MIT 6.S890 — Topics in Multiagent Learning

Lecture 8
Learning in games: Φ-regret minimization

Instructor: Prof. Gabriele Farina

As we have seen, external regret minimization is a very narrow instantiation of Φ-regret minimization—perhaps the smallest sensible instantiation. Then, clearly, the problem of coming up with a regret minimizer for a set $\mathcal{X}$ cannot be harder than the problem of coming up with a Φ-regret minimizer for $\mathcal{X}$ for richer sets of transformation functions Φ. It might then seem surprising that there exists a construction that reduces Φ-regret minimization to regret minimization. We see this general construction in Section 2. First, we start with an example of this principle, whose discovery predates the general construction of Section 2, and which happens to be a special case of it.

1 Blum-Mansour's swap regret minimization algorithm

Let's start by considering the task of constructing a swap regret minimizer for probability simplex $\mathcal{X} = \Delta^n$ over n actions $A = \{1, ..., n\}$. As we mentioned in Lecture 4, when all players in a normal-form game play according to the strategies generated by such an algorithm, the average product strategy converges to the set of correlated equilibria of the game. The swap regret minimization algorithm of [BM07] is a simple and elegant algorithm that minimizes *swap* regret by starting from multiple copies of an *external* regret minimizer.

Recall that in swap regret minimization, the learner wants to minimize the Φ-regret with respect to the set of strategy transformations Φ represented by stochastic matrices

$$\Phi := \left\{ P = \left(\begin{bmatrix} | \\ p_1 \\ \downarrow \end{bmatrix} \begin{bmatrix} | \\ p_2 \\ \downarrow \end{bmatrix} \cdots \begin{bmatrix} | \\ p_n \\ \downarrow \end{bmatrix} \right) : p_1, ..., p_n \in \Delta^n \right\}.$$

In order to construct a swap regret minimizer for $\mathcal{X} = \Delta^n$, we start with $|A|$ copies of an external regret minimizer for Δ^n, denoted by $\mathcal{R}_i$.

- To compute the `NextStrategy()` of the swap regret minimizer, we first call the `NextStrategy()` of each $\mathcal{R}_i$ to obtain a distribution $p_i^{(t)} \in \Delta^n$. We will then assemble all $p_i^{(t)}$ into a stochastic matrix $P^{(t)}$, and will output a fixed point

$$x^{(t)} = P^{(t)} x^{(t)} = \sum_{i=1}^{n} x_i^{(t)} p_i^{(t)} \in \Delta^n.$$

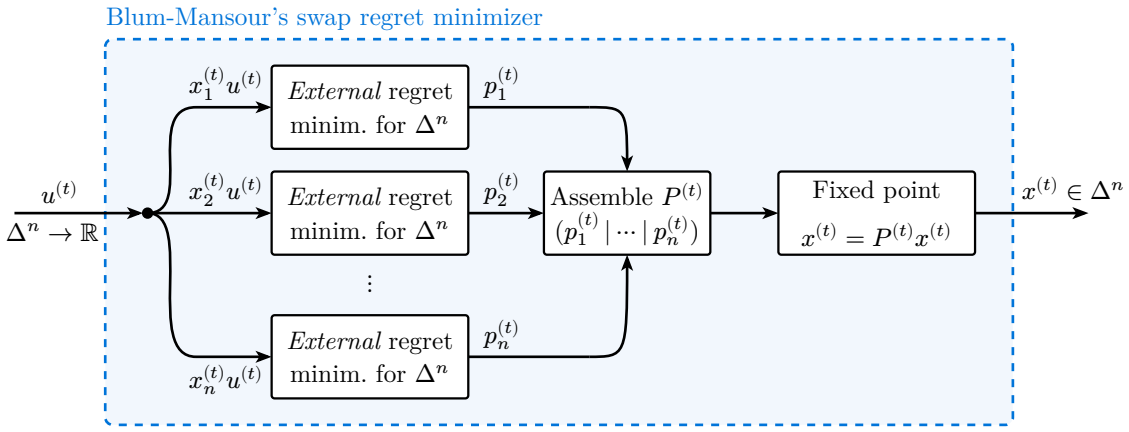
- To compute the `ObserveUtility`$\left(u^{(t)}\right)$ of the swap regret minimizer, we first call the `ObserveUtility` of each $\mathcal{R}_i$ with the linear utility function

$$v_i^{(t)} := x_i^{(t)} u^{(t)}$$

⋆These notes are class material that has not undergone formal peer review. The TAs and I are grateful for any reports of typos.

that is obtained by rescaling $u^{(t)}$ by the component $x_i^{(t)}$ of the last-output strategy $x^{(t)}$.

The process can be depicted pictorially as in Figure 1.



We now claim that the algorithm described above is a swap regret minimizer for Δ^n.

Theorem 1.1. Let $\mathrm{Reg}_i^{(T)}$ denote the regret incurred by each external regret minimizer $\mathcal{R}_i$ for Δ^n in the Blum-Mansour construction described above. Then, the swap regret cumulated by the algorithm satisfies:

$$\mathrm{SwapReg}^{(T)} \leq \sum_{i=1}^{n} \mathrm{Reg}_i^{(T)}.$$

In particular, the swap regret grows sublinearly in T whenever the extenral regret minimizers $\mathcal{R}_i$ guarantee sublinear regret.

Proof. By construction, the external regret incurred by each $\mathcal{R}_i$ is

$$\mathrm{Reg}_i^{(T)} = \max_{\tilde{p}_i \in \Delta^n} \sum_{t=1}^{T} \Big(v^{(t)}(\tilde{p}_i) - v^{(t)}(p_i^{(t)}) \Big). \tag{1}$$

Pick any $\hat{P} = (\hat{p}_1 \mid \ldots \mid \hat{p}_n) \in \Phi$. Then, the swap regret cumulated compared to always transforming strategies according to $\hat{P}$ is, by definition,

$$\begin{aligned}
\sum_{t=1}^{T} u^{(t)}\big(\hat{P}x^{(t)}\big) - u^{(t)}\big(x^{(t)}\big) &= \sum_{t=1}^{T} u^{(t)}\big(\hat{P}x^{(t)}\big) - u^{(t)}\big(P^{(t)}x^{(t)}\big) && \big(x^{(t)} = P^{(t)}x^{(t)}\big) \\
&= \sum_{t=1}^{T} \left[\left(\sum_{i=1}^{n} x_i^{(t)} u^{(t)}(\hat{p}_i) \right) - \left(\sum_{i=1}^{n} x_i^{(t)} u^{(t)}(p_i^{(t)}) \right) \right] && \big(\text{linearity of } u^{(t)}\big) \\
&= \sum_{t=1}^{T} \left(\sum_{i=1}^{n} v_i^{(t)}(\hat{p}_i) - v_i^{(t)}(p_i^{(t)}) \right) && \Big(\text{definition of } v_i^{(t)}\Big) \\
&= \sum_{i=1}^{n} \left(\sum_{t=1}^{T} v_i^{(t)}(\hat{p}_i) - v_i^{(t)}(p_i^{(t)}) \right) && (\text{switching summation order}) \\
&\leq \sum_{i=1}^{n} \mathrm{Reg}_i^{(T)}. && (\text{from } (1))
\end{aligned}$$

Taking a maximum over all $\hat{P} \in \Phi$ concludes the proof. □

2 A general approach: Gordon-Greenwald-Marks's reduction

Blum-Mansour's swap regret minimization algorithm is a special case of a much more general construction. Gordon, G. J., Greenwald, A., & Marks, C. [GGM08] show that Φ-regret minimization for a strategy set $\mathcal{X}$ can be constructed starting from the following two ingredients:

1. an *external* regret minimization for the set Φ; and
2. a *fixed point oracle* Φ, that is, an algorithm that given any $\phi \in \Phi$ outputs a fixed point $\phi(x) = x \in \mathcal{X}$.

Intuitively, the external regret minimizer for Φ has the role of tracking which transformation ϕ the decision maker should focus on at each time time. The linear utility function $U^{(t)} : \Phi \to \mathbb{R}$ observed by the external regret minimizer is constructed from the last-output strategy $x^{(t)}$ and the utility function $u^{(t)}$ observed at time t, according to the formula

$$U^{(t)}(\phi) = u^{(t)}\big(\phi\big(x^{(t)}\big)\big), \tag{2}$$

where $x^{(t)}$ is the last-output strategy. The final construction is as follows:

- Each call to `NextStrategy`() first calls $\mathcal{R}$.`NextStrategy`() to obtain the next transformation $\phi^{(t)}$. Then, a fixed point $x^{(t)} = \phi^{(t)}\big(x^{(t)}\big) \in \mathcal{X}$ is computed and output.
- Each call to `ObserveUtility`$\big(u^{(t)}\big)$ with linear utility function $u^{(t)}$ constructs the linear utility function $U^{(t)} : \phi \mapsto u^{(t)}\big(\phi\big(x^{(t)}\big)\big)$ given in (2), and passes it to $\mathcal{R}$ via $\mathcal{R}$.`ObserveUtility`$\big(U^{(t)}\big)$.

Graphically, we can summarize the process as in the following block diagram.

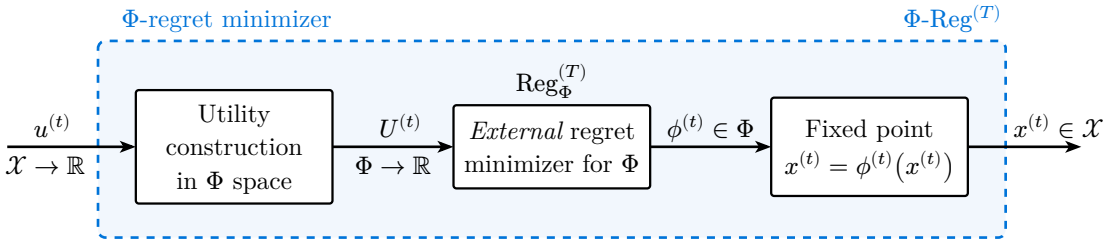


Figure 2: Gordon-Greenwald-Marks's construction of a Φ-regret minimizer.

Theorem 2.1 ([GGM08]). The Φ-regret Φ-Reg$^{(T)}$ cumulated up to time T by we have just defined is exactly equal to the (external) cumulative regret Reg$_\Phi^{(T)}$ cumulated by $\mathcal{R}$:

$$\Phi\text{-Reg}^{(T)} = \text{Reg}_\Phi^{(T)} \qquad \forall T = 1, 2, \dots.$$

Because the regret cumulated by $\mathcal{R}$ grows sublinearly by hypothesis of it being a regret minimizer, then so does the Φ-regret of the Φ-regret minimization algorithm defined above.

Proof. The proof of correctness of the above construction is deceptively simple. Since $\mathcal{R}$ outputs transformations $\phi^{(1)}, \phi^{(2)}, \dots \in \Phi$ and receives utilities $\phi \mapsto u^{(1)}\big(\phi\big(x^{(1)}\big)\big), \phi \mapsto u^{(2)}\big(\phi\big(x^{(2)}\big)\big), \dots$, its cumulative regret $R^{(T)}$ is by definition

$$\text{Reg}_\Phi^{(T)} = \max_{\hat{\phi} \in \Phi} \left\{ \sum_{t=1}^{T} \Big(u^{(t)}\Big(\hat{\phi}\big(x^{(t)}\big)\Big) - u^{(t)}\Big(\phi^{(t)}\big(x^{(t)}\big)\Big) \Big) \right\}.$$

Now, since by construction $x^{(t)}$ is a fixed point of $\phi^{(t)}$, $\phi^{(t)}\big(x^{(t)}\big) = x^{(t)}$, and therefore we can write

$$\text{Reg}_\Phi^{(T)} = \max_{\hat{\phi} \in \Phi} \left\{ \sum_{t=1}^{T} \Big(u^{(t)}\Big(\hat{\phi}\big(x^{(t)}\big)\Big) - u^{(t)}\big(x^{(t)}\big) \Big) \right\},$$

where the right-hand side is exactly the cumulative Φ-regret $\Phi\text{-Reg}^{(T)}$ incurred by $\mathcal{R}_\Phi$. □

Bibliography

[BM07] A. Blum and Y. Mansour, "From external to internal regret.," *Journal of Machine Learning Research*, vol. 8, no. 6, 2007.

[GGM08] G. J. Gordon, A. Greenwald, and C. Marks, "No-regret learning in convex games," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 360–367.

Changelog

- Oct 1: Fixed typo in the description of Gordon–Greenwald–Marks's reduction

6.S890 Topics in Multiagent Learning

Fall 2024