Stochastic Witness Calculus:
Exact Measure Certificates for Mathematical Existence, Learned Proof Search, and Anytime-Valid Empirical Claims

Kenju Tomita
Rochester Institute of Technology; Ephemerent Research

September 4, 2026

Abstract

Probability can establish deterministic existence, but automated reasoning systems typically treat randomized search, formal proof, model counting, and empirical confidence as unrelated objects. We introduce *Stochastic Witness Calculus* (SWC), a machine-oriented certificate interface for proving that a sound verifier accepts a set of witnesses with strictly positive measure. The resulting conclusion is an ordinary deterministic theorem: probability is used to certify nonemptiness, not to weaken truth. We give an instance-indexed semantics, a small proof calculus, family and cover rules for universal statements, support-preserving mixtures, change-of-measure and conditioning rules, product and sequential composition, moment-based mass certificates, a dual obstruction rule, and a compactness extension for continuous witness spaces. We also identify a no-free-lunch boundary: for full-support finite distributions, positive accepted mass is equivalent to satisfiability, and exact mass computation inherits weighted model-counting hardness.

We implement an exact finite kernel and evaluate six settings. On 60 planted 16-variable SAT instances, a learned product proposal raises median exact witness mass from 1/65,536 to 0.939; a 10% uniform mixture preserves support while yielding a median 27,810-fold increase. A non-enumerative dynamic-programming backend certifies weighted mass over witness spaces of size 2^{2048} in about 1.1 seconds and agrees exactly with an independent closed form. A quantifier-audit stress test shows that 10^4 random checks miss a single exceptional instance among 10^6 with probability 0.9900, while 10^5 samples miss a true witness mass of 10^{-6} with probability 0.9048. For the Erdős–Straus case study, exact enumeration certifies finite witnesses for all 82,887 primes $p \equiv 1 \pmod{24}$ below 10^7 using shifts $c \leq 127$; this is finite verification, not a proof of the open conjecture. A proof-carrying integrity stress test accepts all 300 exact certificates and rejects all 1,800 tampered or floating-point-only variants. Finally, optional-stopping simulations contrast a 30.8% null crossing rate for repeated nominal tests with 4.34% for a likelihood-ratio e-process. SWC is a certificate and systems abstraction built on the classical probabilistic method, not a new axiom or a shortcut around proof complexity.

Keywords: probabilistic method; formal verification; automated theorem proving; proof certificates; witness distributions; weighted model counting; quantifier auditing; e-processes; anytime-valid inference; Erdős–Straus conjecture.

1 Introduction

A probabilistic proof can establish a statement with complete mathematical certainty. The standard mechanism is elementary: if a random construction produces a valid object with positive probability, then at least one valid object exists. This idea underlies the probabilistic method throughout combinatorics [1]. Its logical force is exact, not approximate.

Yet three distinct practices are often conflated:

1. *probability inside a proof*, where a measure calculation proves deterministic existence;
2. *randomized proof search*, where a generator samples candidate witnesses but supplies no proof that success mass is positive;
3. *statistical evidence about physical reality*, where conclusions remain conditional on a data-generating model and an error budget.

This paper proposes SWC as a common certificate language that keeps these modes separate. The exact mode is designed for formal mathematics and automated theorem proving. The empirical mode is designed for sequential physical experiments and scientific monitoring. They share a generator–verifier–auditor architecture, but they do not share the same epistemic endpoint.

The logical core is classical. Formal probabilistic reasoning already appears in Coq, Isabelle/HOL, EasyCrypt, probabilistic program logics, and formalizations of the probabilistic method [2, 3, 7, 8]. Recent proof-mining work also studies extraction of computable bounds from probabilistic existence arguments [17]. The proposed contribution is therefore not the implication "positive probability implies existence." It is the following systems-level synthesis:

- an instance-indexed certificate schema for positive-mass existence;
- an explicit trusted-kernel boundary between learned generation and deductive verification;
- support-preserving mixtures that prevent learned policies from erasing rare witnesses;
- a dual obstruction rule for probabilistic discovery of certified counterexamples;
- quantifier-discipline checks that block invalid average-to-pointwise promotion;
- a separate anytime-valid empirical mode for claims about physical systems;
- a reproducible benchmark spanning SAT, an open-number-theory case study, and sequential testing.

Main claim. When all mathematical obligations are formally discharged, an exact SWC certificate is as strong as an ordinary proof because it compiles into one. In physical science, no method can remove assumptions about instruments, interventions, and data generation; the appropriate analogue is instead a theorem about the protocol's false-certification risk under an explicit assumption set.

Contributions. This paper makes six concrete contributions while keeping the classical logical core explicit.

1. We define a proof judgment for positive-mass existence certificates and prove soundness and conservativity relative to an ambient logic.
2. We extend the basic rule to universal families through pointwise, cover, and well-founded reduction rules, making the quantifier obligations explicit.
3. We give compositional certificate rules for mixtures, products, Markov kernels, conditioning, change of measure, deterministic pushforwards, first/second moments, and local-lemma backends.
4. We specify a trusted-kernel boundary and a proof-carrying mass architecture compatible with exact enumeration, dynamic programming, knowledge compilation, and formally verified model counting.
5. We define an adversarial quantifier-audit discipline that rejects average-to-pointwise promotion and zero-hit-to-zero-mass inference.

6. We evaluate the interface in six complementary experiments spanning learned discrete search, non-enumerative exact counting, universal-quantifier stress tests, open-number-theory certificates, proof-carrying integrity, and anytime-valid empirical monitoring.

2 Related Work and Positioning

2.1 The probabilistic method

The classical probabilistic method proves existence by defining a probability space and showing that the set of desired objects has nonzero measure, often through expectation, concentration, deletion, alteration, or the Lovász local lemma [1]. Edmonds and Paulson formalize reusable versions of these arguments in Isabelle/HOL, including a formal Lovász local lemma, and document gaps that informal treatments can hide [8]. SWC packages the same logical pattern as an instance-indexed certificate intended to interact with search models and proof kernels.

2.2 Verification of probabilistic programs

Randomized algorithms and probabilistic programs have been formalized in Coq and Isabelle/HOL [2, 7]. Relational logics and coupling methods support reasoning about uniformity, independence, privacy, and cryptographic games [3]. Systems such as Caesar target deductive verification of probabilistic programs, while reparameterization methods compile randomness into deterministic functions of random seeds [21, 24]. Those systems primarily verify probabilistic computations. SWC instead takes a theorem-facing view: the output of a probabilistic program is treated as a witness distribution, and the central certificate is a lower bound on the mass of deterministically valid witnesses.

2.3 Formal proof assistants, witnessing distributions, and AI-guided search

Lean and Mathlib exemplify the small-kernel model in which generated proof terms are checked independently of the generator [15, 23]. Contemporary theorem-proving agents similarly place exact verifiers in the loop, while recent systems automate proof engineering in specialized probabilistic logics such as EasyCrypt [13]. Paradise et al. use a sparse witnessing distribution as a proof oracle for checking approximate consistency of exponentially many probabilistic claims [18]; their goal is complexity-theoretic verification of a predictor, whereas SWC treats a distribution over mathematical witnesses as an object whose accepted mass can certify deterministic existence. SWC complements these lines by asking a generator to propose not only a witness or proof script, but a distribution plus a checkable lower bound on its accepted mass.

2.4 Certified counting and proof-carrying mass

For a finite Boolean witness space, accepted probability is a weighted model count. Exact counting is generally intractable—the classical counting complexity of Boolean solution sets traces to Valiant's #P framework [25]—but structured representations can make counting and checking tractable. Capelli studies knowledge-compilation languages as proof systems for #SAT [5]. Bryant, Nawrocki, Avigad, and Heule introduce certified partitioned-operation graphs and a Lean-verified toolchain for weighted and unweighted model counting [4]. Tan et al. formally verify the PAC guarantee of an approximate model counter and dynamically certify solver calls [22]. These systems provide natural back ends for SWC mass obligations. Exact proof-producing counts can discharge mathematical certificates directly;

PAC counts remain bounded-error unless a separate deterministic lower-bound certificate removes the residual failure probability.

2.5 Proof-carrying systems and trusted kernels

The architectural principle is close to proof-carrying code: an untrusted producer may perform expensive synthesis, while a small consumer checks a portable certificate [16]. In SWC, the producer may be a language model, optimizer, sampler, model counter, or human mathematician. The trusted consumer checks distribution normalization, verifier soundness, the accepted-mass lower bound, positivity, and quantifier scope. The system is therefore designed so that improvements in search do not enlarge the trusted base.

2.6 Anytime-valid empirical inference

Confidence sequences and e-processes provide time-uniform error guarantees under adaptive monitoring [11, 19]. Their conclusion remains statistical rather than deductive about nature, but the error statement is itself mathematical and can be machine checked. We adopt this distinction in the empirical mode of SWC.

3 Exact Mathematical Mode

3.1 Instance-indexed witness systems

Definition 3.1 (Witness problem)**.** A witness problem is a tuple

$$\mathcal{P} = (I, \{\mathcal{W}_i\}_{i \in I}, R),$$

where I is an instance space, $\mathcal{W}_i$ is the witness space for instance i, and $R(i, w)$ is the target relation. The intended theorem is

$$\forall i \in I \; \exists w \in \mathcal{W}_i \; R(i, w).$$

Definition 3.2 (Exact stochastic witness certificate)**.** For a fixed instance i, an exact SWC certificate is

$$\Pi_i = (\mu_i, V_i, \delta_i, \pi_{\text{sound}}, \pi_{\text{mass}}, \pi_+),$$

where:

1. μ_i is a probability measure on $\mathcal{W}_i$;
2. $V_i : \mathcal{W}_i \to \{0, 1\}$ is a deterministic verifier;
3. π_{sound} proves $V_i(w) = 1 \Rightarrow R(i, w)$;
4. π_{mass} proves $\mu_i(\{w : V_i(w) = 1\}) \geq \delta_i$;
5. π_+ proves $\delta_i > 0$.

Theorem 3.3 (Stochastic witness soundness)**.** *If an exact* SWC *certificate exists for instance i, then $\exists w \in \mathcal{W}_i$ such that $R(i, w)$.*

Proof. Let $A_i = \{w : V_i(w) = 1\}$. If no valid witness existed, verifier soundness would imply $A_i = \varnothing$. Every probability measure satisfies $\mu_i(\varnothing) = 0$, contradicting $\mu_i(A_i) \geq \delta_i > 0$. Thus A_i contains an accepted witness, and soundness converts it into a valid witness. □

The conclusion is deterministic. There is no residual probability that the theorem is false once the certificate obligations are proved.

3.2 Family certificates and universal theorems

A universal statement requires more than many successful instances. It requires one proof object whose obligations range over every instance.

Definition 3.4 (Uniform certificate family)**.** A uniform SWC family for a witness problem $\mathcal{P}$ consists of definable maps

$$i \mapsto (\mu_i, V_i, \delta(i))$$

and proofs, uniform in i, that V_i is sound, that $\mu_i(V_i^{-1}(1)) \geq \delta(i)$, and that $\delta(i) > 0$ for every $i \in I$.

Corollary 3.5 (Pointwise compilation)**.** *A uniform certificate family proves*

$$\forall i \in I \; \exists w \in \mathcal{W}_i \; R(i, w).$$

The lower bound need not be uniform in magnitude. A function such as $\delta(i) = 2^{-2^i}$ is logically sufficient, although it gives an impractical extraction algorithm. The crucial requirement is pointwise positivity, not a positive average over i.

3.3 A proof rule

The exact mode can be exposed as the inference rule

$$\frac{\mu_i \in \mathsf{Prob}(\mathcal{W}_i) \quad \forall w \, [V_i(w) = 1 \Rightarrow R(i, w)] \quad \mu_i(V_i^{-1}(1)) \geq \delta_i \quad \delta_i > 0}{\exists w \in \mathcal{W}_i \; R(i, w)} \; \text{SWC-Exists}.$$

A universal theorem requires the premises for every instance i, not merely on average over an instance distribution.

Proposition 3.6 (Conservativity)**.** *Assume the measure theory and verifier semantics used by* SWC *are formalized in an ambient logic T. Every theorem derived by* SWC-Exists *is derivable in T without adding a new axiom.*

Proof. The rule is an abbreviation for the theorem that a measurable set of positive measure is nonempty, followed by verifier soundness. Both are ordinary derivations in T. □

Proposition 3.7 (Embedding ordinary constructive proofs)**.** *If a conventional proof provides a witness w_i with $R(i, w_i)$, then it induces an* SWC *certificate with the point mass $\mu_i = \delta_{w_i}$ and lower bound $\delta_i = 1$.*

Thus SWC does not replace ordinary proofs; it contains constructive existence as a degenerate case.

3.4 Quantifier discipline

Proposition 3.8 (Pointwise requirement)**.** *The statement*

$$\mathbb{P}_{i \sim \nu}[\exists w \; R(i, w)] = 1$$

does not in general imply $\forall i \; \exists w \; R(i, w)$.

Proof. A probability-one event may exclude a nonempty ν-null set. A universal conclusion requires a certificate for each instance or a separate proof that the null exceptional set is empty. □

This blocks a common invalid transition in open problems: density-zero or almost-sure solvability is not universal solvability.

Proposition 3.9 (Finite cover rule)**.** *Suppose $I = I_1 \cup \cdots \cup I_m$ and, for each j, a uniform certificate family proves $\forall i \in I_j \, \exists w \, R(i, w)$. Then the families compose into a proof over all of I.*

Proof. For any $i \in I$, choose an index j with $i \in I_j$ and apply the corresponding certificate family. In a proof assistant, membership evidence or a decidable classifier supplies the branch tag. □

Proposition 3.10 (Well-founded reduction rule)**.** *Let $\prec$ be a well-founded relation on I. Suppose base instances have exact certificates and every nonbase instance i has a computable reduction $r(i) \prec i$ together with a sound witness transformer*

$$T_i : \mathcal{W}_{r(i)} \to \mathcal{W}_i, \qquad R(r(i), w) \Rightarrow R(i, T_i(w)).$$

Then certificates for reduced instances compile by well-founded induction into certificates for all instances.

This rule clarifies that SWC is compatible with ordinary induction and reduction arguments; probability need only enter at the leaves where nonemptiness is established.

3.5 Support-preserving learned generators

A learned proposal may concentrate mass on likely witnesses but accidentally assign negligible or zero mass to rare valid ones. The following mixture protects sound search support.

Proposition 3.11 (Support-preserving mixture)**.** *Let μ_0 be a baseline distribution, μ_1 any learned distribution, and*

$$\mu_\eta = \eta\mu_0 + (1 - \eta)\mu_1, \qquad 0 < \eta \leq 1.$$

For every measurable accepted set A,

$$\mu_\eta(A) \geq \eta\mu_0(A).$$

Hence any positive baseline witness mass remains positive after learning.

Proof. Immediate from nonnegativity: $\mu_\eta(A) = \eta\mu_0(A) + (1 - \eta)\mu_1(A) \geq \eta\mu_0(A)$. □

The learned policy is therefore outside the trusted logical core. It may improve search efficiency, but the mixture and exact mass proof preserve theorem soundness.

Proposition 3.12 (Change-of-measure transfer)**.** *Let μ and ν be probability measures and let A be the accepted set. If $\mu \ll \nu$ on A and*

$$\frac{d\mu}{d\nu}(w) \leq C < \infty \quad \text{for } \nu\text{-almost every } w \in A,$$

then

$$\nu(A) \geq \frac{\mu(A)}{C}.$$

Consequently, a certified positive mass under μ transfers to ν whenever the density-ratio bound is proved.

Proof. By the Radon–Nikodym representation, $\mu(A) = \int_A (d\mu/d\nu)\, d\nu \le C\nu(A)$. □

Proposition 3.13 (Conditioning and rejection)**.** *Let $A \subseteq B$ with $\mu(B) > 0$. Under the conditioned generator $\mu(\cdot \mid B)$,*

$$\mu(A \mid B) = \frac{\mu(A)}{\mu(B)}.$$

Thus rejection sampling preserves positivity whenever the retained event has positive mass and the accepted subset already has positive mass.

3.6 Certificate combinators

The term "calculus" refers to a small algebra of certificate-preserving transformations. The following rules are elementary consequences of measure theory, but making them explicit permits a checker to assemble large certificates from smaller ones without trusting the search procedure.

Proposition 3.14 (Tagged disjunction)**.** *Let $A_1 \subseteq W_1$ and $A_2 \subseteq W_2$ have certified masses $\mu_1(A_1) \ge \delta_1$ and $\mu_2(A_2) \ge \delta_2$. On the tagged union $W_1 \sqcup W_2$, choose the mixture $\eta\mu_1 + (1-\eta)\mu_2$, where $0 \le \eta \le 1$, and accept the corresponding tagged accepted set. Its mass is at least*

$$\eta\delta_1 + (1-\eta)\delta_2.$$

In particular, any positive component proves the disjunctive existence claim.

Proof. The two tagged components are disjoint, so their accepted masses add under the mixture. □

Proposition 3.15 (Product conjunction)**.** *Suppose $\mu_j(A_j) \ge \delta_j$ for $j = 1, 2$. Under the explicit product measure $\mu_1 \otimes \mu_2$, the product accepted set $A_1 \times A_2$ has mass at least $\delta_1\delta_2$. Thus two existence certificates compose into a certificate for a witness pair satisfying both relations.*

Proof. By the defining property of the product measure, $(\mu_1 \otimes \mu_2)(A_1 \times A_2) = \mu_1(A_1)\mu_2(A_2) \ge \delta_1\delta_2$. □

Proposition 3.16 (Sequential composition)**.** *Let μ be a distribution on W_1, let $A \subseteq W_1$ satisfy $\mu(A) \ge \delta$, and let $K(w_1, \cdot)$ be a probability kernel on W_2. If for every $w_1 \in A$ a measurable accepted set $B(w_1) \subseteq W_2$ satisfies*

$$K(w_1, B(w_1)) \ge \varepsilon,$$

then the joint generator $w_1 \sim \mu$, $w_2 \sim K(w_1, \cdot)$ assigns mass at least $\delta\varepsilon$ to

$$\{(w_1, w_2) : w_1 \in A,\ w_2 \in B(w_1)\}.$$

Proof. Integrating the conditional lower bound over A gives

$$\int_A K(w_1, B(w_1))\, d\mu(w_1) \ge \varepsilon\mu(A) \ge \delta\varepsilon.$$

□

Proposition 3.17 (Sound pushforward)**.** *Let $f : W \to W'$ be measurable. If acceptance of w implies acceptance and soundness of $f(w)$, then the pushforward distribution $f_*\mu$ assigns the target accepted set at least the source accepted mass. Deterministic witness transformations therefore preserve a certified lower bound.*

These rules separate two notions that are frequently conflated. Product and sequential composition use explicitly defined joint measures; they do not infer independence from data. A checker should reject a product lower bound unless the product construction or an equivalent dependence proof is part of the certificate.

Derivation-level soundness. Let certificate derivations be generated from atomic mass certificates by tagged disjunction, product, sequential bind, conditioning, change of measure, pushforward, finite cover, and well-founded reduction. Soundness of the full calculus follows by structural induction: each rule maps valid premises to a valid positive-mass conclusion, and the terminal SWC-Exists rule converts positive accepted mass into ordinary existence. This gives a clean implementation strategy: the trusted kernel checks a small derivation tree rather than re-running the untrusted search.

3.7 Witness extraction

Theorem 3.18 (Repeated-sampling extraction)**.** *Suppose independent samples from μ_i are computable and accepted with probability at least $\delta_i > 0$. Then after k trials,*

$$\mathbb{P}(\textit{no accepted witness}) \le (1-\delta_i)^k,$$

and the expected number of trials to the first accepted witness is at most $1/\delta_i$.

Existence may be mathematically useful even when δ_i is too small for practical extraction. SWC records both logical success and algorithmic cost.

3.8 Classical versus constructive extraction

The logical status of a mass certificate depends on the ambient foundations and on how the measure is represented.

Proposition 3.19 (Finite atomic extraction)**.** *Let $W = (w_1, \dots, w_m)$ be a finite list, let V be decidable, and assign exact nonnegative integer weights a_j with $\sum_j a_j > 0$. If*

$$\sum_{j:V(w_j)=1} a_j > 0,$$

then a linear scan computes an index j with $a_j > 0$ and $V(w_j) = 1$.

Proof. If every accepted entry had weight zero, the accepted-weight sum would be zero. Decidability of V and the finite list permit exhaustive search for the first accepted positive-weight entry. □

Thus the finite rational kernel used in our experiments is constructive: positivity both proves existence and yields an explicit witness. In a general uncountable measure space, the theorem "positive measure implies nonempty" remains an ordinary classical existence argument, but it need not provide an algorithm for locating a point. A formal implementation should therefore record two independent fields: theorem status and extraction status. This distinction prevents a nonconstructive H3 certificate from being advertised as an efficient H4 witness generator.

For countably supported generators, effective extraction additionally requires a computable enumeration and a certified procedure that eventually exposes enough accepted mass. Merely knowing an abstract real number $\mu(A) > 0$ need not reveal where its mass is located.

3.9 Expectation certificates

The usual first-moment method fits the same interface.

Proposition 3.20 (Expectation-to-witness rule)**.** *Let $F : \mathcal{W}_i \to \mathbb{R}$ be measurable and integrable. If $\mathbb{E}_{\mu_i}[F] \geq \tau$, then there exists w with $F(w) \geq \tau$.*

Proof. If $F(w) < \tau$ for every w, then the measurable function $\tau - F$ is strictly positive everywhere. Since $\mathcal{W}_i = \bigcup_{k\geq 1}\{w : \tau - F(w) \geq 1/k\}$ and $\mu_i(\mathcal{W}_i) = 1$, at least one set in the union has positive measure. Hence $\mathbb{E}[\tau - F] > 0$, contradicting $\mathbb{E}[F] \geq \tau$. In a finite space this reduces to the familiar fact that at least one value is no smaller than the average. □

A proof compiler can reduce this rule to nonemptiness of $\{w : F(w) \geq \tau\}$, sometimes with an explicit mass lower bound when F is bounded.

Theorem 3.21 (Paley–Zygmund backend)**.** *Let $X \geq 0$ be square-integrable with $\mathbb{E}[X] > 0$. For $0 \leq \theta < 1$,*

$$\mathbb{P}(X \geq \theta\mathbb{E}[X]) \geq (1-\theta)^2\frac{\mathbb{E}[X]^2}{\mathbb{E}[X^2]}.$$

In particular,

$$\mathbb{P}(X > 0) \geq \frac{\mathbb{E}[X]^2}{\mathbb{E}[X^2]} > 0.$$

Proof. Write $A = \{X \geq \theta\mathbb{E}[X]\}$. Then

$$\mathbb{E}[X] \leq \theta\mathbb{E}[X] + \mathbb{E}[X\mathbf{1}_A],$$

so $(1-\theta)\mathbb{E}[X] \leq \mathbb{E}[X\mathbf{1}_A]$. Cauchy–Schwarz gives $\mathbb{E}[X\mathbf{1}_A]^2 \leq \mathbb{E}[X^2]\mathbb{P}(A)$, yielding the bound. □

This rule turns second-moment calculations into explicit positive-mass certificates and is often stronger than a bare expectation argument.

Proposition 3.22 (Union-bound backend)**.** *Let $B_1, \ldots, B_m$ be measurable bad events. If certified upper bounds satisfy*

$$\sum_{j=1}^{m} \mu(B_j) \leq 1 - \varepsilon \qquad \text{for some } \varepsilon > 0,$$

then the good event $G = \bigcap_j B_j^c$ has $\mu(G) \geq \varepsilon$ and therefore contains a witness.

Proof. The union bound gives $\mu(\bigcup_j B_j) \leq \sum_j \mu(B_j) \leq 1 - \varepsilon$, so $\mu(G) = 1 - \mu(\bigcup_j B_j) \geq \varepsilon > 0$. □

Example 3.23 (Ramsey lower-bound certificate)**.** Choose a uniformly random labeled graph on n vertices. For each k-vertex set S, let B_S be the event that S induces either a clique or an independent set. Since all $\binom{k}{2}$ edges are independent,

$$\mathbb{P}(B_S) = 2^{1-\binom{k}{2}}.$$

Consequently,

$$\mathbb{P}\left(\bigcap_S B_S^c\right) \geq 1 - \binom{n}{k}2^{1-\binom{k}{2}}.$$

Whenever the right side is positive, Theorem 3.22 compiles the mass bound into a deterministic existence theorem for a graph with neither a k-clique nor a k-vertex independent set, hence $R(k,k) > n$. In SWC terms, the graph distribution, the forbidden-subgraph verifier, the exact union-bound arithmetic, and the positive remainder are separate checkable fields.

Local-lemma backend. The symmetric Lovász local lemma is another native mass compiler: if bad events each have probability at most p, each depends on at most d others, and $ep(d+1) \leq 1$, then the probability of avoiding all bad events is positive [1, 10]. A proof-carrying implementation need only certify the event probabilities, the dependency graph, and the inequality. Existing formalizations of the local lemma make this a realistic proof-assistant backend [8].

3.10 Dual certified refutation

Definition 3.24 (Obstruction verifier)**.** An obstruction verifier $U(i, c)$ is sound if

$$U(i, c) = 1 \Rightarrow \forall w \in \mathcal{W}_i \; \neg R(i, w).$$

Theorem 3.25 (Stochastic obstruction rule)**.** *If a probability distribution over pairs (i, c) assigns positive mass to certificates accepted by a sound obstruction verifier, then there exists a counterexample instance i.*

Proof. Positive accepted mass implies at least one accepted pair (i, c). Soundness of U proves that this i has no valid witness. □

Sampling many failed witnesses is not a refutation. The accepted object must certify nonexistence.

3.11 Continuous witness spaces

For continuous distributions, an isolated exact witness often has measure zero. Positive mass is sufficient but not necessary. One remedy uses nested approximate certificates.

Theorem 3.26 (Nested approximation rule)**.** *Let K be compact, let $F : K \to \mathbb{R}_{\geq 0}$ be continuous, and let*

$$A_k = \{w \in K : F(w) \leq 2^{-k}\}.$$

If each A_k is nonempty, then there exists $w^\star \in K$ with $F(w^\star) = 0$.

Proof. The sets A_k are nonempty, closed, and nested. Compactness gives a point in their intersection. Continuity and $F(w) \leq 2^{-k}$ for every k imply $F(w) = 0$. □

An SWC proof may establish nonemptiness of each A_k through positive mass, while interval arithmetic and compactness discharge the limiting step.

4 Exactness Levels and Complexity Boundaries

4.1 Certificate-status lattice

The word "probabilistic" can describe several epistemically different objects. SWC uses explicit status labels:

Level	Object	Mathematical consequence
H0	Heuristic or Monte Carlo success estimate	Evidence only
H1	PAC or confidence lower bound with failure probability $\beta > 0$	Bounded-error claim
H2	Exact mass theorem conditional on an unproved arithmetic or physical hypothesis	Conditional theorem
H3	Exact proof-carrying lower bound $\mu(A) \geq \delta > 0$	Ordinary existence theorem
H4	Extracted witness accepted by a deterministic verifier	Constructive certificate

Only H3 and H4 have unconditional theorem status relative to the ambient axioms. A checker must preserve these labels rather than silently promoting H0–H2.

4.2 Full-support equivalence and no free lunch

Proposition 4.1 (Finite full-support equivalence)**.** *Let W be finite and let $\mu(w) > 0$ for every $w \in W$. For any accepted set $A \subseteq W$,*

$$\mu(A) > 0 \quad \Longleftrightarrow \quad A \neq \varnothing.$$

Thus positive mass is a different proof interface, not an automatic reduction in logical or computational difficulty.

Proposition 4.2 (Complexity boundary)**.** *Let $W = \{0,1\}^n$, let μ be uniform, and let V_φ verify whether an assignment satisfies a CNF formula φ. Deciding whether $\mu(V_\varphi^{-1}(1)) > 0$ is exactly SAT, while computing the exact numerator is #SAT and is #P-complete in general [25].*

The value of SWC therefore comes from structured mass proofs, reusable certificate rules, and independent checking—not from bypassing worst-case complexity. A short certificate may exist when a direct witness is difficult to discover, but no theorem in this paper implies that such certificates are always short.

5 Trusted Kernel and Mass-Certificate Back Ends

For finite witness spaces, the trusted computation can be particularly small. Assign each witness w_j a nonnegative integer weight a_j, not all zero. The exact accepted mass is

$$\delta = \frac{\sum_{j:V(i,w_j)=1} a_j}{\sum_j a_j}.$$

The numerator is positive if and only if at least one positive-weight witness is accepted.

Listing 1: Reference finite-mass certificate checker.

```
check(instance, witnesses, integer_weights, verifier):
    assert len(witnesses) == len(integer_weights)
    assert all(weight >= 0)
    assert sum(integer_weights) > 0

    accepted_weight = 0
    first_witness = None
    for w, weight in zip(witnesses, integer_weights):
        if verifier(instance, w):
```

```
            accepted_weight += weight
            if weight > 0 and first_witness is None:
                first_witness = w

    delta = accepted_weight / sum(integer_weights) # exact rational
    return delta, first_witness
```

This enumerator is a reference semantics, not a scalability claim. In serious applications, π_{mass} should be supplied by symbolic counting, concentration, local lemmas, couplings, algebraic identities, proof-producing model counters, or a proof assistant.

5.1 Proof objects and kernel judgments

A portable derivation can be represented by the grammar

$$\begin{aligned}\pi ::= {} & \mathsf{atom}(C) \mid \mathsf{mix}(\eta, \pi_1, \pi_2) \mid \mathsf{product}(\pi_1, \pi_2) \mid \mathsf{bind}(\pi_1, K, \pi_2) \\ & \mid \mathsf{condition}(B, \pi) \mid \mathsf{change}(C, \pi) \mid \mathsf{push}(f, \pi) \mid \mathsf{cover}(\pi_1, \ldots, \pi_m) \mid \mathsf{reduce}(r, T, \pi).\end{aligned}$$

The kernel judgment

$$\Gamma \vdash \pi : (i, R, \delta)$$

means that, under assumptions Γ, the derivation certifies a sound accepted set for instance i with mass at least δ. The terminal rule may emit an existence theorem only after checking $\delta > 0$. This syntax makes dependency, conditioning, and quantifier scope explicit in the proof object rather than implicit in prose.

Theorem 5.1 (Kernel soundness by structural induction)**.** *If every atomic certificate is sound and each derivation constructor is checked against its corresponding rule in this paper, then every accepted derivation $\Gamma \vdash \pi : (i, R, \delta)$ with $\delta > 0$ proves $\exists w\, R(i, w)$ in the ambient logic.*

Proof. Induct on the syntax of π. The atomic case is Theorem 3.3. Each constructor preserves a valid lower bound by the associated mixture, product, sequential, conditioning, change-of-measure, pushforward, cover, or reduction proposition. The terminal positivity check and verifier soundness yield ordinary existence. □

5.2 Trusted computing base

The intended exact kernel contains only:

1. a parser for the instance, distribution, verifier, and derivation tree;
2. exact arithmetic or proof-term checking for normalization and mass bounds;
3. a deterministic verifier soundness theorem;
4. implementations of the small certificate rules; and
5. a quantifier-scope checker.

Language models, neural proposal policies, Monte Carlo estimators, SAT solvers, compilers, and theorem-search agents remain untrusted unless their outputs carry independently checkable certificates.

5.3 Proof-producing counting back ends

For Boolean witnesses, an accepted mass is a weighted model count. Certified knowledge compilation can translate a CNF into a tractable graph together with a proof of equivalence; a small verified checker then computes the exact weighted count [4]. This is a direct large-scale backend for H3 certificates. Approximate counting with formally verified PAC guarantees [22] is valuable for search and bounded-error evidence, but by itself remains H1: a nonzero failure budget cannot be silently compiled into theoremhood.

Proposition 5.2 (Certified interval positivity)**.** *Suppose a sound interval checker proves*

$$0 < L \le \mu(A) \le U$$

with exact rational endpoints L, U. Then L is an H3 mass certificate and A is nonempty. A floating-point estimate $\widehat{\mu}(A) > 0$ without a proved enclosure is not an H3 certificate.

This permits scalable numerical back ends provided every rounding and truncation error is enclosed with directed arithmetic or a proof term. It also makes a sharp interface distinction: floating point may guide search, but only an exact positive lower endpoint may cross the trusted boundary.

5.4 Threat model

The kernel rejects certificates with floating-point-only positivity, unproved normalization, hidden conditioning, unverifiable randomness, circular references, mismatched verifier versions, inferred independence, or a universal conclusion supported only by sampled instances. Cryptographic hashes and reproducible builds protect provenance, but they do not replace mathematical soundness.

6 AI-Assisted Stochastic Proof Search

6.1 Architecture

Let $G_\theta(w \mid i)$ be a learned witness generator. SWC separates four roles:

1. **Generator:** proposes witnesses and structured distributions.
2. **Verifier:** deterministically checks candidate validity.
3. **Bound prover:** derives a symbolic lower bound δ_i on accepted mass.
4. **Adversarial auditor:** searches for zero-mass instances, hidden independence assumptions, quantifier swaps, and verifier mismatches.

A useful training objective is not merely empirical solve rate, but certifiable mass:

$$\max_\theta \ \mathbb{E}_{i\sim\mathcal{D}}\big[\log \underline{\delta}_\theta(i)\big],$$

where $\underline{\delta}_\theta(i)$ is a formally justified lower bound rather than a Monte Carlo estimate. The support mixture in Theorem 3.11 allows aggressive learning without deleting the baseline proof route.

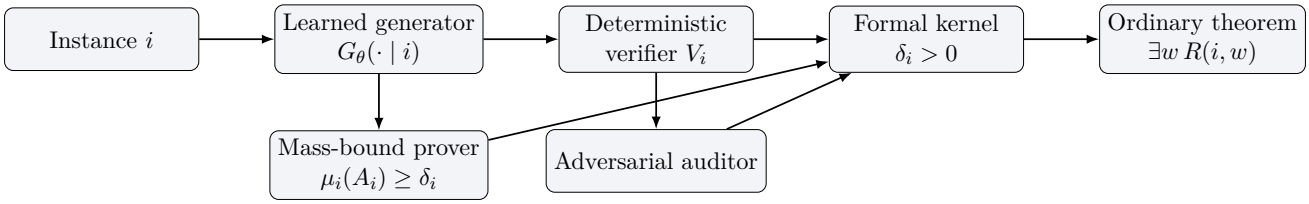

Figure 1: The generator is untrusted. Soundness comes from a deterministic verifier and a checked positive-mass certificate.

6.2 Failure modes

The framework explicitly rejects:

- estimating δ_i from samples and treating the estimate as exact;
- assuming independence because empirical correlations are small;
- proving $\mathbb{E}_i[\delta_i] > 0$ and concluding $\delta_i > 0$ for every i;
- accepting a probabilistic verifier with nonzero false-acceptance risk as an exact theorem checker;
- defining a distribution whose normalization or measurability is not proved;
- moving from a continuous density around an approximate solution to existence of an exact point without a limiting theorem.

7 Empirical Reality Mode

Mathematical truth is relative to axioms and formal semantics. Claims about physical systems additionally depend on assumptions about measurement, intervention, stationarity, confounding, and instrument integrity. No statistical method converts these assumptions into assumption-free certainty.

We define an empirical certificate

$$\Pi_{\text{emp}} = (\mathcal{A}, \mathcal{P}, \mathcal{D}, E_t, \alpha, \pi_{\text{cal}}),$$

where $\mathcal{A}$ is the declared model class, $\mathcal{P}$ a preregistered protocol, $\mathcal{D}$ a signed data record, E_t an evidence process, and π_{cal} a proof of risk calibration.

Theorem 7.1 (Anytime-valid certification)**.** *Suppose that under every distribution P in a null model $\mathcal{H}_0$, (E_t) is a nonnegative supermartingale with $E_0 \leq 1$. Then*

$$\sup_{P \in \mathcal{H}_0} \mathbb{P}_P\left(\sup_{t \geq 0} E_t \geq \frac{1}{\alpha}\right) \leq \alpha.$$

This is Ville's inequality and is central to e-process methodology [19]. The certificate proves a statement about the protocol's false-certification frequency under $\mathcal{A}$, even under optional stopping. It does not prove an assumption-free proposition about nature.

Physical analogue of proof strength. The strongest honest physical endpoint is therefore:

$$\text{formal protocol validity} + \text{auditable assumptions} + \text{bounded risk} + \text{independent replication}.$$

This is not logically identical to a mathematical theorem, but it is a rigorous, machine-checkable empirical guarantee.

8 Experiments

All experiments use fixed seeds and are included in the accompanying source archive. Exact mathematical experiments recompute accepted mass from integer weights or certified recurrences; sampling is used only to train proposals or illustrate what sampling cannot prove. We organize the evaluation around six questions:

1. Can an untrusted learned generator increase exact witness mass while a baseline mixture preserves support?
2. Can a non-enumerative checker certify mass over exponentially large witness spaces?
3. Do explicit stress tests catch average-to-pointwise and zero-hit-to-zero-mass errors?
4. Can the interface audit a model-led investigation of an open mathematical problem without promoting finite evidence to a theorem?
5. Does the empirical mode preserve its declared risk under optional stopping?
6. Can a small proof-carrying checker accept exact certificates while rejecting semantic tampering and floating-point-only claims?

8.1 Experiment 1: learned witness distributions for SAT

Setup. We generated 60 planted 3-CNF instances with 16 variables and 112 clauses, plus 20 deliberately unsatisfiable controls formed by adding contradictory unit clauses. The witness space contains all $2^{16} = 65{,}536$ assignments. A deterministic verifier evaluates every clause.

The baseline generator is uniform. A cross-entropy method trains an independent Bernoulli proposal from clause-satisfaction scores. Final probabilities are quantized to multiples of $1/256$, allowing exact accepted mass to be computed as an integer numerator over the common denominator 256^{16}. To protect support, we also evaluate

$$\mu_{\text{mix}} = 0.1\mu_{\text{uniform}} + 0.9\mu_{\text{learned}}.$$

Results. All 60 planted instances had exactly certified positive mass; all 20 controls had exactly zero accepted mass under exhaustive enumeration. The median uniform witness mass was $1/65{,}536 \approx 1.526 \times 10^{-5}$. The learned proposal's median exact mass was 0.9393, and the support-preserving mixture's median was 0.8454. Relative to uniform, the mixture produced a median mass increase of 27,809.8× and a geometric-mean increase of 4,346.3×. Its worst-case ratio was exactly 0.1, matching Theorem 3.11.

To test literal support collapse, we also hardened each learned product proposal to a point mass at its coordinate-wise mode. The resulting deterministic proposal landed on a satisfying assignment for 50 of the 60 satisfiable instances and assigned *exactly zero* accepted mass on the remaining 10. Mixing this hard proposal with 10% uniform mass restored certified positive mass on all 60 instances; the minimum restored mass was 1.5259×10^{-6}. The soft learned proposal itself retained formal full support because

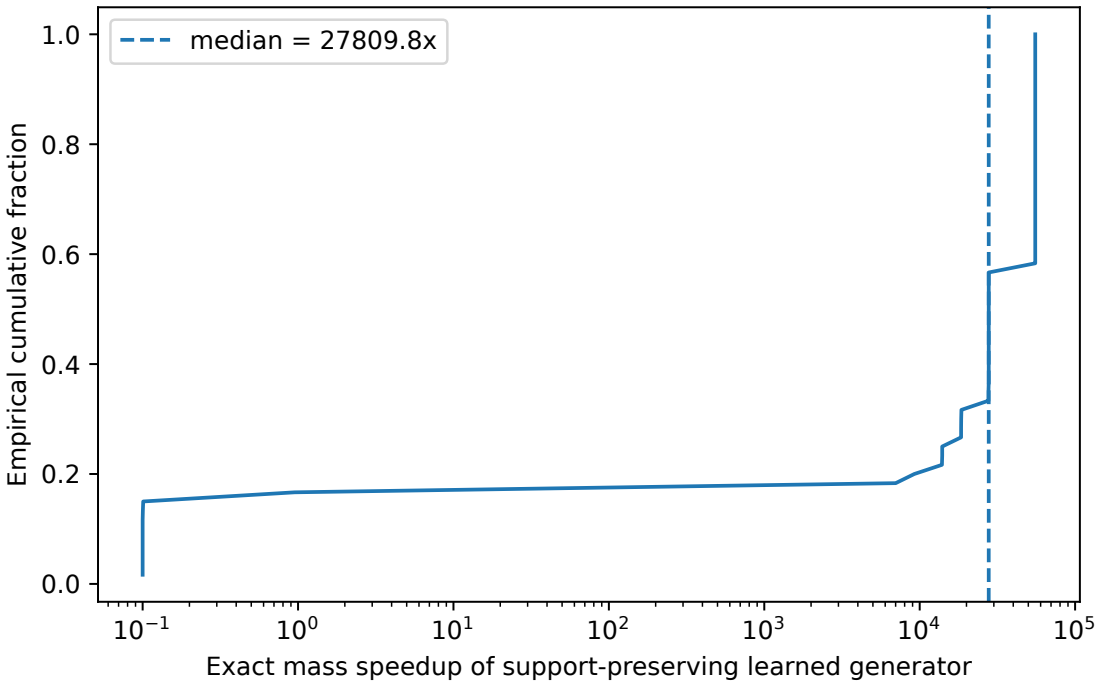

Figure 2: Exact accepted-mass speedup for the support-preserving learned SAT generator. Learning often concentrates almost all mass on satisfying assignments, while the 10% uniform component enforces a certified floor.

its rational Bernoulli parameters were clipped away from zero and one, but it performed worse than uniform on 10 instances. The ablation therefore separates concentration from support: optimization can improve typical extraction while a theorem-level baseline component protects rare witnesses against model hardening or truncation.

This experiment does not show that learning makes proof easier in every case. It shows that an untrusted optimizer can greatly improve extraction while a simple mixture theorem preserves the logical route.

8.2 Experiment 2: a non-enumerative dynamic-programming certificate

Witness problem. For the path graph P_n, a bit vector encodes an independent set. We accept exactly those vectors with $k = \lfloor n/4 \rfloor$ selected vertices and no adjacent selected pair. The witness space has size 2^n, while the number of accepted witnesses is

$$\binom{n-k+1}{k}.$$

Under a product Bernoulli generator with integer one-weight w and common denominator Q, every accepted string has weight $w^k(Q-w)^{n-k}/Q^n$.

Proof-carrying recurrence. The checker recomputes exact integer numerators using

$$A_{i,j,0} = (A_{i-1,j,0} + A_{i-1,j,1})(Q-w), \qquad A_{i,j,1} = A_{i-1,j-1,0}w,$$

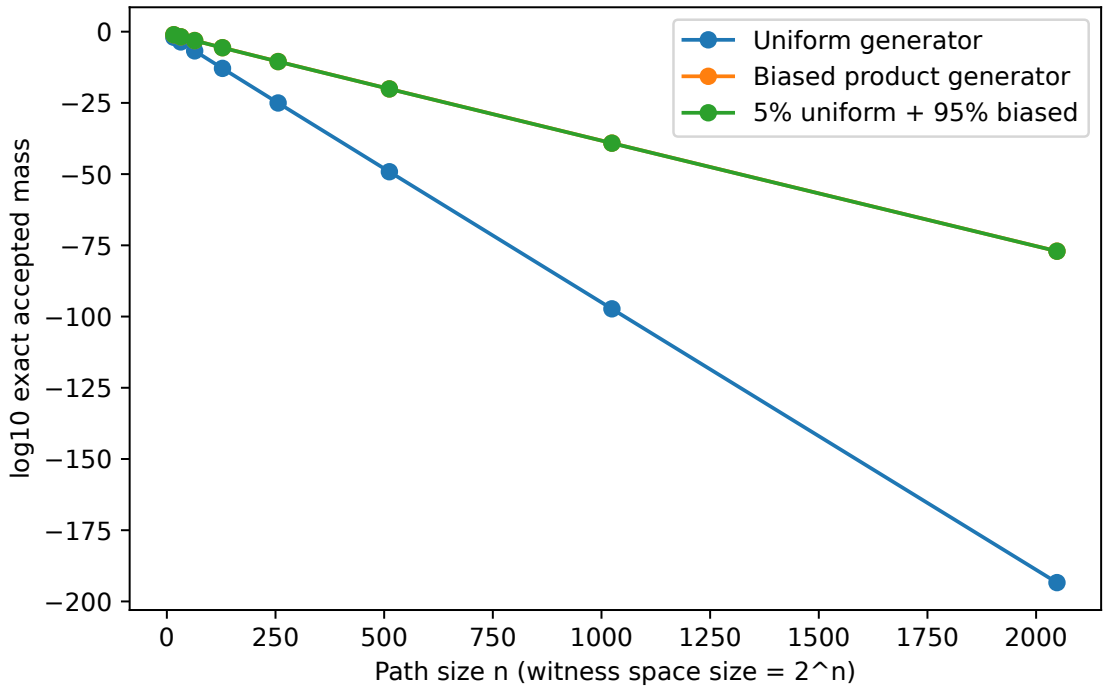

Figure 3: Exact accepted mass for path independent-set witnesses. The checker handles witness spaces of size 2^n without enumeration; the support-preserving proposal changes extraction scale while exact recurrence checking remains unchanged.

with $A_{0,0,0} = 1$. The final numerator is $A_{n,k,0} + A_{n,k,1}$. This dynamic program requires $O(nk)$ exact updates rather than enumerating 2^n vectors. We independently cross-check it against the closed form for every tested size.

Results. For $n = 16, 32, \ldots, 2048$, every recurrence/closed-form comparison matched exactly. At $n = 2048$ the witness space contains 2^{2048} strings, the common denominator has 4,933 decimal digits, and the checker completed in 1.04 seconds for uniform weights and 1.10 seconds for a product proposal with $\mathbb{P}(X_i = 1) = 1/4$. The uniform accepted mass was approximately $10^{-193.40}$, while the biased proposal achieved $10^{-77.05}$. A trajectory-level mixture with 5% uniform mass preserved a certified baseline floor and yielded a mass ratio of 2.12×10^{116} relative to uniform.

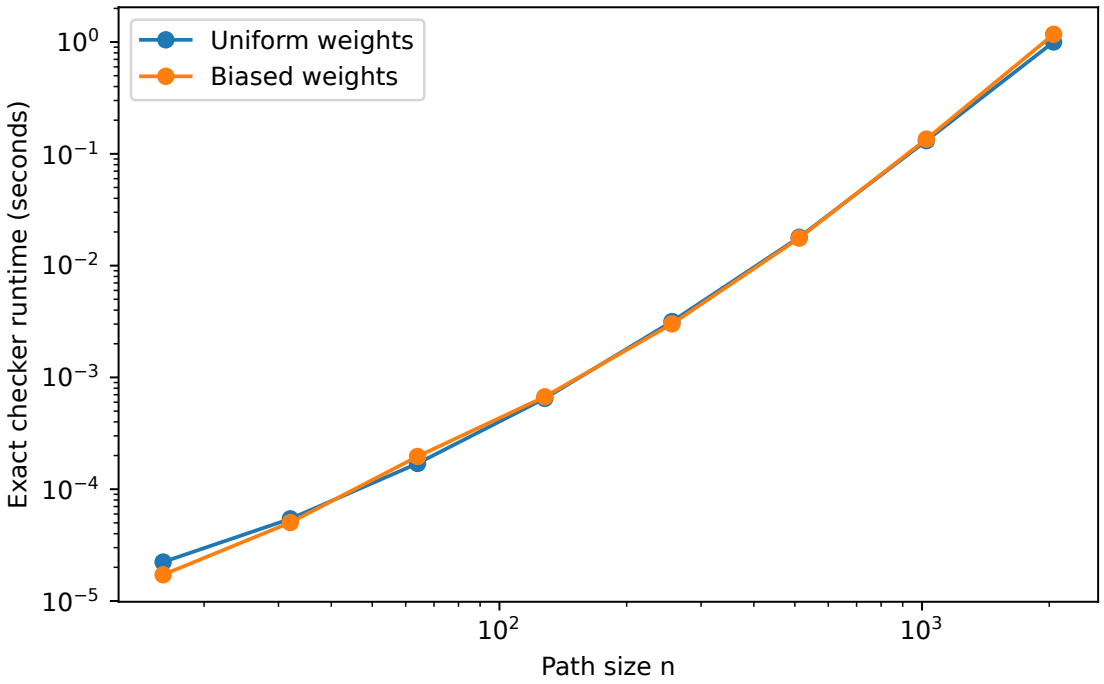

Figure 4: Runtime of the exact dynamic-programming checker. This is a structured benchmark rather than a general #SAT solver, but it demonstrates that mass certificates need not enumerate the witness space.

8.3 Experiment 3: adversarial quantifier auditing

This experiment targets two invalid inferences that repeatedly appear in open-problem reasoning.

Hidden exceptional instance. Consider a finite family of N instances in which every instance except one has positive witness mass. A random benchmark drawing k instances with replacement misses the unique zero-mass exception with exact probability

$$\left(1 - \frac{1}{N}\right)^{k}.$$

For $N = 10^6$, $k = 10^4$ random checks miss the exception with probability 0.99005; even $k = 10^5$ checks miss it with probability 0.90484. Seeded simulations with 20,000 repetitions per cell agreed with the exact curves to maximum absolute error 0.0085.

Zero observed hits. If a witness generator has true positive mass $\delta > 0$, the probability of observing no accepted witness in k independent samples is $(1-\delta)^k$. For $\delta = 10^{-6}$ and $k = 10^5$, zero hits occur with probability 0.90484; for $\delta = 10^{-8}$ and $k = 10^6$, the probability is 0.99005. Thus finite failure-to-sample cannot establish zero mass.

A finite pointwise checker catches the planted exception exactly, but this is only a toy illustration. For infinite families, the required object is a uniform symbolic family certificate, a cover, or a well-founded reduction—not an arbitrarily large benchmark.

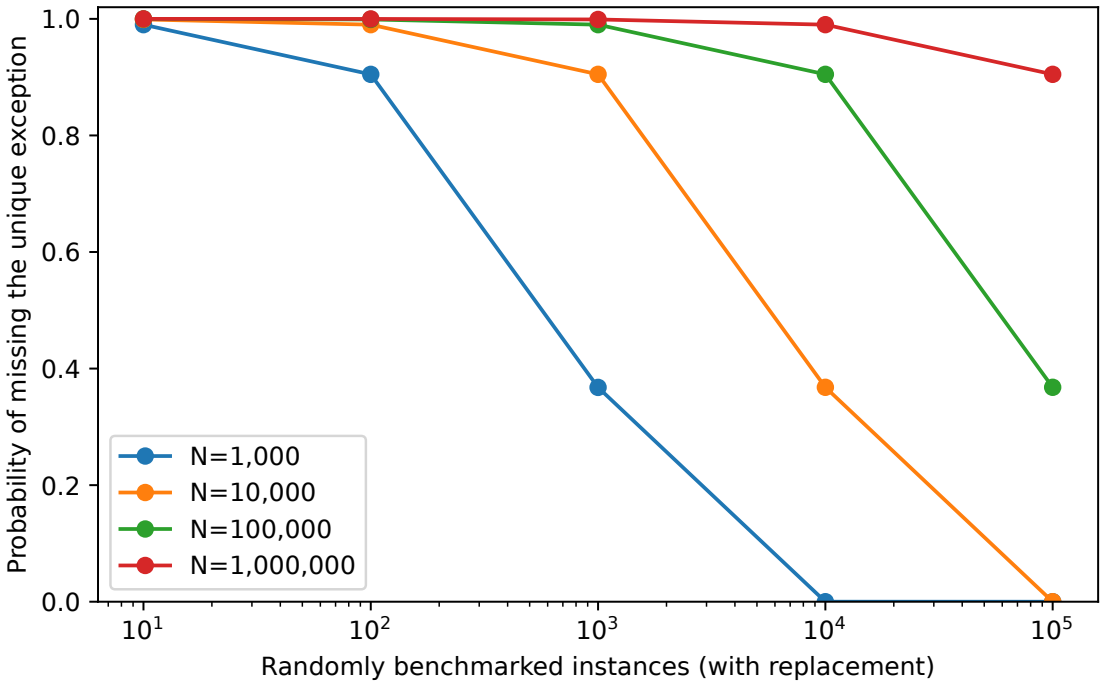

Figure 5: A high benchmark success rate cannot certify a universal theorem. When one exceptional instance is hidden among N, random testing can miss it with high probability even for large test budgets.

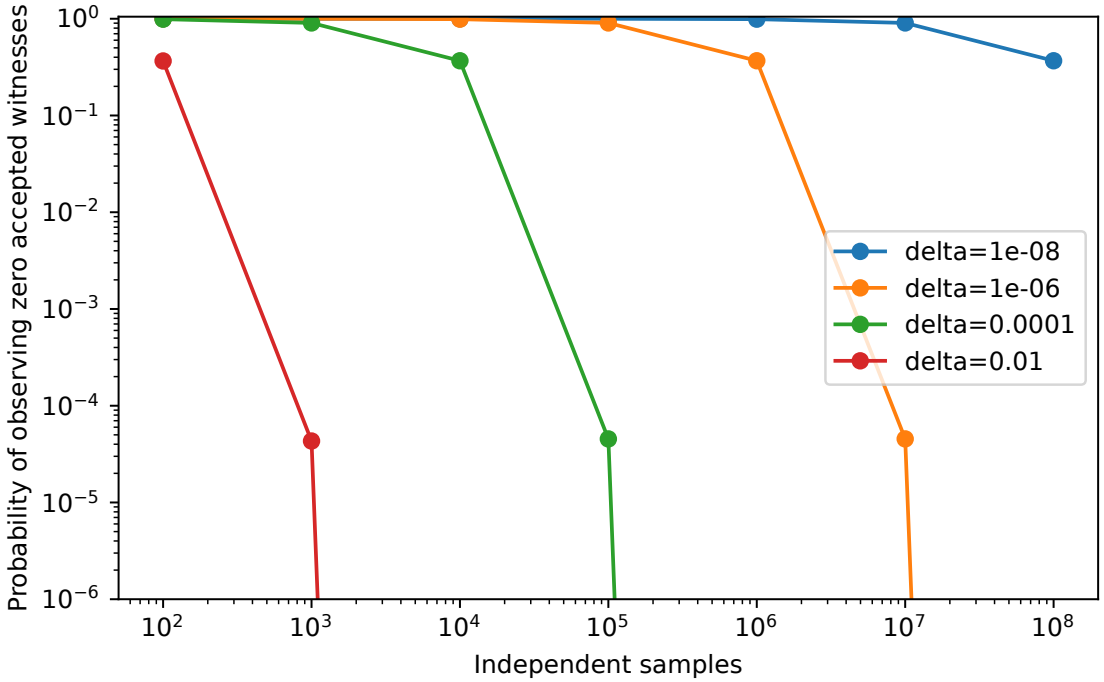

Figure 6: Zero sampled witnesses is compatible with strictly positive mass. Exact or symbolic lower bounds are required to distinguish an empty accepted set from a rare one.

8.4 Experiment 4: finite Erdős–Straus certificates

The Erdős–Straus conjecture asks whether

$$\frac{4}{n} = \frac{1}{x} + \frac{1}{y} + \frac{1}{z}$$

has positive integer solutions for every $n \geq 2$. It remains open [6, 9]. For primes $p \equiv 1 \pmod 4$, the anchor-shift framework sets

$$a_c = \frac{p+c}{4}, \qquad c \equiv 3 \pmod 4,$$

and the exact fixed-shift divisor criterion accepts when some $d \mid a_c^2$ satisfies

$$d \equiv -a_c \pmod c \qquad \text{or} \qquad 4d \equiv -1 \pmod c.$$

An accepted pair reconstructs explicit denominators and is checked by integer cross multiplication. A self-contained derivation of the criterion and of the bound $c \leq 2p$ appears in Section A.

Distribution. For each prime $p \equiv 1 \pmod{24}$ below 10^7, we use the 32 shifts

$$c \in \{3, 7, 11, \ldots, 127\}.$$

The uniform generator chooses a shift uniformly and then a divisor of a_c^2 uniformly. We also test a support-preserving small-shift prior with integer shift weight $\lfloor 128/c \rfloor$.

Results. There are 82,887 tested primes. Exact enumeration found positive witness mass for every one, and the first reconstructed identity for every prime passed exact verification. The largest least successful shift was $c = 107$, attained at $p = 8{,}803{,}369$. The median uniform accepted mass was 0.048398 (about 20.66 expected independent draws); the small-shift prior increased the median to 0.176253 (about 5.67 expected draws). The minimum uniform mass was $1/3600$ at the same hardest prime.

This experiment is an exact proof for the stated finite range and witness space. It is not evidence that the same 32 shifts suffice universally, and it is not a proof of the conjecture. Salez verified the conjecture through 10^{17}, later work reports 10^{18}, and Dahan proves strong depth-dependent exceptional-set bounds while explicitly leaving the universal conjecture open [6, 14, 20].

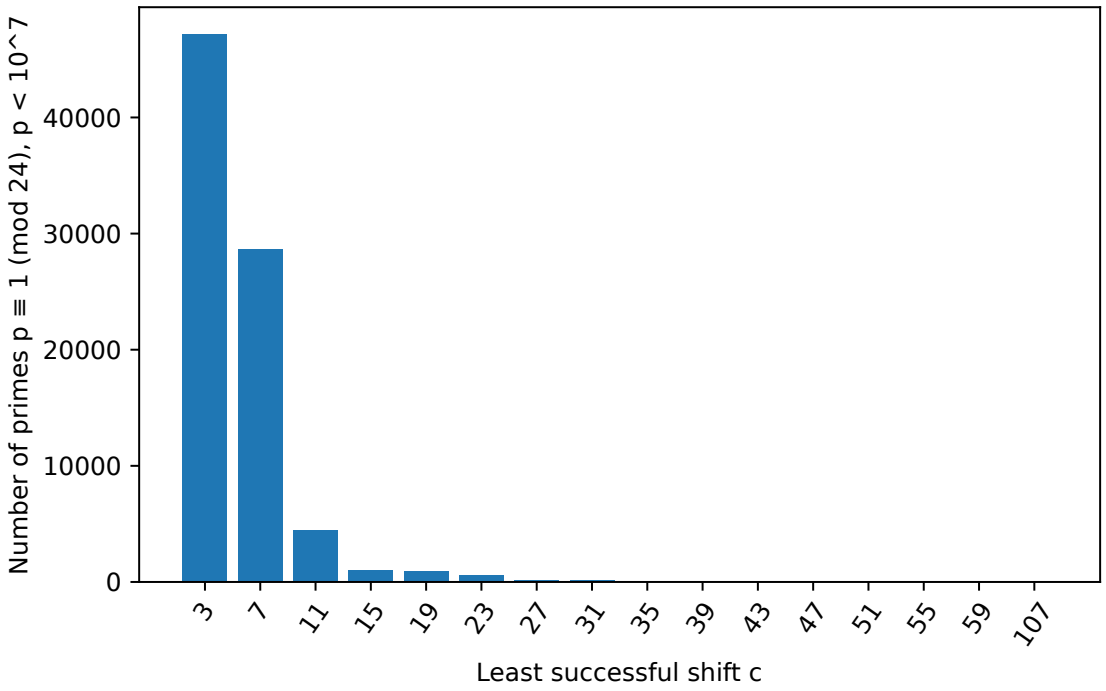

Figure 7: Least successful shift for all primes $p \equiv 1 \pmod{24}$ below 10^7. The isolated $c = 107$ case is visible at the extreme right.

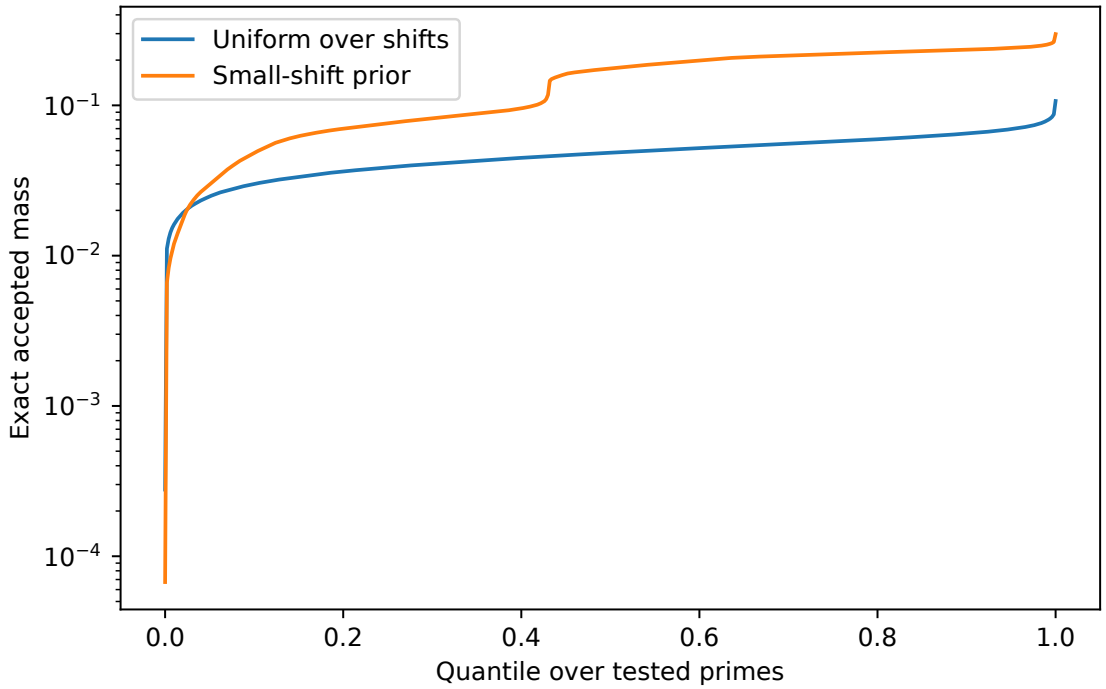

Figure 8: Exact finite accepted-mass distributions. A small-shift prior improves typical extraction but slightly reduces mass on the rare hardest cases; full support preserves positivity.

8.5 Arbiter v23 contribution to the case study

Arbiter v23 is an internal looped 27B-parameter reasoning model used in the preceding project that motivated this paper. According to the audited project logs, Arbiter:

- derived the bounded multiplicative-box reformulation of the anchor shift;
- tightened a provisional shift bound to $c \leq 2p$ using the least denominator;
- generated 122 machine-verified residue families during abductive search;
- produced proof drafts for two central structural theorems;
- proposed many false families that exact tools rejected before inclusion.

The current artifact independently reimplements the finite divisor criterion and the $p < 10^7$ survey, but it does not reproduce all 122 family-generation runs. Arbiter is treated as a discovery system, not as an author or trusted verifier. Its relevance to SWC is architectural: it proposes witness distributions and structural hypotheses, while exact tools determine what survives.

8.6 Experiment 5: optional stopping in empirical mode

Setup. We simulated 50,000 Bernoulli paths of length 500 under the simple null $p_0 = 0.5$ and 50,000 under $p_1 = 0.6$. We compare:

1. a one-sided nominal 5% z-test repeated at every time $t \geq 20$ without multiplicity correction;
2. the likelihood-ratio e-process
$$E_t = \left(\frac{0.6}{0.5}\right)^{S_t} \left(\frac{0.4}{0.5}\right)^{t-S_t},$$
stopped when $E_t \geq 1/0.05 = 20$.

Under the simple null, (E_t) is a nonnegative martingale of mean one, so its crossing probability is at most 0.05.

Results. The repeated nominal test crossed on 30.826% of null paths (Wilson 95% half-width 0.405 percentage points), whereas the e-process crossed on 4.342% (half-width 0.179 points). At $p = 0.6$, the corresponding detection probabilities by time 500 were 99.946% and 97.738%; median stopping times conditional on crossing were 41 and 115 observations. Thus the anytime-valid method pays some detection delay but maintains its risk interpretation under continuous monitoring.

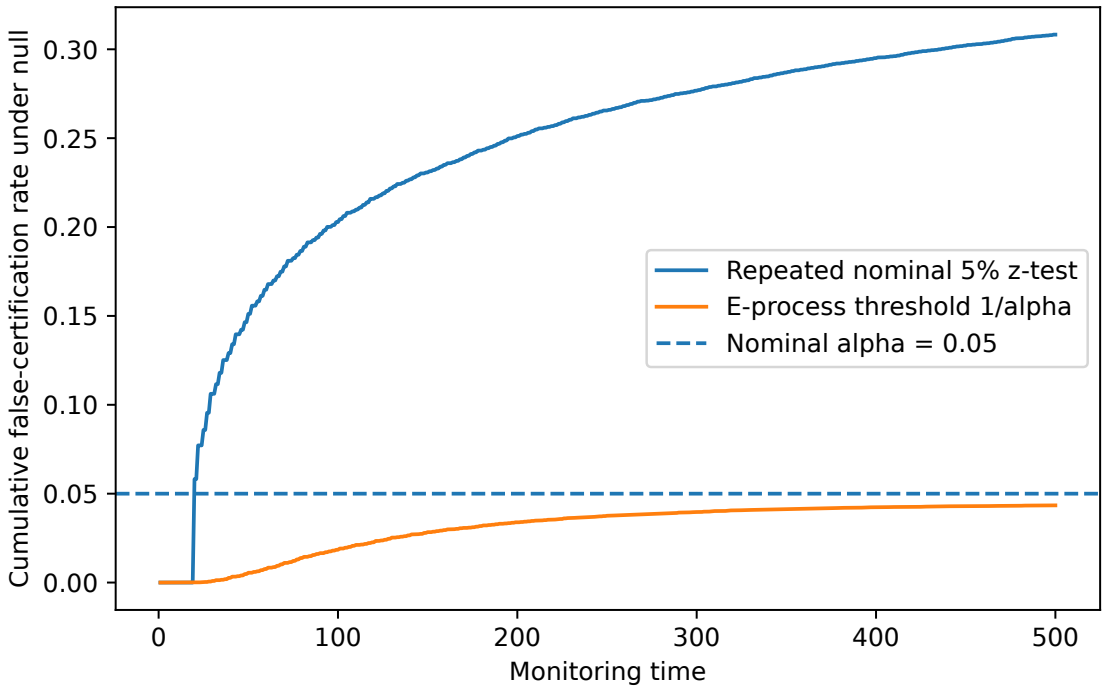

Figure 9: Optional-stopping behavior under the null. Repeated nominal testing accumulates false certifications; the e-process remains below the 5% time-uniform bound in the simulation.

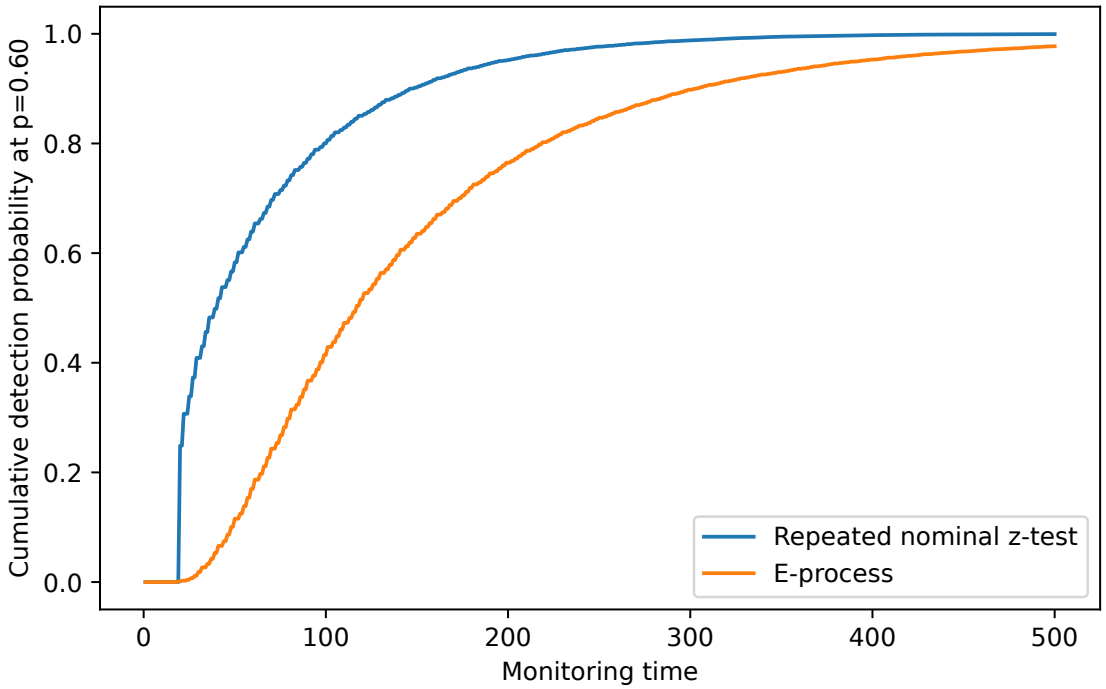

Figure 10: Detection probability under the alternative $p = 0.6$. The anytime-valid e-process is slower than repeated nominal testing but retains a valid interpretation under optional stopping.

8.7 Experiment 6: proof-carrying certificate integrity

Setup. We implemented a portable JSON certificate for the path-graph mass problem in Experiment 2. Each certificate binds the semantic claim (n, k, w, Q) to a canonical SHA-256 digest and carries the claimed exact accepted numerator, common denominator, reduced fraction, backend identifier, and positivity flag. The artifact includes a machine-readable schema and a complete example certificate. An independent checker validates the schema, recomputes the digest, reruns the exact $O(nk)$ recurrence, verifies normalization and fraction reduction, and refuses floating-point-only positivity.

We generated 300 valid certificates over path sizes $16 \le n \le 256$. For each certificate, an untrusted mutation harness produced six altered variants: an incremented accepted numerator, an incremented denominator, a changed target size, a changed distribution weight, an unsupported schema version, and a pseudo-certificate containing only a decimal mass estimate.

Results. The checker accepted all 300 valid certificates and rejected all 1,800 altered variants. Median checking time for valid certificates was 0.433 ms, and the 99th percentile across all checked objects was 4.06 ms on the reference machine. The experiment is not a cryptographic security proof and SHA-256 does not authenticate an author; it tests a narrower systems claim: semantic fields, exact arithmetic, and proof status are bound tightly enough that the specified corruptions cannot silently cross the kernel boundary.

Table 1: Proof-carrying integrity stress test.

Quantity	Result
Valid exact certificates	300 / 300 accepted
Tampered or float-only variants	1,800 / 1,800 rejected
Mutation classes per certificate	6
Median valid-certificate check	0.433 ms
99th percentile over all objects	4.06 ms
Exact arithmetic	Python integers and rational reduction

9 What the Experiments Establish

The experiments test different layers of the methodology rather than one aggregate claim. The SAT study shows that an untrusted learned proposal can increase exact witness mass by orders of magnitude while a theorem-level mixture floor protects support; the hard-mode ablation makes the protection concrete by turning 10 satisfiable instances from zero learned support back into positive certified mass. The path-graph study shows that exact mass certificates can be checked over exponentially large witness spaces using structure rather than enumeration. The quantifier audit demonstrates, with exact probabilities, why benchmark coverage and zero observed hits cannot substitute for pointwise mass proofs. The Erdős–Straus study shows how an open-problem investigation can produce exact finite certificates while the framework refuses a universal promotion. The sequential study validates the separate empirical endpoint: a theorem about time-uniform risk, not certainty about nature. The proof-carrying integrity study shows that the proposed trust boundary can be instantiated as a small recomputing checker that rejects altered semantics, altered mass claims, unsupported schemas, and floating-point-only evidence.

No experiment proves that short mass certificates exist for arbitrary theorems. Exact model counting is hard in general, learned generators can collapse, structured recurrences may be unavailable, open

problems may resist all known lower bounds, and empirical guarantees remain model-conditional. The experiments are therefore reference implementations and falsification tests for the interface, not evidence that SWC automatically resolves hard conjectures. The integrity experiment covers a declared mutation suite, not arbitrary implementation bugs, side channels, collision attacks, or compiler compromise.

10 Discussion

10.1 Positioning and novelty boundary

The nearest ingredients of SWC already exist separately. The classical probabilistic method supplies the logical implication from positive probability to existence; probabilistic program logics verify distributional computations; model counters compute accepted weight; proof-carrying systems separate untrusted production from trusted checking; and e-processes provide anytime-valid empirical risk control. The paper's claim is that these pieces can be organized into one theorem-facing certificate interface with explicit pointwise quantifiers and proof-status boundaries.

Table 2: Relationship to adjacent paradigms. The final column states the interface layer proposed here, not a claim that the preceding field lacks formal rigor.

Paradigm	Native strength	Layer emphasized by SWC
Probabilistic method	Proves existence from a positive-probability construction	Instance-indexed machine certificate, explicit mass object, and compiler to verifier-backed existence
Probabilistic program verification	Proves quantitative properties of randomized programs	Treats program output as a theorem-witness distribution with a deterministic target relation
Weighted model counting	Computes or certifies accepted weight for Boolean constraints	Connects the count to verifier soundness, universal quantifier scope, and certificate composition
Proof-carrying code	Untrusted producer emits a small independently checked object	Specializes the object to distributions, mass lower bounds, positivity, and theorem extraction
AI theorem search	Generates candidate proofs or witnesses checked by a formal environment	Allows the learned object to be a distribution or structural mass argument while keeping it outside the kernel
Anytime-valid inference	Controls false certification under adaptive monitoring	Places empirical guarantees in a separate assumption-indexed mode rather than calling them mathematical truth

Accordingly, the novelty claim is deliberately architectural and methodological. We do not claim a new axiom, a new proof of the probabilistic method, or a general complexity reduction. We propose a typed certificate calculus, a trusted-boundary specification, a status lattice, adversarial quantifier checks, proof-carrying mass back ends, and an evaluation protocol for learned proof search. Whether this synthesis proves useful at scale is an empirical research question, not a consequence of the soundness theorem alone.

10.2 Why positive mass is a useful target

Proof search normally asks for one valid path through an enormous discrete space. A mass certificate asks a different question: can a tractable region of the distribution be shown to intersect the accepted

set? This can expose tools unavailable to direct search, including expectation arguments, symmetry, couplings, local lemmas, inclusion–exclusion, algebraic counting, and concentration.

Even a tiny lower bound proves existence. Computational usefulness, however, depends on its scale. SWC therefore separates the Boolean field `proves_existence` from the quantitative field `expected_trials`.

10.3 Why learned mass must not be trusted directly

Monte Carlo success rates are not mass proofs. A model may overfit tested instances or assign zero mass to an exceptional class. The trusted object must be a symbolic or exact lower bound. Learning is most naturally used to suggest decompositions, symmetries, partitions, change-of-measure arguments, and candidate inequalities that a proof kernel can check.

10.4 Relation to derandomization

A positive-mass proof establishes existence without necessarily constructing a witness efficiently. Conditional expectations, pseudorandomness, and algorithmic local lemmas may extract deterministic constructions. Formal work on pseudorandomness and randomized algorithms provides a natural implementation substrate [12]. SWC records derandomization as an optional compiler stage rather than a prerequisite for theoremhood.

10.5 Mathematical and physical reality

In mathematical mode, an accepted certificate is an ordinary proof relative to its axioms. In empirical mode, the strongest defensible statement is conditional:

$$\text{under assumptions } \mathcal{A}, \quad \mathbb{P}(\text{false certification}) \leq \alpha.$$

The distinction is not a weakness of SWC; it is a boundary between deductive and empirical knowledge that the framework is designed to expose.

10.6 When a stochastic certificate is scientifically useful

A mass certificate is most valuable when at least one of the following holds: the accepted set has exploitable symmetry; a second-moment or local-lemma argument is easier than explicit construction; a knowledge compiler produces a compact counting graph; a learned generator exposes a tractable high-mass region; or a reduction decomposes a universal theorem into families with separate certificates. It is least useful when positivity is exactly as hard as the original satisfiability problem and no concise lower-bound proof exists.

10.7 Evaluation criteria for future systems

We recommend reporting six quantities separately: verifier soundness status, mass-certificate status level H0–H4, minimum pointwise mass over the audited family, expected extraction cost, certificate-checking cost, and coverage of the intended instance domain. Average mass or benchmark solve rate should never replace the minimum/coverage fields for a universal claim.

11 Limitations and Threats to Validity

1. **Classical logical core.** Positive probability implying existence is not new. The contribution is the certificate interface, compositional rules, quantifier discipline, AI integration, and reference implementation.
2. **No proof-assistant implementation yet.** The trusted kernel is Python/C++ with exact arithmetic, not a fully formal Lean, Isabelle, or Coq development. The paper proves the rules on paper; machine-checked metatheory remains future work.
3. **Worst-case hardness remains.** By Theorem 4.2, deciding positive mass under a full-support Boolean generator is SAT, and exact mass is #P-hard. SWC reorganizes proof obligations but does not remove complexity barriers.
4. **Structured scalability only.** The new dynamic-programming benchmark scales because the witness constraint has bounded local structure. It is not representative of arbitrary industrial #SAT.
5. **Synthetic SAT distribution.** Planted formulas and contradictory controls are suitable for testing semantics and support collapse, not for claiming state-of-the-art SAT performance.
6. **Quantifier audit is deliberately adversarial and simple.** Its value is logical calibration, not algorithmic novelty.
7. **Finite number-theory evidence.** The Erdős–Straus survey certifies only the stated range and shifts. It cannot justify a universal extrapolation.
8. **Simple sequential null.** The e-process experiment uses a simple Bernoulli null and fixed alternative. Composite nulls, nuisance parameters, dependence, and distribution shift need richer constructions.
9. **Certificate provenance.** Arbiter's historical contributions are based on internal audited logs; independent reproduction of the complete 122-family search requires public release of those prompts, hashes, and certificates.
10. **Scope of empirical mode.** A formal risk theorem is only as relevant as its declared physical assumptions. Instrument failure, unmodeled confounding, and adversarial data corruption remain external threats unless explicitly modeled.

12 Research Agenda

Machine-checked metatheory. Formalize the certificate judgment, derivation rules, and finite-weight kernel in Lean 4 or Isabelle/HOL. The first milestone is a proof-producing checker whose trusted result is an ordinary existential theorem.

Certified knowledge-compilation backend. Integrate CPOG-style proof-producing weighted model counting [4]. This would replace exhaustive SAT enumeration with independently checked compiled circuits on realistic formulas.

Moment and local-lemma compilers. Build tactic support that converts first/second-moment calculations, dependency graphs, and local-lemma inequalities into accepted-mass lower bounds. This targets the classical probabilistic method more directly than raw counting.

Distribution synthesis. Train theorem-proving agents to maximize *certified* rather than empirical witness mass. Candidate objectives should reward symbolic density-ratio bounds, decomposable support, and short mass proofs, not just sampled success.

Adversarial quantifier auditing. Automate checks for average-to-pointwise promotion, hidden independence, measure-zero exceptions, unproved normalization, and finite-search extrapolation. The stress-test suite should include adversarially hidden exceptional classes.

Continuous certificates. Implement rational interval and compactness certificates for nested approximate witnesses, including exact proof objects for closure, nesting, and diameter convergence.

Empirical certificate standard. Combine protocol hashes, immutable raw data, instrument metadata, e-processes, confidence sequences, sensitivity analysis, and replication records into portable assumption-indexed artifacts.

Open-mathematics benchmarks. Use problems with exact finite verifiers but unknown universal theorems. Score systems on new witnesses, certified mass lower bounds, obstruction discovery, certificate size, pointwise coverage, and honest localization of the remaining quantifier.

13 Conclusion

Stochastic reasoning does not need to weaken proof. If a sound verifier accepts a set of candidates with formally certified positive probability, then the accepted set is nonempty and the resulting mathematical conclusion is deterministic. SWC makes that classical principle explicit as a certificate boundary suitable for learned generators, exact auditors, and proof assistants.

The same language cannot make physical claims assumption-free. It can, however, produce machine-checkable statements about false-certification risk under explicit models and adaptive protocols. This yields a unified architecture with two honest endpoints: ordinary theoremhood in mathematical mode, and calibrated, auditable evidence in empirical mode.

The experiments show both promise and restraint. Learned distributions can increase exact witness mass by orders of magnitude; exact finite certificates can audit a model-led investigation of an open conjecture; and e-processes can preserve risk under optional stopping. At the same time, none of these mechanisms licenses promotion of finite evidence or probabilistic intuition into an unproved universal claim. That separation is the central design principle of Stochastic Witness Calculus.

Reproducibility Statement

The accompanying archive contains source code, fixed random seeds, raw CSV/JSON results, figures, tests, and the LaTeX source. The reported exact counts and probability values were regenerated in a clean rerun and matched the manuscript. Runtime measurements are environment-dependent and are not expected to be byte-for-byte identical. The principal commands are:

```
python experiments/exp_sat.py
python experiments/exp_chain_wmc.py
python experiments/exp_quantifier_audit.py
```

```
python experiments/exp_certificate_integrity.py
g++ -O3 -std=c++17 experiments/exp_erdos.cpp -o experiments/exp_erdos
./experiments/exp_erdos 10000000 127 results/erdos_results.csv results/erdos_summary.json
python experiments/exp_eprocess.py
python experiments/make_figures.py
python -m unittest discover -s tests
```

The exact finite results can be regenerated without network access.

Acknowledgments and Contribution Disclosure

Arbiter v23, an internal looped 27B-parameter reasoning model, contributed hypotheses, reformulations, and proof drafts to the Erdős–Straus case study. All mathematical claims retained from those runs were subjected to deterministic symbolic or computational checks. Arbiter is not listed as an author and is not part of the trusted verification base. GPT-5.6 Pro assisted with literature synthesis, code and figure generation, and manuscript drafting. The human author directed the project, selected and audited the retained claims, and is responsible for the manuscript.

References

[1] Noga Alon and Joel H. Spencer. *The Probabilistic Method*. 4th ed. Wiley, 2016.

[2] Philippe Audebaud and Christine Paulin-Mohring. "Proofs of Randomized Algorithms in Coq". In: *Mathematics of Program Construction*. Vol. 4014. Lecture Notes in Computer Science. Springer, 2006, pp. 49–68. DOI: 10.1007/11783596_6.

[3] Gilles Barthe, Thomas Espitau, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. "Proving Uniformity and Independence by Self-Composition and Coupling". In: *arXiv preprint arXiv:1701.06477* (2017).

[4] Randal E. Bryant, Wojciech Nawrocki, Jeremy Avigad, and Marijn J. H. Heule. "Certified Knowledge Compilation with Application to Formally Verified Model Counting". In: *arXiv preprint arXiv:2501.12906* (2025).

[5] Florent Capelli. "Knowledge Compilation Languages as Proof Systems". In: *arXiv preprint arXiv:1903.04039* (2019).

[6] Benjamin Dahan. "Sieve Dimension and Search Depth for the Erdős–Straus Conjecture, $n \equiv 1 \pmod{24}$". In: *arXiv preprint arXiv:2608.24035* (2026).

[7] Manuel Eberl, Johannes Hölzl, and Tobias Nipkow. "A Verified Compiler for Probability Density Functions". In: *arXiv preprint arXiv:1707.06901* (2017).

[8] Chelsea Edmonds and Lawrence C. Paulson. "Formal Probabilistic Methods for Combinatorial Structures using the Lovász Local Lemma". In: *arXiv preprint arXiv:2310.00513* (2023).

[9] Christian Elsholtz and Terence Tao. "Counting the Number of Solutions to the Erdős–Straus Equation on Unit Fractions". In: *Journal of the Australian Mathematical Society* 94.1 (2013), pp. 50–105. DOI: 10.1017/S1446788712000468.

[10] Paul Erdős and László Lovász. "Problems and Results on 3-Chromatic Hypergraphs and Some Related Questions". In: *Infinite and Finite Sets* 10 (1975), pp. 609–627.

[11] Steven R. Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. "Time-Uniform, Nonparametric, Nonasymptotic Confidence Sequences". In: *The Annals of Statistics* 49.2 (2021), pp. 1055–1080. DOI: 10.1214/20-AOS1991.

[12] Emin Karayel. "Derandomization with Pseudorandomness". In: *arXiv preprint arXiv:2404.16614* (2024).

[13] Yiping Ma, Yu-Lin Tsai, Mayank Rathee, Deevashwer Rathee, François Dupressoir, Pierre-Yves Strub, and Raluca Ada Popa. "ShannonProver: Towards Automating Formal Cryptographic Proofs". In: *arXiv preprint arXiv:2607.02847* (2026).

[14] Spiridon Mihnea and Dumitru C. Bogdan. "Further Verification and Empirical Evidence for the Erdős–Straus Conjecture". In: *arXiv preprint arXiv:2509.00128* (2025).

[15] Leonardo de Moura and Sebastian Ullrich. "The Lean 4 Theorem Prover and Programming Language". In: *Automated Deduction – CADE 28*. Vol. 12699. Lecture Notes in Computer Science. Springer, 2021, pp. 625–635. DOI: 10.1007/978-3-030-79876-5_37.

[16] George C. Necula. "Proof-Carrying Code". In: *Proceedings of the 24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. 1997, pp. 106–119. DOI: 10.1145/263699.263712.

[17] Morenikeji Neri, Paulo Oliva, and Nicholas Pischke. "A Systematic Way of Analysing Proofs in Probability Theory". In: *arXiv preprint arXiv:2604.08078* (2026).

[18] Orr Paradise, Oliver Richardson, Yoshua Bengio, and Shafi Goldwasser. "How to Verify Consistency of Probabilistic Claims". In: *arXiv preprint arXiv:2608.11181* (2026).

[19] Aaditya Ramdas, Peter Grünwald, Vladimir Vovk, and Glenn Shafer. "Game-Theoretic Statistics and Safe Anytime-Valid Inference". In: *Statistical Science* 38.4 (2023), pp. 576–601. DOI: 10.1214/23-STS894.

[20] Serge E. Salez. "The Erdős–Straus Conjecture: New Modular Equations and Checking up to $N = 10^{17}$". In: *arXiv preprint arXiv:1406.6307* (2014).

[21] Philipp Schröer et al. "A Deductive Verifier for Probabilistic Programs – Caesar". In: *arXiv preprint arXiv:2605.15827* (2026).

[22] Yong Kiam Tan, Jiong Yang, Mate Soos, Magnus O. Myreen, and Kuldeep S. Meel. "Formally Certified Approximate Model Counting". In: *arXiv preprint arXiv:2406.11414* (2024).

[23] The mathlib Community. "The Lean Mathematical Library". In: *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*. 2020, pp. 367–381. DOI: 10.1145/3372885.3373824.

[24] Jean-Baptiste Tristan et al. "Verification of Machine Learning Systems via Reparameterization". In: *arXiv preprint arXiv:2007.06776* (2020).

[25] Leslie G. Valiant. "The Complexity of Enumeration and Reliability Problems". In: *SIAM Journal on Computing* 8.3 (1979), pp. 410–421. DOI: 10.1137/0208032.

A Erdős–Straus Criterion Used in the Case Study

This appendix makes the number-theory verifier in Experiment 4 self-contained. It does not prove the universal conjecture; it proves that every accepted finite certificate reconstructs a valid solution and that every prime solution can be represented by some admissible shift.

Theorem A.1 (Anchor-shift and two-target divisor equivalence)**.** *Let $p \equiv 1 \pmod 4$ be prime. Then $4/p$ has a representation by three positive unit fractions if and only if there is an integer*

$$c \equiv 3 \pmod 4, \qquad 0 < c \le 2p,$$

for which, with $a = (p+c)/4$, some divisor $d \mid a^2$ satisfies

$$d \equiv -a \pmod c \qquad \text{or} \qquad 4d \equiv -1 \pmod c.$$

Each accepted pair (c, d) yields explicit positive denominators.

Proof. Suppose first that

$$\frac{4}{p} = \frac{1}{x} + \frac{1}{y} + \frac{1}{z}, \qquad x \le y \le z.$$

Because the other two terms are positive, $x > p/4$. Also $4/p \le 3/x$, so $x \le 3p/4$. Set

$$c = 4x - p, \qquad a = x, \qquad M = ap.$$

Then $c \equiv 3 \pmod 4$ and $0 < c \le 2p$, while

$$\frac{c}{M} = \frac{1}{y} + \frac{1}{z}.$$

Multiplication and completion of the rectangle give

$$(cy - M)(cz - M) = M^2.$$

Both factors are positive because each of $1/y$ and $1/z$ is smaller than their sum c/M. Thus, with

$$e = cy - M, \qquad e' = cz - M,$$

we have $ee' = M^2$ and

$$c \mid M + e, \qquad c \mid M + e'.$$

Conversely, any positive divisor pair $ee' = M^2$ satisfying these congruences defines

$$x = a, \qquad y = \frac{M+e}{c}, \qquad z = \frac{M+e'}{c},$$

and direct substitution gives $1/x + 1/y + 1/z = 4/p$.

It remains to collapse the divisors of $M^2 = a^2p^2$. Since c is odd and $4a = p + c$, we have

$$\gcd(a, c) = \gcd(p, c) = 1,$$

so $\gcd(M, c) = 1$. Hence one congruence $e \equiv -M \pmod c$ automatically implies the complementary one: from $ee' = M^2$,

$$(-M)e' \equiv M^2 \pmod c \quad \Longrightarrow \quad e' \equiv -M \pmod c.$$

Every divisor $e \mid M^2$ is uniquely $e = p^j d$ with $j \in \{0, 1, 2\}$ and $d \mid a^2$. Since $p \equiv 4a \pmod c$:

- if $j = 1$, then $e \equiv -M$ is equivalent to $d \equiv -a \pmod c$;
- if $j = 2$, it is equivalent to $4d \equiv -1 \pmod c$;

- if $j = 0$, then $d \equiv -4a^2 \pmod{c}$; replacing d by the complementary divisor a^2/d gives the preceding $4d \equiv -1$ target.

The reverse constructions choose $e = pd$ in the first branch and $e = p^2 d$ in the second branch. The displayed formulas for x, y, z then give positive integers and the exact identity. This proves both directions. □

The experiment enumerates the stated finite shift set and all divisors of a^2. Its C++ verifier reconstructs e, $e' = M^2/e$, and (x, y, z), then checks the integer identity

$$4xyz = p(xy + xz + yz)$$

using 128-bit integer arithmetic.

B Exact Finite Certificate Schema

A portable certificate may contain:

```
claim_id
instance_encoding
witness_space_encoding
generator_family
exact_weight_representation
deterministic_verifier
verifier_soundness_reference
accepted_mass_numerator
accepted_mass_denominator
positivity_proof
assumptions
software_and_formalization_hashes
```

For non-enumerative certificates, the accepted-mass fields may be replaced by a theorem reference and proof term establishing the lower bound.

C Additional Experimental Results

Table 3: SAT exact-mass results. "Speedup" denotes accepted-mass ratio, not wall-clock runtime.

Quantity	Value
Satisfiable instances	60
Unsatisfiable controls	20
Variables / clauses	16 / 112
Median uniform mass	1.5259×10^{-5}
Median learned mass	0.9393
Median 10% support-mixture mass	0.8454
Median support-mixture mass ratio	27,809.8×
Geometric-mean mass ratio	4,346.3×
Worst support-mixture ratio	0.1×
Hardened modes satisfying	50/60
Hardened modes with zero witness mass	10/60
Hard-mode mixtures with positive mass	60/60
Minimum hard-mode mixture mass	1.5259×10^{-6}

Table 4: Erdős–Straus finite certificate results.

Quantity	Value
Prime range	$p < 10^7$, $p \equiv 1 \pmod{24}$
Number of primes	82,887
Shifts	$3, 7, \ldots, 127$ (32 total)
Certified positive	82,887
Exact first-witness checks passed	82,887
Maximum least shift	107 at $p = 8{,}803{,}369$
Median uniform accepted mass	0.048398
Median small-shift-prior mass	0.176253
Minimum uniform accepted mass	1/3600

Table 5: Sequential monitoring simulation, 50,000 paths per condition.

Method	Null crossing	Power at $p = 0.6$
Repeated nominal 5% z-test	30.826%	99.946%
Likelihood-ratio e-process	4.342%	97.738%

Table 6: Non-enumerative path-graph mass certificate.

Quantity	Value
Largest path	$n = 2048$, $k = 512$
Witness-space size	2^{2048}
DP / closed-form cross-checks	all passed
Uniform exact mass	$10^{-193.396}$
Biased proposal exact mass	$10^{-77.047}$
5% support-mixture mass ratio	2.12×10^{116}
Uniform / biased checker runtime	1.04 s / 1.10 s

Table 7: Quantifier-audit examples.

Scenario	Test budget	Probability of misleading observation
One exception among 10^6 instances	10^4 draws	0.99005 miss
One exception among 10^6 instances	10^5 draws	0.90484 miss
True mass 10^{-6}	10^5 samples	0.90484 zero hits
True mass 10^{-8}	10^6 samples	0.99005 zero hits

Table 8: Proof-carrying certificate integrity results.

Quantity	Value
Valid certificates accepted	300 / 300
Altered certificates rejected	1,800 / 1,800
Mass-field mutations	600 / 600 rejected
Semantic claim mutations	600 / 600 rejected
Schema and float-only mutations	600 / 600 rejected
Median valid-certificate check	0.433 ms
99th percentile over all objects	4.06 ms