Efficient Regret Minimization Algorithm for Extensive-Form Correlated Equilibrium

Anonymous Author(s)
Affiliation
Address
email

Abstract

In the past few years, self-play methods based on regret minimization have become the state of the art for computing Nash equilibria in large two-players zero-sum extensive-form games. These methods fundamentally rely on the hierarchical structure of the players' sequential strategy spaces to construct a regret minimizer that recursively minimizes regret at each decision point in the game tree. In this paper, we introduce the first efficient regret minimization algorithm for computing extensive-form *correlated* equilibria in large two-player *general-sum* games with no chance moves. Designing such an algorithm is significantly more challenging than designing one for the Nash equilibrium counterpart, as the constraints that define the space of correlation plans lack the hierarchical structure and might even form cycles. We show that some of the constraints are redundant and can be excluded from consideration, and present an efficient algorithm that generates the space of extensive-form correlation plans incrementally from the remaining constraints. This structural decomposition is achieved via a special convexity-preserving operation that we coin *scaled extension*. We show that a regret minimizer can be designed for a scaled extension of any two convex sets, and that from the decomposition we then obtain a global regret minimizer. Our algorithm produces feasible iterates. Experiments show that it significantly outperforms prior approaches—the LP-based approach and a very recent subgradient descent algorithm—and for larger problems it is the only viable option.

1 Introduction

In recent years, self-play methods based on regret minimization, such as counterfactual regret minimization (CFR) (Zinkevich et al., 2007) and its faster variants (Tammelin et al., 2015; Brown et al., 2017; Brown & Sandholm, 2019) have emerged as powerful tools for computing Nash equilibria in large extensive-form games, and have been instrumental in several recent milestones in poker (Bowling et al., 2015; Brown & Sandholm, 2017a,b; Moravčík et al., 2017). These methods exploit the hierarchical structure of the sequential strategy spaces of the players to construct a regret minimizer that recursively minimizes regret locally at each decision point in the game tree. This has inspired regret-based algorithms for other solution concepts in game theory, such as extensive-form perfect equilibria (Farina et al., 2017), Nash equilibrium with strategy constraints (Farina et al., 2017, 2019a,b; Davis et al., 2019), and quantal-response equilibrium (Farina et al., 2019a).

In this paper, we give the first *efficient* regret-based algorithm for finding an *extensive-form correlated equilibrium (EFCE)* (von Stengel & Forges, 2008) in two-player general-sum games with no chance moves. EFCE is a natural extension of the correlated equilibrium (CE) solution concept to the setting of extensive-form games. Here, the strategic interaction of rational players is complemented by a

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.

mediator that privately recommends behavior, but does not *enforce it*: it is up to the mediator to make recommendations that the players are incentivized to follow.

Designing a regret minimization algorithm that can efficiently search over the space of extensive-form correlated strategies (known as *correlation plans*) is significantly more difficult than designing one for the Nash equilibrium counterpart. This is because the constraints that define the space of correlation plans lack the hierarchical structure of sequential strategy spaces and might even form cycles. Existing general-purpose regret minimization algorithms, such as follow-the-regularized-leader (Shalev-Shwartz & Singer, 2007) and mirror descent, as well as those proposed by Gordon et al. (2008) in the context of convex games, are not practical: they require the evaluation of proximal operators (generalized projections problems) or the minimization of linear functions on the space of extensive-form correlation plans. In the former case, no distance-generating function is known that can be minimized efficiently over this space, while in the latter case current linear programming technology does not scale to large games, as we show in the experimental section of this paper. The regret minimization algorithm we present in this paper computes the next iterate in *linear* time in the dimension of the space of correlation plans.

We show that some of the constraints that define the polytope of correlation plans are redundant and can be eliminated, and present an efficient algorithm that generates the space of extensive-form correlation plans incrementally from the remaining constraints. This structural decomposition is achieved via a special convexity-preserving operation that we coin *scaled extension*. We show that a regret minimizer can be designed for a scaled extension of any two convex sets, and that from the decomposition we then obtain a global regret minimizer. Experiments show that our algorithm significantly outperforms prior approaches—the LP-based approach (von Stengel & Forges, 2008) and a very recent subgradient descent algorithm (Farina et al., 2019c)—and for larger problems it is the only viable option.

2 Preliminaries

2.1 Extensive-Form Games

Extensive-form games (EFGs) are played on a game tree. They can capture sequential and simultaneous moves as well as private information. Each node in the game tree belongs to a player, who acts at that node; for the purpose of this paper, we focus on two-player games only. Edges leaving a node correspond to actions that can be taken at that node. In order to capture private information, the game tree is supplemented with *information sets*. Each node belongs to exactly one information set, and each information set is a nonempty set of tree nodes for the same Player i, which are the set of nodes that Player i cannot distinguish among, given what they have observed so far. We will focus on *perfect-recall* EFGs, that is, EFGs where no player forgets what the player knew earlier. We denote by $\mathcal{I}_1$ and $\mathcal{I}_2$ the sets of all information sets that belong to Player 1 and 2, respectively. All nodes that belong to an information set $I \in \mathcal{I}_1 \cup \mathcal{I}_2$ share the same set of available actions (otherwise the player acting at those nodes would be able to distinguish among them); we denote by A_I the set of actions available at information set I. We define the set of *sequences* of Player i as the set $\Sigma_i := \{(I, a) : I \in \mathcal{I}_i, a \in A_I\} \cup \{\varnothing\}$, where the special element $\varnothing$ is called *empty sequence*. Given an information set $I \in \mathcal{I}_i$, we denote by $\sigma(I)$ the *parent sequence* of I, defined as the last pair $(I', a') \in \Sigma_i$ encountered on the path from the root to any node $v \in I$; if no such pair exists (that is, Player i never acts before any node $v \in I$), we let $\sigma(I) = \varnothing$. We (recursively) define a sequence $\tau \in \Sigma_i$ to be a *descendent* of sequence $\tau' \in \Sigma_i$, denoted by $\tau \succeq \tau'$, if $\tau = \tau'$ or if $\tau = (I, a)$ and $\sigma(I) \succeq \tau'$. We use the notation $\tau \succ \tau'$ to mean $\tau \succeq \tau' \wedge \tau \neq \tau'$. Figure 1 shows a small example EFG; black round nodes belong to Player 1, white round nodes belong to Player 2, action names are not shown, gray round sets define information sets, and the numbers along the edges define concise names for sequences (for example, '7' denotes sequence (D, a) where a is the leftmost action at D).

Sequence-Form Strategies In the sequence-form representation (Romanovskii, 1962; Koller et al., 1996; von Stengel, 1996), a strategy for Player i is compactly represented via a vector $\boldsymbol{x}$ indexed by sequences $\sigma \in \Sigma_i$. When $\sigma = (I, a)$, the entry $x[\sigma] \geq 0$ defines the product of the probabilities according to which Player i takes their actions on the path from the root to information set I, up to and including action a; furthermore, $x[\varnothing] = 1$. Hence, in order to be a valid sequence-form strategy, x must satisfy the 'probability mass conservation' constraint: for all $I \in \mathcal{I}_i$, $\sum_{a \in A_I} x[(I, a)] = x[\sigma(I)]$. That is, every information sets partitions the

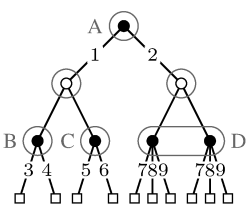


Figure 1: Small example.

probability mass received from the parent sequence onto its actions. In this sense, the constraints that define the space of sequence-form strategies naturally exhibit a hierarchical structure.

2.2 Extensive-Form Correlated Equilibria

Extensive-form correlated equilibrium (EFCE) (von Stengel & Forges, 2008) defines a natural extension of the solution concept of *correlated equilibrium (CE)* (Aumann, 1974) to the world of extensive-form games. In EFCE, a mediator privately reveals recommendations to the players as the game progresses. These recommendations are *incremental*, in the sense that recommendations for the move to play at each decision point of the game are revealed only if and when the decision point is reached. This is in contrast with CE, where recommendations *for the whole game* are privately revealed upfront when the game starts. Players are free to not follow the recommended moves, but once a player does not follow a recommendation, he will not receive further recommendations. In an EFCE, the recommendations are incentive-compatible—that is, the players are motivated to follow all recommendations. EFCE and CE are good candidates to model strategic interactions in which intermediate forms of centralized control can be achieved (Ashlagi et al., 2008).

In a recent preprint, Farina et al. (2019c) show that in two-player perfect-recall extensive-form games, an EFCE that guarantees a social welfare (that is, sum of player's utilities) at least τ can be expressed as the solution to a bilinear saddle-point problem, that is an optimization problem of the form $\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{y}\in\mathcal{Y}} \boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are convex and compact sets and $\boldsymbol{A}$ is a matrix of real numbers. In the case of EFCE, $\mathcal{X} = \Xi$ is known as the *polytope of correlation plans* (see also Section 2.3) and $\mathcal{Y}$ is the convex hull of certain sequence-form strategy spaces. In general, Ξ cannot be captured by a polynomially small set of constraints, since computing a social-welfare-maximizing EFCE in a two-player perfect-recall game is computationally hard (von Stengel & Forges, 2008, Section 3.7).[1] However, in the special case of games with no chance moves, this is not the case, and Ξ is the intersection of a polynomial (in the input game tree size) number of constraints, as discussed in the next subsection. In fact, most of the current paper is devoted to studying the structure of Ξ. We will largely ignore $\mathcal{Y}$, for which an efficient regret minimizer can already be built, for instance by using the theory of *regret circuits* (Farina et al., 2019b). Similarly, we will not use any property of matrix $\boldsymbol{A}$ (except that it can be computed and stored efficiently).

2.3 Polytope of Extensive-Form Correlation Plans in Games with no Chance Moves

In their seminal paper, von Stengel & Forges (2008) characterize the constraints that define the space of extensive-form correlation plans Ξ in the case of two-player perfect-recall games *with no chance moves*. The characterization makes use of the following two concepts:

Definition 1 (Connected information sets, $I_1 \rightleftharpoons I_2$)**.** *Let I_1, I_2 be information sets for Player 1 and 2, respectively. We say that I_1 and I_2 are* connected*, denoted $I_1 \rightleftharpoons I_2$, if there exist two nodes $u \in I_1, v \in I_2$ such that u is on the path from the root to v, or v is on the path from the root to u.*

Definition 2 (Relevant sequence pair, $\sigma_1 \bowtie \sigma_2$)**.** *Let $\sigma_1 \in \Sigma_1, \sigma_2 \in \Sigma_2$. We say that (σ_1, σ_2) is a* relevant sequence pair*, and write $\sigma_1 \bowtie \sigma_2$, if any of σ_1 or σ_2 is the empty sequence, or if the information sets to which σ_1 and σ_2 belong are connected (in the sense of Definition 1).*

Definition 3 (von Stengel & Forges (2008))**.** *In a two-player perfect-recall extensive-form game with no chance moves, the space Ξ of* correlation plans *is a convex polytope containing* nonnegative *vectors indexed over relevant sequences pairs, and is defined as*

$$\Xi := \left\{ \boldsymbol{\xi} : \begin{array}{ll} \bullet\ \xi[\varnothing, \varnothing] = 1 & \\ \bullet\ \sum_{a\in A_I} \xi[(I,a),\sigma_2] = \xi[\sigma(I),\sigma_2] & \forall I \in \mathcal{I}_1 \text{ s.t. } (I,a) \bowtie \sigma_2 \ \ \forall a \in A_I \\ \bullet\ \sum_{a\in A_J} \xi[\sigma_1,(J,a)] = \xi[\sigma_1,\sigma(J)] & \forall J \in \mathcal{I}_2 \text{ s.t. } \sigma_1 \bowtie (J,a) \ \ \forall a \in A_J \end{array} \right\}.$$

2.4 Regret Minimization and Relationship with Bilinear Saddle-Point Problems

A regret minimizer is a device that supports two operations: (i) RECOMMEND, which provides the next decision $\boldsymbol{x}^{t+1} \in \mathcal{X}$, where $\mathcal{X}$ is a nonempty, convex, and compact subset of a Euclidean space $\mathbb{R}^n$; and (ii) OBSERVELOSS, which receives/observes a convex loss function ℓ^t that is used to evaluate decision $\boldsymbol{x}^t$ (Zinkevich, 2003). For the purposes of this paper, we will only be interested in linear loss functions, which at all times t we will represent in the form of a vector $\boldsymbol{\ell}^t \in \mathbb{R}^n$. A regret minimizer is an *online* decision maker in the sense that each decision is made by taking into account only past

[1] *One* feasible EFCE can be done in polynomial time (Huang & von Stengel, 2008; Huang, 2011) using the *ellipsoid-against-hope* algorithm (Papadimitriou & Roughgarden, 2008; Jiang & Leyton-Brown, 2015). Unfortunately, that algorithm is known to not scale beyond small games.

decisions and their corresponding losses. The quality metric for the regret minimizer is its *cumulative regret* R^T, which is defined as the difference between the loss cumulated by the sequence of decisions $\boldsymbol{x}^1, \dots, \boldsymbol{x}^T$ and the loss that would have been cumulated by the *best-in-hindsight time-independent* decision $\hat{\boldsymbol{x}}$. Formally, $R^T := \sum_{t=1}^T \langle \boldsymbol{\ell}^t, \boldsymbol{x}^t \rangle - \min_{\hat{\boldsymbol{x}} \in \mathcal{X}} \sum_{t=1}^T \langle \boldsymbol{\ell}^t, \hat{\boldsymbol{x}} \rangle$. A 'good' regret minimizer has R^T sublinear in T; this property is known as *Hannan consistency*. Hannan consistent regret minimizers can be used to converge to a solution of a *bilinear saddle-point problem* (Section 2.2). To do so, two regret minimizers, one for $\mathcal{X}$ and one for $\mathcal{Y}$, are set up so that at each time t they observe loss vectors $\boldsymbol{\ell}_x^t := -\boldsymbol{A}\boldsymbol{y}^t$ and $\boldsymbol{\ell}_y^t := \boldsymbol{A}^\top \boldsymbol{x}^t$, respectively, where $\boldsymbol{x}^t \in \mathcal{X}$ and $\boldsymbol{y}^t \in \mathcal{Y}$ are the decisions output by the two regret minimizers. A well-known folk theorem asserts that in doing so, at time T the average decisions $(\bar{\boldsymbol{x}}^T, \bar{\boldsymbol{y}}^T) := (\frac{1}{T}\sum_{t=1}^T \boldsymbol{x}^t, \frac{1}{T}\sum_{t=1}^T \boldsymbol{y}^t)$ have *saddle-point gap* (a standard measure of how close a point is to being a saddle-point) $\gamma(\bar{\boldsymbol{x}}^T, \bar{\boldsymbol{y}}^T) := \max_{\hat{\boldsymbol{x}} \in \mathcal{X}} \hat{\boldsymbol{x}}^\top \boldsymbol{A} \bar{\boldsymbol{y}}^T - \min_{\hat{\boldsymbol{y}} \in \mathcal{Y}} (\bar{\boldsymbol{x}}^T)^\top \boldsymbol{A} \hat{\boldsymbol{y}}$ bounded above by $\gamma(\bar{\boldsymbol{x}}^T, \bar{\boldsymbol{y}}^T) \leq (R_{\mathcal{X}}^T + R_{\mathcal{Y}}^T)/T$ where $R_{\mathcal{X}}^T$ and $R_{\mathcal{Y}}^T$ are the cumulative regrets of the two regret minimizers. Since the regrets grow sublinearly, this is enough to conclude that $\gamma(\bar{\boldsymbol{x}}^T, \bar{\boldsymbol{y}}^T) \to 0$ as $T \to +\infty$. As discussed in the introduction, this approach has proved extremely successful in computational game theory.

3 Scaled Extension: A Convexity-Preserving Operation for Incrementally Constructing Strategy Spaces

In this section, we introduce a new convexity-preserving operation between two sets. We show that it provides an alternative way of constructing the strategy space of a player in an extensive-form game that is different from the construction based on convex hulls and Cartesian products described by Farina et al. (2019b). Our new construction enables one to *incrementally extend* the strategy space in a top-down fashion, whereas the construction by Farina et al. (2019b) was bottom-up. Most importantly, as we will show in Section 3.1, this new operation enables one to incrementally, recursively construct the extensive-form correlated strategy space (again in a top-down fashion).

Definition 4. *Let $\mathcal{X}$ and $\mathcal{Y}$ be nonempty, compact and convex sets, and let $h : \mathcal{X} \to \mathbb{R}_+$ be a nonnegative affine real function. The* scaled extension *of $\mathcal{X}$ with $\mathcal{Y}$ via h is defined as the set*

$$\mathcal{X} \overset{h}{\triangleleft} \mathcal{Y} := \{(\boldsymbol{x}, \boldsymbol{y}) : \boldsymbol{x} \in \mathcal{X},\ \boldsymbol{y} \in h(\boldsymbol{x})\mathcal{Y}\}.$$

Since we will be composing multiple scaled extensions together, it is important to verify that the operation above not only preserves convexity, but also preserves the non-emptiness and compactness of the sets (a proof of the following Lemma is available in Appendix A):

Lemma 1. *Let $\mathcal{X}, \mathcal{Y}$ and h be as in Definition 4. Then $\mathcal{X} \overset{h}{\triangleleft} \mathcal{Y}$ is nonempty, compact and convex.*

3.1 Construction of the Set of Sequence-Form Strategies

The scaled extension operation can be used to construct the polytope of a perfect-recall player's strategy in sequence-form in an extensive-form game. We illustrate the approach in the small example of Figure 1; the generalization to any extensive-form strategy space is immediate. As noted in Section 2.1, any valid sequence-form strategy must satisfy probability mass constraints, and can be constructed incrementally in a top-down fashion, as follows (in the following we refer to the same naming scheme as in Figure 1 for the sequences of Player 1):

i. First, the empty sequence is set to value $x[\varnothing] = 1$.
ii. (Info set A) Next, the value $x[\varnothing]$ is partitioned into the two non-negative values $x[1]{+}x[2]{=}x[\varnothing]$.
iii. (Info set B) Next, the value $x[1]$ is partitioned into two non-negative values $x[3] + x[4] = x[1]$.
iv. (Info set C) Next, the value $x[1]$ is partitioned into two non-negative values $x[5] + x[6] = x[1]$.
v. (Info set D) Next, the value $x[2]$ is partitioned into 3 non-negative values $x[7]{+}x[8]{+}x[9]{=}x[2]$.

The incremental choices in the above recipe can be directly translated—in the same order—into set operations by using scaled extensions, as follows:

i. First, the set of all feasible values of sequence $x[\varnothing]$ is the singleton $\mathcal{X}_0 := \{1\}$.
ii. Then, the set of all feasible values of $(x[\varnothing], x[1], x[2])$ is the set $\mathcal{X}_1 := \mathcal{X}_0 \times \Delta^2 = \mathcal{X}_0 \triangleleft^{h_1} \Delta^2$, where h_1 is the linear function $h_1 : \mathcal{X}_0 \ni x[\varnothing] \mapsto x[\varnothing]$ (the identity function).
iii. In order to characterize the set of all feasible values of $(x[\varnothing], \dots, x[4])$ we start from $\mathcal{X}_1$, and *extend* any element $(x[\varnothing], x[1], x[2]) \in \mathcal{X}_1$ with the two sequences $x[3]$ and $x[4]$, drawn from the set $\{(x[3], x[4]) \in \mathbb{R}_2^+ : x[3] + x[4] = x[1]\} = x[1]\Delta^2$. We can express this extension using scaled extension: $\mathcal{X}_2 := \mathcal{X}_1 \triangleleft^{h_2} \Delta^2$, where $h_2 : \mathcal{X}_1 \ni (x[\varnothing], x[1], x[2]) \mapsto x[1]$.

iv. Similarly, we can extend every element in $\mathcal{X}_2$ to include $(x[5], x[6]) \in x[1]\Delta^2$: in this case, $\mathcal{X}_3 := \mathcal{X}_2 \lhd^{h_3} \Delta^2$, where $h_3 : \mathcal{X}_2 \ni (x[\varnothing], x[1], x[2], x[3], x[4]) \mapsto x[1]$.

v. The set of all feasible $(x[\varnothing], .., x[9])$ is $\mathcal{X}_4 := \mathcal{X}_3 \lhd^{h_4} \Delta^3$, where $h_4 : \mathcal{X}_3 \ni (x[\varnothing], \ldots, x[6]) \mapsto x[2]$.

Hence, the polytope of sequence-form strategies for Player 1 in Figure 1 can be expressed as $\{1\} \overset{h_1}{\lhd} \Delta^2 \overset{h_2}{\lhd} \Delta^2 \overset{h_3}{\lhd} \Delta^2 \overset{h_4}{\lhd} \Delta^3$, where the scaled extension operation is intended as left associative.

3.2 Regret Minimizer for Scaled Extension

It is always possible to construct a regret minimizer for $\mathcal{Z} = \mathcal{X} \overset{h}{\lhd} \mathcal{Y}$, starting from a regret minimizer for $\mathcal{X} \subseteq \mathbb{R}^{n_1}$ and $\mathcal{Y} \subseteq \mathbb{R}^{n_2}$. The fundamental technical insight of the construction is that, given any vector $\boldsymbol{\ell} = (\boldsymbol{\ell}_x, \boldsymbol{\ell}_y) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$, the minimization of a linear function over $\mathcal{Z}$ can be broken down into two separate minimization subproblems (again of linear functions) over $\mathcal{X}$ and $\mathcal{Y}$:

$$\begin{aligned}\min_{\boldsymbol{z} \in \mathcal{Z}} \langle \boldsymbol{\ell}, \boldsymbol{z} \rangle &= \min_{\boldsymbol{x} \in \mathcal{X}, \boldsymbol{y} \in \mathcal{Y}} \left\{ \langle \boldsymbol{\ell}_x, \boldsymbol{x} \rangle + h(\boldsymbol{x}) \langle \boldsymbol{\ell}_y, \boldsymbol{y} \rangle \right\} = \min_{\boldsymbol{x} \in \mathcal{X}} \left\{ \langle \boldsymbol{\ell}_x, \boldsymbol{x} \rangle + h(\boldsymbol{x}) \min_{\boldsymbol{y} \in \mathcal{Y}} \langle \boldsymbol{\ell}_y, \boldsymbol{y} \rangle \right\} \\ &= \min_{\boldsymbol{x} \in \mathcal{X}} \left\{ \left\langle \boldsymbol{\ell}_x + \boldsymbol{a} \cdot \min_{\boldsymbol{y} \in \mathcal{Y}} \langle \boldsymbol{\ell}_y, \boldsymbol{y} \rangle, \boldsymbol{x} \right\rangle \right\} + b \cdot \min_{\boldsymbol{y} \in \mathcal{Y}} \langle \boldsymbol{\ell}_y, \boldsymbol{y} \rangle,\end{aligned}$$

where $\boldsymbol{a} \in \mathbb{R}^{n_1}$ and $b \in \mathbb{R}$ are the (unique) vectors such that the affine function h can be written as $h : \mathcal{X} \ni \boldsymbol{x} \mapsto \langle \boldsymbol{a}, \boldsymbol{x} \rangle + b$. Thus, it is possible to break the problem of minimizing regret over $\mathcal{Z}$ into two regret minimization subproblems over $\mathcal{X}$ and $\mathcal{Y}$ (more details in Appendix B). In particular:

Proposition 1. *Let $\text{RM}_{\mathcal{X}}$ and $\text{RM}_{\mathcal{Y}}$ be two regret minimizer over $\mathcal{X}$ and $\mathcal{Y}$ respectively, and let $R^T_{\mathcal{X}}, R^T_{\mathcal{Y}}$ denote their cumulative regret at time T. Then, Algorithm 1 provides a regret minimizer over $\mathcal{Z}$ whose cumulative regret $R^T_{\mathcal{Z}}$ is bounded above as $R^T_{\mathcal{Z}} \leq R^T_{\mathcal{X}} + h^* R^T_{\mathcal{Y}}$, where $h^* := \max_{\boldsymbol{x} \in \mathcal{X}} h(\boldsymbol{x})$.*

Algorithm 1 Regret minimizer over the scaled extension $\mathcal{X} \lhd^h \mathcal{Y}$.

```
function RECOMMEND()
    x^t ← RM_X.RECOMMEND()
    y^t ← RM_Y.RECOMMEND()
    return (x^t, h(x^t) y^t)

function OBSERVELOSS(ℓ^t = (ℓ^t_x, ℓ^t_y))
    y^t ← RM_Y.RECOMMEND()
    ℓ̃^t_y ← ℓ^t_x + ⟨ℓ^t_y, y^t⟩ · a
    RM_X.OBSERVELOSS(ℓ^t_x)
    RM_Y.OBSERVELOSS(ℓ̃^t_y)
```

Algorithm 1 can be composed recursively to construct a regret minimizer for any set that is expressed via a chain of scaled extensions, such as the polytope of sequence-form strategies (Section 3.1).

4 Unrolling the Structure of the Correlated Strategy Polytope

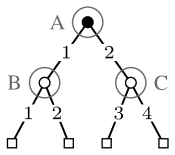
In this section, we study the combinatorial structure of the polytope of correlated strategies (Section 2.3) of a two-player perfect-recall extensive-form game with no chance moves. The central result of this section, Theorem 1, asserts that the correlated strategy polytope Ξ can be expressed via a chain of scaled extensions. This matches the similar result regarding the sequence-form strategy polytope that we discussed in Section 3.1. However, unlike the sequence-form strategy polytope, the constraints that define the correlated strategy polytope do not exhibit a natural hierarchical structure: the constraints that define Ξ (Definition 3) are such that the same entry of the correlation plan $\boldsymbol{\xi}$ can appear in multiple constraints, and furthermore the constraints will in general form cycles. This makes the problem of unrolling the structure of Ξ significantly more challenging.

The key insight is that some of the constraints that define Ξ are redundant (that is, implied by the remaining constraints) and can therefore be safely eliminated. Our algorithm identifies one such set of redundant constraints, and removes them. The set is chosen in such a way that the remaining constraints can be laid down in a hierarchical way that can be captured via a chain of scaled extensions.

4.1 Example

Before we delve into the technical details of the construction, we illustrate the key idea of the algorithm in a small example. In particular, consider the small game tree of Figure 2 (left), where we used the same conventions as in Section 2.1 and Figure 1. All sequence pairs are relevant; the set of constraints that define Ξ is shown in Figure 2 (middle).

In order to generate all possible correlation plans $\boldsymbol{\xi} \in \Xi$, we proceed as follows. First, we assign $\xi[\varnothing, \varnothing] = 1$. Then, we partition $\xi[\varnothing, \varnothing]$ into two non-negative values $(\xi[1, \varnothing], \xi[2, \varnothing]) \in \xi[\varnothing, \varnothing]\Delta^2$ in accordance with the constraint $\xi[1, \varnothing] + \xi[2, \varnothing] = \xi[\varnothing, \varnothing]$. Next, using the constraints $\xi[\sigma_1, 1] + \xi[\sigma_1, 2] = \xi[\sigma_1, \varnothing]$ and $\xi[\sigma_1, 3] + \xi[\sigma_1, 4] = \xi[\sigma_1, \varnothing]$, we pick values $(\xi[\sigma_1, 1], \xi[\sigma_1, 2]) \in \xi[\sigma_1, \varnothing]\Delta^2$ and $(\xi[\sigma_1, 2], \xi[\sigma_1, 3]) \in \xi[\sigma_1, \varnothing]\Delta^2$ for $\sigma_1 \in \{1, 2\}$. So far, our strategy for filling the



In this game, Ξ is defined by the following constraints:

$$\begin{cases} \xi[\varnothing,\varnothing] = 1, & \\ \xi[\sigma_1,1] + \xi[\sigma_1,2] = \xi[\sigma_1,\varnothing] & \forall \sigma_1 \in \{\varnothing,1,2\}, \\ \xi[\sigma_1,3] + \xi[\sigma_1,4] = \xi[\sigma_1,\varnothing] & \forall \sigma_1 \in \{\varnothing,1,2\}, \\ \xi[1,\sigma_2] + \xi[2,\sigma_2] = \xi[\varnothing,\sigma_2] & \forall \sigma_2 \in \{\varnothing,1,2,3,4\}. \end{cases}$$

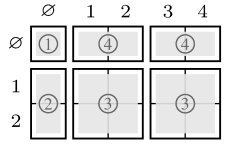


Figure 2: (Left) Example game (Section 4.1). (Middle) Constraints that define Ξ in the example game. (Right) Fill-in order of $\boldsymbol{\xi}$. The cell at the intersection of row σ_1 and column σ_2 represents the entry $\xi[\sigma_1,\sigma_2]$ of $\boldsymbol{\xi}$.

correlation plan has been to *split* entries according to the information structure of the players. As shown in Section 3.1, these steps can be expressed via scaled extension operations.

Next, we fill in the four remaining entries in ξ, that is $\xi[\varnothing,\sigma_2]$ for $\sigma_2 \in \{1,2,3,4\}$, in accordance with constraint $\xi[1,\sigma_2] + \xi[2,\sigma_2] = \xi[\varnothing,\sigma_2]$. In this step, we are not splitting any value; rather, we fill in $\xi[\varnothing,\sigma_2]$ in the only possible way (that is, $\xi[\varnothing,\sigma_2] = \xi[1,\sigma_2] + \xi[2,\sigma_2]$), by means of a linear combination of already-filled-in entries. This operation can be also expressed via scaled extensions, with the singleton set $\{1\}$: $\{(\xi[1,\sigma_2],\xi[2,\sigma_2],\xi[\varnothing,\sigma_2])\} = \{(\xi[1,\sigma_2],\xi[2,\sigma_2])\} \triangleleft^h \{1\}$, where $h : (\xi[1,\sigma_2],\xi[2,\sigma_2]) \mapsto \xi[1,\sigma_2] + \xi[2,\sigma_2]$ (note that h respects the requirements of Definition 4). This way, we have filled in all entries in ξ. However, only 9 out of the 11 constraints have been taken into account in the construction, and we still need to verify that the two leftover constraints $\xi[\varnothing,1] + \xi[\varnothing,2] = \xi[\varnothing,\varnothing]$ and $\xi[\varnothing,3] + \xi[\varnothing,4] = \xi[\varnothing,\varnothing]$ are automatically satisfied by our way of filling in the entries of $\boldsymbol{\xi}$. Luckily, this is always the case: by construction, $\xi[\varnothing,1]+\xi[\varnothing,2] = (\xi[1,1]+\xi[1,2])+(\xi[2,1]+\xi[2,2]) = \xi[1,\varnothing]+\xi[2,\varnothing] = \xi[\varnothing,\varnothing]$ (the proof for $\xi[\varnothing,3] + \xi[\varnothing,4]$ is analogous). We summarize the construction steps pictorially in Figure 2 (right).

Remark 1. *Similar construction that starts from assigning values for $\xi[\varnothing,\sigma_2]$ ($\sigma_2 \in \{1,2,3,4\}$ using constraints $\xi[\varnothing,1] + \xi[\varnothing,2] = \xi[\varnothing,\varnothing]$, $\xi[\varnothing,3] + \xi[\varnothing,4] = \xi[\varnothing,\varnothing]$ and fills out $\xi[\sigma_1,\sigma_2]$ for $(\sigma_1,\sigma_2) \in \{1,2\} \times \{1,2,3,4\}$ would have not been successful: if $(\xi[1,\varnothing],\xi[1,1])$ and $(\xi[1,2],\xi[1,3])$ are filled in independently, there is no way of guaranteeing that $\xi[1,1] + \xi[1,2] = \xi[1,3] + \xi[1,4]$ ($= \xi[1,\varnothing]$) as required by the constraints.*

4.2 An Unfavorable Case that Cannot Happen in Games with No Chance Moves

We now show that there exist game instances in which the general approach used in the previous subsection fails. In particular, consider a relevant sequence pair (σ_1,σ_2) such that both σ_1 and σ_2 are parent sequences of two information sets of Player 1 and Player 2 respectively, and assume that all sequence pairs in the game are relevant. Then, no matter what the order of operations is, the situation described in Remark 1 cannot be avoided. Luckily, in two-player perfect-recall games with no chance moves, one can prove that this occurrence never happens (see Appendix C for a proof):

Proposition 2. *Consider a two-player perfect-recall game with no chance moves, and let (σ_1,σ_2) be a relevant sequence pair, let I_1, I_1' be two distinct information sets of Player 1 such that $\sigma(I_1) = \sigma(I_1') = \sigma_1$, and let I_2, I_2' be two distinct information sets of Player 2 such that $\sigma(I_2) = \sigma(I_2') = \sigma_2$. It is not possible that both $I_1 \rightleftharpoons I_2$ and $I_1' \rightleftharpoons I_2'$.*

In other words, if $I_1 \rightleftharpoons I_2$, then any pair of sequences (σ_1',σ_2') where σ_1' belongs to I_1' and σ_2' belongs to I_2' is *irrelevant*. As we show in the next subsection, this is enough to yield a polynomial-time algorithm to 'unroll' the process of filling in the entries of $\boldsymbol{\xi} \in \Xi$ in any two-player perfect-recall extensive-form game with no chance moves. The following definition is crucial for that algorithm:

Definition 5. *Let (σ_1,σ_2) be a relevant sequence pair, and let $I_1 \in \mathcal{I}_1$ be an information set for Player 1 such that $\sigma(I_1) = \sigma_1$. Information set I_1 is called* critical for σ_2 *if there exists at least one $I_2 \in \mathcal{I}_2$ with $\sigma(I_2) = \sigma_2$ such that $I_1 \rightleftharpoons I_2$. (A symmetric definition holds for an $I_2 \in \mathcal{I}_2$.)*

It is a simple corollary of Proposition 2 that for any relevant sequence pair, *at least one* player has *at most one* critical information set for the opponent's sequence. We call such a player *critical* for that relevant sequence pair.

4.3 A Polynomial-Time Algorithm that Decomposes Ξ using Scaled Extensions

In this section, we present the central result of the paper: an efficient algorithm that expresses Ξ as a chain of scaled extensions of simpler sets. In particular, as we have already seen in Section 4.1, each set in the decomposition is either a simplex (when *splitting* an already-filled-in entry) or the singleton set $\{1\}$ (when *summing* already filled-in entries and assigning the result to a new entry of $\boldsymbol{\xi}$).

The algorithm consists of a recursive function, DECOMPOSE, which takes three arguments: a relevant sequence pair (σ_1, σ_2), a subset $\mathcal{S}$ of the set of all relevant sequence pairs, and a set $\mathcal{D}$ of vectors with entries indexed by the elements in $\mathcal{S}$. $\mathcal{S}$ represents the set of indices of ξ that have already been filled in, while $\mathcal{D}$ is the set of all partially-filled-in correlation plans (see Section 4.1). The decomposition for the whole polytope Ξ is obtained by evaluating $\text{DECOMPOSE}((\varnothing, \varnothing), \mathcal{S} = \{(\varnothing, \varnothing)\}, \mathcal{D} = \{(1)\})$, which corresponds to the starting situation in which only the entry $\xi[\varnothing, \varnothing]$ has been filled in (with the value 1 as per Definition 3). Each call to DECOMPOSE returns a pair $(\mathcal{S}', \mathcal{D}')$ of updated indices and partial vectors, to reflect the new entries that were filled in during the call.

Each call to $\text{DECOMPOSE}((\sigma_1, \sigma_2), \mathcal{S}, \mathcal{D})$ works as follows:

- First, the algorithm finds one critical player for the relevant sequence pair (σ_1, σ_2) (see end of Section 4.2). Assume without loss of generality that Player 1 is critical (the other case is symmetric), and let $\mathcal{I}^* \subseteq \mathcal{I}_1$ be the set of critical information sets for σ_2 that belong to Player 1.
- For each $I \in \mathcal{I}_1$ such that $\sigma(I) = \sigma_1$, we:
  - Fill in all entries $\{\xi[(I, a), \sigma_2] : a \in A_I\}$ by splitting $\xi[\sigma_1, \sigma_2]$. This is reflected by updating the set of filled-in-indices $\mathcal{S} \leftarrow \mathcal{S} \cup \{((I, a), \sigma_2)\}$ and extending $\mathcal{D}$ via a scaled extension: $\mathcal{D} \leftarrow \mathcal{D} \lhd^h \Delta^{|A_I|}$ where h extracts $\xi[\sigma_1, \sigma_2]$ from any partially-filled-in vector.
  - Then, for each $a \in A_I$ we assign $(\mathcal{S}, \mathcal{D}) \leftarrow \text{DECOMPOSE}(((I, a), \sigma_2), \mathcal{S}, \mathcal{D})$.

  After this step, all the indices in $\{(\sigma_1', \sigma_2') : \sigma_1' \succ \sigma_1, \sigma_2' \succeq \sigma_2\} \cup \{(\sigma_1, \sigma_2)\}$ have been filled in, and none of the indices in $\{(\sigma_1, \sigma_2') : \sigma_2' \succ \sigma_2\}$ have been filled in yet.
- Finally, we fill out all indices in $\{(\sigma_1, \sigma_2') : \sigma_2' \succ \sigma_2\}$. We do so by iterating over all information sets $J \in \mathcal{I}_2$ such that $\sigma(J) \succeq \sigma_2$. For each such J, we split into two cases, according to whether $\mathcal{I}^* = \{I^*\}$ (for some $I^* \in \mathcal{I}_1$, as opposed to $\mathcal{I}^*$ being empty) and $J \rightleftharpoons I^*$, or not:
  - If $\mathcal{I}^* = \{I^*\}$ and $J \rightleftharpoons I^*$, then for all $a \in A_J$ we fill in the sequence pair $\xi[\sigma_1, (J, a)]$ by assigning its value in accordance with the constraint $\xi[\sigma_1, (J, a)] = \sum_{a^* \in I^*} \xi[(I^*, a^*), (J, a)]$ via the scaled extension $\mathcal{D} \leftarrow \mathcal{D} \lhd^h \{1\}$ where the linear function h maps a partially-filled-in vector to the value of $\sum_{a^* \in I^*} \xi[(I^*, a^*), (J, a)]$.
  - Otherwise, we fill in the entries $\{\xi[\sigma_1, (J, a)] : a \in A_J\}$, by splitting the value $\xi[\sigma_1, \sigma(J)]$. In other words, we let $\mathcal{D} \leftarrow \mathcal{D} \lhd^h \Delta^{|A_J|}$ where h extracts the entry $\xi[\sigma_1, \sigma(J)]$ from a partially-filled-in vector in $\mathcal{D}$.
- At this point, all the entries corresponding to indices $\tilde{\mathcal{S}} = \{(\sigma_1', \sigma_2') : \sigma_1' \succeq \sigma_1, \sigma_2' \succeq \sigma_2\}$ have been filled in, and we return $(\mathcal{S} \cup \tilde{\mathcal{S}}, \mathcal{D})$.

Every call to DECOMPOSE increases the cardinality of $\mathcal{S}$ by at least one unit. Since $\mathcal{S}$ is a subset of the set of relevant sequence pairs, and since the total number of relevant sequence pair is polynomial in the input game tree size, the algorithm runs in polynomial time. Furthermore, since every change to $\mathcal{D}$ is done via scaled extensions (with either a simplex or the singleton set $\{1\}$), we conclude that:

Theorem 1. *In a two-player perfect-recall EFG with no chance moves, the space of correlation plans Ξ can be expressed via a sequence of scaled extensions with simplexes and singleton sets. An exact algorithm exists to compute such expression in polynomial time.*

See Appendix D for a proof of correctness of the algorithm.

5 Experimental Evaluation

We experimentally evaluate the scalability of our regret-minimization algorithm for computing an extensive-form correlated equilibrium. In particular, we implement a regret minimizer for the space of correlation plans by computing the structural decomposition of Ξ into a chain of scaled extensions (Section 4.3) and repeatedly applying the construction of Section 3.2. This regret minimizer is then used on the saddle-point formulation of an EFCE (Section 2.2) as explained in Section 2.4, with two modifications that are standard in the literature on regret minimization algorithms for game theory (Tammelin et al., 2015): (i) alternation and (ii) linear averaging. We use *regret-matching-plus* (Tammelin et al., 2015) to minimize the regret over the simplex domains in the structural decomposition. These variants are known to be beneficial in the case of Nash equilibrium, and we observed the same for EFCE. We compare our algorithm to two known algorithms in the literature. The first is based on linear programming (von Stengel & Forges, 2008). The second is a very recent subgradient descent algorithm for this problem (Farina et al., 2019c), which leverages a recent subgradient descent technique (Wang & Bertsekas, 2013). All algorithms were run on a machine with 16 GB of RAM and an Intel i7 processor with 8 cores. We used the Gurobi commercial solver

Board size	Num turns	Ship length	$\lvert\Sigma_1\rvert$	$\lvert\Sigma_2\rvert$	Num. rel. seq. pairs
(3, 2)	3	1	15k	47k	3.89M
(3, 2)	4	1	145k	306k	26.4M
(3, 2)	4	2	970k	2.27M	111M

Table 1: Game metrics for the different instances of the Battleship game that we test on.

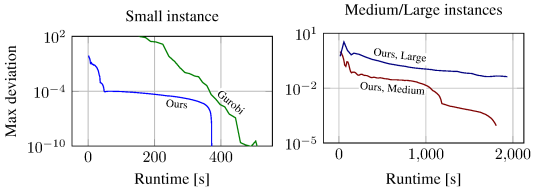


Figure 3: Experimental results. The y axis shows the maximum utility increase upon deviation.

(while allowing it to use any number of threads) to solve the LP when evaluating the scalability of the LP-based method proposed by von Stengel & Forges (2008).

Game instances. We test the scalability of our algorithm in a benchmark game for EFCE that was recently proposed by Farina et al. (2019b): a parametric variant of the classical war game *Battleship*. Table 1 shows some statistics about the three game instances that we use, including the number of relevant sequence pairs in the game (Definition 2). 'Board size' refers to the size of the Battleship playfield; each player has a field of that size in which to place his ship. 'Num turns' refers to the maximum number of shots that each player can take (in turns). 'Ship length' is the length of the one ship that each player has. Despite the seemingly small board sizes and the presence of only one ship per player, the game trees for these instances are quite large, with each player having tens of thousands to millions of sequences.

Scalability of the Linear Programming Approach (von Stengel & Forges, 2008). Only the small instance could be solved by Gurobi, Figure 3 (left). (Out of the LP algorithms provided by Gurobi, the barrier method was faster than the primal- and dual-simplex methods.) On the medium and large instance, Gurobi was killed by the system for trying to allocate too much memory. Farina et al. (2019c) report that the large instance needs more than 500GB of memory in order for Gurobi to run.

Scalability of the Very Recent Subgradient Technique (Farina et al., 2019c). The very recent subgradient descent algorithm for this problem was able to solve the small and medium instances if the algorithm's step size was tuned well. An advantage of our technique is that it has no parameters to tune. Another issue is that the iterates Ξ of the subgradient algorithm are not feasible while ours are. Furthermore, on the large instance, the subgradient technique was already essentially unusable because each iteration took over an hour (mainly due to computing the projection).

Scalability of Our Approach. We implemented the structural decomposition algorithm of Section 4.3. Our parallel implementation using 8 threads has a runtime of 2 seconds on the small instance, 6 seconds on the medium instance, and 40 seconds on the large instance (each result was averaged over 10 runs). Finally, we evaluated the performance of the regret minimizer constructed according to Section 3.2; the results are in Figure 3 (left) for the small instance and Figure 3 (right) for the medium and large instance. As expected, on the small instance, the rate of convergence of our regret minimizer (a first-order method) is slower than that of the barrier method (a second-order method). However, the barrier method incurs a large overhead at the beginning, since Gurobi spends time factorizing the constraint matrix and computing a good ordering of variables for the elimination tree. The LP-based approach could not solve the medium or large instance, while ours could. Even on the largest instance, no more than 2GB of memory was reserved by our algorithm.

6 Conclusions

We introduced the first efficient regret minimization algorithm for finding an extensive-form correlated equilibrium in large two-player general-sum games with no chance moves. This turned out to be more challenging than designing an algorithm for Nash equilibrium because the constraints that define the space of correlation plans lack the hierarchical structure of sequential strategy spaces and might even form cycles. We showed that some of the constraints are redundant and can be excluded from consideration, and presented an efficient algorithm that generates the space of extensive-form correlation plans incrementally from the remaining constraints. We achieved this decomposition via a special convexity-preserving operation that we coined *scaled extension*. We showed that a regret minimizer can be designed for a scaled extension of any two convex sets, and that from the decomposition we then obtain a global regret minimizer. Our algorithm produces feasible iterates. Experiments showed that it significantly outperforms prior approaches—the LP-based approach and a very recent subgradient descent algorithm—and for larger problems it is the only viable option.

References

Ashlagi, I., Monderer, D., and Tennenholtz, M. On the value of correlation. *Journal of Artificial Intelligence Research*, 33:575–613, 2008.

Aumann, R. Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, 1:67–96, 1974.

Bowling, M., Burch, N., Johanson, M., and Tammelin, O. Heads-up limit hold'em poker is solved. *Science*, 347(6218), January 2015.

Brown, N. and Sandholm, T. Safe and nested subgame solving for imperfect-information games. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 689–699, 2017a.

Brown, N. and Sandholm, T. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, pp. eaao1733, Dec. 2017b.

Brown, N. and Sandholm, T. Solving imperfect-information games via discounted regret minimization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

Brown, N., Kroer, C., and Sandholm, T. Dynamic thresholding and pruning for regret minimization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

Davis, T., Waugh, K., and Bowling, M. Solving large extensive-form games with strategy constraints. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

Farina, G., Kroer, C., and Sandholm, T. Regret minimization in behaviorally-constrained zero-sum games. In *International Conference on Machine Learning (ICML)*, 2017.

Farina, G., Kroer, C., and Sandholm, T. Online convex optimization for sequential decision processes and extensive-form games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2019a.

Farina, G., Kroer, C., and Sandholm, T. Regret circuits: Composabilty of regret minimizers. In *International Conference on Machine Learning (ICML)*, 2019b.

Farina, G., Ling, C. K., Fang, F., and Sandholm, T. Correlation in extensive-form games: Saddle-point formulation and benchmarks. *arXiv preprint*, 2019c.

Gordon, G. J., Greenwald, A., and Marks, C. No-regret learning in convex games. In *Proceedings of the 25th international conference on Machine learning*, pp. 360–367. ACM, 2008.

Huang, W. *Equilibrium computation for extensive games*. PhD thesis, London School of Economics and Political Science, January 2011.

Huang, W. and von Stengel, B. Computing an extensive-form correlated equilibrium in polynomial time. In *International Workshop On Internet And Network Economics (WINE)*, pp. 506–513. Springer, 2008.

Jiang, A. X. and Leyton-Brown, K. Polynomial-time computation of exact correlated equilibrium in compact games. *Games and Economic Behavior*, 91:347–359, 2015.

Koller, D., Megiddo, N., and von Stengel, B. Efficient computation of equilibria for extensive two-person games. *Games and Economic Behavior*, 14(2), 1996.

Moravčík, M., Schmid, M., Burch, N., Lisý, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M., and Bowling, M. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337), May 2017.

Papadimitriou, C. H. and Roughgarden, T. Computing correlated equilibria in multi-player games. *Journal of the ACM*, 55(3):14, 2008.

Romanovskii, I. Reduction of a game with complete memory to a matrix game. *Soviet Mathematics*, 3, 1962.

Shalev-Shwartz, S. and Singer, Y. A primal-dual perspective of online learning algorithms. *Machine Learning*, 69(2-3):115–142, 2007.

Tammelin, O., Burch, N., Johanson, M., and Bowling, M. Solving heads-up limit Texas hold'em. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

von Stengel, B. Efficient computation of behavior strategies. *Games and Economic Behavior*, 14(2): 220–246, 1996.

von Stengel, B. and Forges, F. Extensive-form correlated equilibrium: Definition and computational complexity. *Mathematics of Operations Research*, 33(4):1002–1022, 2008.

Wang, M. and Bertsekas, D. P. Incremental constraint projection-proximal methods for nonsmooth convex optimization. *SIAM J. Optim.(to appear)*, 2013.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *International Conference on Machine Learning (ICML)*, pp. 928–936, Washington, DC, USA, 2003.

Zinkevich, M., Bowling, M., Johanson, M., and Piccione, C. Regret minimization in games with incomplete information. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2007.