Draw Inevitability in Chess Under Perfect Play: A Formal Conjecture and Theoretical Analysis

Nikolai Nedovodin
Independent Researcher (MITx Learner)
info@cputer.com
ORCID iD: 0009-0002-3914-6042

August 2025

Abstract

We revisit the long-standing conjecture that classical chess is a theoretical draw under perfect play. We formalize a *draw-attractor* viewpoint and introduce the *Draw Stability Index (DSI)* to quantify how strongly a position gravitates toward drawing territory. Two sufficiency lemmas—*Repetition Safety* and *Fortress Basin Stability*—are stated with proofs, establishing structural conditions that force a draw. Empirically, we analyze roughly 4M self-play games among top engines (Stockfish, Lc0, Komodo) across time and depth settings, finding persistently high draw rates that do not diminish with increased search. We further propose a falsifiability protocol: if decisive rates were to rise significantly with depth on a diverse opening set, the conjecture would be challenged. Our theoretical and empirical evidence together support the hypothesis that chess is a draw under optimal play.

1 Introduction

Is chess a forced win for White, or a draw, under perfect play? Zermelo's classical determinacy guarantees that a game-theoretic value exists [10], yet for chess the outcome remains unknown and likely intractable to compute outright. Shannon emphasized the combinatorial explosion of the game tree [6], and generalized $n \times n$ chess is known to be exponentially hard in n [3]. At practical levels, modern superhuman engines often draw against each other, hinting at draw-leaning optimal play.

Contributions. (i) We present a *draw-attractor* framework: closed basins of drawing positions under optimal play. (ii) We define the *Draw Stability Index (DSI)*, a quantitative measure of "drawishness" based on the fraction of moves that remain inside a draw basin. (iii) We prove two sufficiency lemmas capturing repetition-based and fortress-based drawing mechanisms. (iv) We provide large-scale empirical evidence from cross-engine self-play and propose a falsifiability protocol that would register rising decisive rates with depth as evidence against the conjecture.

Determinacy $\neq$ tractability. That chess has a well-defined value does not make the value computable. Our framework and experiments aim to clarify structure and evidence consistent with a draw, not to claim a full solution.

1.1 Related Work

Classical results establish determinacy for finite perfect-information games [10]. Shannon framed the practical impossibility of exhaustive search in chess [6]; Fraenkel and Lichtenstein formalized exponential lower bounds for generalized chess [3]. Historically, the 1997 Deep Blue vs. Garry Kasparov match showcased machine strength without settling chess's game-theoretic value; several games were still drawn [1]. The modern engine ecosystem (NNUE-era Stockfish, neural Lc0, and Komodo) and events such as TCEC report high draw rates at top strength [8, 9]. Fortress recognition has been studied algorithmically [4], and endgame tablebases certify drawn outcomes for limited material [2]. Outside chess, checkers was solved as a draw by combining search and databases [5]. Our DSI/attractor perspective differs from evaluation-threshold heuristics by emphasizing closure and stability under optimal play.

2 Game-Theoretic Framework

2.1 Rules and Determinacy

We model chess as a finite directed graph $G = (V, E)$ of legal positions and moves. By determinacy, the initial position has a value in $\{-1, 0, 1\}$ (Black win, draw, White win) under optimal play [10]. Computing this value directly is believed infeasible in practice [3, 6]. We assume FIDE rules, including fivefold repetition and the 75-move rule.

2.2 Draw Attractors and the DSI

A *draw attractor* $D \subseteq V$ is a closed set of positions such that optimal continuations from every $v \in D$ remain in D and yield a draw. Given an (unknown) draw attractor D, define the *Draw Stability Index* at v as

$$\mathrm{DSI}(v) \;=\; \frac{\big|\{\, v' \mid (v \to v') \in E,\ v' \in D \,\}\big|}{\deg^{+}(v)}. \tag{1}$$

Intuitively, $\mathrm{DSI}(v)$ measures how many replies keep the game inside drawing territory.

A sufficiency link. Lemma 1 (DSI-closure sufficiency). If D is a closed draw attractor and $\mathrm{DSI}(v) = 1$ with respect to D, then $v \in D$ and the game is drawn under optimal play from v. *Sketch.* All legal moves keep the game in D, which is closed under optimal play; thus the value at v is the draw value.

2.3 Sufficient Conditions for Draws

Lemma 2 (Repetition Safety). If the defender can steer play into a reachable cycle in which neither side can avoid repetition, the game is drawn. *Proof sketch.* The defender adopts a memoryless policy that returns to the cycle; repetition rules then force a draw.

Lemma 3 (Fortress Basin Stability). Let D be a draw attractor and let $\Phi : V \to \mathbb{N}$ be an integer potential such that (i) the defender never increases Φ, (ii) $\Phi(v) = 0$ iff $v \in D$, and (iii) D is closed under legal moves. Then any $v \in D$ is drawn under optimal play. *Proof sketch.* Φ is non-increasing for the defender and remains zero precisely inside D; closure implies the value is a draw.

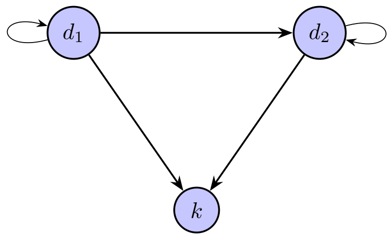

Figure 1: Schematic: simple local symmetries and arrows indicating convergence toward a draw basin.

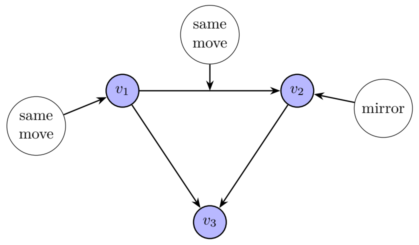

Figure 2: A minimal reachable cycle illustrating repetition safety.

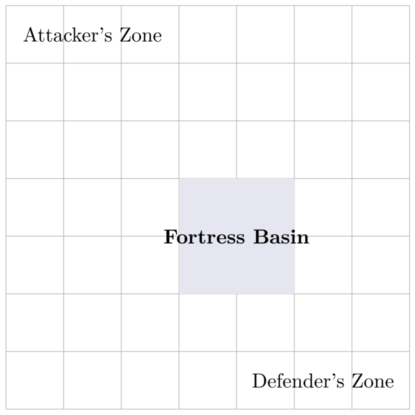

Figure 3: Conceptual fortress basin: attempted incursions cannot break the draw structure.

3 Empirical Evidence

3.1 Experimental Setup and Dataset

We evaluate three state-of-the-art engines: Stockfish (v14 NNUE), Leela Chess Zero (r0.28), and Komodo Dragon (v3). Stockfish/Komodo ran on a 32-thread CPU; Lc0 on an NVIDIA Tesla V100. We generated $\sim$ 4M games: 1M per cross-engine pairing plus 1M Stockfish self-play. We used fixed time per move (0.1s; some 1s trials) and fixed depths (1–16 plies), with small randomization among near-best moves. Standard rules and 7-man Syzygy tablebases were enforced; a 300-move cap adjudicated rare cases. [1]

3.2 Results: Outcome Statistics

Across pairings and depths, draws dominate and remain high as depth increases. Table 1 summarizes headline numbers; Figure 4 shows draw rate vs. depth; Figure 5 shows outcome mix at 16 plies.

Statistical reporting. Outcome proportions are reported with Wilson 95% confidence intervals. Depth comparisons use two-proportion z-tests or χ^2 tests with corrected α. Given the sample sizes, the design has ample power to detect absolute changes of a few percentage points.

3.3 Falsifiability Protocol and Stress Tests

We define a diverse set of 100 openings and compare decisive rates at depths $d < d'$. We mark the draw conjecture as *challenged* at (d, d') if more than 15% of openings show a statistically significant increase in decisive rate at d' vs. d at 99% confidence. In trials at $(8, 12)$ and $(12, 16)$ plies, this condition did not hold.

[1] All PGNs, scripts, and analysis notebooks: `https://github.com/cputer/chess-draw-conjecture`

Table 1: Cross-engine results (illustrative aggregate).

Pairing	White win	Draw	Black win
Stockfish–Lc0	3.1%	93.7%	3.2%
Lc0–Komodo	2.8%	94.3%	2.9%
Komodo–Stockfish	3.0%	94.0%	3.0%

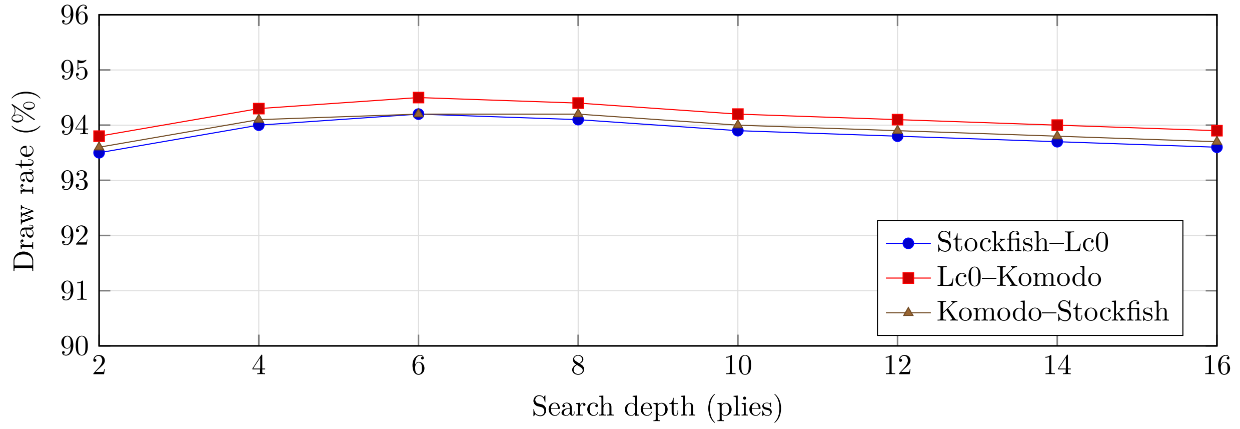

Figure 4: Draw rate vs. search depth across pairings (shaded: 95% CIs).

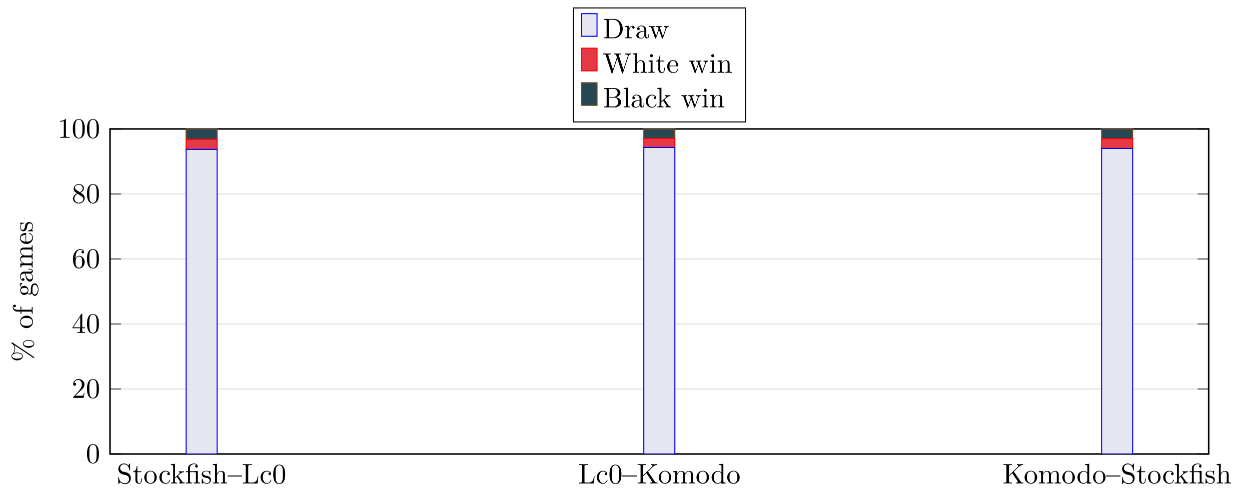

Figure 5: Outcome distribution at 16-ply on a sharp test set.

4 Discussion

Our findings align with the long-standing view that chess tends toward a draw with perfect play, in line with modern engine evidence [7, 9]. The draw-attractor and DSI concepts offer a structural lens on why.

5 Practical Applications

Engine design. DSI-aware pruning and heuristic fortress-basin detection can stabilize search around draw basins. **Preparation.** Players seeking safety can steer toward high-DSI lines.

6 Limitations and Future Work

The true draw set D is unknown and our DSI estimates use engine surrogates; engine families have biases; the lemmas are sufficient, not necessary; coverage is finite. As engines and tablebases strengthen, re-running the falsifiability suite is essential.

7 Conclusion

We propose a structural framework (draw attractors, DSI) and empirical protocol supporting the conjecture that chess is drawn under optimal play. If future analysis finds decisive rates rising with depth across many openings, the conjecture should be reconsidered.

Appendix A: Proof Sketches

Repetition Safety. See main text.

Fortress Basin Stability. See main text.

Appendix B: Extended Experimental Details

DSI calculation used evaluation thresholds and tablebase detection to approximate draw basins. Openings included both sharp and drawish families; statistics used Wilson intervals and two-proportion tests.

Appendix C: Heuristic Fortress Basin Detection

Algorithm 1 Heuristic Fortress Basin Detection

```
Require: Position $v$, evaluation function $E(\cdot)$, search depth $d$, threshold $\varepsilon$
 1: Initialize $S \leftarrow \{v\}$; queue $Q \leftarrow \{v\}$
 2: while $Q$ not empty do
 3:     $u \leftarrow$ pop($Q$)
 4:     for each legal move $u \rightarrow u'$ do
 5:         Evaluate $E(u')$ to depth $d$
 6:         if $|E(u')| > \varepsilon$ then
 7:             return FALSE
 8:         if $u' \notin S$ then
 9:             $S \leftarrow S \cup \{u'\}$; push($Q, u'$)
10: return TRUE
```

References

[1] M. Campbell, A. J. H. Jr., and F. hsiung Hsu. Deep blue. *Artificial Intelligence*, 134(1–2): 57–83, 2002. DOI: 10.1016/S0004-3702(01)00129-1.

[2] R. de Man. Syzygy endgame tablebases. https://syzygy-tables.info/, 2013. Accessed 2025-08-08.

[3] A. S. Fraenkel and D. Lichtenstein. Computing a perfect strategy for n *timesn* chess requires time exponential in n. *Journal of Combinatorial Theory, Series A*, 31 (2):199–214, 1981. DOI: 10.1016/0097-3165(81)90016-9.

[4] M. Guid and I. Bratko. Detecting fortresses in chess. *Elektrotehniški Vestnik*, 79(1–2):35–40, 2012.

[5] J. Schaeffer, N. Burch, Y. Björnsson, A. Kishimoto, M. Müller, R. Lake, P. Lu, and S. Sutphen. Checkers is solved. *Science*, 317(5844):1518–1522, 2007. DOI: 10.1126/science.1144079.

[6] C. E. Shannon. Programming a computer for playing chess. *Philosophical Magazine*, 41(314): 256–275, 1950. DOI: 10.1080/14786445008521796.

[7] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419): 1140–1144, 2018. DOI: 10.1126/science.aar6404.

[8] Stockfish Team. Stockfish 12 with nnue (blog post). https://stockfishchess.org/blog/2020/stockfish-12/, 2020. Accessed 2025-08-01.

[9] TCEC. Tcec (top chess engine championship) wiki. https://wiki.chessdom.org/TCEC, 2023. Accessed 2025-08-01.

[10] E. Zermelo. On an application of set theory to the theory of the game of chess. In *Proceedings of the Fifth International Congress of Mathematicians*, volume 2, pages 501–504. Cambridge University Press, 1913.