Modern Algorithmic Game Theory

Martin Schmid
Department of Applied Mathematics
Charles University in Prague

December 17, 2025


Discounted CFR

Motivation: Regret Inertia

- Consider a simple game with three actions and payoffs: $0, 1,$ and $-1,000,000.$
- In iteration 1, CFR/CFR+ plays uniformly ($1/3$ each) and the expected utility is $\approx -333,333$, leading to positive regrets of $333,333$ for a_1 and $333,334$ for a_2
- The regret difference between the best action (a_2) and the mediocre action (a_1) is only 1.
- The agent will play a_1 and a_2 with nearly $50/50$ probability for a long time
- It takes **471,407 iterations** for CFR+ to converge to the optimal action a_2!
- The idea behind DCFR is to discount regrets from early iterations to reduce the weight of these initial massive regrets, allowing the true difference between a_1 and a_2 to emerge much faster

Linear CFR (LCFR)

- Linear CFR (LCFR) addresses inertia by weighing iteration t proportional to t
- This means iteration 100 has $100\times$ more influence than iteration 1
- The algorithm is identical to vanilla CFR, but at iteration t, it
  - weights the strategy by a weight of t (similarly to CFR+)
  - weights the immediate regret by a weight of t (or equivalently multiply the cumulative regrets by $\frac{t}{t+1}$)
- LCFR is particularly effective in games with **large mistakes** (where bad actions have massive negative penalties)
- It also retains the same asymptotic convergence rate as CFR
- In the motivating example, LCFR chooses the second action with 100% probability only after **970 iterations!**

Combining CFR+ and LCFR?

- Since CFR+ and LCFR solve two orthogonal problems and both algorithms are empirically faster than CFR, it is natural to try to combine the two into **Linear CFR+**
- This variant would use:
  1. Regret Matching+
  2. Alternating updates
  3. Linear weighting of both cumulative regrets and average strategies
- However, this combination performs **very poorly** in practice
- While theoretically sound, the aggressive clipping of CFR+ combined with the aggressive weighting of LCFR leads to instability

Discounted CFR (DCFR)

- DCFR generalizes the weighting idea using three parameters (α, β, γ) to control the forgetting rate
- At each iteration t, we multiply accumulated values by a discount factor:
  - positive regrets: $\frac{t^\alpha}{t^\alpha+1}$
  - negative regrets: $\frac{t^\beta}{t^\beta+1}$
  - average strategy: $\left(\frac{t}{t+1}\right)^\gamma$
- Additionally, DCFR uses RM and alternating? updates as they empirically lead to stronger performance
- This allows us to treat positive and negative regrets differently (e.g., dampening negative regrets less aggressively to retain information about bad actions)
- Both CFR+ and Linear CFR are special cases of Discounted CFR with parameters $(\infty, -\infty, 1)$ and $(1, 1, 1)$, respectively

Empirically Optimal Values

- The authors (Brown & Sandholm, 2019) performed extensive hyperparameter sweeps to find the best default values
- The optimal values vary depending on the game
- However, the following values led to consistently stronger performance than CFR+
  - $\alpha = 1.5$ (discount positive regrets moderately)
  - $\beta = 0$ (do not discount negative regrets)
  - $\gamma = 2$ (discount average strategy quadratically)
- **Discounted CFR is still considered one of the best tabular algorithms for solving imperfect information games!**

Monte Carlo CFR

Monte Carlo CFR

- All CFR variants we have seen so far traversed the whole game tree in every iteration
- This limits the applicability of these algorithms to larger games
- Monte Carlo CFR only considers a subset of terminal histories in each iteration
- This leads to faster per-iteration time at the cost of increased variance due to sampling

Monte Carlo CFR

- Technically, MCCFR is a family of algorithms defined by a partition $\mathcal{Q} = \{Q_1, \ldots, Q_r\}$ of terminal histories $\mathcal{Z}$
- The partition has to cover the whole set $\mathcal{Z}$, i.e. $\bigcup_i Q_i = \mathcal{Z}$
- At each iteration, the algorithm samples a block $Q_i \in \mathcal{Q}$ with probability q_i ($\sum_i q_i = 1$) and updates the cumulative regrets only for information sets leading to terminal histories $z \in Q_i$
- With proper importance sampling, we can guarantee that the cumulative regrets are in expectation the same as in vanilla CFR
- Notably, when we have a partition containing only a single block, $\mathcal{Q} = \{\mathcal{Z}\}$, we recover vanilla CFR

Variants of Terminal History Partitions

- There are three notable examples of partitions $\mathcal{Q}$
  - **Chance-sampled CFR**. In this variant, each block contains all histories that share the same sequence of chance moves, e.g. all histories in Kuhn Poker where Player 1 was dealt `Q` and Player 2 was dealt `J`
  - **External-sampling CFR**. In this variant, we sample all actions of the chance player and our opponent, and we traverse every single of our actions
  - **Outcome-sampling CFR**. We will explain this variant in more details in the remainder of the lecture

External-Sampling vs. Outcome-Sampling

- Let's draw this on the blackboard. Gemini and Nano Banana weren't cooperating yesterday

Outcome-Sampling CFR

- In Outcome-sampling CFR, each block Q_i contains a single terminal history z_i
- In each iteration, we sample a single terminal history z_i according to a **sampling policy** ξ and update only information states on the trajectory from the root to the sampled history z_i
- We require $\xi(s, a) > 0$ to ensure that each terminal history is sampled with non-zero probability to guarantee an unbiased regret estimates
- One possible choice of ξ is $\xi(s, a) = (1 - \epsilon)\pi(s, a) + \epsilon \frac{1}{|\mathcal{A}(s)|}$
- Instead of counterfactual value functions, the algorithm now uses **sampled counterfactual value functions** $\tilde{v}^{\pi}_{i,c}$ and $\tilde{q}^{\pi}_{i,c}$ which include an importance sampling term

Sampled Counterfactual Value Functions

- The **sampled counterfactual state** value $\tilde{v}^{\pi}_{i,c}(s|z)$ of an information state $s \in \mathcal{S}_i$ given a terminal history z is defined as

$$\tilde{v}^{\pi}_{i,c}(s|z) = \tilde{v}^{\pi}_{i,c}(h|z) = \begin{cases} \frac{P^{\pi}_{-i}(h) v^{\pi}_i(h)}{P^{\xi}(z)} & \text{for } h \in s \text{ and } h \sqsubseteq z \\ 0 & \text{otherwise} \end{cases}$$

- The **sampled counterfactual state-action** value $\tilde{q}^{\pi}_{i,c}(s,a)$ of an information state $s \in \mathcal{S}_i$ and action $a \in \mathcal{A}_i(s)$ given a terminal history z is defined as

$$\tilde{q}^{\pi}_{i,c}(s,a|z) = \tilde{q}^{\pi}_{i,c}(h,a|z) = \begin{cases} \frac{P^{\pi}_{-i}(h) q^{\pi}_i(h,a)}{P^{\xi}(z)} & \text{for } h \in s \text{ and } h \sqsubseteq z \\ 0 & \text{otherwise} \end{cases}$$

Outcome-Sampling CFR

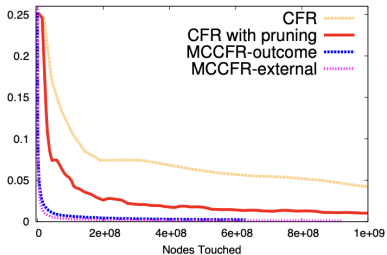
- Note that the sampled counterfactual value functions are simply counterfactual value functions computed from a single history h with the importance sampling term $P^{\xi}(z)$
- Such an estimate is provably unbiased $\mathbb{E}_{z\sim\xi}[\tilde{v}^{\pi}_{i,c}(s|z)] = v^{\pi}_{i,c}(s)$
- As a result, we can simply replace counterfactual values with their sampled variants and arrive at

$$\tilde{R}^{t}(s,a) = \tilde{q}^{\pi}_{i,c}(s,a) - \tilde{v}^{\pi}_{i,c}(s)$$

- The rest of the algorithm stays the same as in vanilla CFR

Empirical Performance of MCCFR

- The following figure shows the exploitability as a function of the number of visited nodes during the run of each algorithm in a generalized version of Kuhn Poker
- We can see that MCCFR variants produce less exploitable strategies while touching a lot fewer nodes than CFR
- Unfortunately, in practice MCCFR algorithms usually converge slower (in terms of wall-clock time) than their full-tree traversal counterparts

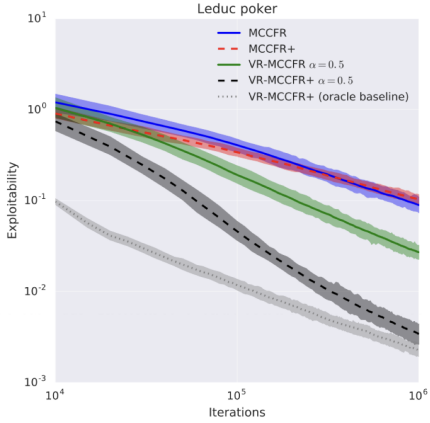


Combining MCCFR and CFR+

- We have already seen the difference in convergence speeds between CFR and CFR+, thus it is natural to ask if the same (or similar) difference is retained even after we introduce sampling
- Unfortunately, CFR+ loses its magic when combined with Monte Carlo sampling and MCCFR usually outperforms MCCFR+
- One possible way to explain this is that it is unlikely to get the convergence rate near $\frac{1}{T}$ in the sampling settings, as the Central Limit Theorem says that the average value converges at the rate of $\frac{\sigma}{\sqrt{T}}$
- In the next course, we will see methods (VR-MCCFR) that drastically reduce the sampling variance and make Monte Carlo sampling and CFR+ work

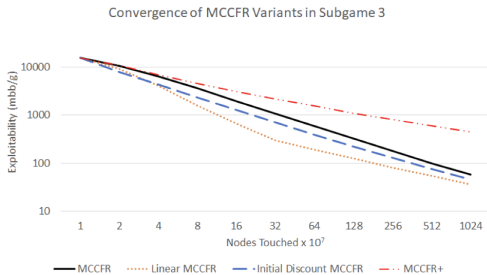
Comparison of MCCFR and MCCFR+

- The following figure compares the performance of MCCFR, MCCFR+ and their variance-reduced variants in Leduc Poker



Combining MCCFR and Discounted CFR

- Unlike CFR+, Linear CFR in combination with Monte Carlo sampling leads to a faster convergence than vanilla MCCFR in No-limit Hold'em Poker, as shown in the figure below
- Unfortunately, the authors didn't show the combination of Discounted CFR and Monte Carlo sampling, so it is (probably) unknown if other configurations scale comparably well



Week 10 Homework

You can find more detailed descriptions of homework tasks in the GitHub repository.

1. Discounted CFR
2. Monte Carlo CFR