CSCE 631 — Algorithmic Game Theory

Lecture 17: Poker Case Study — Libratus and Pluribus

Alan Kuhnle

Summer 2026

Texas A&M University

Today's Plan

History of Computer Poker

Libratus: Heads-Up No-Limit

Pluribus: Multiplayer No-Limit

Key Techniques and Insights

Lessons for AGT

Summary

History of Computer Poker

Why Poker?

- Poker has been the principal benchmark for imperfect-information game solving since the 1970s
- Unlike chess/Go: hidden information, stochastic outcomes, bluffing
- Poker requires *all* the techniques from this course:
  - Nash equilibrium computation
  - Extensive-form game representation
  - Counterfactual regret minimization
  - Game abstraction (Lectures 15–16)
- Solving poker = solving imperfect-information games at scale

Timeline of Major Milestones

Year	Milestone	Significance
2003	PsOpti4	First competitive limit hold'em bot
2007	Polaris	Beat human pros in limit hold'em (exhibition)
2008	CFR introduced	Zinkevich et al. — scalable equilibrium finding
2015	Cepheus	Essentially solved heads-up limit hold'em
2017	DeepStack	Beat pros in heads-up no-limit (online)
2017	Libratus	Beat top pros in heads-up no-limit (live)
2019	Pluribus	Beat pros in 6-player no-limit

Heads-Up Limit Hold'em: Solved

Bowling et al. (2015), *Science*

Cepheus **essentially solved** heads-up limit Texas hold'em (HULHE):

$$\text{exploitability} < 0.986 \text{ mbb/h}$$

This is less than one milli-big-blind per hand — indistinguishable from a Nash equilibrium over a human lifetime of play.

Key technical elements:

- CFR+ with regret-matching$^+$ (faster convergence than vanilla CFR)
- Compressed strategy storage: 262 TB reduced to 11 TB
- Computed over 68 CPU-years across 200 nodes
- 3.19×10^{14} information sets in the full game

The No-Limit Challenge

Why is no-limit so much harder than limit?

	Limit hold'em	**No-limit hold'em**
Bet sizes	Fixed (2 options)	Any amount (continuous)
Game states	$\sim 3 \times 10^{14}$	$\sim 10^{161}$
Information sets	$\sim 3 \times 10^{14}$	$\sim 10^{161}$
Strategic depth	Moderate	Deep (stack dynamics)

- The continuous bet space means action abstraction is *essential*
- Stack-to-pot ratio creates complex multi-street dynamics
- Cannot solve the full game — must use abstraction + real-time solving

Libratus: Heads-Up No-Limit

Libratus: Overview

Brown & Sandholm (2018), *Science*

Libratus defeated four top human professionals in heads-up no-limit Texas hold'em over 120,000 hands, winning by a total of \$1.77 million in chips (with statistical significance $p < 0.0002$).

Three-module architecture:

1. **Pre-computed blueprint** strategy (offline)
2. **Nested safe subgame solving** (real-time)
3. **Self-improvement** module (overnight)

Each module addresses a different limitation of abstraction-based play.

Module 1: Blueprint Strategy

- Computed via **MCCFR** (Monte Carlo CFR) on an abstracted game
- Information abstraction: $\sim$200 card buckets per round
- Action abstraction: $\sim$14 bet sizes
- Total abstract game: $\sim 10^{12}$ information sets (10^{149} times smaller than the full game)
- Computation: $\sim$15 million core-hours over months

Role of the Blueprint

The blueprint is **not** what Libratus plays. It serves as:

1. The default strategy at the start of each hand
2. The initialization (warm start) for subgame solving
3. The leaf-value estimator for depth-limited solves

Module 2: Nested Safe Subgame Solving

The core innovation enabling superhuman play:

1. At **each decision point** during actual play, Libratus solves the remaining subgame in real time
2. The subgame is solved at a *finer* abstraction (or no abstraction) than the blueprint
3. Safety constraint: the opponent's counterfactual value at the subgame root must not increase

$$v_{-i}^{\text{subgame}}(I_{-i}) \leq v_{-i}^{\text{blueprint}}(I_{-i}) \quad \forall\ I_{-i} \text{ at the boundary}$$

4. Nested: each subsequent subgame solve builds on the previous one

Hardware: the Pittsburgh Supercomputing Center's Bridges cluster, using up to 46 seconds of computation per decision.

Safe Subgame Solving: Key Guarantee

Theorem (Brown & Sandholm, 2017)

If the blueprint has exploitability ϵ and we apply safe subgame solving with safety margins $\delta_I \geq 0$ at each boundary information set I, then the resulting strategy has exploitability at most $\epsilon - \sum_I w_I \delta_I$, where $w_I > 0$ are reach-probability weights.

- Safe subgame solving can only *decrease* exploitability
- More computation at each decision $\Rightarrow$ larger safety margins $\Rightarrow$ lower exploitability
- In the limit of infinite computation, nested safe subgame solving converges to a Nash equilibrium of the full game

- After each day of play (15,000 hands), Libratus identified situations where the humans found unexploited weaknesses
- Overnight: re-solved the blueprint in regions where the humans gained value
- Added finer action abstractions for bet sizes the humans used frequently

Effect

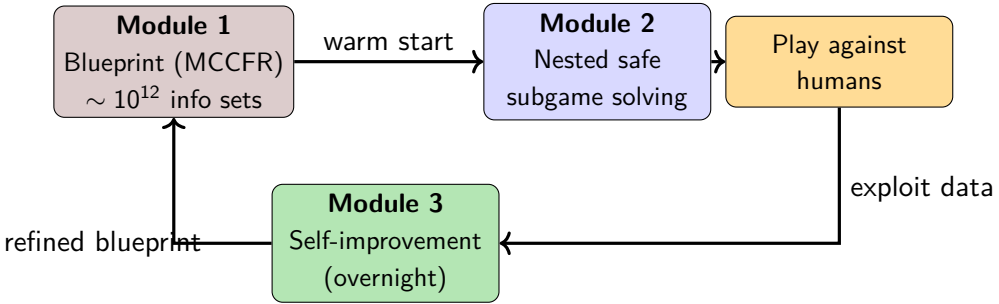
The humans initially found exploitable patterns (e.g., unusual bet sizes not in the abstraction). By the end of the 20-day match, these exploits were closed. The professionals reported that Libratus "kept getting better."

Libratus: Results

Metric	Libratus	Humans (best)
Total chip winnings	$+1{,}766{,}250$	$-1{,}766{,}250$
Win rate (mbb/h)	$+147$	-147
Hands played	$120{,}000$	
p-value	< 0.0002	

- All four professionals lost overall
- The margin (147 mbb/h) is large: professional-level edges are typically $\sim$ 5–50 mbb/h
- Quote from Daniel McAulay: "Libratus is the best NLHE player in the world right now"

Libratus: Architectural Diagram



Pluribus: Multiplayer No-Limit

The Multiplayer Challenge

Heads-up (2-player) zero-sum games:

- Nash equilibrium is the gold standard (minimax theorem)
- CFR provably converges to Nash equilibrium
- Safe subgame solving provides formal guarantees

Multiplayer ($n > 2$):

- No longer zero-sum (even in poker)
- Nash equilibrium may not be unique or meaningful
- CFR has **no convergence guarantee** to Nash equilibrium
- Safe subgame solving guarantees break down
- New solution concept needed

Pluribus: Overview

Brown & Sandholm (2019), *Science*

Pluribus achieved superhuman performance in 6-player no-limit Texas hold'em, defeating elite human professionals in two formats:

- 5 humans + 1 bot ($p < 0.002$)
- 1 human + 5 bots ($p < 0.008$)

Key innovations:

1. Blueprint via MCCFR with **linear averaging**
2. **Depth-limited** search (not full subgame solving)
3. Search against a **small set of opponent policies**, not a single equilibrium
4. Remarkably efficient: ran on a single server with 64 cores

Pluribus: Blueprint Computation

- Computed via **MCCFR** with **linear CFR** averaging:

$$\bar{\sigma}^T(I, a) = \frac{\sum_{t=1}^{T} t \cdot \sigma^t(I, a) \cdot \pi_i^{\sigma^t}(I)}{\sum_{t=1}^{T} t \cdot \pi_i^{\sigma^t}(I)}$$

- Later iterations are weighted more heavily (linear in t)
- This dramatically improves convergence speed compared to uniform averaging

Computational Efficiency

- 8 days on a 64-core server with 512 GB RAM
- Cost: $\sim$\$150 of cloud compute
- Compare Libratus: millions of core-hours on a supercomputer
- The blueprint is much smaller (abstracted more aggressively) because real-time search compensates

Pluribus: Depth-Limited Search

- Full subgame solving is infeasible with 6 players (subgame trees are too large)
- Instead: solve to a **limited depth**, using blueprint values at leaves

Depth-Limited Solving in Pluribus

1. At each decision, build a subgame to depth d (typically one or two betting rounds ahead)
2. At depth-d leaves, estimate payoffs using the blueprint strategy
3. Solve this truncated subgame using a modified CFR variant
4. Play the resulting strategy at the current decision

The blueprint acts as a "value function" for positions beyond the search horizon — analogous to the evaluation function in chess engines.

Pluribus: Search Against Multiple Policies

Key insight: in multiplayer, there is no single "opponent."

- Pluribus searches against a **small set of different opponent policies**, not a single equilibrium:
  1. The blueprint strategy (default)
  2. The blueprint biased toward folding
  3. The blueprint biased toward calling
  4. The blueprint biased toward raising
- The search finds a strategy that performs well against *all* of these variants
- This is more robust than optimizing against a single opponent model

Why This Works

The opponent policies capture a range of plausible human strategies. By performing well against all of them, Pluribus is robust to uncertainty about opponent behavior.

Pluribus: Results

Metric	Result
Format 1 (5 humans + 1 bot)	+48 mbb/h ($p < 0.002$)
Format 2 (1 human + 5 bots)	Average human lost significantly
Hands played (Format 1)	10,000
Number of pros	15 (including 2 WSOP bracelets)
Computation per hand	$\sim$20 seconds on 2 CPUs

- First AI to achieve superhuman performance in a major multiplayer benchmark game
- The win rate of 48 mbb/h is significant against world-class players
- The computational efficiency is remarkable: a single commodity server

Libratus vs. Pluribus: Comparison

	Libratus	**Pluribus**
Players	2	6
Game	HU NLHE	6-max NLHE
Year	2017	2019
Blueprint algo	MCCFR	MCCFR + linear avg
Real-time search	Nested safe subgame	Depth-limited
Safety guarantee	Yes	No (not applicable)
Hardware	Supercomputer	64-core server
Blueprint cost	Millions core-hours	8 days, $150
Search per decision	46 sec	20 sec

Key Techniques and Insights

CFR+ and Linear CFR: Why They Matter

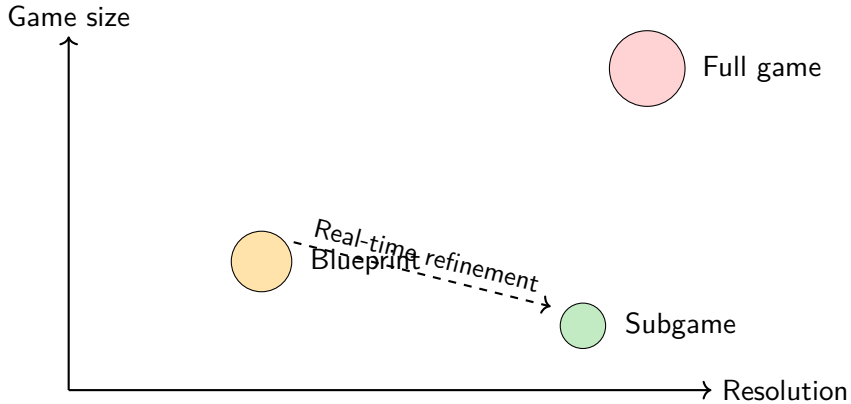
CFR+ (Tammelin, 2014):

- Replace $R_i^T(I,a) = \max(0, R_i^{T-1}(I,a) + r_i^T(I,a))$
- Cumulative regrets are floored at zero (no negative regret)
- Converges an order of magnitude faster than vanilla CFR
- Used in Cepheus (limit hold'em)

Linear CFR (Brown & Sandholm, 2019):

- Weight iteration t by t in the average strategy
- Recent iterations count more (appropriate when early iterations are far from equilibrium)
- Also discount positive regrets: multiply by $\frac{t}{t+1}$ each iteration
- Used in Pluribus

The Role of Abstraction in Modern Solvers



- The blueprint compresses the full game to a manageable size
- Real-time solving recovers resolution where it matters

What Made Libratus/Pluribus Succeed?

1. **Separation of offline and online computation**: months of offline blueprint + seconds of online refinement
2. **Safe subgame solving** (Libratus): formal guarantee that refinement cannot hurt
3. **Linear CFR** (Pluribus): $10\times$ faster convergence enables practical multiplayer blueprints
4. **Depth-limited search** (Pluribus): makes multiplayer real-time solving feasible
5. **Robustness over optimality**: search against multiple opponent models rather than a single equilibrium
6. **Self-improvement** (Libratus): adapt to opponent exploits

Lessons for AGT

Beyond Poker: General Principles

The Libratus/Pluribus methodology is not poker-specific:

1. **Abstraction + refinement**: solve a compressed version, then refine in real time. Applicable to any large imperfect-information game.
2. **Blueprint as value function**: the abstracted solution provides heuristic values for deeper search. Same principle as evaluation functions in game tree search.
3. **No special domain knowledge**: Libratus/Pluribus use no poker-specific heuristics — only the game rules.
4. **Safe refinement**: theoretical guarantees on online improvement are possible and practical.

Applications Beyond Poker

Domain	Connection
Cybersecurity	Attacker/defender with hidden capabilities
Negotiation	Multi-round, imperfect information
Military strategy	Sequential, hidden information, stochastic
Auctions	Strategic bidding with private valuations
Financial trading	Imperfect info, sequential decisions
Autonomous driving	Multi-agent, imperfect info, real-time

- The abstraction–refinement paradigm transfers directly
- The main requirement: a well-defined game with computable utilities
- Open challenge: games with continuous state/action spaces (beyond discrete abstraction)

Current Limitations and Open Problems

1. **3+ player games**: no convergence guarantee for CFR; Pluribus's practical success lacks theoretical backing
2. **Opponent modeling**: Libratus/Pluribus play near-equilibrium, but against weak opponents, exploitation is better
3. **Continuous action spaces**: current abstractions discretize; function approximation (next lecture) is an alternative
4. **Long-horizon games**: depth-limited solving degrades when the horizon is very long (imprecise leaf values)
5. **Partial observability beyond cards**: games where the hidden information is richer than a fixed-size signal

Next lecture: Deep CFR — replacing tabular CFR with neural networks to handle games too large for even aggressive abstraction.

Summary

Summary

1. **HULHE solved**: Cepheus (2015), essentially zero exploitability via CFR+ at scale
2. **Libratus** (2017): three-module architecture (blueprint + nested safe subgame solving + self-improvement) beat top pros in heads-up no-limit
3. **Pluribus** (2019): depth-limited search with linear CFR and multi-policy opponent modeling achieved superhuman 6-player no-limit on a single server
4. **Key principle**: coarse offline blueprint + fine-grained online refinement
5. **Safe subgame solving**: formal guarantee that refinement cannot increase exploitability
6. **General methodology**: the abstraction–refinement paradigm extends beyond poker to any large imperfect-information game