Continual Re-Solving for HUNL Poker: Theoretical Backbone and Implementation Manual

Abstract

This document consolidates the theoretical backbone of a practical, continual re-solving system for heads-up no-limit Texas Hold'em (HUNL) under a sparse action menu and depth-limited search. The goals are twofold: (i) to serve as an instruction manual for implementing the method end-to-end, and (ii) to present the elementary proofs used to verify core invariants in my implementation. Building upon the foundational work of DeepStack, I provide formal statements and proofs for the specific adaptations and invariants required in my system. All foundational results regarding continual re-solving are cited to DeepStack.

1 Problem Setting and High-Level Design

Game model. HUNL is a two-player zero-sum extensive-form game with imperfect information, chance nodes (dealing), and alternating moves. Play proceeds across streets (preflop, flop, turn, river), with public-card revelation between betting rounds.

Continual re-solving. At each decision, I re-solve the current public subtree to obtain a local strategy, act from that strategy, and carry forward only (a) the acting player's range and (b) the opponent's counterfactual value (CFV) vector, updated after own/chance/opponent transitions. Depth is limited for tractability; beyond the limit, leaf utilities are provided by learned CFV functions. (Foundational soundness and the re-solving protocol are due to DeepStack.)

Sparse actions and depth limits. I employ a first layer of {Fold, Call/Check, Pot, All-in} with lean replies to balance strength and tractability; depth limits at the end of street on preflop and flop invoke CFV networks; the turn proceeds to terminal with exact endgame. These choices match the implementation and invariant suite I have developed.

2 Notation and Objects

- K buckets (clusters) per street; $r_1, r_2 \in \Delta^K$ denote bucketed ranges.
- $P > 0$ current pot; Pnorm$P/(S_1^{(0)} + S_2^{(0)}) \in (0, 1]$.
- $\varphi_{\text{board}} \in \{0,1\}^{52}$ one-hot encoding of public cards.
- CFV networks (flop/turn) output per-bucket pot-fraction CFVs for both players.

3 CFV Networks and Outer Zero-Sum Layer

3.1 Input/Output contract (instruction)

For each depth-limit query (end of preflop or flop), I build

$$x \;=\; \big[\,\text{Pnorm}\;\big|\;\varphi_{\text{board}}\;\big|\;r_1\;\big|\;r_2\,\big] \in \mathbb{R}^{1+52+2K},$$

and obtain unadjusted predictions $(v_1, v_2) \in \mathbb{R}^K \times \mathbb{R}^K$ in fractions of pot. I multiply by P only if chip units are needed downstream.

3.2 Per-sample outer zero-sum adjustment (statement and proof)

Proposition 1 (Outer zero-sum identity)**.** *Let* $s_1 = \langle r_1, v_1\rangle$*,* $s_2 = \langle r_2, v_2\rangle$*, and* $\delta\frac{1}{2}(s_1 + s_2)$*. Define*

$$f_1 v_1 - \delta\,\mathbf{1}, \qquad f_2 v_2 - \delta\,\mathbf{1}.$$

Then $\langle r_1, f_1\rangle + \langle r_2, f_2\rangle = 0$ *(up to numerical roundoff).*

Proof.

$$\langle r_1, f_1\rangle + \langle r_2, f_2\rangle = \langle r_1, v_1\rangle - \delta\langle r_1, \mathbf{1}\rangle + \langle r_2, v_2\rangle - \delta\langle r_2, \mathbf{1}\rangle = (s_1 + s_2) - \delta(1 + 1).$$

Since $\delta = \frac{1}{2}(s_1 + s_2)$, the expression reduces to $(s_1 + s_2) - (s_1 + s_2) = 0$. □

Implementation note. I apply the adjustment samplewise after the linear heads and before loss computation; I report all validation metrics in pot units.

4 Follow/Terminate Gadget and Monotone Carry-Forward

Let S be a boundary state (end-of-street) with opponent range $r_{\text{opp}} \in \Delta^K$ and constraint vector $c^{(t)} \in \mathbb{R}^K$ (opponent CFVs) carried from the previous boundary.

4.1 Gadget (instruction)

For each bucket i, the opponent chooses *Terminate* (receive $c_i^{(t)}$) or *Follow* (enter the re-solved subgame and receive $\tilde{v}_i^{(t)}$). I define

$$v_{\text{opp},i} \max\{c_i^{(t)},\ \tilde{v}_i^{(t)}\}, \qquad v_{\text{hero},i} - v_{\text{opp},i}.$$

This preserves zero-sum per bucket and range-weights to zero overall (consistency with the outer layer).

4.2 Monotone carry-forward (statement and proof)

Proposition 2 (Protection & monotonicity)**.** *With the update* $c^{(t+1)} \max\{c^{(t)}, \tilde{v}^{(t)}\}$ *(componentwise), it holds that*

(i) $v_{opp,i} \geq c_i^{(t)}$ *for all* i *(*Terminate is never worse than Follow*).*

(ii) $c_i^{(t+1)} \geq c_i^{(t)}$ *for all* i *(*non-decreasing lower bound across boundaries*).*

Proof. (i) Since $v_{\text{opp},i} = \max\{c_i^{(t)}, \tilde{v}_i^{(t)}\}$, $v_{\text{opp},i} \geq c_i^{(t)}$ by definition. (ii) By the update, $c_i^{(t+1)} = \max\{c_i^{(t)}, \tilde{v}_i^{(t)}\} \geq c_i^{(t)}$. □

Implementation note. I use self-play CFVs for $c^{(t)}$ in practice for stability; the termination option guarantees safety while damping oscillations between boundaries.

5 Bayes-Consistent Range Updates

Let $i \in \{1, \ldots, K\}$ index opponent buckets. After observing an opponent action a, I update

$$p'(i) \;=\; \frac{p(i)\,\pi(a \,|\, i)}{\sum_{j=1}^{K} p(j)\,\pi(a \,|\, j)}.$$

Proposition 3 (Normalization and invariances)**.** *For any prior $p \in \Delta^K$ and likelihoods $\pi(a \,|\, i) \geq 0$ with $\sum_i p(i)\,\pi(a \,|\, i) > 0$,*

(i) $\sum_i p'(i) = 1$ (posterior is a probability vector);

(ii) If $\pi(a \,|\, i)$ is constant in i, then $p' = p$ (uninformative action leaves the prior unchanged);

(iii) For a sequence of independent observations $a_1, \ldots, a_T$, the posterior equals the one-shot update with product likelihoods.

Proof. (i) Summing the numerator over i yields the denominator; division gives 1. (ii) If $\pi(a \,|\, i) = \kappa$ for all i, then $p'(i) = p(i)\kappa/(\kappa \sum_j p(j)) = p(i)$. (iii) Follows by multiplication of likelihoods and a single normalization at the end. □

Instruction. I maintain strictly positive mass by construction; a small uniform mixture is acceptable as a safety fallback when external priors are degenerate.

6 Clustering and Bucket Interface

Feature sketch. My clustering framework operates in a low-dimensional feature space capturing (i) showdown equity, (ii) equity potential, and (iii) a payoff proxy, normalized before Euclidean k-means. Bucket count is $K = \min\{K_{\text{target}}, |H|\}$ for candidate hand set H. Drift-triggered re-clustering avoids unnecessary recomputation.

7 Depth-Limit Policy and Endgame

Depth-limit placement. I place the depth limit at end of street on preflop/flop (invoking CFV networks) and solve-to-terminal from turn through river (exact endgame). This matches the documented interface between search and value functions and preserves the pot-fraction semantics.

8 Engine Invariants (Sanity Layer)

Mass and pot monotonicity. Range mass is conserved to numerical tolerance, and public pot changes are non-negative modulo explicit refunds, with street transitions restricted to "stay or advance by one". I enforce these as testable invariants.

9 Implementation Checklist (Instruction Manual)

1. **State & legality.** I implement a single public-state engine: actor order per street; sparse legal menu; street advance on equalization; all-in lock fast-forward.

2. **CFV I/O.** Input $[\mathrm{Pnorm}\,|\,\varphi_{\mathrm{board}}\,|\,r_1\,|\,r_2]$; output per-bucket pot-fraction CFVs; apply outer zero-sum adjustment before loss and logging.

3. **Re-solver.** CFR-style iterations with *Follow/Terminate* gadget at boundary; self-play constraints; carry-forward via componentwise max; statistical action pruning with reactivation (variance-aware).

4. **Preflop cache.** Content-addressed signature; require bit-identical reuse on hits.

5. **Sanity harness.** Enforce zero-sum residual $\leq 10^{-6}$ (post-adjust), range-mass tolerance, and non-negative pot deltas on synthetic transitions.

References

[1] M. Moravčík, M. Schmid, N. Burch, V. Lisý, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, and M. Bowling. *DeepStack: Expert-Level Artificial Intelligence in Heads-Up No-Limit Poker*. *Science*, 356(6337):508–513, 2017. See also arXiv:1701.01724 (technical appendix).