MIT Theory Reading Group — Fri, Apr 19th 2024

Near-optimal learning in imperfect-information sequential games

(and other combinatorial convex settings)

Gabriele Farina (✉ gfarina@mit.edu)*

1 Learning dynamics in normal-form games

The theory of learning in games fundamentally tackles the following natural question: *Can agents discover strong strategies in a game by repeating the game many times, over time improving their strategies by moving in the direction of the gradient of their utility?*

In particular, the theory of no-regret dynamics gives a quantitative meaning to the notion of "learning" by demanding that each player keep their *regret* as small as possible. To fix ideas, let's consider the usual model of normal-form games. These are multiplayer games in which each player has a finite set of strategies $\mathcal{A}_i$, and the utility of each player is a function of the strategies chosen by all players. At each repetition $t = 1, 2, ...$ of the game, each player i selects independently a *distribution* $\lambda_i^{(t)} \in \Delta(\mathcal{A}_i)$ over their strategies. Since the players' distributions are independent, the expected utility of each player is a multilinear function

$$u_i\left(\lambda_1^{(t)}, ..., \lambda_n^{(t)}\right) = U_i \cdot \lambda_1^{(t)} \cdot ... \cdot \lambda_n^{(t)}.$$

Given a node of play, the *regret* of each player i is the difference between the utility they would have obtained by playing a single, fixed distribution in hindsight, and the utility they actually obtained by playing a distribution over strategies, that is,

$$\mathrm{Reg}_i^{(T)} \coloneqq \max_{\hat{\lambda}_i \in \Delta(\mathcal{A}_i)} \left\{ \sum_{t=1}^{T} u_i\left(\hat{\lambda}_i, \lambda_{-i}^{(t)}\right) - u_i\left(\lambda_i^{(t)}, \lambda_{-i}^{(t)}\right) \right\}.$$

The attractiveness of regret as a natural performance metric for learning in games stems from the following connections with equilibrium computation.

Theorem 1.1. In a two-player zero-sum normal-form game, the product of the average distributions of play is an ε-approximate *Nash equilibrium*, where

$$\varepsilon \le \frac{1}{T} \sum_{i \in \{1,2\}} \mathrm{Reg}_i^{(T)}.$$

This connection was pivotal in the development of superhuman AI in games, since learning dynamics are typically first-order iterative methods that can be implemented and scaled much more efficiently than linear programming.

*I am grateful for any reports of typos.

However, the connection between regret and equilibrium is not limited to zero-sum games. In fact, the connection extends to general-sum games as well, albeit with a slightly different guarantee.

Theorem 1.2. In an n-player general-sum normal-form game, the average product distribution of play is an ε-approximate *coarse-correlated equilibrium*, where

$$\varepsilon \leq \frac{1}{T} \max_{i \in [n]} \mathrm{Reg}_i^{(T)}.$$

(We remark the inversion between "average" and "product", and switch from sum to maximum.)

1.1 Multiplicative weights update and optimism

Arguably, the best-studied algorithm for learning in games is the *multiplicative weights update* (MWU) algorithm. MWU prescribes that each player updates their distribution over strategies by multiplying it by a factor that is proportional to the exponential of the utility gradient. The algorithm is simple and has been shown to have strong guarantees in the context of no-regret learning in games. It is optimal assuming that the other players might be playing adversarially, that is, with the sole intent of increasing player i's regret.

However, learning in games is usually a more benign setting, despite its obvious nonstationarity due to the concurrent learning of all agents. In particular, when all players are trying to learn, one would expect that their strategies are somewhat *predictable*, and that faster rates might be developed. This intuition started a long line of research, which today is best represented by the theory of *optimism* in learning dynamics. (More bibliographic remarks are available at the end.)

To fast forward to the current state of the art, the *optimistic* multiplicative weights update (OMWU) algorithm is a simple extension of MWU, with superior regret guarantees. The difference with MWU simply resides in the addition of a *momentum term* to the gradient. We present it next.

OMWU(player i):

$\lambda_i^{(1)} :=$ uniform distrib. over the player's strategy set $\mathcal{A}_i$

$g_i^{(0)} := 0 \in \mathbb{R}^{\mathcal{A}_i}$

for $t = 1, 2, \ldots$ **do**

  play strategy $\lambda_i^{(t)}$

  compute gradient $g_i^{(t)} := \nabla_{\lambda_i} u_i\left(\lambda_1^{(t)}, \ldots, \lambda_n^{(t)}\right)$

  compute optimistic gradient $\tilde{g}_i^{(t)} := 2g_i^{(t)} - g_i^{(t-1)}$ // Non-optimistic is $\tilde{g}_i^{(t)} := g_i^{(t)}$

  update $\lambda_i^{(t+1)}[a] \propto \lambda_i^{(t)}[a] \cdot \exp\left\{\eta \cdot \tilde{g}_i^{(t)}[a]\right\} \quad \forall a \in \mathcal{A}_i$

1.2 Guarantees and near-optimal regret rates

When each player $i \in [n]$ learns using OMWU with the same learning rate $\eta > 0$, the following strong properties are known to hold. For simplicity, we assume without loss of generality that all utilities in the game are in the range $[0, 1]$ (if not, rescaling all utilities to satisfy the condition leaves the equilibria unchanged).

Theorem 1.3 ($O(T^{1/4})$ per-player regret; [Syr+15]). For all T, if $\eta = \frac{T^{-1/4}}{\sqrt{n-1}}$, the regret of each player $i \in [n]$ is bounded as

$$\mathrm{Reg}_i^{(T)} \leq (4 + \log|\mathcal{A}_i|)\sqrt{n-1} \cdot T^{1/4}.$$

By Theorem 1.2, this implies convergence to the set of coarse-correlated equilibria at a rate of $O_T(T^{-3/4})$.

Theorem 1.4 (Near-optimal per-player regret; [DFG21]). There exist universal constants $C, C' > 1$ so that, for all T, if $\eta \leq \frac{1}{Cn\log^4 T}$, the regret of each player $i \in [n]$ is bounded as

$$\mathrm{Reg}_i^{(T)} \leq \frac{\log|\mathcal{A}_i|}{\eta} + C' \log T.$$

By Theorem 1.2, this implies convergence to the set of coarse-correlated equilibria at a rate of $O_T(\log^4(T)/T)$.

Theorem 1.5 (Optimal regret sum; [Syr+15]). If $\eta \leq \frac{1}{\sqrt{8}(n-1)}$, at all times T the *sum* of the players' regrets satisfies

$$\sum_{i\in[n]} \mathrm{Reg}_i^{(T)} \leq \frac{n}{\eta} \max_{i\in[n]} \log|\mathcal{A}_i|.$$

By Theorem 1.1, this implies convergence to the set of Nash equilibria in two-player zero-sum games at a rate of $O_T(1/T)$.

1.3 The question for this talk

The analysis of the OMWU algorithm, especially when it comes to the more advanced results such as the one by Daskalakis, C., Fishelson, M., & Golowich, N. [DFG21] mentioned in Theorem 1.4, is extremely tied to the structure of normal-form games. What can be said about learning dynamics for more expressive classes of games? Is it possible to extend the guarantees of OMWU to more general settings?

When I set out to answer these questions, I was particularly interested in extending the guarantees of OMWU to the setting of *imperfect-information sequential games*. These games are significantly more complex than normal-form games, in that they also capture turns and imperfect information. Equilibria in these games are much more difficult to compute and require misdirecting behavior (such as bluffing) at equilibrium. *Can we extend the guarantees of OMWU to the setting of imperfect-information sequential games and other combinatorially structured games?*

2 Extensive-form games (EFGs)

The standard representation of an imperfect-information extensive-form game is through its *game tree*, which formalizes the interaction of the players as a directed tree. In the game tree, each non-terminal node belongs to exactly one player, who acts at the node by picking one of the outgoing edges (each labeled with an action name). Imperfect information is captured in this representation by partitioning the nodes of each player into sets (called *information sets*) of nodes that are indistinguishable to that player given his or her observations.

2.1 The model

We now provide more details and notation regarding extensive-form games.

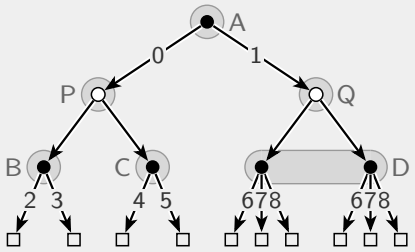
■ **Histories, actions, and payoffs**. The *game tree* represents the strategic interaction of players as a finite directed tree. Each node that is not a leaf of the game tree is associated with a unique acting player. In an n-player game, the set of valid players is the set $[n] = \{1, ..., n\}$. One of these players might serve the role of a *chance* (or *nature*) *player*—a fictitious player that selects actions according to a *known, fixed* probability distribution and models exogenous stochasticity of the environment (for example, a roll of the dice, or drawing a card from a deck). Such a chance player does not participate in the learning and has no incentives (that is, it is excluded from consideration when defining equilibria).

The players keep acting until a leaf of the game tree—called a *terminal node*—is reached. Terminal nodes are not associated with any acting player; the set of terminal nodes is denoted $\mathcal{Z}$. When the game transitions to a terminal node $z \in \mathcal{Z}$, each player $i \in [n]$ receives a *payoff* according to some payoff function $\mathcal{Z} \to \mathbb{R}$.

■ **Imperfect information and information sets**. To model imperfect information, the nodes of each player $i \in [n]$ are partitioned into a collection $\mathcal{I}_i$ of so-called *information sets*. Each information set $I \in \mathcal{I}_i$ groups together nodes that Player i cannot distinguish between when he or she acts there. In the limit case in which all information sets are a singleton, the player never has any uncertainty about which node they are acting at, and the game is said to have *perfect information*.

Since a player always knows what actions are available at a decision node, any two nodes h, h' belonging to the same information set I must have the same set of available actions. Correspondingly, we can write $\mathcal{A}_I$ to denote the set of actions available at any node that belongs to information set I.

Example 2.1. For example, consider the following two-player game, where black nodes belong to Player 1 and white nodes belong to Player 2.



The information set D for Player 1 encodes the fact that Player 1 does not observe Player 2′s action at Q.

Since nodes in the same information set are indistinguishable to the player, every strategy for the player must select the *same action* at all nodes in the same information set.

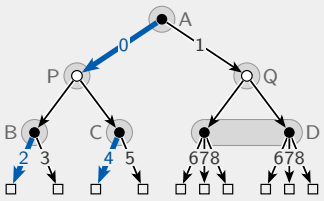
■ **Perfect recall**. As is standard in the literature, we assume that the game has *perfect recall*, that is, information sets satisfy the the fact that no player forgets about their actions, and about information once acquired.[1]

[1]The condition of perfect recall is formalized as follows. A player $i \in [n]$ is said to have *perfect recall* if, for any information set $I \in \mathcal{I}_i$, for any two nodes $h, h' \in I$ the sequence of Player i's actions encountered along the path from the root to h and from the root to h' must coincide (or otherwise, Player i would be able to distinguish among the nodes, since the player remembers all of the actions they played in the past). The game is perfect recall if all players have perfect recall.

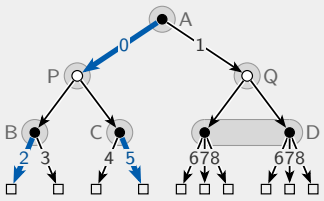
2.2 Reducing an extensive-form game to a normal-form game

What does it mean to have a mixed strategy for an extensive-form game? One classical answer is the following. Consider a player, and imagine enumerating all their deterministic strategies for the tree. A mixed strategy is then a probability distribution over these deterministic strategies.

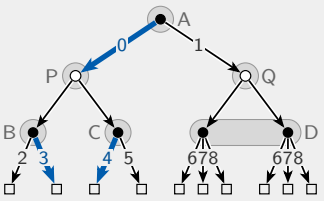
Example 2.2. In the small game of Example 2.1, a mixed strategy for Player 1 is a probability distribution over the following 7 strategies $v_1, ..., v_7$.



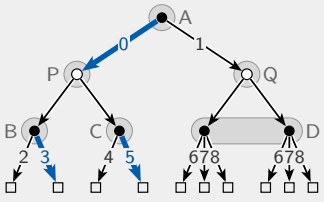
1 2 3 4 5 6 7 8 9
$v_1 := (1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0)$



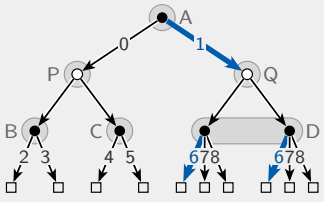
1 2 3 4 5 6 7 8 9
$v_2 := (1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0)$



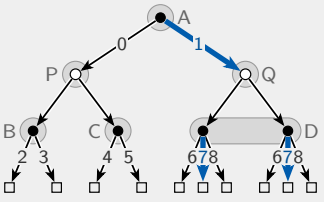
1 2 3 4 5 6 7 8 9
$v_3 := (1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0)$



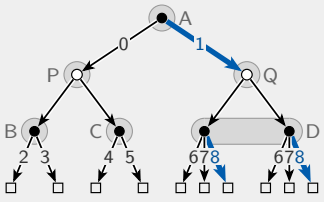
1 2 3 4 5 6 7 8 9
$v_4 := (1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0)$



1 2 3 4 5 6 7 8 9
$v_5 := (0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0)$



1 2 3 4 5 6 7 8 9
$v_6 := (0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0)$



1 2 3 4 5 6 7 8 9
$v_7 := (0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1)$

These strategies are called the *reduced*[2] *normal-form plans* of the player. We denote the set of all deterministic strategies of Player i as $\mathcal{V}_i$.

By considering the normal-form game in which each player's strategy space is the set of all deterministic strategies in the tree, we have converted the extensive-form game into its *normal-form equivalent*.

Of course, the most glaring issue with this representation is that the number of strategies in the normal-form game is exponential in the size of the game tree. For this reason, it was long believed that operating with this normal-form representation was computationally infeasible. Historically, this led to the development of specialized algorithms for extensive-form games, such as CFR and its variants. As we show next, this belief was unfounded. In fact, it is possible to simulate the OMWU dynamics in the normal-form equivalent of an extensive-form game exactly in polynomial time.

[2]The term "reduced" refers to the fact that no actions are specified for parts of the games not reached due to decisions of the player in higher parts of the game tree.

3 Learning in the normal form: Kernelized multiplicative weights for 0/1-polyhedral games

Contrary to the common belief that learning in the normal-form equivalent of an extensive-form game is computationally infeasible, we show that the OMWU algorithm can be implemented in polynomial time in the normal-form equivalent of an extensive-form game.

3.1 The convex game structure of the normal-form equivalent

The key insight is that the payoff matrix of the normal-form equivalent is heavily structured. In particular, we show that the specific representation of strategies used in Example 2.2 allows us to write the expected utility of each player as a *multilinear function* of the expected strategies of all players. (We call a game with this property a *convex* game.)

Theorem 3.1. Given any choice of deterministic strategies $v_1, ..., v_n$ for each player, the utility of each player in the game is a multilinear function

$$u_i(v_1, ..., v_n) = U_i \cdot v_1 \cdot ... \cdot v_n$$

of the strategies, where the tensor U_i has a number of nonzeros upper bounded by the number of leaves of the game tree.

Proof. For any player's strategy $v \in \mathcal{V}_i$, we have that $v[k] = 1$ if and only if all actions of Player i on the path from the root of the game down to edge k included are selected. Hence, the indicator of whether a terminal node $z \in \mathcal{Z}$ is reached is the product of one entry of v_j for all $j \in [n]$. Considering that the utility of the player is defined as the sum over all terminal nodes of the indicator of whether the terminal node is reached times the player's utility at the terminal node, we obtain the result. □

Let now $\lambda_i \in \Delta(\mathcal{V}_i)$ be independent mixed strategies for each player i. Then, the expected utility of each player i is the mulitilinear function of the *expected normal-form strategies*

$$\begin{aligned}\mathbb{E}_{v_j \sim \lambda_j}[u_i(v_1, ..., v_n)] &= U_i \cdot \mathbb{E}_{v_1 \sim \lambda_1}[v_1] \cdot ... \cdot \mathbb{E}_{v_n \sim \lambda_n}[v_n] \\ &= U_i \cdot \left(\sum_{v_1 \in \mathcal{V}_1} \lambda_1[v_1] v_1 \right) \cdot ... \cdot \left(\sum_{v_n \in \mathcal{V}_n} \lambda_n[v_n] v_n \right).\end{aligned} \tag{1}$$

So, in particular, the gradient of the expected utility of each player i with respect to their mixed strategy λ_i is

$$\nabla_{\lambda_i} \mathbb{E}_{v_j \sim \lambda_j}[u_i(v_1, ..., v_n)] = \begin{pmatrix} \vdots \\ \langle g_i, v \rangle \\ \vdots \end{pmatrix}_{v \in \mathcal{V}_i},$$

where

$$\mathbb{R}^{d_i} \ni g_i \coloneqq U_i \cdot \overline{v}_1 \cdot ... \cdot \overline{v}_{i-1} \cdot \overline{v}_{i+1} \cdot ... \cdot \overline{v}_n, \qquad \text{where} \qquad \overline{v}_j \coloneqq \left(\sum_{v_j \in \mathcal{V}_j} \lambda_j[v_j] v_j \right).$$

With this in mind, the OMWU algorithm over the strategies in the extensive-form games can be written in the following form.

EFG-OMWU(player i):

$\lambda_i^{(1)} :=$ uniform distribution over $\mathcal{V}_i$

$g_i^{(0)} := 0 \in \mathbb{R}^{\mathcal{V}_i}$

for $t = 1, 2, \ldots$ **do**

  play strategy $\lambda_i^{(t)} \in \Delta(\mathcal{V}_i)$

  compute expectations $\overline{v}_j^{(t)} := \sum_{v_j \in \mathcal{V}_j} \left(\lambda_j^{(t)}[v_j] \cdot v_j \right) \ \forall j \in [n]$ // ???

  compute gradient $g_i^{(t)} := U_i \cdot \overline{v}_1^{(t)} \ldots \cdot \overline{v}_{i-1}^{(t)} \cdot \overline{v}_{i+1}^{(t)} \cdot \ldots \cdot \overline{v}_n^{(t)}$ // Poly time in tree size

  compute optimistic gradient $\tilde{g}_i^{(t)} := 2g_i^{(t)} - g_i^{(t-1)}$ // Poly time in tree size

  update $\lambda_i^{(t+1)}[v] \propto \lambda_i^{(t)}[v] \cdot \exp\left\{ \eta \cdot \left\langle \tilde{g}_i^{(t)}, v \right\rangle \right\}$ for all $v \in \mathcal{V}_i$ // ???

3.2 Main result

For ease of notation, from now we will focus on a generic player $i \in [n]$, and drop all subscripts i from the notation. The main result that we seek to prove is the following.

Theorem 3.2 (Main theorem). There exists a *kernel function* $K_{\mathcal{V}} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, which depends on the combinatorial structure of the reduced normal-form strategies $\mathcal{V}$ of the player, such that the EFG-OMWU can be implemented using $d + 1$ evaluations of $K_{\mathcal{V}}$ at each iteration.

Note that Theorem 3.2 is crucially independent on the number of strategies $|\mathcal{V}|$, and only depends on the *number of player's actions* d (which is polynomial in the size of the input game tree)!

Hence, as long as the kernel function can be evaluated efficiently, then EFG-OMWU can be simulated efficiently too.

3.3 The 0/1-polyhedral kernel

In order to introduce the notion of 0/1-polyhedral kernel, we need to first introduce the notion of 0/1-polyhedral feature map.

Definition 3.1 (0/1-polyhedral feature map). The 0/1-polyhedral feature map $\phi_{\mathcal{V}}$ is defined as

$$\phi_{\mathcal{V}} : \mathbb{R}^d \to \mathbb{R}^{\mathcal{V}}, \qquad \phi_{\mathcal{V}}(x)[v] := \prod_{k : v[k]=1} x[k] \qquad \forall v \in \mathcal{V}.$$

As is common with kernels, we define the 0/1-polyhedral kernel as the inner product of the feature maps.

Definition 3.2 (0/1-polyhedral kernel).

$$K_{\mathcal{V}} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}, \qquad K_{\mathcal{V}}(x, y) := \langle \phi_{\mathcal{V}}(x), \phi_{\mathcal{V}}(y) \rangle = \sum_{v \in \mathcal{V}} \prod_{k : v[k]=1} x[k]y[k].$$

3.4 Keeping track of the distribution over vertices

We start from showing how the 0/1-polyhedral kernel can help implement Line 8 of EFG-OMWU efficiently. The key insight is that the strategies $\lambda^{(t)}$ computed by EFG-OMWU are fully captured by the feature map of a low-dimensional vector at all iterations.

Theorem 3.3. At all times t, the distribution $\lambda^{(t)}$ over strategies $\mathcal{V}$ computed by EFG-OMWU is proportional to the feature map of the vector

$$b^{(t)} := \exp\left\{\eta \sum_{\tau=1}^{t-1} \tilde{g}^{(\tau)}\right\}.$$

Proof. We prove the result by induction over the time t.

- Base case. At time $t = 1$, we have

$$b^{(1)} = 1 \qquad \Longrightarrow \qquad \phi_{\mathcal{V}}\big(b^{(1)}\big) = 1 \propto \frac{1}{|\mathcal{V}|} = \lambda^{(1)}.$$

- Inductive step. At time $t+1$, the probability of the strategy $v \in \mathcal{V}$ computed by EFG-OMWU is

$$\lambda^{(t+1)}[v] \propto \lambda^{(t)}[v] \cdot \exp\big\{\eta \big\langle \tilde{g}^{(t)}, v \big\rangle\big\}.$$

  The key insight now is that since $v \in \{0,1\}^d$, then

$$\exp\big\{\eta \big\langle \tilde{g}^{(t)}, v \big\rangle\big\} = \exp\left\{\eta \sum_{k : v[k]=1} \tilde{g}^{(t)}[k]\right\} = \prod_{k : v[k]=1} \exp\big\{\eta \tilde{g}^{(t)}[k]\big\}.$$

  Substituting the inductive hypothesis,

$$\begin{aligned}
\lambda^{(t+1)}[v] &\propto \phi_{\mathcal{V}}\big(b^{(t)}\big)[v] \cdot \prod_{k : v[k]=1} \exp\big\{\eta \tilde{g}^{(t)}[k]\big\} \\
&= \left(\prod_{k : v[k]=1} b^{(t)}[k]\right) \cdot \left(\prod_{k : v[k]=1} \exp\big\{\eta \tilde{g}^{(t)}[k]\big\}\right) \\
&= \prod_{k : v[k]=1} b^{(t+1)}[k] \\
&= \phi_{\mathcal{V}}\big(b^{(t+1)}\big)[v],
\end{aligned}$$

  completing the proof. □

In fact, we can even slightly refine the previous result by quantifying exactly the proportionality constant. We do so in the next corollary.

Corollary 3.1. At all times t, one has

$$\lambda^{(t)} = \frac{\phi_{\mathcal{V}}\big(b^{(t)}\big)}{K_{\mathcal{V}}\big(b^{(t)}, 1\big)}.$$

Proof. Since we know from Theorem 3.3 that $\lambda^{(t)}[v] \propto \phi_{\mathcal{V}}(b^{(t)})[v]$, and the sum $\sum_{v\in\mathcal{V}} \lambda^{(t)}[v] = 1$, the proportionality constant must be the inverse of

$$\sum_{v\in\mathcal{V}} \phi_{\mathcal{V}}(b^{(t)})[v] = \sum_{v\in\mathcal{V}} \phi_{\mathcal{V}}(b^{(t)})[v]\cdot 1 = \sum_{v\in\mathcal{V}} \phi_{\mathcal{V}}(b^{(t)})[v]\cdot \phi_{\mathcal{V}}(1)[v] = K_{\mathcal{V}}(b^{(t)},1),$$

completing the proof. □

3.5 Reconstructing the expectation

We now show how one can reconstruct the expectation

$$\sum_{v\in\mathcal{V}} (\lambda^{(t)}[v]\cdot v)$$

which is needed to compute the utility gradient in the EFG-OMWU algorithm (Line 5). The key is provided in the next theorem, which extends a nice insight by Takimoto, E., & Warmuth, M. K. [TW03] .

Theorem 3.4. At all times t, the expectation $\sum_{v\in\mathcal{V}} \lambda^{(t)}[v]\cdot v$ can be computed via $d+1$ kernel computations according to the formula

$$\sum_{v\in\mathcal{V}} (\lambda^{(t)}[v]\cdot v) = \left(1 - \frac{K_{\mathcal{V}}(b^{(t)},\overline{e}_1)}{K_{\mathcal{V}}(b^{(t)},1)}, ..., 1 - \frac{K_{\mathcal{V}}(b^{(t)},\overline{e}_d)}{K_{\mathcal{V}}(b^{(t)},1)}\right),$$

where

$$\overline{e}_k := (1,...,1,0,1,...,1) = 1 - e_k \in \mathbb{R}^d$$

is the vector that is equal to 1 in all coordinates except the i-th one where it is zero.

Proof. The key insight is that, for all $k \in [d]$,

$$\phi_{\mathcal{V}}(\overline{e}_k)[v] = \prod_{j:v[j]=1} \overline{e}_{k[j]} = \begin{cases} 0 \text{ if } v[k]=1 \\ 1 \text{ otherwise} \end{cases}$$
$$= 1 - v[k].$$

So,

$$(\phi_{\mathcal{V}}(1) - \phi_{\mathcal{V}}(\overline{e}_k))[v] = \phi_{\mathcal{V}}(1)[v] - \phi_{\mathcal{V}}(\overline{e}_k)[v] = 1 - (1 - v[k]) = v[k],$$

and we can write, for all $k \in [d]$,

$$\left(\sum_{v\in\mathcal{V}} \lambda^{(t)}[v]\cdot v\right)[k] = \sum_{v\in\mathcal{V}} (\lambda^{(t)}[v]\cdot(\phi_{\mathcal{V}}(1) - \phi_{\mathcal{V}}(\overline{e}_k))[v])$$
$$= \frac{1}{K_{\mathcal{V}}(b^{(t)},1)} \sum_{v\in\mathcal{V}} \phi_{\mathcal{V}}(b^{(t)})[v]\cdot(\phi_{\mathcal{V}}(1)[v] - \phi_{\mathcal{V}}(\overline{e}_k)[v])$$
$$= \frac{1}{K_{\mathcal{V}}(b^{(t)},1)} (K_{\mathcal{V}}(b^{(t)},1) - K_{\mathcal{V}}(b^{(t)},\overline{e}_k)) = 1 - \frac{K_{\mathcal{V}}(b^{(t)},\overline{e}_k)}{K_{\mathcal{V}}(b^{(t)},1)},$$

which is the statement. □

3.6 Kernel computation in EFGs

Shows that, as long as the kernel can be computed efficiently, then the expectation can be computed efficiently as well. In particular, as we will see in `todo`{thm:komwu in imperfect-information extensive-form games}, KOMWU can be implemented with linear-time iterations in the number of sequences d_i.

■ **Worst-case linear complexity for a single evaluation.** We start by verifying that the sequence-form kernel can be evaluated in linear time for any pair of points $x, y \in \mathbb{R}^{d_i}$. To do so, we introduce a *partial kernel function* $K_I : \mathbb{R}^{d_i} \times \mathbb{R}^{d_i} \to \mathbb{R}$ for every information set $I \in \mathcal{I}$,

$$K_I : \mathbb{R}^{d_i} \times \mathbb{R}^{d_i} \to \mathbb{R}, \qquad K_I(x, y) \coloneqq \sum_{v \in \mathcal{V}_{\succcurlyeq I}} \prod_{k : v[k]=1} x[k]y[k].$$

where $\mathcal{V}_{\succcurlyeq I}$ denotes the projection of the strategy set $\mathcal{V}$ onto only those actions at I or below (removing duplicates). We have the following.

Theorem 3.5. For any vectors $x, y \in \mathbb{R}^{d_i}$, the two following recursive relationships hold:

$$K_{\mathcal{V}}(x, y) = x[\varnothing]y[\varnothing] \prod_{I \in \mathcal{I}} K_I(x, y), \tag{2}$$

and, for all information sets $I \in \mathcal{I}$,

$$K_I(x, y) = \sum_{a \in \mathcal{A}(I)} \left(x[Ia]y[Ia] \prod_{I' \in \mathcal{C}(I)} K_{I'}(x, y) \right), \tag{3}$$

where $\mathcal{C}(I)$ denotes those information sets that are immediate successors of I. In particular, (2) and (3) give a recursive algorithm to evaluate the polyhedral kernel $K_{\mathcal{V}}$ associated with the strategy space of any player i in an imperfect-information extensive-form game in linear time in d_i.

Corollary 3.2. For each player i, EFG-OMWU can be implemented in polynomial time in the size of the game tree. As a consequence, all the regret guarantees of OMWU can be extended to the setting of imperfect-information extensive-form games as a black box.

We also make the following tangential remark.

Remark 3.1. Surprisingly, even for the non-optimistic version the kernelized MWU algorithm achieves better regret bounds than all prior algorithms for learning in extensive-form games.

■ **Amortizated constant-time kernel computation.** We can actually refine the result above by showing an implementation of EFG-OMWU with *linear-time* per-iteration complexity in the size of the game tree, by exploiting the structure of the particular set of kernel evaluations needed at every iteration and amortizing computation of the $d + 1$ kernels required by EFG-OMWU at each iteration.

3.7 Applicability beyond extensive-form games

The kernelized approach we have introduced is not limited to extensive-form games. In fact, it can be applied to any 0/1-polyhedral game, that is, a game where

- Each player i has a strategy set $\mathcal{V}_i \subseteq \{0, 1\}^{d_i}$; and

- The expected utility of each player is multilinear in the expectation of the

strategies of all players.

Kernelized OMWU guarantees that for each player, $d_i + 1$ kernel computations are sufficient at each iteration to recover the expectated strategy and simulate OMWU in the normal-form game defined by the $\mathcal{V}_i$. In the original paper, we show that the kernel can be computed efficiently for a number of combinatorial domains.

■ **Binary strings**. When $\mathcal{V} = \{0,1\}^d$, then

$$K_{\mathcal{V}}(x, y) = (1 + x[1]\, y[1])(1 + x[2]\, y[2]) \cdot \ldots \cdot (1 + x[d]\, y[d])$$

can be evaluated in linear time in d.

■ **N-sets**. When $\mathcal{V} = \left\{x \in \{0,1\}^d : \|x\|_1 = n\right\}$, the kernel can be computed efficiently via dynamic programming.

■ **Set of flows in a DAG**. In this case, the kernel can be computed in linear time in the number of DAG edgs by performing dynamic programming on the topological order of the DAG. This case was already covered by Takimoto, E., & Warmuth, M. K. [TW03], though we remark that in kernelized OMWU the concept of weight pushing is not needed, so the final algorithms are slightly different.

4 Further material and bibliography

The utilities $u^{(t)}$ that arise while learning games are structured and often have nice properties—for example changing slowly over time. It is then natural to wonder what improved guarantees can be achieved in games specifically, and consequently how fast convergence to equilibrium can be guaranteed. This fundamental question was first formulated and addressed by Daskalakis, C., Deckelbaum, A., & Kim, A. [DDK11] within the context of *zero-sum games*. Since then, there has been a considerable interest in extending their guarantee to more general settings (Chen, X., & Peng, B. [CP20]; Daskalakis, C., & Golowich, N. [DG22]; Foster, D. J., Li, Z., Lykouris, T., Sridharan, K., & Tardos, É. [Fos+16]; Piliouras, G., Sim, R., & Skoulakis, S. [PSS22]; Rakhlin, S., & Sridharan, K. [RS13]; Syrgkanis, V., Agarwal, A., Luo, H., & Schapire, R. E. [Syr+15]). As mentioned, Daskalakis, C., Fishelson, M., & Golowich, N. [DFG21] established that when all players in a general *normal-form* game employ OMWU, the regret of each player grows *nearly-optimally* as $O(\log^4 T)$ after T repetitions of the game, leading to an *exponential improvement* over the guarantees obtained using traditional techniques within the no-regret framework.

The kernelized OMWU algorithm we have introduced in this talk was introduced in Farina, G., Lee, C.-W., Luo, H., & Kroer, C. [Far+22].

Bibliography

[Syr+15] V. Syrgkanis, A. Agarwal, H. Luo, and R. E. Schapire, "Fast convergence of regularized learning in games," in *Advances in Neural Information Processing Systems*, 2015.

[DFG21] C. Daskalakis, M. Fishelson, and N. Golowich, "Near-Optimal No-Regret Learning in General Games," *CoRR*, 2021.

[TW03] E. Takimoto and M. K. Warmuth, "Path kernels and multiplicative updates," *Journal of Machine Learning Research*, vol. 4, pp. 773–818, 2003.

[DDK11] C. Daskalakis, A. Deckelbaum, and A. Kim, "Near-optimal no-regret algorithms for zero-sum games," in *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2011.

[CP20] X. Chen and B. Peng, "Hedging in games: Faster convergence of external and swap regrets," in *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[DG22] C. Daskalakis and N. Golowich, "Fast rates for nonparametric online learning: from realizability to learning in games," in *Proceedings of the Annual Symposium on Theory of Computing (STOC)*, 2022.

[Fos+16] D. J. Foster, Z. Li, T. Lykouris, K. Sridharan, and É. Tardos, "Learning in Games: Robustness of Fast Convergence," in *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2016.

[PSS22] G. Piliouras, R. Sim, and S. Skoulakis, "Beyond Time-Average Convergence: Near-Optimal Uncoupled Online Learning via Clairvoyant Multiplicative Weights Update," in *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[RS13] S. Rakhlin and K. Sridharan, "Optimization, learning, and games with predictable sequences," in *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2013.

[Far+22] G. Farina, C.-W. Lee, H. Luo, and C. Kroer, "Kernelized Multiplicative Weights for 0/1-Polyhedral Games: Bridging the Gap Between Learning in Extensive-Form and Normal-Form Games," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2022. [Online]. Available: `https://proceedings.mlr.press/v162/farina22a/farina22a.pdf`