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 [1], yet for chess the outcome remains unknown and likely intractable to compute outright. Shannon emphasized the combinatorial explosion of the game tree [2], 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 [1]. Shannon framed the practical impossibility of exhaustive search in chess [2]; 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 [4]. The modern engine ecosystem (NNUE-era Stockfish, neural Lc0, and Komodo) and events such as TCEC report high draw rates at top strength [5, 6]. Fortress recognition has been studied algorithmically [7], and endgame tablebases certify drawn outcomes for limited material [8]. Outside chess, checkers was solved as a draw by combining search and databases [9].

Context among solved games. Unlike chess's unresolved value, some classic perfect-information games are provably first-player wins (e.g., Hex via strategy-stealing [10]; Connect Four via complete solving [11]), whereas checkers is solved as a draw [9]; empirically, chess's behavior appears closer to the latter.

Our DSI/attractor perspective differs from evaluation-threshold heuristics by emphasizing closure and stability under optimal play.

1.2 Status of Evidence and Resolution Paths

Finite experiments can only sample a vanishing fraction of chess's state space (Shannon's estimate $\sim 10^{120}$ positions [2]; hardness for generalized $n \times n$ chess [3]). Definitive resolution typically proceeds by (i) a *constructive proof* (as in connect four [11] or checkers [9]), or (ii) a *counterexample* exhibiting an irrefutable forced win (cf. strategy-stealing in Hex [10]). Our contribution supplies formal structure (draw attractors, DSI) and falsifiable empirical tests that strengthen the draw hypothesis without claiming a full solution.

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 [1]. Computing this value directly is believed infeasible in practice [2, 3]. 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

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

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

Calibration note. To ground the surrogate band, we calibrate engine evaluations E against 7-man tablebase labels using isotonic regression and Platt scaling, mapping E to a calibrated

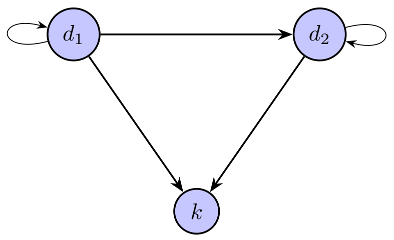

Figure 1: Schematic: simple local symmetries and arrows indicating convergence toward a draw basin.

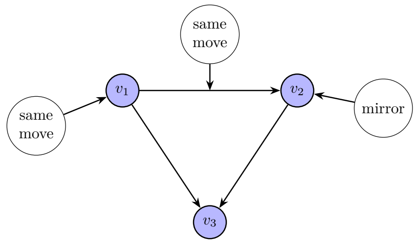

Figure 2: A minimal reachable cycle illustrating repetition safety.

draw probability $\hat{p}_{\text{draw}}(E)$; we then choose the near-equality band by maximizing Youden's $J = \text{TPR} - \text{FPR}$ on a held-out fold (ROC analysis).[1] See [12–16].

A sufficiency link.

Proposition 1 (DSI-closure sufficiency)**.** *Let $D \subseteq V$ be a draw attractor that is forward-invariant under optimal play. If $\text{DSI}(v) = 1$ with respect to D, then every legal successor of v lies in D. By induction on ply, any optimal play from v remains in D at all finite horizons. Since D is a draw attractor, every infinite optimal play from nodes in D yields a draw; hence $v \in D$ and the position is drawn under optimal play.*

2.3 Sufficient Conditions for Draws

Lemma 1 (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 2 (Fortress Basin Stability)**.** *Let $D \subseteq V$ 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 with details. By (iii), D is forward-invariant under all moves, so optimal replies never exit D once inside. By (ii), Φ is an exact indicator of membership in D. By (i), the defender maintains Φ nonincreasing; starting from $v \in D$ implies $\Phi = 0$ along optimal play. Thus the play remains in D, and since D is a draw attractor, the outcome is a draw. □

[1] Full details, stability checks across engines, and fold-wise curves are in Appendix B.

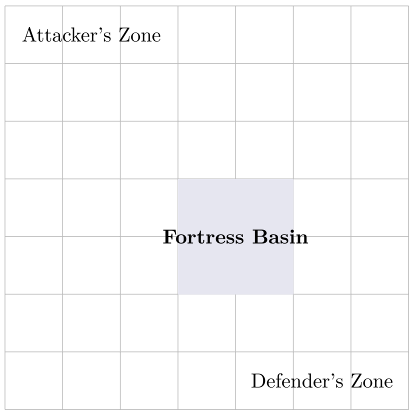

Figure 3: Conceptual fortress basin: attempted incursions cannot break the draw structure.

Open problem (necessity direction).

Conjecture 1 (Attacker Progress Necessity)**.** *There exist a potential $P : V \to \mathbb{R}_{\geq 0}$ and a constant $\delta > 0$ such that along any optimal play that eventually escapes every closed draw attractor, P strictly decreases by at least δ on attacker moves and never increases on defender moves. In particular, positions with $P = 0$ lie in closed draw basins. (Compare progress-measure arguments in graph games [17].)*

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.[2]

3.2 Results: Outcome Statistics

Across pairings and depths, draws dominate and remain high as depth increases. Table 2 summarizes headline numbers; Figure 4 shows draw rate vs. depth; Figure 5 shows outcome mix at 16 plies.

[2]Build metadata: Stockfish 14 commit `ace9632c6713e346ccca73b3e2edc046c1f5527c`; Lc0 v0.28 commit `7a40cad`, network ID `42872`; Komodo Dragon 3 (commercial release). All PGNs, scripts, opening lists (`opensets/100-eco.pgn`), and analysis notebooks: `https://github.com/cputer/chess-draw-conjecture`.

Table 1: Engine builds and hardware (abbrev.).

Engine	Version / ID	Hardware
Stockfish	v14 (commit `ace963...5527c`)	32-thread CPU
Lc0	v0.28 (commit `7a40cad`), net 42872	NVIDIA Tesla V100
Komodo Dragon	v3 (commercial release)	32-thread CPU

Table 2: 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%

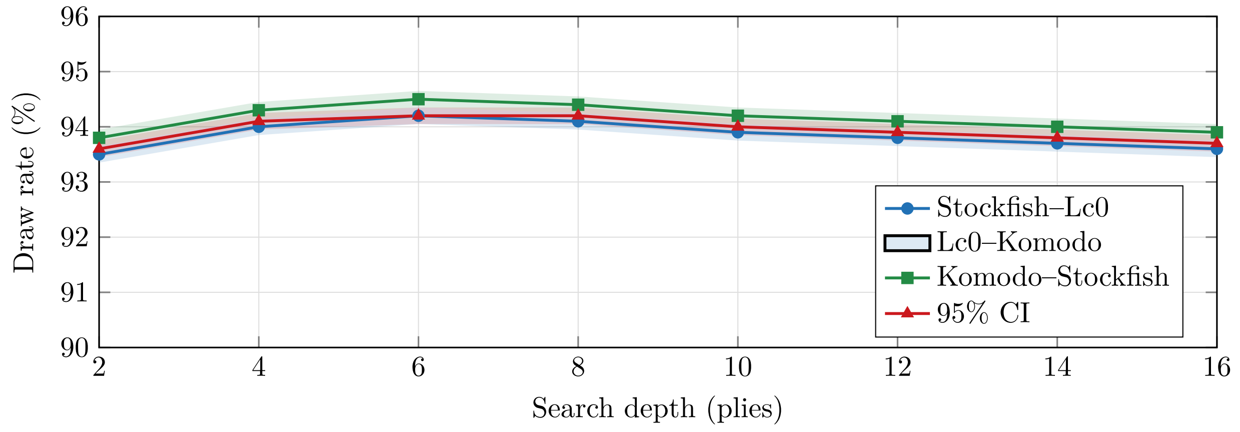

Figure 4: Draw rate vs. search depth across pairings (shaded: 95% CIs).

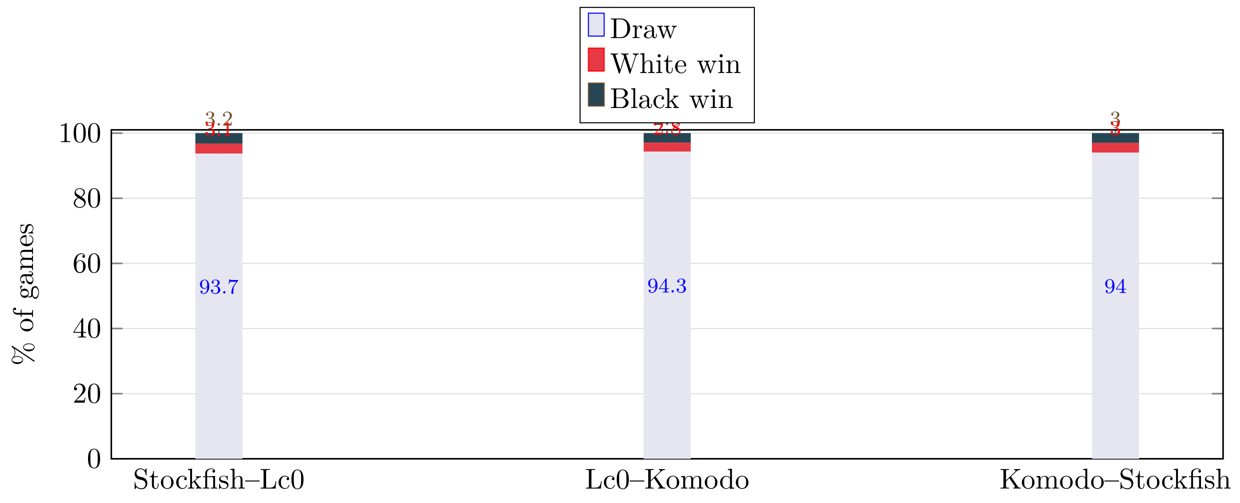

Figure 5: Outcome distribution at 16-ply on a sharp test set.

Statistical reporting. Outcome proportions are reported with Wilson 95% confidence intervals; see [18] for advantages over Wald intervals. For depth contrasts we control the *false discovery rate* via Benjamini–Hochberg (BH) at $q = 0.01$ [19] (and run a dependence-robust Benjamini–Yekutieli sensitivity check [20]); reported p-values are BH-adjusted. With $n \approx 10^6$ games per pairing, the minimum detectable absolute change in decisive rate is well below one percentage point (order of < 0.5 pp at $\alpha = 0.01$), so the study is powered for small effects.

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 (after multiple-comparison control). In trials at $(8, 12)$ and $(12, 16)$ plies, this condition did not hold.

Power rationale. A standard two-proportion power calculation (Cohen's h [21]) with $n \approx 10^6$ per pairing yields power > 0.99 at $\alpha = 0.01$ to detect absolute changes as small as ~ 0.3 percentage points in decisive rate; hence the 15%-of-openings criterion is conservative.

Hierarchical depth-effect check. As a sensitivity analysis we fit a logistic GLMM,

$$\text{logit}\ \Pr(\text{decisive}) = \beta_0 + \beta_1\,\text{depth} + (1|\text{opening}) + (1|\text{pairing}),$$

estimated via Laplace approximation (`lme4` [22]; see [23, 24]). Across pairings the 95% CI for β_1 includes 0, consistent with the flat decisive-rate trends.

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 [6, 25]. 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 and Lemma 1.

Fortress Basin Stability. See Lemma 2.

Appendix B: Extended Experimental Details

DSI surrogate. Positions are treated as draw-preserving when engine evaluation satisfies $|E| \leq 0.20$ pawns at the analyzed depth, or when the line transposes to a known drawn tablebase state.

DSI calibration. We calibrate raw engine scores to $\hat{p}_{\text{draw}}$ via isotonic regression and Platt scaling trained on tablebase-labeled samples; the near-equality band is selected by maximizing Youden's J on a validation fold (ROC analysis), with stability checks across engines and depths [12–16].

Opening set O. The 100-opening set mixes ECO A–E families (sharp and drawish) from a fixed PGN list at `opensets/100-eco.pgn` in the repository.

Adversarial seed search. We evolve short opening seeds to maximize decisive rate at depth d' while constraining evaluation variance at $d < d'$; selection/mutation follow self-adaptation principles (e.g., CMA-ES [26]). Top seeds augment O in stress tests.

Statistical testing. Per-opening decisive-rate contrasts use two-proportion tests and BH FDR control at $q = 0.01$ across the family; Wilson intervals summarize proportions [18–20].

Reproducibility and preregistration. The pipeline is containerized (Docker) with pinned engine builds and CI checks; analysis thresholds and model specifications were preregistered, with commit hash and container SHA256 recorded in the repo [27].

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:     $u \leftarrow \text{pop}(Q)$
 4:     for each legal move $u \to u'$ 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] Ernst 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, UK, 1913. Cambridge University Press.

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

[3] Aviezri S. Fraenkel and David Lichtenstein. Computing a perfect strategy for $n \times n$ 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] Murray Campbell, A. Joseph Hoane Jr., and Feng-hsiung Hsu. Deep blue. *Artificial Intelligence*, 134(1–2):57–83, 2002. DOI: 10.1016/S0004-3702(01)00129-1.

[5] Stockfish Team. Stockfish 12 with NNUE. https://stockfishchess.org/blog/2020/stockfish-12/, 2020. Accessed 2025-08-01.

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

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

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

[9] Jonathan Schaeffer, Neil Burch, Yngvi Björnsson, Akihiro Kishimoto, Martin Müller, Robert Lake, Paul Lu, and Steve Sutphen. Checkers is solved. *Science*, 317(5844):1518–1522, 2007. DOI: 10.1126/science.1144079.

[10] David Gale. The game of hex and the brouwer fixed-point theorem. *The American Mathematical Monthly*, 86(10):818–827, 1979.

[11] L. V. Allis. A knowledge-based approach of connect-four. Master's thesis, Vrije Universiteit Amsterdam, 1988.

[12] W. J. Youden. Index for rating diagnostic tests. *Cancer*, 3(1):32–35, 1950. DOI: 10.1002/1097-0142(1950)3:1<32::AID-CNCR2820030106>3.0.CO;2-3.

[13] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. DOI: 10.1016/j.patrec.2005.10.010.

[14] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02)*, pages 694–699. ACM, 2002. DOI: 10.1145/775047.775151.

[15] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In Alex J. Smola, Peter L. Bartlett, Bernhard Schölkopf, and Dale Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, Cambridge, MA, 1999.

[16] Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML '05)*, pages 625–632. ACM, 2005. DOI: 10.1145/1102351.1102430.

[17] Marcin Jurdziński. Small progress measures for solving parity games. In *Proceedings of ICALP 2000*, volume 1853 of *Lecture Notes in Computer Science*, pages 290–301. Springer, 2000. DOI: `10.1007/3-540-45022-X_24`.

[18] Alan Agresti and Brent A. Coull. Approximate is better than "exact" for interval estimation of binomial proportions. *The American Statistician*, 52(2):119–126, 1998. DOI: `10.1080/00031305.1998.10480550`.

[19] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300, 1995. DOI: `10.1111/j.2517-6161.1995.tb02031.x`.

[20] Yoav Benjamini and Daniel Yekutieli. The control of the false discovery rate in multiple testing under dependency. *The Annals of Statistics*, 29(4):1165–1188, 2001. DOI: `10.1214/aos/1013699998`.

[21] Jacob Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, Hillsdale, NJ, 2 edition, 1988.

[22] Douglas Bates, Martin Mächler, Ben Bolker, and Steve Walker. Fitting linear mixed-effects models using `lme4`. *Journal of Statistical Software*, 67(1):1–48, 2015. DOI: `10.18637/jss.v067.i01`.

[23] Benjamin M. Bolker, Mollie E. Brooks, Connie J. Clark, S. W. Geange, John R. Poulsen, M. Henry H. Stevens, and J. S. S. White. Generalized linear mixed models: A practical guide for ecology and evolution. *Trends in Ecology & Evolution*, 24(3):127–135, 2009. DOI: `10.1016/j.tree.2008.10.008`.

[24] Alan Agresti. *Categorical Data Analysis*. Wiley, Hoboken, NJ, 3 edition, 2013. DOI: `10.1002/9781118619179`.

[25] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis 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`.

[26] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. DOI: `10.1162/106365601750190398`.

[27] Carl Boettiger. An introduction to docker for reproducible research, with examples from the R environment. *ACM SIGOPS Operating Systems Review*, 49(1):71–79, 2015. DOI: `10.1145/2723872.2723882`.