Chapter 5
TreeSwap: Efficient Swap Regret Minimization

Ioannis Anagnostides, Gabriele Farina, and Brian Hu Zhang*

So far, we have seen that no-regret learning, and efficient equilibrium computation, are essentially possible whenever the set of deviations Φ is "efficiently representable", in particular, whenever *external* regret minimization is possible over Φ. What about beyond? In particular, we have not yet addressed the *strongest* possible notion of Φ-regret, namely, the case where Φ contains *all* functions $\varphi : \mathcal{X} \to \mathcal{X}$—known as *swap regret*.

The fact that algorithms for Φ-regret minimization grow increasingly complex as Φ itself grows in complexity may lead one to believe that efficient swap regret minimization is hopeless. For example, the set of all functions $\varphi : \mathcal{X} \to \mathcal{X}$ has *infinite* dimension, so the techniques we have covered so far do not work. If $\mathcal{X}$ is a polytope, we could run the Blum-Mansour algorithm over its vertices—this would indeed succeed, but it has the drawback that the number of vertices of $\mathcal{X}$ is generally not polynomial in the dimension d of $\mathcal{X}$, and hence the algorithm is not efficient. Hence, for a long time, the possibility of swap regret minimization —and correspondingly, correlated equilibrium computation beyond normal-form games— remained a major open question.

The question has since essentially been fully resolved. In a simultaneous major breakthrough, Peng and Rubinstein [PR24] and Dagan, Daskalakis, Fishelson and Golowich [DDFS24], using essentially an identical algorithm now known as `TreeSwap`, showed that, if there is a regret minimizer on $\mathcal{X}$ that achieves *external* regret ϵ after M rounds, then there is a regret minimizer on $\mathcal{X}$ that achieves ϵT *swap* regret after $M^{1/\epsilon}$ rounds, and therefore efficient swap regret minimization *ks* in fact possible, at least when ϵ is a constant. They also showed nearly-matching lower bounds for normal-form games, which Daskalakis, Farina, Golowich, Sandholm and Zhang [DFG+24] later extended to extensive-form games as well, thereby precluding the possibility of $\text{poly}(d, 1/\epsilon)$-time algorithms for swap regret beyond normal-form games. In this chapter, we will go over the upper bound.

5.1 The `TreeSwap` Algorithm

We now describe the `TreeSwap` algorithm, following Peng and Rubinstein [PR24] and Dagan, Daskalakis, Fishelson and Golowich [DDFS24]. Let $\mathcal{R}$ be a no-regret algorithm on $\mathcal{X}$ whose

*✉ ianagnos@cs.cmu.edu, {gfarina,zhangbh}@mit.edu. These tutorial notes have not undergone formal peer review. We are grateful for any feedback or reports of typos.

regret is bounded by ϵT after M rounds. Of course, $\mathcal{R}$ will not necessarily have small swap regret. So, how can we hope to bound swap regret?

One thing to try might be the following. Instead of having $\mathcal{R}$ update its strategy on *every* iteration, it will be lazy, and only update its strategy every M iterations, using the average utility on those M iterations. That is, at time M, it receives utility $\frac{1}{M}\sum_{t=1}^{M} \boldsymbol{u}^{(t)}$, at time $2M$ it receives utility $\frac{1}{M}\sum_{t=M+1}^{2M} \boldsymbol{u}^{(t)}$, and so on. Then, between those M iterations, that is, while $\mathcal{R}$ is playing the same strategy, we initialize *another copy of the same regret minimizer*, which we will call $\mathcal{R}'$, which updates its strategy on *every* iteration.

The purpose of $\mathcal{R}'$ is to bound the value of the most profitable deviation from every single strategy that $\mathcal{R}$ plays. Therefore, $\mathcal{R}'$ can ensure that no strategy played by $\mathcal{R}$ incurs large swap regret. Of course, $\mathcal{R}'$ itself will have high swap regret… but this can be solved by introducing yet more layers!

The `TreeSwap` algorithm uses K layers of regret minimizers $\mathcal{R}_0, ..., \mathcal{R}_{K-1}$, each responsible for ensuring that there is no profitable swap deviation for the regret minimizers in the next layer. $\mathcal{R}_0$ updates on every iteration. Each regret minimizer $\mathcal{R}_k$ updates M times slower than the previous regret minimizer $\mathcal{R}_{k-1}$, and resets every M updates. Hence, this algorithm runs for M^K steps. The strategy output by `TreeSwap` is the uniform distribution (not to be confused with the average) over the strategies $(\boldsymbol{x}_0^{(t)}, ..., \boldsymbol{x}_{K-1}^{(t)})$ output by the K regret minimizers. For simplicity of notation, we will zero-index timesteps and use the notation $[n] = \{0, ..., n-1\}$.

Algorithm: `TreeSwap`

Input: Regret minimizers $\mathcal{R}_1, ..., \mathcal{R}_K$
for timesteps $t \in [T] \coloneqq [M^K]$ **do**
  for layers $k \in [K]$ **do**
    if $M^{k+1} \mid t$ **then** reset $\mathcal{R}_k$
    else if $M^k \mid t$ **then** pass to $\mathcal{R}_k$ the average of the last M^k utilities,

$$\frac{1}{M^k}\sum_{\tau=t-M^k}^{t-1} \boldsymbol{u}^{(\tau)}$$

    $\boldsymbol{x}_k^{(t)} \leftarrow$ current strategy of $\mathcal{R}_k$
  play mixed strategy $\mathrm{Unif}(\boldsymbol{x}_0^{(t)}, ..., \boldsymbol{x}_{K-1}^{(t)}) \in \Delta(\mathcal{X})$
  receive utility $\boldsymbol{u}^{(t)} \in \mathbb{R}^d$

Theorem 5.1. The regret of `TreeSwap` is bounded by $T \cdot (\epsilon + 1/K)$. In particular, taking $K = 1/\epsilon$ and $M = \mathrm{poly}(d, 1/\epsilon)$ (as is achieved by any reasonable external-regret-minimizing algorithm), `TreeSwap` achieves swap regret ϵT after $M^K = d^{\tilde{O}(1/\varepsilon)}$ iterations.

Remark 5.2. Theorem 5.1 only applies when T is a power of M. Handling the case where T is not a power of M requires a bit of care and results in a looser bound of $T \cdot (\epsilon + 3/K)$, but the main ideas are captured by the above result and its proof. [DDFS24]

Remark 5.3. We are mainly interested in the implications of Theorem 5.1 for its implications for swap regret in high-dimensional settings. However, it is worth noting that Theorem 5.1 achieves a new result even in the normal-form setting: its regret bound for normal form, when instantiated with multiplicative weights, is $\log(N)^{\tilde{O}(1/\epsilon)}$, which, for large N and ϵ, is better than the bound of Blum-Mansour.

We now sketch a proof of Theorem 5.1.

Proof Sketch. In this proof, timesteps are zero-indexed, and $[n] \coloneqq \{0, ..., n-1\}$. Let $\varphi : \mathcal{X} \to \mathcal{X}$ be any function. Then the (time-averaged) regret against φ is given by

$$\begin{aligned}
\frac{1}{T}\operatorname{Reg}(T, \varphi) &= \frac{1}{K}\sum_{k=0}^{K-1} \mathop{\mathbb{E}}_{t\in[T]} \left\langle \boldsymbol{u}^{(t)}, \varphi(\boldsymbol{x}_k^{(t)}) \right\rangle - \left\langle \boldsymbol{u}^{(t)}, \boldsymbol{x}_k^{(t)} \right\rangle \\
&= \frac{1}{K}\sum_{k=0}^{K-1} \mathop{\mathbb{E}}_{t\in[T]} \left\langle \boldsymbol{u}^{(t)}, \varphi(\boldsymbol{x}_k^{(t)}) \right\rangle - \frac{1}{K}\sum_{k=1}^{K} \mathop{\mathbb{E}}_{t\in[T]} \left\langle \boldsymbol{u}^{(t)}, \boldsymbol{x}_{k-1}^{(t)} \right\rangle \\
&= \frac{1}{K}\sum_{k=1}^{K-1} \underbrace{\mathop{\mathbb{E}}_{t\in[T]} \left\langle \boldsymbol{u}^{(t)}, \varphi(\boldsymbol{x}_k^{(t)}) - \boldsymbol{x}_{k-1}^{(t)} \right\rangle}_{\leq \epsilon} + \frac{1}{K} \underbrace{\mathop{\mathbb{E}}_{t\in[T]} \left\langle \boldsymbol{u}^{(t)}, \varphi(\boldsymbol{x}_0^{(t)}) - \boldsymbol{x}_{K-1}^{(t)} \right\rangle}_{\leq 1} \\
&\leq \epsilon + \frac{1}{K}
\end{aligned}$$

where, in the second-to-last line, the bound on the first term comes from the regret bound of each phase between resets (of length M updates), noting that $\varphi(\boldsymbol{x}_k^{(t)})$ is constant within any given phase and thus the regret against $\varphi(\boldsymbol{x}_k^{(t)})$ is bounded by the external regret during the phase. □

Bibliography for this chapter

[PR24] B. Peng and A. Rubinstein. (2024). Fast Swap Regret Minimization and Applications to Approximate Correlated Equilibria. *Symposium on Theory of Computing (STOC)*.

[DDFS24] Y. Dagan, C. Daskalakis, M. Fishelson and N. Golowich. (2024). From External to Swap Regret 2.0: An Efficient Reduction for Large Action Spaces. *Symposium on Theory of Computing (STOC)*.

[DFG+24] C. Daskalakis, G. Farina, N. Golowich, T. Sandholm and B. H. Zhang. (2024). A Lower Bound on Swap Regret in Extensive-Form Games. *arXiv preprint arXiv:2406.13116*.