Beyond Machine Persistence

Weaknesses of Single-Premise Proof Swarms and the Design of a Dual-Premise, Obstruction-Exchange Architecture for Automated Mathematical Discovery

Working draft — July 2026

Beyond Machine Persistence: Weaknesses of Single-Premise Proof Swarms and the Design of a Dual-Premise, Obstruction-Exchange Architecture

Working draft, July 2026. Prepared through human–AI collaboration; in line with the disclosure norms urged by the Leiden Declaration [16], the division of labor was: problem framing, architectural direction, and the dual-premise concept by the human author; drafting, prior-art survey, and adversarial self-review by an AI system under explicit instruction to attack novelty claims and treat accuracy as a hard constraint.

Abstract

In July 2026, OpenAI reported that a swarm-configured model produced a complete proof of the Cycle Double Cover Conjecture (CDC), open for roughly fifty years, in under an hour [1, 2]. The system's public artifacts — a released prompt and paper [2, 3] — reveal an architecture of engineered persistence: a hardcoded premise that a proof exists, blinding from the literature, a completion gate rejecting all partial output, 64 information-isolated generation agents, adversarial checker agents equipped with a hand-written error checklist, and a minimum compute budget. We argue this architecture, while sufficient for a class of problems we characterize as *patience-bounded*, carries nine structural weaknesses, the most serious being (i) a hardcoded truth-value that fails catastrophically on false conjectures, (ii) verification acting as a one-shot filter rather than a feedback signal, and (iii) the absence of any novelty or attribution gate. We then survey the piecemeal state of the art — kernel-verified provers (AlphaProof, Aristotle, Seed-Prover, AxiomProver, DeepSeek-Prover-V2), informal reasoning swarms, ML pattern-discovery, conjecture generators, counterexample search, and prover–verifier games — and locate the couplings each lacks. Finally we specify, in full, a **Dual-Premise, Obstruction-eXchange architecture (DPOX)**: two swarms operating under opposite credulous premises (provable / refutable), coupled through a typed, versioned obstruction ledger; a refutation-to-generation feedback loop that conditions each generation wave on the localized failures of the last; a premise controller that maintains a live posterior over the conjecture's truth-value in place of any hardcoded assumption; and a verification stack whose primary output, in the expected case where no proof is found, is an *obstruction atlas* — a characterization of where and why the problem resists, including the type signature of missing objects. We attack our own design, state its inherited ceilings (formalization coverage, semantic faithfulness, concept generation), and

give an evaluation protocol with preregistered reporting of failures. Novelty accounting is explicit throughout; the claimed contribution is confined to the couplings, not the components.

1. Introduction: persistence was the easy half

The public evidence of 2024–2026 splits AI mathematics into two demonstrated capabilities that have never cohabited in one system. Machine-learning pattern detectors have surfaced genuinely new mathematical structure — the murmurations of elliptic curves [4], the hypercube decompositions behind progress on the combinatorial invariance conjecture [5, 6], the natural slope in knot theory [5] — with humans doing all subsequent proving. Separately, reasoning models under heavy test-time compute have constructed proofs and disproofs of named open problems — the disproof of Erdős's 1946 unit-distance conjecture [7], and now the CDC proof [1] — entirely within the technique space of a single domain.

The CDC result is the purest demonstration to date of what we will call *engineered persistence*. Thomas Bloom's assessment of the proof — "short, elementary, and could have been discovered in the 1980s" [8] — and his diagnosis of why it wasn't (a human tries the natural approach, watches it fail, and concludes the easy route is closed, while the machine "does not get discouraged and keeps trying small variations" [8]) together define a stratum of open problems whose difficulty was never conceptual: problems open because the search among known techniques stalled on human discouragement, not because the required mathematics was missing. Bloom's own caveat defines the stratum's limit: such problems are "likely only a small proportion of open problems, and we don't know in advance which they are" [8].

This paper takes the CDC architecture seriously as an engineering artifact, which its public prompt makes possible for the first time [3]. Section 2 reconstructs it. Section 3 states nine weaknesses. Section 4 surveys the piecemeal landscape and identifies the missing couplings. Sections 5–7 specify DPOX in full, attack it, and give an evaluation protocol. Throughout, the organizing principle is the asymmetry that the components themselves keep demonstrating: generation is cheap; verification is the bottleneck; and the epistemic *disposition* of a system — credulous or skeptical — is a configurable parameter whose correct value depends on where in the pipeline it is installed.

2. The CDC architecture, reconstructed

From the released prompt and reporting [1, 2, 3], the system comprises seven mechanisms.

M1 — Premise injection. The prompt instructs the model to assume a complete proof of the conjecture exists, foreclosing the honest default response that the conjecture is open.

M2 — Blinding. The model is barred from internet search, preventing both the retrieval-of-existing-solution failure (the mode behind the October 2025 episode in which claimed GPT-5 solutions to Erdős problems turned out to exist in the literature [9]) and the "this is a known open problem" exit.

M3 — Completion gate. Partial results, reductions to other unproven conjectures, surveys of the state of the art, and explanations of difficulty are all rejected as insufficient; the model cannot respond until a complete candidate proof exists.

M4 — Parallel generation with information isolation. 64 subagents run concurrently; most are deliberately kept ignorant of which approach currently looks most promising, to force independent exploration.

M5 — Adversarial checking. Dedicated checker agents test each candidate against a hand-written list of domain-typical errors (e.g., closed walks misidentified as cycles; reductions that create new bridges).

M6 — Compute floor. The system must compute for at least eight hours before it may consider giving up. (It finished in one [1].)

M7 — Human verification, post hoc. Community verification remained pending at the time of reporting; Bloom's positive but preliminary review is the most detailed public assessment [1, 8].

Two observations frame everything that follows. First, M1–M6 already constitute a crude hybrid of dispositions: M1–M4 install credulous persistence in generation; M5 installs skepticism in verification. The architecture works *because* the dispositions are separated by role. Second, every mechanism is static: the premise never updates, the checklist never grows, the isolation never becomes coordination, and a kill verdict from M5 carries no information back into M4 beyond "try again."

3. Nine weaknesses

W1 — Hardcoded truth-value. M1 answers the open question by fiat. Pointed at a *false* conjecture, the architecture either burns its budget generating nothing acceptable to M3, or — worse — produces the most sophisticated available wrongness, since M3 rejects every honest output and M5's checklist tests local validity, not global truth. The failure is not hypothetical in class: OpenAI's own most celebrated prior result was a *disproof* [7]. A system that must be told the answer's sign before it starts is not resolving conjectures; it is elaborating an assumption. The unit-distance and CDC results were, on this axis, produced by architectures configured with the luck of the right premise.

W2 — Verification as one-shot filter. An M5 kill discards the candidate and nothing else. If sixty candidates die at the same structural point — say, every reduction attempt creates a new bridge — the architecture neither notices the coincidence nor exploits it. The swarm restarts blind. Persistence without memory of *where* the wall stands is a random walk with extra steps; the single most valuable byproduct of mass failure, the localization of the obstruction, is computed implicitly and thrown away. (Bloom's reconstruction of the human failure mode — try, fail, conclude wrongly — is an inference *from* failure location; the machine that out-persisted humans does not perform it.)

W3 — No novelty or attribution gate. Bloom traces the proof's core ideas to a 1983 paper of Bermond, Jackson and Jaeger, uncited; he doubts the strategy was independently invented, "given that its first problem-solving instinct is generally to search for all related papers on a problem and read them" [10]. M2's blinding operates at inference time only; the training weights contain the literature. The architecture therefore cannot distinguish invention from recall, and its output paper misrepresents provenance by omission — the precise failure the Leiden Declaration names among its five threats [16]. A verification stack with no retrieval-attack gate certifies validity while laundering attribution.

W4 — Isolation is not diversity. M4's information isolation prevents groupthink *within* the run but cannot decorrelate what was correlated before the run began: 64 instances of the same weights share the same priors, the same training distribution, the same characteristic blind spots. The swarm's effective diversity is the diversity of one model's temperature samples. Against a problem

whose solution lies outside the model's prior — rather than in a low-probability corner of it — all 64 agents fail *together*, and the compute multiplier buys nothing.

W5 — Checker circularity and static checklists. M5's checkers are the same base intelligence as the generators; the "First Proof" challenge demonstrated the resulting risk profile at scale, with the bulk of submissions constituting "very convincing nonsense" whose flaws surfaced only under expert human reading [11]. A checker sharing the generator's blind spots passes exactly the errors the generator characteristically makes. The hand-written checklist mitigates this for *anticipated* error classes, but it is static (it does not grow from observed failures), domain-specific (it must be re-authored per problem), and human-authored (it scales with expert attention, the resource the system exists to conserve).

W6 — Verifiability selection. The CDC output was trustworthy-in-practice because it was short and elementary — cheap for humans to check. This is not an accident of the problem; it is a selection effect of the architecture. Outputs of engineered persistence over known techniques are *systematically* biased toward the elementary, which is also the only regime where post-hoc human verification (M7) remains affordable. The architecture thus succeeds precisely where its verification debt is small, and would silently accumulate unpayable debt on any problem whose resolution ran to hundreds of pages. Its success class and its safe class coincide today; nothing in the design keeps them coincident.

W7 — Survivorship reporting. Labs attack many problems in parallel and report the hits [8]. The failure distribution — which problems resisted, for how long, at what obstruction — is scientifically the more valuable dataset (it is the raw material of the patience-bounded/concept-bounded classifier) and is currently proprietary or discarded. The public record therefore overstates hit rates and understates the informativeness of failure.

W8 — No stopping theory. The eight-hour floor is arbitrary. The architecture maintains no estimate of problem depth, no calibrated posterior on resolvability-with-current-techniques, and no principled distinction between "not yet" and "not this way." Its budget policy encodes exactly one bit of prior knowledge (persist longer than a discouraged human), which happened to suffice.

W9 — Premise–weights inconsistency. M1 instructs the model to believe what its weights know to be unestablished (the conjecture's openness is in the training corpus). The behavioral effect of running inference against an injected belief inconsistent with parametric knowledge is unmeasured; plausible failure modes include leakage of the true status into reasoning (undermining M1) and systematic overconfidence in candidate proofs (aggravating W5). The architecture treats prompt-level belief as if it overwrote weight-level knowledge; no evidence supports that assumption.

4. The piecemeal landscape and its missing couplings

Every component a better architecture needs exists somewhere, in a system that lacks the others.

Kernel-verified provers. AlphaProof reached IMO 2024 silver by reinforcement learning against the Lean verifier, including test-time RL on problem variants [12]; Aristotle combines an informal reasoning model, MCTS-guided formal search, and a geometry subsystem to reach IMO 2025 gold with machine-checked solutions [13]; Seed-Prover reached formal gold via whole-proof generation with iterative refinement [13]; AxiomProver and the open Numina-Lean-Agent solve 12/12 Putnam 2025 problems, and Aristotle and AxiomProver have each formalized solutions to open Erdős problems [14]; DeepSeek-Prover-V2 holds 88.9% on miniF2F as an open 671B model [15]. The kernel supplies what no LLM checker can: verification that cannot be persuaded. Tao's formulation is

exact: RL "is just so good at finding these backdoors... the kernel is the one thing they can't fake" [15]. Three deficiencies confine these systems. *Substrate*: they prove only what the formal library can express; mathlib's coverage is a rounding error against mathematics, so the legibility ceiling that bounds every gate-based scheme binds hardest exactly at research frontiers. *Specification faithfulness*: the kernel certifies that the stated formal theorem is proved, not that the formal statement means the informal conjecture — a mis-autoformalized statement yields a certified proof of the wrong thing, the formal-layer analog of semantic hallucination, and no deployed system carries a faithfulness gate stronger than back-translation heuristics. *Search, not formation*: MCTS over tactic space explores the interior of existing formal theory; nothing in the loop mints a new definition worth having.

Informal persistence swarms. The CDC and unit-distance systems, per Sections 2–3: natural-language proofs, no kernel, checker circularity (W5), single premise (W1), filter-only verification (W2).

ML pattern discovery. Murmurations [4] and the combinatorial-invariance pipeline [5, 6] establish that learned detectors surface structure humans missed, including structure later *proved* by humans building on the AI-derived objects [6]. Deficiencies: no proof engine (output is a phenomenon, not a theorem); dependence on a human champion to carry the finding to a structural home; and a coverage bias toward computable, dataset-friendly invariants — the discovery-side face of the legibility ceiling.

Conjecture generators. The Ramanujan Machine and PSLQ-style pipelines emit candidate identities at scale [17]. Deficiency: no importance ranking and no proof coupling; the output stream is itself a flood of the kind the Leiden authors fear [16].

Counterexample search. Wagner's RL constructions refuted several open conjectures in combinatorics [18]; SAT/ILP encodings settle finite cases; AlphaEvolve-style program search improves extremal bounds. Deficiency: this community's systems run *disjoint* from proof systems — refutation and proof are institutionally separate machines, so neither side's failures inform the other's search. The unit-distance disproof [7] emerged from the proof-swarm side almost by accident of premise, underscoring W1 rather than curing it.

Prover–verifier games and debate. Training protocols in which a prover is optimized against a checkable-by-weaker-verifier objective [19], and debate-style setups in which adversarial agents argue before a judge [20], install the right *relationship* between generation and verification — but at training time, for legibility and judge-accuracy, on problems with known answers. Deficiency: no deployed discovery loop; the adversarial pressure shapes the model, not the live search on an open problem.

Summary of missing couplings. Across the landscape, three couplings appear nowhere: (C1) refutation feeding generation as a standing signal (every system discards failure location); (C2) proof-search and counterexample-search exchanging information on the same problem under opposite premises (the two communities' machines never meet); (C3) novelty/attribution verification as a first-class gate (everywhere either absent or post hoc). DPOX is the assembly of exactly these three couplings around components that all exist.

5. DPOX: full architecture

5.0 Design principles

P1. *Disposition placement*: credulity in generation, skepticism in verification, never both at one locus. P2. *No hardcoded truth-value*: the system maintains a posterior over the conjecture's status and spends compute proportionally. P3. *Failure is the product*: in the expected case (no resolution), the deliverable is the localized obstruction, at publication grade. P4. *Verification is layered and heterogeneous*: no single checker class, and where formalizable, the kernel is final. P5. *Provenance is a verification dimension*: a result is not "a result" until its relation to prior art is certified.

5.1 Objects

Problem object Π: the conjecture in dual representation — informal statement Π_I and, where autoformalization succeeds, formal statement Π_F with a recorded *faithfulness dossier* (independent back-translations, semantic diffs, and a human sign-off bit for research-grade runs; see 5.6 and the limit in 6.3).

Candidate objects. Proof sketches σ (structured natural language with explicit lemma DAG); formal fragments ϕ (Lean code, possibly containing `sorry` placeholders); construction candidates κ (explicit objects, generators, or programs emitting witnesses); reductions ρ (claimed implications between Π and other statements, kept as first-class *conditional* objects rather than rejected outright as in M3).

Obstruction records — the central data structure. An obstruction is a typed, versioned record

$$\omega = (\ell,\ \tau,\ F,\ E,\ v)$$

where ℓ is the *location*: the minimal lemma/subgoal in a candidate's DAG at which failure occurred; τ the *type*: which gate killed it (5.5) and the structural class of the failure (invariant mismatch, no-go violation, kernel error class, checker counterexample-to-step, faithfulness doubt); F the *forgetting annotation*: for transfer-based attempts, the structure the attempted map provably fails to preserve — the principled seed of the next attack, per the gate-engine design of which DPOX is the closed-loop extension; E an *evidence weight*: how many independent candidates, from how many distinct strategy arms, died at ℓ; and v version stamps binding the record to the library snapshot and checker versions that produced it (enabling un-blessing; see 5.7).

The ledger L: an append-only, versioned store of obstruction records, near-miss records (from the B-swarm, below), exhausted-arm signatures, and attribution findings. L is the shared state through which all coupling flows; agents otherwise share nothing.

5.2 The A-swarm (proof-hunting)

Premise: Π is provable with existing theory plus at most small local inventions. Disposition: credulous persistence — M1–M4 and M6 are retained *inside this swarm*, where they belong. Two corrections to CDC design:

Strategy-seeded decorrelation (vs. W4). The swarm is partitioned into arms $A_1 \dots A_k$, each pinned by prompt to a distinct attack family — induction/structural, extremal/counting, probabilistic, algebraic reformulation, analytic, and a *transfer arm* running the functorial gate-engine (propose a witnessing map from a structurally similar formalized domain; attack it at its forgetting). Pinning by strategy forces the exploration diversity that information isolation merely permits. Where budget

allows, arms run on heterogeneous base models; weight-level decorrelation is the only full cure for W4 and is priced in 6.1.

Ledger conditioning (the standing loop; vs. W2). Each generation wave is prompted with a digest of L: the top open obstructions ranked by attack value $E \cdot \text{centrality}(\ell)$; forbidden moves (failure signatures with E above threshold, each carried *with its reason*, so that an arm may deliberately target a forbidden move's load-bearing assumption rather than merely avoid it); and exhausted-arm declarations. The prompt template makes the target explicit: "prior attempts concentrate failure at ℓ under type τ; vary there." This converts Bloom's "keeps trying small variations" from undirected to obstruction-directed variation — Lakatos's lemma-incorporation [21] as a running process rather than a philosophical description.

5.3 The B-swarm (counterexample-hunting)

Premise: Π is false. Disposition: equally credulous persistence toward refutation. Arms: explicit small-case search (SAT/ILP/CP encodings where Π admits finite certificates); generative construction (Wagner-style RL and program search over object families [18]); asymptotic violation (hunting parameter regimes where Π's quantitative form degrades); and *hypothesis-deletion probes* — systematic search for objects satisfying all but one hypothesis of Π.

The B-swarm's characteristic output below a full counterexample is the **near-miss record**: an explicit object failing Π's conclusion while satisfying all hypotheses except a named one, or satisfying all hypotheses while failing a step of some A-candidate proof (in which case it enters L as a checker-grade kill with a concrete witness — the strongest possible τ). Near-misses carry the same (ℓ, τ, F, E, v) typing: the hypothesis that blocked the construction is a *location*, and the density of near-misses against a hypothesis is evidence about where Π's truth is load-bearing.

5.4 The obstruction-exchange protocol (coupling C2)

Exchange runs through L on a fixed cadence, in both directions, by two translation rules.

A→B (failure loci as habitat constraints). A high-E obstruction at lemma ℓ with quantifier signature Q compiles to a search constraint for B: any counterexample, if one exists, must realize $\neg\ell$'s existential content. Concretely: if all proof attempts die trying to establish "every bridgeless graph admits decomposition property P," then B's generators are re-targeted to the family of bridgeless graphs stressing $\neg P$ — the failure of the provers describes the *habitat* of the counterexample. The translation is mechanical: negate the failing lemma, extract its witnesses' type, and emit a generator specification.

B→A (near-miss boundaries as candidate lemmas). When B's constructions die repeatedly against hypothesis h or emergent property P (every near-miss family gets destroyed by the same structural feature), that feature compiles to a conjecture-grade subgoal for A: "prove that h forces P." The A-swarm's transfer and algebraic arms receive it as a targeted lemma; if proved (kernel-checked where possible), it both hardens future A-attempts and permanently prunes B's search space. This is the mechanization of the working mathematician's oscillation between proof attempt and counterexample hunt, which no deployed system currently performs across its own boundary.

Convergence behavior. Under exchange, A's obstruction set and B's near-miss frontier trace the same boundary from opposite sides: the *live boundary* of Π — the minimal structural locus where the problem's truth is actually contested. The architecture's central empirical claim, testable per Section 7, is that dual-premise search with exchange localizes this boundary in strictly fewer

candidate-generations than either premise alone, because each side's failures are the other side's map.

5.5 Verification stack (vs. W5, W3)

Gates in cost order; every kill emits a typed obstruction into L rather than a boolean.

G1. *Static structural gates*: type/dimension consistency; invariant matching; the negative library of no-go and obstruction theorems (a candidate contradicting a known impossibility dies at cost near zero). G2. *Formal witnessing*: autoformalize the candidate's lemma DAG; attempt kernel verification of whatever fragment formalizes; partial type-check localizes exactly (the failing subterm is ℓ for free). Where Π_F exists, a fully kernel-checked proof or a kernel-checked counterexample is *final* — modulo the faithfulness dossier, which is the stack's one non-kernel dependency (6.3). G3. *Cross-model adversarial review*: checker agents drawn from base models disjoint from the generating arm (decorrelating W5's shared blind spots), armed with (a) the static checklist seeded from domain expertise as in M5, and (b) — unlike M5 — the *learned* checklist: every τ ever recorded in L for this problem class, so the checklist grows from observed failure instead of remaining hand-authored. G4. *Novelty/attribution gate* (coupling C3): a retrieval adversary with full literature access — deliberately the inverse of M2's blinding, safe here because it runs in verification, where recall is the point — executes the attack "where does this already exist?" via citation search, embedding similarity over the candidate's lemma structure, and targeted reading. Verdicts: *new / known (cite) / known-variant (cite + delta)*. Output attaches to the result as a provenance annotation; a proof that is valid but substantially anticipated ships as such. This gate is cheap, high-precision, plays to demonstrated model strength (literature search and error-hunting are precisely what current systems do best), and directly discharges the Leiden attribution requirement [16] architecturally rather than normatively. G5. *Human attention router*: the scarce resource, spent last, on exactly two object classes — candidates surviving G1–G4, and high-E obstruction clusters whose atlas entry (5.8) merits specialist reading.

5.6 The premise controller (vs. W1, W8, W9)

No agent is ever told the truth-value of Π as a fact about the world; each swarm receives its premise as a *working stance*, which is consistent with weight-level knowledge that the problem is open (dissolving W9's inconsistency: "assume for the search that a proof exists" is a coherent instruction where "a proof exists" is not).

The controller maintains a posterior $p_t = P(\Pi \text{ true} \mid L_t)$ updated on ledger evidence: kernel-checked lemmas hardening one side; near-miss density and its gradient; obstruction hardness growth (an obstruction whose E grows superlinearly while attack diversity saturates is evidence of a wall, i.e., of concept-boundedness rather than falsity); and reduction objects ρ linking Π's probability to neighbor problems. Compute allocates to swarms as a bandit over arms with mixing weights tied to p_t (never to 0/1: a floor fraction keeps the minority premise alive, since the posterior is itself fallible). Stopping is a decision rule over $(p_t, \text{atlas maturity}, \text{budget})$: the run ends with a resolution, or with a *verdict object* — "patience-bounded, unresolved, boundary at [atlas]" versus "evidence of concept-boundedness: all arms saturate at obstruction ω^*; the missing object has type signature [F-derived characterization]" — with calibrated confidence. W8's arbitrary floor becomes a measurable quantity; W7's discarded failure distribution becomes the primary published artifact.

5.7 Ledger integrity (un-blessing)

L is load-bearing shared state, hence the design's most attractive corruption target (6.2). Mitigations: obstruction records above an E-threshold require G3 re-verification by a disjoint checker before they may condition generation ("verified obstruction" tier); all records carry v stamps, and a library update, checker patch, or faithfulness revision triggers *un-blessing* — automatic demotion of every record whose stamps are invalidated, with downstream conditioning recomputed. Verdicts are therefore always "as of version," never flat — the cache-invalidation discipline that any live-updated formal substrate forces.

5.8 Outputs

Three product classes, in descending expected frequency: (i) the **obstruction atlas** — the versioned map of the live boundary: verified obstructions ranked by E and centrality, near-miss families, forced-lemma results, and, where the transfer arm ran, the type-characterization of missing objects (for the Riemann Hypothesis this atlas would *contain* the Hilbert–Pólya specification — self-adjoint operator, unitary symmetry class, prescribed spectral statistics, no known construction — as its terminal entry, which is the honest ceiling of the architecture on concept-bounded problems); (ii) the **classifier verdict** with calibrated confidence, the direct answer to "we don't know in advance which they are" [8]; (iii) **resolutions** — kernel-checked where Π_F exists, G3/G4-certified otherwise — always shipped with the G4 provenance annotation.

6. Attacking DPOX

6.1 Decorrelation is bought, not designed. Strategy pinning diversifies *search direction*; it cannot diversify the prior generating the searches. If every arm runs one base model, a solution outside that model's representational reach defeats all arms together, exchange or no exchange (W4 survives in weakened form). The full cure — heterogeneous frontier models across arms — multiplies cost and engineering surface, and the residual correlation of models trained on overlapping corpora is unmeasured. DPOX narrows W4; it does not close it.

6.2 Ledger poisoning. A wrong obstruction record with high E — e.g., a checker systematically misjudging one error class — misdirects *both* swarms simultaneously; coupling turns a local checker bias into a global search bias. The verified-obstruction tier and un-blessing bound but do not eliminate this: G3 re-verification inherits whatever blind spot is shared across checker models. The coupling that is the design's contribution is also its distinctive attack surface; single-premise swarms cannot be poisoned this way because they remember nothing.

6.3 The faithfulness dependency. G2's finality holds only relative to the $\Pi_I \leftrightarrow \Pi_F$ correspondence, and no strong faithfulness gate exists anywhere in the field: back-translation catches gross mistranslation, not subtle semantic drift. Every kernel-checked claim DPOX emits is conditional on a dossier whose strongest current entry is human sign-off — the deepest unsolved dependency, inherited untouched.

6.4 The legibility ceiling, inherited. G1–G2 reach only formalized substrate; the transfer arm's library is mathlib-bounded; the atlas over-resolves boundaries in formalized regions and blurs where formalization thins. DPOX inherits, in full, the coverage limit that bounds every gate scheme: it makes the engine's *judgments* sharper without making its *sight* wider. Concept-bounded problems remain out of reach by construction — DPOX characterizes missing objects (via F) and cannot build them. On the Riemann Hypothesis, DPOX's honest promise is the best-localized statement

of why it cannot succeed.

6.5 Goodhart on the atlas. If atlas entries become academically valuable (creditable, citable), generation drifts toward manufacturing impressive-looking obstructions — failure theater. Mitigation (verified tier, human router) costs the scarce resource; unmitigated, the system optimizes its own primary product into noise.

6.6 Economics. Dual premises, heterogeneous models, standing verification, and B-swarm construction search (SAT instances are not cheap) multiply CDC-style cost severalfold per problem. The design bets that obstruction-directed search saves more generations than the coupling costs; Section 7 makes the bet falsifiable. If it loses, DPOX is a more honest but slower CDC, valuable only for its verdict objects.

7. Evaluation protocol

Retrospective. Re-run resolved cases with the resolution excluded from conditioning: CDC (true premise correct) and unit-distance (false premise correct). Primary metric: candidate-generations to resolution, DPOX vs. single-premise ablations (A-only, B-only, A+B without exchange, A+B with exchange but no ledger conditioning). The design's central claim (5.4) predicts a strict ordering.

Prospective. A preregistered slate of open problems spanning predicted patience-bounded and concept-bounded strata (FrontierMath's Open Problems tier supplies vetted candidates [22]), with *all* outcomes published — resolutions, verdicts, and atlases alike — as the direct correction to W7. Metrics: resolution rate; atlas quality under blinded specialist review (does the obstruction map match expert understanding of why the problem is hard, and does it teach the expert anything); calibration of the premise controller's posterior against eventual resolutions; G4 attribution precision/recall against expert literature audit.

Falsifiers. The architecture fails its own test if exchange yields no generation savings over the uncoupled ablation; if atlas quality does not beat a baseline of "LLM essay on why the problem is hard"; or if the premise controller's posterior is uncalibrated on the retrospective set.

8. Relation to prior art, and an honest novelty accounting

Every component is standing art: kernel-verified search [12, 13, 14, 15]; engineered persistence [1, 3]; RL counterexample construction [18]; prover–verifier training and debate [19, 20]; conjecture generation [17]; ML pattern discovery [4, 5, 6]; graded gate sieves and functorial transfer checking (developed in the working sessions preceding this draft); and the entire philosophical frame is Popper's conjectures-and-refutations [23] and, more exactly, Lakatos's proofs-and-refutations [21], of which DPOX is an attempted mechanization. Automated conjecture–refutation loops have antecedents as old as Lenat's AM and Fajtlowicz's Graffiti.

The claimed contribution, held at low-to-moderate confidence and confined to couplings: (a) dual-premise swarms over the *same* open problem governed by a live posterior rather than a hardcoded truth-value; (b) the typed, versioned obstruction ledger with the A↔B translation rules of 5.4; (c) refutation-to-generation conditioning as a standing loop; (d) the obstruction atlas and calibrated boundedness verdict as primary deliverables, inverting the field's current output hierarchy. We could not locate this assembly in the literature; absence of evidence at search depth is weak evidence of absence, and specialists in automated reasoning are explicitly invited to deflate any of (a)–(d) with citations. Under this paper's own G4 standard, the design does not count as a result until that gate has run.

References

[1] M. Bastian, "OpenAI's GPT-5.6 Sol Ultra reportedly solves a 50-year-old math problem in under an hour," The Decoder, July 11, 2026. [2] OpenAI, CDC proof paper, cdn.openai.com/pdf/04d1d1e4-bc75-476a-97cf-49055cd98d31/cdc_proof.pdf, 2026. [3] OpenAI, CDC system prompt, cdn.openai.com/pdf/04d1d1e4-bc75-476a-97cf-49055cd98d31/cdc_prompt.pdf, 2026. [4] Y.-H. He, K.-H. Lee, T. Oliver, A. Pozdnyakov, "Murmurations of elliptic curves," arXiv:2204.10140, 2022. [5] A. Davies, P. Veličković, L. Buesing, et al., "Advancing mathematics by guiding human intuition with AI," Nature 600, 2021. [6] G. Barkley, C. Gaetz, et al., work proving cases of the combinatorial invariance conjecture via hypercube decompositions, 2023–2026. [7] Coverage of OpenAI's disproof of the Erdős unit-distance conjecture, with supporting remarks by N. Alon, M. Wood, T. Bloom, May 2026. [8] T. Bloom, public assessment of the CDC proof, x.com/thomasfbloom, July 2026. [9] Reporting on retracted claims that GPT-5 solved open Erdős problems (solutions located in existing literature), October 2025. [10] T. Bloom, on the CDC paper's missing citations to Bermond–Jackson–Jaeger (1983), x.com/thomasfbloom, July 2026. [11] "First Proof" challenge, February 2026: expert assessment of AI submissions to ten research problems. [12] AlphaProof team, "Olympiad-level formal mathematical reasoning with reinforcement learning," Nature, 2025. [13] Harmonic (T. Achim et al.), "Aristotle: IMO-level automated theorem proving," arXiv:2510.01346, 2025; ByteDance Seed-Prover, reported therein. [14] Survey of Lean 4 agents: AxiomProver, Numina-Lean-Agent, Putnam 2025 results, and formalized Erdős-problem solutions, arXiv:2604.03789, 2026. [15] DeepSeek-Prover-V2 miniF2F results and T. Tao on kernel soundness, ETAPS 2026 materials (L. de Moura). [16] Leiden Declaration on Artificial Intelligence and Mathematics, June 2, 2026. [17] G. Raayoni et al., "Generating conjectures on fundamental constants with the Ramanujan Machine," Nature 590, 2021. [18] A. Z. Wagner, "Constructions in combinatorics via neural networks," arXiv:2104.14516, 2021. [19] OpenAI, prover–verifier games for LLM legibility, 2024. [20] G. Irving, P. Christiano, D. Amodei, "AI safety via debate," arXiv:1805.00899, 2018. [21] I. Lakatos, *Proofs and Refutations*, Cambridge University Press, 1976. [22] Epoch AI, FrontierMath and the Open Problems tier; benchmark paper arXiv:2411.04872, 2024. [23] K. Popper, *Conjectures and Refutations*, 1963.