively. Finally, given two strategies \( x \in X \) and \( y \in Y \), it is easy to check that player \( i \)'s expected payoff can be written as the bilinear form \( x^T U_i y \), where \( U_i \in \mathbb{R}^{[\Sigma_i] \times [\Sigma_j]} \) is player \( i \)'s sequence-form utility matrix, defined as:

\[
U_i[\sigma_i, \sigma_j] := \sum_{z \in Z; \sigma_i(z) = \sigma_i, \sigma_j(z) = \sigma_j} p_c(z) u_i(z) \quad \forall \sigma_i \in \Sigma_i, \forall \sigma_j \in \Sigma_j.
\]

Analogously, the sequence-form utility matrix of player \( j \) is \( U_j \in \mathbb{R}^{[\Sigma_i] \times [\Sigma_j]} \), and, thus, \( x^T U_j y \) is his/her expected payoff given two sequence-form strategies \( x \in X \) and \( y \in Y \).

**Additional notation.** Given two sequences \( \sigma_i, \sigma_j' \in \Sigma_i \) of player \( i \), we write \( \sigma_i \subseteq \sigma_j' \) to denote that \( \sigma_i \) is a sub-sequence of \( \sigma_j' \); formally, this is the case whenever the ordered list of actions identified by \( \sigma_i \) is a prefix of that of \( \sigma_j' \). Similarly, we use notation \( \sigma_j \subseteq \sigma_j' \) for two sequences \( \sigma_j, \sigma_j' \in \Sigma_j \) of player \( j \). Moreover, given a player \( i \)'s sequence-form strategy \( x \in X \) and a player \( j \)'s infoset \( J \), we let \( \rho_{-j}(J, x) \) be the probability of reaching \( J \) given that player \( j \) plays so as to reach it and player \( i \) plays \( x \) (also accounting for chance probabilities); formally \( \rho_{-j}(J, x) := \sum_{h \in J} x[\sigma_i(h)] p_c(h) \).

### 3 Exploiting opponents under utility constraints

We study settings in which the agent player \( i \) repeatedly faces the human opponent \( j \) in a two-player EFG. Our goal is the design of agents that learn the strategy of the human so as to effectively exploit it, while at the same time guaranteeing that the human’s expected utility remains under control during the entire repeated interaction. In the rest of this section, we formally introduce our problem and provide a general overview of the approach we undertake to tackle it.

We let \( T \) be the number of times the EFG is played. Player \( j \) plays according to the same sequence-form strategy \( y^* \in Y \) at each iteration \( t \in [T] \). This strategy is unknown to player \( i \). On the other hand, at iteration \( t \), player \( i \) selects and plays a strategy \( x^t \in X \). Then, at the end of the iteration, player \( i \) receives as feedback a sequence of player \( j \)'s actions \( \sigma_j^t \in \Sigma_j \), which is defined by the path in the game tree followed during game playing at that iteration (notice that \( \sigma_j^t \) is made of actions sampled according to player \( j \)'s strategy \( y^* \)). In the following, for \( t \in [T] \), we let \( \mathcal{H}^t \) be the history of feedbacks received by the agent player \( i \) up to iteration \( t \) (included), namely \( \mathcal{H}^t := (\sigma_j^1, \sigma_j^2, \ldots, \sigma_j^t) \).

To ensure that player \( j \)'s utility is kept under control during the repeated interaction, player \( i \) must play strategies \( x^t \) such that the resulting player \( j \)'s expected utilities are within some given thresholds. The lower limit of the range ensures that the agent does not over-exploit the human. On the other hand, the upper limit of the range guarantees that the expected payoff of the human is not too high. Formally, we require \( x^t \in \mathcal{X}^t \), where the subset \( \mathcal{X}^t \subseteq X \) is defined as follows:

**Definition 1 (Utility-constrained Strategy Set).** Let \( t \in [T] \) and \( \delta \in (0, 1) \). Given a lower limit \( \alpha \in \mathbb{R} \) and an upper limit \( \beta \in \mathbb{R} \), we define the utility-constrained strategy set \( \mathcal{X}^t \subseteq X \) at iteration \( t \) as the set of player \( i \)'s sequence-form strategies \( x \in X \) such that \( \mathbb{P}(\alpha \leq x^T U_j y^* \leq \beta) \geq 1 - \delta \), with respect to the randomness of the history \( \mathcal{H}^{t-1} \) of feedbacks observed by player \( i \) up to iteration \( t - 1 \) (included).\(^3\)

\(^3\)The methodology that we propose in this paper can also be adapted to control the agent’s expected utility, rather than the one of the human player.

\(^4\)In this work, given \( n \in \mathbb{N}_+ \), we denote by \( [n] \) the set \( \{1, \ldots, n\} \) of the first \( n \) natural numbers.

\(^5\)In the rest of this work, we make implicit the dependency of \( \mathcal{X}^t \) from \( \delta, \alpha, \) and \( \beta \), as the values of these parameters will be clear from context.
After \( T \) game repetitions, given the strategies \( x^t \in \mathcal{X}^t \) played by the agent player \( i \) during iterations \( t \in [T] \), we measure his/her performance by means of the following notion of regret:

\[
R^T := \sum_{t=1}^{T} \max_{x^t \in \mathcal{X}^t} \left( (x^t)^\top U_i y^* - (x^t)^\top U_i y^* \right),
\]

which represents how much player \( i \) would have gained in expectation by playing a utility-maximizing strategy in the utility-constrained strategy set \( \mathcal{X}^t \) rather than \( x^t \), at each iteration \( t \in [T] \). The goal that we pursue in the rest of this work is to achieve sublinear regret, that is \( R^T = o(T) \), while at the same time guaranteeing that the played strategies \( x^t \) satisfy the constraints defined by the sets \( \mathcal{X}^t \).

**Overview of our results.** In what follows, we give a brief sketch of the approach we adopt to tackle the problem. We propose a learning algorithm for the agent player \( i \), which we call *Constrained Opponent eXploitation with Upper Confidence Bounds* (COX-UCB). It builds on two core components. The first one deals with the construction of the utility-constrained strategy set \( \mathcal{X}^t \) at each iteration \( t \in [T] \). It works by building a confidence region \( \mathcal{Y}^{t-1} \subseteq \mathcal{Y} \) for player \( j \)'s strategy, using the history \( \mathcal{H}^{t-1} \) of feedbacks observed up the previous iteration \( t - 1 \). This is such that the true (unknown) strategy \( y^* \) lies within \( \mathcal{Y}^{t-1} \) with probability at least \( 1 - \delta \), for some fixed confidence level \( \delta \in (0, 1) \). Then, the utility-constrained strategy set \( \mathcal{X}^t \) can be characterized by a set of linear inequalities that exploits the structure of the confidence region \( \mathcal{Y}^{t-1} \). A detailed formal treatment of this first component is provided in Section 4. The second core component consists in a rule to select the strategy \( x^t \in \mathcal{X}^t \) to play at each iteration \( t \in [T] \). We propose an approach based on the *optimism in face of uncertainty* principle. More details on this second component can be found in Section 5 together with the regret bounds attained by the algorithm. We refer the reader to Algorithm 1 for a general sketch of our COX-UCB algorithm, where two alternative implementations of the procedure \( \text{SELECTSTRATEGY}(\mathcal{X}^t, \mathcal{Y}^{t-1}) \) will be given in Algorithms 2 and 3 in Section 5.

**Algorithm 1 COX-UCB**

1: \( t \leftarrow 1 \)
2: while \( t \leq T \) do
3: \( \text{Build confidence region } \mathcal{Y}^{t-1} \text{ from history of past feedbacks } \mathcal{H}^{t-1} \) \hspace{1cm} \( \triangleright \) Subsection 4.1
4: \( \text{Use } \mathcal{Y}^{t-1} \text{ to build the utility-constrained strategy set } \mathcal{X}^t \) \hspace{1cm} \( \triangleright \) Subsection 4.2
5: \( x^t \leftarrow \text{SELECTSTRATEGY}(\mathcal{X}^t, \mathcal{Y}^{t-1}) \) \hspace{1cm} \( \triangleright \) Section 5
6: \( \text{Play the game according to strategy } x^t \)
7: Observe player \( j \)'s sequence \( \sigma_j \), obtained from the path in the game tree followed during play
8: \( \mathcal{H}^t \leftarrow \mathcal{H}^{t-1} \cup \{ \sigma_j \} \)
9: \( t \leftarrow t + 1 \)

4 **How to control the human’s expected utility**

In this section, we provide the formal details on the construction of the utility-constrained strategy set \( \mathcal{X}^t \) at each iteration \( t \in [T] \) (Definition 1). We split the section into two main parts:

- **Subsection 4.1** shows how to use the history \( \mathcal{H}^t \) of feedbacks observed by player \( i \) up to iteration \( t \) to derive a confidence region \( \mathcal{Y}^t \subseteq \mathcal{Y} \) for player \( j \)'s strategy \( y^* \), such that \( y^* \) lies within \( \mathcal{Y}^t \) with probability at least \( 1 - \delta \) for some fixed confidence \( \delta \in (0, 1) \);

- **Subsection 4.2** describes how to exploit the confidence region \( \mathcal{Y}^{t-1} \) built using feedbacks observed up to iteration \( t - 1 \) to construct a set of linear constraints that fully characterize sequence-form strategies in the utility-constrained strategy set \( \mathcal{X}^t \) at iteration \( t \).

All the proofs omitted from this section are reported in Appendices A and B.

4.1 **Building a confidence region for the human’s strategy**

Let us recall that, at each iteration \( t \in [T] \), player \( i \) observes a player \( j \)'s sequence \( \sigma_j \) determined by selecting actions to play during the game according to the sequence-form strategy \( y^* \). We build the desired high-probability confidence region \( \mathcal{Y}^t \) by exploiting information provided by observed
sequences to derive, for each player \( j \)'s infoset \( J \in \mathcal{J} \), appropriate confidence intervals for the realization probabilities \( y^*[\sigma_j(J)a] \) of sequences \( \sigma_j(J)a \) terminating with an action \( a \in A(J) \) at \( J \).

**The case of a single infoset.** Before showing our general technique, it is useful to present the easier setting in which player \( j \) has a unique infoset, and, thus, his/her strategy is defined as a probability distribution \( p \in \Delta^{|A|} \), where, with an abuse of notation, \( A \) denotes the finite set of actions available at the infoset. In this case, player \( i \) observes \( t \) actions \( a^1, \ldots, a^t \in A \) sampled independently according to \( p \). Then, a natural estimator for \( p \) is the empirical frequency of actions \( p^t \in \Delta^{|A|} \), defined so that \( p^t[a] := \frac{1}{t} \sum_{\tau=1}^{t} \mathbb{1}\{a^\tau = a\} \) for every \( a \in A \). By noticing that \( t p^t \) is a random variable following a multinomial distribution with parameters \( t \) and \( p \), that is \( t p^t \sim \mathcal{M}(t; p) \), the following lemma by [Devroye] can be used to derive the desired confidence intervals for the probabilities \( p[a] \).

**Lemma 1** (Lemma 3 by [Devroye] [1983]). Let \( p \in \Delta^{|A|} \) and \( a^1, \ldots, a^t \in A \) be \( t \) actions sampled independently according to \( p \). Then, for any \( 0 < \delta \leq 3 \exp(-4|A|/5) \), it holds:

\[
P \left( \sum_{a \in A} \left| p^t[a] - p[a] \right| \geq 5 \sqrt{\frac{\ln(3/\delta)}{t}} \right) \leq 1 - \delta.
\]

By exploiting the fact that \( p \in \Delta^{|A|} \), we can refine the result in Lemma 1 by giving bounds that hold for each component of \( p \) separately (see Lemma 2 below). This additional step is crucial when building confidence intervals for realization probabilities \( y^*[\sigma_j(J)a] \) at infoset \( J \) in general.

**Lemma 2.** Let \( p \in \Delta^{|A|} \) and \( a^1, \ldots, a^t \in A \) be \( t \) actions sampled independently according to \( p \). Then, for any \( 0 < \delta \leq 3 \exp(-4|A|/5) \), it holds:

\[
P \left( \bigcap_{a \in A} \left\{ \left| p^t[a] - p[a] \right| \leq \frac{5}{2} \sqrt{\frac{\ln(3/\delta)}{t}} \right\} \right) \geq 1 - \delta.
\]

Next, we generalize the approach described above to the general case of any infosets structure.

First, we introduce some useful random variables. For every iteration \( t \in [T] \), player \( j \)'s infoset \( J \in \mathcal{J} \), and action \( a \in A(J) \), we let \( O^t(J, a) := \mathbb{1}\{\sigma_j(J)a \subseteq \sigma^*_j\} \) be a random variable that is equal to 1 if and only if player \( j \) played action \( a \) at infoset \( J \) during iteration \( t \), while it is equal to 0 otherwise. It is easy to check that \( O^t(J, a) \) follows a Bernoulli distribution with parameter \( p^t(J, a) := y^*[\sigma_j(J)a] \rho^t_{-j}(J) \), where, for the ease of presentation, we let \( \rho^t_{-j}(J) := \rho_{-j}(J, x^t) \) be the contribution to the probability of reaching infoset \( J \) due to player \( i \)'s strategy \( x^t \) and chance probabilities. The random variables \( O^t(J, a) \) are instrumental for defining \( N^t(J, a) := \sum_{\tau=1}^{t} O^t(J, a) \), which represents the number of times \( a \) is played at \( J \) up to iteration \( t \). Intuitively, variables \( N^t(J, a) \) of infoset \( J \in \mathcal{J} \) play the same role as the random vector \( t p^t \) in the single-infoset case.

We follow an approach analogous to that of the single-infoset case at each player \( j \)'s infoset, and, then, put all the resulting confidence intervals together to define \( \mathcal{Y}^t \). To do so, we need to circumvent the following issue: at each infoset \( J \in \mathcal{J} \), the random variables \( N^t(J, a) \) for \( a \in A(J) \) are not jointly distributed as a multinomial, preventing a direct application of Lemma 1. We deal with this by adding a fictitious action at each infoset, so that random variables \( N^t(J, a) \) are multinomially distributed for each \( J \in \mathcal{J} \). Then, the fictitious action can be easily factored out by using Lemma 2.

Let \( A_o(J) := A(J) \cup \{a_o\} \) be the new action set at \( J \in \mathcal{J} \), with \( a_o \) denoting the fictitious action. For \( t \in [T] \), we define \( \mathcal{O}^t(J, a_o) := \mathbb{1}\{\sigma_j(J)a_o \not\subseteq \sigma^*_j\} \) as a random variable equal to 1 if and only if...
We show how to construct the utility-constrained strategy set while it is easy to check that for every player

\[ \text{Theorem 1.} \]

let at least

\[ y \]

Definition 1 for such condition) for every

By exploiting the high-probability confidence region

\[ \sum_{j \in J} \rho^*_{J,j}(J) \]

confidence region

\[ Y \]

\[ J \]

\[ \rho^*_{J,j}(J) \]

\[ y \]

\[ J \]

\[ t \sum_{j \in J} \rho^*_{J,j}(J) \]

we have:

\[ \mathbb{E} [\mathcal{N}^t(J,a)] = t \sum_{j \in J} \mathbb{E} [\mathcal{O}^t(J,a)] = y^*[\sigma_j(J)a] \]

\[ \sum_{j \in J} \rho^*_{J,j}(J) y^*[\sigma_j(J)a] \]

while it is easy to check that \( \mathbb{E} [\mathcal{N}^t(J,a)] = t - t \sum_{j \in J} \rho^*_{J,j}(J) y^*[\sigma_j(J)a] \). As a result, we conclude that \( n^t_j \) follows a multinomial distribution, circumventing our initial issue. Formally:

\[ n^t_j \sim \mathcal{M}(t,v), \quad v \in \Delta^{|A_o(J)|} \]

\[ v[a] = \begin{cases} \rho^*_{J,j}(J) y^*[\sigma_j(J)a] & \text{if } a \in A(J) \\ 1 - \rho^*_{J,j}(J) \sum_{a \in A(J)} y^*[\sigma_j(J)a] & \text{if } a = a_o \\ \end{cases} \]

By exploiting this last observation and using Lemmas 1 and 2, we provide confidence intervals defined locally at each infoset \( J \in \mathcal{J} \) for the realization probabilities \( y^*[\sigma_j(J)a] \) of sequences \( \sigma_j(J)a \) terminating with an action \( a \in A(J) \) at \( J \). By letting \( \mathcal{Y}^t \in \mathbb{R}^{|A(J)|} \) be such that:

\[ y^*[\sigma_j(J)a] := \mathcal{N}^t(J,a) \quad \forall J \in \mathcal{J}, \forall a \in A(J), \]

we have the following lemma:

\[ \text{Lemma 3. Let } J \in \mathcal{J} \text{ be a player } j \text{'s infoset. Then, for any } 0 < \delta \leq 3 \exp(-4|A(J)|/5) \text{ it holds:} \]

\[ \mathbb{P} \left( \bigcap_{a \in A(J)} \left\{ y^*[\sigma_j(J)a] - y^*[\sigma_j(J)a] \leq \frac{5}{2\rho^*_{J,j}(J)} \sqrt{\frac{\ln(3/\delta)}{2}} \right\} \right) \geq 1 - \delta. \]

Lemma 3 shows that one can use \( y^*[\sigma_j(J)a] \) to estimate the realization probability \( y^*[\sigma_j(J)a] \) of some sequence \( \sigma_j(J)a \), where \( y^*[\sigma_j(J)a] \) represents the observed frequency of action \( a \) at infoset \( J \), adjusted by the average probability of reaching \( J \) due to other players, namely \( \rho^*_{J,j}(J) \). Thus, \( y^* \) generalizes the empirical frequency \( p^* \) in the single-infoset case to the sequence-form setting. Moreover, Lemma 3 gives high-probability confidence intervals for probabilities \( y^*[\sigma_j(J)a] \) at every infoset \( J \in \mathcal{J} \), which is used in the following theorem that gives us the desired high-probability confidence region \( \mathcal{Y}^t \).

\[ \text{Theorem 1. For every player } j \text{'s infoset } J \in \mathcal{J}, \text{ let } \delta_j \in (0,1) \text{ be such that the condition in Lemma 3 is satisfied and } \sum_{J \in \mathcal{J}} \delta_j < 1. \text{ Then, } \mathbb{P}(\mathcal{Y}^t) \geq 1 - \delta, \text{ where } \delta := \sum_{J \in \mathcal{J}} \delta_j \text{ and} \]

\[ \mathcal{Y}^t := \left\{ y \in \mathcal{Y} : \left| y^*[\sigma_j(J)a] - y^*[\sigma_j(J)a] \right| \leq \frac{5}{2\rho^*_{J,j}(J)} \sqrt{\frac{\ln(3/\delta_j)}{2}} \quad \forall J \in \mathcal{J}, \forall a \in A(J) \right\}. \]

4.2 Constructing the utility-constrained strategy set

We show how to construct the utility-constrained strategy set \( \mathcal{X}^t \)—for some \( \delta \in (0,1) \) and \( \alpha, \beta \in \mathbb{R} \)—by exploiting the high-probability confidence region \( \mathcal{Y}^t \) (see Theorem 1). Our approach is to force the condition that each \( x \) in the utility-constrained strategy set must satisfy with high probability (see Definition 1 for such condition) for every \( y \in \mathcal{Y}^t \). Formally, we require \( \rho \leq x^\top U_j y \leq \beta \) for all \( y \in \mathcal{Y}^t \). By definition of \( \mathcal{Y}^t \), this guarantees that such constraint holds also for \( y^* \) with probability at least \( 1 - \delta \). Formally, we define

\[ \mathcal{X}^t := \left\{ x \in \mathcal{X} : \max_{y \in \mathcal{Y}^t} x^\top U_j y \leq \beta \land \min_{y \in \mathcal{Y}^t} x^\top U_j y \geq \alpha \right\}, \]

so that, for any \( x \in \mathcal{X}^t \), the upper limit \( \beta \) and the lower limit \( \alpha \) are satisfied for every \( y \in \mathcal{Y}^t \).
In what follows, we characterize the set $\mathcal{X}^t$ by means of a set of linear inequalities. We formulate the constrained maximization problem, i.e., $\max_{y \geq 0} x^T U_j y$, using the following linear program:

$$\max_{y \geq 0} x^T U_j y \quad \text{s.t.} \quad \begin{align*}
F_j y &= f_j \\
y &\leq y^{t-1} + \epsilon^{t-1} \\
y &\geq y^{t-1} - \epsilon^{t-1},
\end{align*}$$

(3a) (3b) (3c) (3d)

where we let $\epsilon^{t-1} \in \mathbb{R}_+^{\left| J^{t-1} \right|}$ be such that $\epsilon^{t-1}[\sigma_J(J) a] := \frac{5}{2\phi_{J}(\delta_J)} \sqrt{\frac{\ln(3\delta_J)}{t}}$ for $J \in \mathcal{J}$ and $a \in A(J)$, for some $\delta_J \in (0, 1)$ that satisfy the condition in Lemma 3 and $\sum_{J \in \mathcal{J}} \delta_J \leq \delta$.

Notice that, by recalling the definition of $y^{t-1}$ in Equation (2) and that of $y^{t-1}$ in Theorem 1, Constraints (3b), (3c), and (3d) correctly encode the fact that $y$ must belong to the set $\mathcal{Y}^{t-1}$.

The dual of Problem (3) reads as: $\min_{u \geq 0} b^T u$ s.t. $A^T u \geq U_j^T x$, where $u \in \mathbb{R}^{(\left| J^{t} \right|+1) \times \left| J^{t-1} \right|}$ is a vector of dual variables, while $b$ and $A$ are suitably-defined vector and matrix, respectively. By strong duality, the optimal dual objective equates the optimal primal objective, and, thus, given any player $i$’s strategy $x \in \mathcal{X}$, the condition $\max_{y \in \mathcal{Y}^{t-1}} x^T U_j y \leq \beta$ holds if there exists a dual feasible solution $u$ that satisfies the additional constraint that $b^T u \leq \beta$.

By following an analogous reasoning for $\min_{u \in \mathcal{Y}^{t-1}} x^T U_j y$, and letting $\omega \in \mathbb{R}^{(\left| J^{t} \right|+1) \times \left| J^{t-1} \right|}$ be a vector of variables of the dual problem corresponding to its linear programming formulation (similar to Problem (3)), we state the following main result:

**Theorem 2.** Let $t \in [T]$ and $\delta \in (0, 1)$. Given $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}$, it holds:

$$\mathcal{X}^t = \mathcal{X} \cap \left\{ (x, u, \omega) : u, \omega \geq 0, b^T u \leq \beta, A^T u \geq U_j^T x, -b^T \omega \geq \alpha, -A^T \omega \leq U_j^T x \right\}.$$

From Theorem 2, it follows that $\mathcal{X}^t$ can be characterized by a polynomially-sized set of linear constraints, which achieves our initial goal. Notice that, in general, the set $\mathcal{X}^t$ defined in Theorem 2 could be empty. This occurs when the linear program used to build $\mathcal{X}^t$ is unfeasible. In practice, this is not an issue, since it is always guaranteed to be feasible for $t$ large enough. A complete discussion on this point is deferred to Appendix B.

### 5 How to select the strategy to play

In this section, we provide the implementation of the procedure SELECTSTRATEGY($\mathcal{X}^t, \mathcal{Y}^{t-1}$) in Algorithm 1. To guarantee that COX-UCB attains sub-linear regret after $T$ iterations, i.e., $R^T = o(T)$ (with high probability), we adopt an approach inspired from arm-selection strategies used in linear multi-armed bandit problems [Auer, 2002]. In particular, we propose one that uses upper confidence bounds, which is inspired by the LinUCB algorithm [Abbasi-Yadkori et al., 2011].

As a first step, we need to ensure that the algorithm performs enough exploration during game playing. This is crucial to lower bound the probabilities $\beta_J(J)$ that appear in the bounds defining the set $\mathcal{Y}^t$ (see Theorem 1), and, ultimately, to obtain sub-linear regret. In particular, at each $t \in [T]$, the COX-UCB algorithm selects a strategy $x^t$ that belongs to the following subset of $\mathcal{X}^t$:

$$\hat{\mathcal{X}}^t := \left\{ x \in \mathcal{X}^t : x[\sigma_z(z)] \geq \alpha^t \forall z \in Z \right\},$$

where the $\alpha^t$’s are suitably-defined parameters that decrease with the iteration number $t$.\footnote{Notice that a naïve way of adding exploration to the algorithm would be to use an $\epsilon$-greedy policy that plays the selected strategy $x^t \in \mathcal{X}^t$ with probability $1 - \epsilon$ and a random one with probability $\epsilon$. However, this approach does not work in our setting, as it would result in a violation of the constraints on the human’s expected utility. On the other hand, picking $x \in \mathcal{X}^t$ assures that such constraints are satisfied.}
We empirically evaluate the convergence of our COX-UCB algorithm on a standard testbed of Kuhn which can be done efficiently by cutting-hedge solvers (see Section 6).

Algorithm 2 Strategy selection of COX-UCB

1: function SEL-STRAT-UCB($X^t, Y^{t-1}$)
2: \[ x^t \leftarrow \arg\max_{x \in X^t} \max_{y \in Y^{t-1}} x^T U_i y \]
3: return $x^t$

The strategy selection mechanism implemented by COX-UCB is provided in Algorithm 2. It is based on the optimism in face of uncertainty principle, and, thus, it selects a strategy $x^t \in X^t$ that maximizes player $i$’s expected payoff $x^T U_i y$ under the assumption that, for every $x \in X^t$, strategy $y$ is an optimistic estimate of $y^*$ taken from the confidence region $Y^{t-1}$, that is, $y \in Y^{t-1}$ maximizes the same player $i$’s expected payoff $x^T U_i y$. The following theorem provides a high-probability sub-linear regret guarantee for COX-UCB (its complete formal proof can be found in Appendix C).

Theorem 3. Let $\alpha^t := \eta^2 \frac{2\ln t + \ln t + 1}{\ln t (\ln t + 1)^2}$ for every $t \in [T]$, where $\eta \in (0, 1)$, and let $\delta \in (0, 1)$. The COX-UCB algorithm attains the following regret bound with probability at least $1 - \delta$:

\[ R^T \leq \frac{5}{2\eta^2} K_{U_i} C \left( 1 + 2\sqrt{T \ln T} \right), \]

where $K_{U_i} := \|U_i\|_\infty$ and $C$ is a suitably-defined constant.

6 Experimental evaluation

We empirically evaluate the convergence of our COX-UCB algorithm on a standard testbed of Kuhn and Leduc poker games [Kuhn 2016]. We report here the evaluation on rank 7 Kuhn poker ($\text{kuhn}_7$) and rank 2 Leduc poker ($\text{leduc}_2$), while other experiments and details are deferred to Appendix D.

6.1 Approximated version of the COX-UCB algorithm

The COX-UCB algorithm presented in Section 5 (Algorithm 2) solves a bilinear optimization problem at every iteration, with the optimization done over the whole utility-constrained strategy set. Furthermore, let us recall that such set is directly built from the high-confidence region for the human’s strategy, which is specified at time $t$ by an estimate $y^t$ and bounds $\epsilon^t$. Empirical evidence from early experiments showed that, despite having the same worst-case convergence rate, in practice the estimate $y^t$ converges much faster to $y^*$ than the bound $\epsilon^t$ does to 0. We propose a simple modification of COX-UCB, called $\psi$-approximated COX-UCB ($\psi$-COX-UCB), exploiting the faster convergence of the strategy estimation. The only difference between COX-UCB and $\psi$-COX-UCB is in the strategy selection procedure, as specified in Algorithm 3. This simple variation has a twofold effect on the overall algorithm: (i) in some iterations it avoids solving the bilinear program, by solving a much simpler linear program instead, and (ii) it leverages the fast empirical convergence of $y^t$ to converge faster. In the remaining part of this section, we empirically evaluate COX-UCB and $\psi$-COX-UCB for different values of the parameter $\psi$.

6.2 Experimental results

Experimental setting. In order to compare the performances of COX-UCB with those of its approximated version, we consider three versions of $\psi$-COX-UCB, respectively with $\psi = 0.5$, $\psi = 0.7$ and $\psi = 0.9$. We evaluate the performances of the algorithms against 10 different randomly generated strategies for each game instance considered. For each strategy, we execute 5 different algorithm runs. The values $(\alpha, \beta)$ needed for the utility constraints are set to $\alpha = -0.3$ and $\beta = 0.3$ in
Figure 1: Performances of COX-UCB in Kuhn poker with 5 ranks (first row) and Leduc poker with 2 ranks (second row). From left to right: player j’s utility, cumulative regret, and cumulative regret divided by $\sqrt{t \ln t}$.

Convergence results. Figure 1 shows the results of the experiments on kuhn_7 and leduc_2. We observe that, in kuhn_7, the performances of COX-UCB and those of its approximated versions are comparable, thus showing that the rate of convergence of the opponent’s strategy estimation plays a negligible role in this instance. On the other hand, the analysis of leduc_2 highlights a different behavior, suggesting that, in bigger game instances, the dimension of the confidence bound on the opponent’s strategy and the rate with which such dimension reduces become crucial in order to achieve good performances. Furthermore, Figures 1a and 1d show how the algorithm manages to maintain the expected utility of the opponent in the utility range $[-0.3, 0.3]$. On the other hand, we can see how the random algorithm, which plays a random strategy at each iteration, does indeed maintain the utility constraints satisfied (as they are below the threshold of $\beta = 0.3$), but it does not result in sublinear regret $R^T$, as one can observe from Figures 1b and 1c. Interestingly, in Figures 1c and 1f the ratio between the regret and $\sqrt{t \ln t}$, representing its asymptotic dependence on $t$, converges to a constant as $t$ increases, thus showing that our upper bound on the regret is tight.

7 Conclusions and future directions

In this paper, we studied, for the first time, the problem of designing artificial agents that learn how to exploit an unknown human opponent in an online fashion while, at the same time, guaranteeing that the human’s expected utility remains bounded within some upper and lower thresholds. This framework finds application in several real-world human-agent interactions, such as repeated human-agent game playing in which the agent’s goal is also to keep the human engaged in the repeated interaction, and human teaching by means of serious games. Our results hold up under the assumption that the human player adopts a fixed (unknown) stochastic strategy. This is a first crucial step towards more complex models of the human behavior. Nevertheless, even this basic case begets considerable technical challenges, which make the theoretical study of the problem interesting in its own, while also constituting a starting point for the analysis of more complex scenarios.

An interesting direction for future research is assuming a more complex human behavior, such as the case in which the human is learning over time. Another line of research is the enhancement of the scalability of the COX-UCB algorithm, so as to enable its application in real-world sequential games. Finally, it could be of general interest framing our algorithm in serious games scenarios, which have been largely ignored by the research community in artificial intelligence so far.
Acknowledgments and Disclosure of Funding

This work has been partially supported by the Italian MIUR PRIN 2017 Project ALGADIMAR “Algorithms, Games, and Digital Market”.

References


