Sampling-Based Counterfactual Regret Minimization

Scaling CFR to Large Games

Intelligent Agents: Computational Game Solving

October 23, 2025

Recall: Vanilla CFR

What we've seen:

- CFR iteratively updates regrets at every information set
- Each iteration: full tree traversal for both players
- Complexity per iteration: $\mathcal{O}(|Z|)$ where Z = terminal nodes
- Converges to Nash: exploitability $\mathcal{O}(1/\sqrt{T})$

The problem:

- Kuhn poker: $\sim$10 terminals $\rightarrow$ easy!
- Leduc poker: $\sim$10,000 terminals $\rightarrow$ manageable
- Texas Hold'em: $\sim 10^{18}$ terminals $\rightarrow$ **intractable**

Key insight: Do we really need to visit *every* node *every* iteration?

The Core Idea: Sampling

Vanilla CFR:

$$v_i^\sigma(I) = \sum_{a \in A(I)} \sigma(I, a) \cdot v_i^\sigma(I, a)$$

Computed by enumerating all actions, all opponent responses, all chance outcomes...

Sampling CFR:

$$\tilde{v}_i^\sigma(I) = \text{unbiased estimate of } v_i^\sigma(I)$$

Computed by sampling a subset of the game tree

Trade-off:

- Pro: Much cheaper per iteration (sample a few paths instead of all $|Z|$)
- Con: Higher variance $\rightarrow$ need more iterations
- Net result: Often much faster wall-clock time!

Three Flavors of Sampling

Method	What to sample?	What to enumerate?
Outcome Sampling	Everything: chance, both players	Nothing (single path to z)
External Sampling	Chance + opponent actions	Our own actions
Chance Sampling	Chance only	Both players' actions

Intuition:

- **Outcome:** cheapest per iteration, highest variance
- **External:** good balance (most popular in practice)
- **Chance:** lowest variance, but still cheaper than vanilla

Outcome Sampling: The Basic Idea

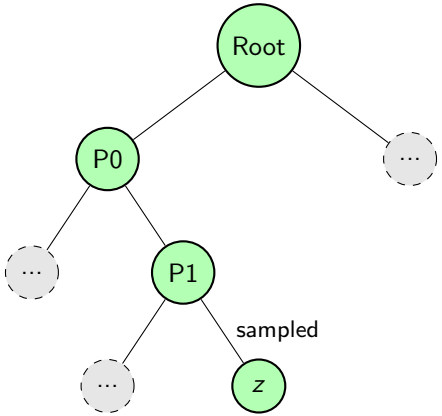
Algorithm (one iteration):

1. Sample a complete trajectory $h_1, a_1, h_2, a_2, \ldots, z$ according to σ^t
   - At chance nodes: sample according to chance distribution
   - At player nodes: sample action $a \sim \sigma^t(I, \cdot)$
2. Walk down that *single path*
3. At each infoset I visited by updating player:
   - Compute **sampled** counterfactual values for all actions
   - Update regrets using these sampled estimates

Key property: Sampled values are *unbiased* estimates

$$\mathbb{E}[\tilde{v}_i(I)] = v_i(I)$$

Outcome Sampling: Example Path



Vanilla CFR: visits all gray + green nodes (entire tree)
Outcome sampling: visits only green nodes (one path)
Cost: $\mathcal{O}(\text{depth})$ instead of $\mathcal{O}(|Z|)$

Outcome Sampling: The Math

At an infoset I on the sampled path, for action a:
Vanilla CFR computes:

$$v_i^\sigma(I,a) = \sum_{z:(I,a)\rightsquigarrow z} \pi_{-i}^\sigma(z \mid I,a) \cdot u_i(z)$$

Outcome sampling estimates:

$$\tilde{v}_i^\sigma(I,a) = \begin{cases} \dfrac{u_i(z)}{\pi^\sigma(z \mid I,a)} \cdot \pi_{-i}^\sigma(z \mid I,a) & \text{if } a \text{ on sampled path} \\ 0 & \text{otherwise} \end{cases}$$

Why this is unbiased:

$$\mathbb{E}[\tilde{v}_i(I,a)] = \sum_z \pi^\sigma(z \mid I,a) \cdot \frac{u_i(z) \cdot \pi_{-i}^\sigma(z \mid I,a)}{\pi^\sigma(z \mid I,a)} = \sum_z \pi_{-i}^\sigma(z \mid I,a) \cdot u_i(z) = v_i(I,a)$$

Outcome Sampling: Importance Weights

The key trick: Divide by sampling probability to correct bias

If we sample z with probability $q(z)$, the unbiased estimator is:

$$\tilde{v} = \frac{f(z)}{q(z)}$$

In CFR, we sample according to σ^t, so:

$$q(z \mid I, a) = \pi^{\sigma}(z \mid I, a)$$

And we want to estimate the counterfactual value:

$$v_i(I, a) = \sum_z \pi_{-i}(z \mid I, a) \cdot u_i(z)$$

So we use importance weight:

$$w(z) = \frac{\pi_{-i}(z \mid I, a)}{\pi^{\sigma}(z \mid I, a)} = \frac{1}{\pi_i(z \mid I, a)}$$

External Sampling: More Stability

Problem with outcome sampling:

- High variance when $\pi_i(z \mid I, a)$ is small
- Rare actions get very large weights $\rightarrow$ noisy updates

External sampling solution:

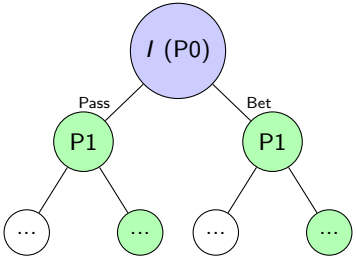
- **Sample:** opponent actions + chance
- **Enumerate:** *our own* actions at each infoset
- No importance weights needed for our actions!

Result:

- Lower variance than outcome sampling
- Higher cost per iteration (enumerate own actions at each I)
- Sweet spot for most applications

External Sampling: Visual Comparison

At P0's infoset I:



- Blue node: our infoset (enumerate both actions)
- Green nodes: opponent infosets (sample one action each)
- Gray nodes: subtrees we'll explore recursively

Cost: $\mathcal{O}(|A| \cdot \text{depth})$ per sampled trajectory

External Sampling: Pseudocode

function ExternalCFR(h, i, π_{-i})
  if h is terminal **then**
    return $u_i(h)$
  end if
  if chance node **then**
    Sample action $a \sim p_c(\cdot \mid h)$
    return ExternalCFR($h \cdot a$, i, π_{-i})
  end if
  $I \leftarrow$ infoset at h
  if player(h) $\neq i$ **then**
    Sample $a \sim \sigma(I, \cdot)$ ▷ Opponent: sample
    return ExternalCFR($h \cdot a$, i, $\pi_{-i} \cdot \sigma(I, a)$)
  else
    $v \leftarrow 0$
    for $a \in A(I)$ **do** ▷ Our actions: enumerate
      $v_a \leftarrow$ ExternalCFR($h \cdot a$, i, π_{-i})
      $v \leftarrow v + \sigma(I, a) \cdot v_a$
      regret[I][a] $\leftarrow$ regret[I][a] $+ \pi_{-i} \cdot (v_a - v)$

Chance Sampling: The Middle Ground

Idea: Only sample chance nodes, enumerate both players' actions

Use case: Games with:

- Many chance outcomes (e.g., card deals)
- Relatively few actions per player

Example: Poker

- Texas Hold'em: $\binom{52}{2} \times \binom{50}{5} = 1.3 \times 10^9$ board combinations
- But only 2–4 actions per decision (fold/call/raise)
- Chance sampling: pick one board, explore all action sequences

Variance:

- Lower than external sampling (no player action sampling)
- Higher cost per iteration than external sampling

Convergence: Theory vs. Practice

Theoretical rates:

Method	**Regret bound**	**Iterations for ϵ-Nash**
Vanilla CFR	$\mathcal{O}(1/\sqrt{T})$	$\mathcal{O}(1/\epsilon^2)$
Outcome Sampling	$\mathcal{O}(1/T^{1/4})$	$\mathcal{O}(1/\epsilon^4)$
External Sampling	$\mathcal{O}(1/T^{1/3})$	$\mathcal{O}(1/\epsilon^3)$
Chance Sampling	$\mathcal{O}(1/\sqrt{T})$	$\mathcal{O}(1/\epsilon^2)$

But wait! Cost per iteration varies dramatically:

$$\text{Total cost} = \underbrace{\text{iterations}}_{\text{higher for sampling}} \times \underbrace{\text{cost/iter}}_{\text{much lower for sampling}}$$

In practice, external sampling often wins by 10^3–$10^6\times$ in large games!

Practical Considerations

Variance reduction techniques:

1. **Baseline subtraction:**

$$\tilde{v}_i(I, a) - b(I) \quad \text{where } \mathbb{E}[b(I) \mid I] = 0$$

   Reduces variance without introducing bias

2. **Stratified sampling:**
   - Ensure rare but important branches get sampled
   - E.g., force sampling of fold vs. call separately

3. **Importance sampling corrections:**
   - Sample from different distribution q, reweight by π/q
   - Useful for exploration vs. exploitation

Case Study: Pluribus (Superhuman Poker AI)

Game: Six-player No-Limit Texas Hold'em
Challenges:

- $\sim 10^{161}$ game states (larger than atoms in universe!)
- Imperfect information + multiplayer
- Real-time play required

Pluribus approach (Brown & Sandholm, 2019):

- **Linear CFR** (improved regret updates)
- **External sampling** for blueprint strategy
- **Depth-limited search** during live play
- **Abstraction:**
  - Action abstraction (limited bet sizes)
  - Information abstraction (bucketing similar hands)

Result: First AI to beat human pros in multiplayer poker!

Implementation Tips

When implementing sampling CFR:

1. **Start simple:** Implement vanilla CFR first, verify correctness
2. **Test on small games:** Kuhn poker is great for debugging
   - Can compare sampled vs. vanilla convergence
   - Should reach same equilibrium (with more iterations)
3. **Track variance:** Log regret estimates over time
   - If too noisy, consider external sampling over outcome
   - Or add variance reduction
4. **Use CFR+:** Regret floor + linear weighting help sampling too!
5. **Parallelize:** Sample trajectories are independent
   - Easy to distribute across cores/machines

Comparison Table: Which to Use?

	Vanilla CFR	**External Sampling**	**Outcome Sampling**
Best for	Small games ($< 10^6$ nodes)	Medium-large games	Huge games with limited compute
Variance	None (deterministic)	Low	High
Iterations needed	Fewest	Moderate	Many
Cost per iter	Highest	Medium	Lowest
Implementation	Simplest	Moderate	Simple but tricky weights
Parallelizes?	No (shared tree)	Yes (independent samples)	Yes (independent samples)

Rule of thumb: External sampling is the default choice for most applications

Summary: Sampling CFR

Key ideas:

- Replace full tree traversal with sampling $\rightarrow$ much cheaper per iteration
- Use importance weighting to maintain unbiased estimates
- Trade-off: more iterations needed, but much lower cost per iteration

Three flavors:

- **Outcome:** sample everything (cheapest, highest variance)
- **External:** sample opponent + chance (best balance)
- **Chance:** sample only chance (lowest variance)

Impact:

- Enabled solving games with $10^{12}+$ nodes
- Foundation of superhuman poker AIs (Libratus, Pluribus)
- General technique: applicable beyond poker

Further Reading

Key papers:

- **Lanctot et al. (2009):** "Monte Carlo Sampling for Regret Minimization in Extensive Games" (NIPS)
  - Original sampling CFR paper
- **Brown & Sandholm (2019):** "Superhuman AI for Multiplayer Poker" (Science)
  - Pluribus: external sampling + depth-limited search
- **Schmid et al. (2019):** "Variance Reduction in Monte Carlo Counterfactual Regret Minimization" (AAAI)
  - Advanced variance reduction techniques

Implementations:

- OpenSpiel (DeepMind): `github.com/deepmind/open_spiel`
- PokerRL: `github.com/TinkeringCode/PokerRL`