

CLARK Criterion: Certificate-Aware Active Evaluation of AI Workflows under Reliability Constraints

Lloyd Clark, PhD

Federal AI · October 27, 2026 · research release 0.35

Abstract

Reliable evaluation can cost more than executing an AI workflow. We study how to allocate reference measurements when predictions and independently checked facts are also available. We present CLARK Criterion, an executable composition of certificate-assisted prediction-powered e-processes, paired workflow comparisons and constrained selection. Sound support certificates can determine individual losses or contrasts without an audit. For remaining targets, a shared randomized audit and a positivity constraint preserve nonnegative betting processes even when a cheap predictor is biased. Simultaneous bounds support a declared failure ceiling, comparison tolerance and abstention. We provide proofs of the composition, conservative finite-grid inversion and a coupled cost identity; we do not claim a new general inference principle or an optimal acquisition algorithm. In 3,840 synthetic trials, exact certificate substitution reduced mean acquisition cost from 385.28 to 141.21 units in the useful-certificate condition (63.3%), but expensive proofs increased cost and 599 of 600 close-candidate trials remained unresolved. A retrospective replay of 1,500 published coding-workflow outcomes found that all three selective policies cost more than full auditing under the declared acquisition model. No new model calls were made. An independent OCaml core replays the single-regime archive decision using exact rational factors and outward integer arithmetic. It checks a narrowly specified numerical claim without authenticating the measurements or machine-verifying the implementation. Its guarantees depend on the measurement and sampling assumptions, certificate soundness where used, and the stated software dependencies.

1 Introduction

Selecting an AI assistant requires a decision about a complete workflow. A model name alone does not specify the prompts, retrieval, tools, reasoning budget or escalation policy that determine the user’s experience. A planning assistant may produce a valid schedule from an obsolete rule, recommend action when information is missing, or abstain on nearly every task. Mean answer quality conceals these distinctions. A useful evaluation asks whether a frozen workflow satisfies a declared reliability requirement and whether its performance and deployment cost justify selecting it over eligible alternatives.

Evaluation itself consumes resources. Executing every workflow on every task, grading every output and repeatedly simulating every proposed intervention may dominate the cost of a deployment decision. Cheap model predictions can guide this expenditure but can also be systematically wrong. Exact checks offer a different opportunity: a verified witness may settle several targets on the same

case. An impossible resource target can rule out multiple plans; a checked source-version condition can settle whether a response violates a stated benchmark rule. The scope of the witness must match the measured outcome. A unit test does not prove all semantic properties of a program, and a simulator proof does not certify the physical world.

This paper asks how to use such evidence within a reproducible constrained evaluator. We combine a support-aware audit rule with established prediction-powered e-process machinery, pair workflow outcomes on the same case, and project simultaneous confidence bounds onto a risk-plus-regret decision. CLARK Frontier provides the workbench; CLARK Criterion is its mathematical engine. The runner and workbench preserve the distinction between an observed descriptive ranking and a conditionally supported selection.

The contribution is a specified implementation and empirical study of that composition. The main mathematical results establish its validity under explicit assumptions, rather than a new universal rate. A simple cost identity explains when exact certificates save acquisition work at a fixed sample count. It does not establish that adaptive auditing always reduces cost to a decision. The experiments therefore include biased predictions, useless and expensive certificates, close candidates and infeasible menus. The resulting evidence is mixed: exact substitution is effective under favorable assumed proof costs, whereas selective auditing and adaptive policy complexity do not guarantee savings. The public-outcome replay supplies a concrete negative efficiency result and a provenance exercise, not a fresh evaluation of current models. A subsequent engineering extension adds a small independent OCaml verifier for archive decisions. Its arithmetic-enclosure argument and tested replay behavior are separate from the frozen empirical study; the earlier experiments have not been retroactively relabeled as verified by that core.

2 Related work

Active testing selects evaluation data to reduce measurement cost (Kossen et al. 2021). Berrada and colleagues show that a cheap frozen surrogate can make this practical for language models (Berrada et al. 2025). SELECT-LLM chooses informative annotation queries using similarities between candidate responses (Durmazkeser et al. 2025). These works already establish that economical evaluation should be compared with stronger baselines than exhaustive testing alone.

Prediction-powered inference uses predictions together with measured corrections (Angelopoulos et al. 2023). Prediction-Powered E-Values extends that principle to sequential inference (Csillag, Struchiner, and Goedert 2025). Ao and colleagues combine biased LLM judges, selective reference audits, inverse-propensity correction and anytime best-arm identification (Ao et al. 2026). Our factor construction is a support-conditioned specialization of this established machinery. The contribution is not the use of a judge, an audit or an anytime confidence sequence.

Multi-fidelity identification already has instance-dependent cost bounds and asymptotically optimal allocation in its own setting (Poiani et al. 2024). Cai and Kandasamy separately allocate performance and feasibility measurements (Cai and Kandasamy 2026). Li and Zhu construct sequential confidence sets for a constrained model-selection target (Li and Zhu 2026). Shared side observations and identification lower bounds are studied through feedback graphs (Russo, Song, and Pacchiano 2025). Our exact support constraints can depend on an observed case and can determine a contrast without both marginal losses. That distinction motivates this implementation, but does not establish priority over all joint-feedback models.

The recent SCALE framework chooses between selective AI scoring and human escalation while

accounting for their costs (Ham et al. 2026). This prior work already studies how to allocate evaluation effort. The policies studied here are transparent heuristics and ablations. They are not exact reproductions of SELECT-LLM, SCALE, TaS-FG or the full allocation algorithms in the cited best-arm papers; the experiments make no head-to-head superiority claim over those methods.

Agent evaluation must also identify what is being compared. Kapoor and colleagues emphasize cost, application suitability, holdouts and reproducibility (Kapoor et al. 2024). Hardy and colleagues distinguish rankings of fixed model-scaffold systems from rankings of underlying models (Hardy et al. 2026). We adopt the fixed-workflow estimand and retain a case hierarchy in the trace schema. These are evaluation disciplines, not independent novelty claims.

The independent verifier follows a conventional small-checker design. OCaml’s abstract module types restrict construction of validated internal values (OCaml contributors 2026), but neither static typing nor a language change establishes statistical soundness. Verified extraction from Rocq to OCaml has a substantially stronger and separately specified proof obligation (Forster, Sozeau, and Tabareau 2024). We do not implement that extraction pipeline or claim machine-checked correctness. The contribution here is an explicit replay contract and conservative arithmetic implementation integrated into the evaluation workbench.

3 Evaluation contract

3.1 Frozen workflows and case-level outcomes

Let $\mathcal{A} = \{1, \dots, A\}$ be a finite set of eligible workflows. Each workflow binds a model identifier, prompt, source-processing policy, tools, budget and escalation rule. These components are fixed before confirmatory evidence is used. A changed component creates a new workflow identity. Eligibility rules, such as local-only execution, are applied before the candidate menu is frozen.

For a fresh case W from regime $h \in \{1, \dots, H\}$, let $Y_a(W) \in [0, 1]$ be the loss defined by a frozen reference evaluator, and write $\mu_{ha} = \mathbb{E}_h Y_a$. A case can contain a declared number of stochastic attempts. Their prespecified mean is one bounded case outcome; the attempts are not counted as independent cases. For unsupported-action risk, the reference rule must specify how abstentions, missing outputs and invalid responses are scored. A separate completion or coverage measure prevents an always-abstaining workflow from being mistaken for a useful solution.

The measured workflow outcome differs from uncertainty inside a planning tool. Hundreds of simulation worlds evaluating one generated schedule do not constitute hundreds of independent language-model cases. If a nested simulator decision remains unresolved, the outer evaluator must retain that state or apply a separately justified error-budget rule. The present API does not transform an unresolved inner point estimate into a true outer label.

The case vector couples workflows through the same task and exogenous environment. It need not share language-model randomness. A provider seed does not establish a mathematically valid common-random-number coupling across different models.

3.2 Reliability and relative loss

Fix a nominal regime distribution q^0 , a total-variation radius ρ , a risk ceiling κ , a regret tolerance ϵ , deployment costs c_a and a nonnegative cost weight γ . Define

$$\mathcal{Q}_\rho = \{q \geq 0 : \mathbf{1}^\top q = 1, \frac{1}{2}\|q - q^0\|_1 \leq \rho\}, \quad g(a) = \sup_{q \in \mathcal{Q}_\rho} \sum_h q_h \mu_{ha}.$$

The true feasible set is $\mathcal{F} = \{a : g(a) \leq \kappa\}$. For a nonempty feasible menu, define constrained regret

$$R(a) = \max \left\{ 0, \max_{b \in \mathcal{F}} \sup_{q \in \mathcal{Q}_\rho} \left[\sum_h q_h (\mu_{ha} - \mu_{hb}) + \gamma(c_a - c_b) \right] \right\}.$$

Deployment cost is the cost of using a selected workflow. Acquisition cost is the separate expense of obtaining decision evidence. A low acquisition cost does not make a workflow cheap to deploy. The numerical studies specify both quantities and do not translate abstract acquisition units into provider dollars.

3.3 Shared support certificates

After a case is drawn, cheap information S_t may include inputs, workflow outputs, scores and verified witnesses. An independent checker returns a set

$$\mathcal{C}_t \subseteq [0, 1]^A, \quad Y(W_t) \in \mathcal{C}_t \quad \text{almost surely.}$$

The implementation accepts marginal intervals and optional pairwise contrast intervals. These are a restricted representation of joint support, not a solver for arbitrary logical proof systems. It validates their numerical consistency and can detect a false support claim when an audited label contradicts it. Such checking does not prove the truth of an unaudited imported witness. Soundness remains the responsibility of the task-specific independent checker.

A detected contradiction quarantines the inference instance. Continuing after simply discarding the contradictory case would preserve potentially invalid evidence from earlier unchecked witnesses and would selectively censor an adverse observation. A new recommendation requires checker review and a fresh replay under a valid contract.

The target set contains each risk $X_a = Y_a$ and each normalized contrast $X_{ab} = (1 + Y_a - Y_b)/2$, so every target lies in $[0, 1]$. Let $J = A + \binom{A}{2}$. A certificate gives a target interval $I_{tj} = [\ell_{tj}, u_{tj}]$. A singleton determines that target exactly. For example, $Y_a - Y_b = 0$ determines a comparison while leaving both risks unknown. The evaluator does not infer feasibility from that comparison alone.

3.4 Assumptions

The validity statements use the following contract. First, workflows, targets, losses, regimes, mixture set and error allocations are fixed before confirmatory cases. Second, the current regime is selected using past information, before drawing a fresh case with its declared regime law. Third, each target's conditional mean within a regime equals its fixed parameter. Fourth, the support checker is sound for the same reference loss. Fifth, an audit reveals that reference measurement, and its random coin is conditionally independent of the hidden outcome given available cheap information. Sixth, unresolved targets have positive logged audit propensity. Seventh, any predictor may learn from previous cases and use current cheap information, but may not see a concealed current reference label before choosing its prediction and propensity.

These assumptions do not make human raters infallible, establish independence of a real workload, validate a simulator or preserve evidence after a model update. A human-evaluation application needs a declared adjudication or noisy-measurement model. A finite fixed task bank requires an appropriate finite-population analysis or an explicitly stated empirical-distribution target.

4 Certificate-assisted sequential evidence

4.1 Base factors and corrected observations

Fix one target and regime stream, suppress its subscripts, and let θ denote a proposed mean. Choose a finite collection of positive magnitudes $\Lambda \subset (0, 1)$ before observing the current case. Each signed base factor is

$$e_t^\lambda(x; \theta) = 1 + \lambda(x - \theta), \quad \lambda \in \Lambda \cup (-\Lambda).$$

Under the mean null $\mathbb{E}[X_t \mid \mathcal{F}_{t-1}] = \theta$, the base factor has conditional mean one. Its choice is predictable. In particular, choosing a bet after reading the current judge score would need a new conditional-validity argument.

After cheap information arrives, select a clipped prediction $p_t \in [0, 1]$ and let $\widehat{e}_t^\lambda(\theta) = 1 + \lambda(p_t - \theta)$. For an unresolved support interval, draw a shared audit coin $B_t \sim \text{Bernoulli}(\pi_t)$ with $\pi_t > 0$. Define

$$Z_t = p_t + \frac{B_t}{\pi_t}(X_t - p_t), \quad G_t^\lambda(\theta) = 1 + \lambda(Z_t - \theta).$$

The corrected observation Z_t can lie outside $[0, 1]$. It is therefore not a Bernoulli observation and must not be passed to the older beta-mixture engine. This release provides a separate inference primitive.

To preserve nonnegativity, the audit probability must satisfy every active target, sign, bet and parameter. For one factor, a sufficient condition is

$$\pi_t \geq \left[1 - \frac{\inf_{x \in I_t} e_t^\lambda(x; \theta)}{\widehat{e}_t^\lambda(\theta)} \right]_+.$$

The implementation takes the maximum of these requirements and a positive audit floor. Since the factors are affine in x and θ , endpoint calculations control the whole parameter interval, not merely the displayed grid. If every relevant target is exactly certified, no audit is needed and its exact factor is used. A zero lower bound in the displayed inequality does not authorize zero propensity for an unresolved target.

The audit coin is part of the evidence record. If reference acquisition fails, retrying the same pending case preserves the original coin; it must not redraw until a convenient outcome occurs. The replay interface consumes logged coins and propensities but cannot authenticate how a third party originally generated them.

4.2 Validity of the audit correction

Proposition 1 (certificate-assisted factors). Under the evaluation contract and positivity rule, each product $E_t^\lambda(\theta) = \prod_{i=1}^t G_i^\lambda(\theta)$ is a nonnegative martingale under the mean null. The same conclusion holds when exact certificates substitute the corresponding base factor. A fixed convex mixture of these products is a nonnegative martingale and hence an e-process.

Proof. If no audit occurs, the corrected factor is $\hat{e}_t^\lambda > 0$. If an audit occurs, it equals

$$\frac{e_t^\lambda(X_t; \theta) - (1 - \pi_t)\hat{e}_t^\lambda(\theta)}{\pi_t} \geq 0.$$

Conditional on the cheap information and latent reference outcome, averaging over the independent audit coin gives exactly $e_t^\lambda(X_t; \theta)$. A sound singleton certificate gives the same factor without randomness. Applying the tower rule and the base mean-null identity gives conditional expectation one given $\mathcal{F}_{t-1}$. Multiplication proves the martingale statement, and fixed nonnegative weights summing to one preserve it. The argument allows biased predictions and outcome-dependent certificates because the corrected conditional expectation is the original base factor. It does not allow an outcome-dependent audit coin with an incorrectly logged propensity. $\square$

This proposition is an explicit specialization of established prediction-powered e-value reasoning (Csillag, Struchiner, and Goedert 2025), not a foundational novelty claim. A shared audit serves multiple targets without requiring the targets to be independent; its probability must satisfy all their positivity conditions.

The correction explains why fewer audits need not mean cheaper decisions. Conditional on cheap information, let $m = \mathbb{E}[X \mid S]$ and $v = \text{Var}(X \mid S)$. Direct expansion gives

$$\text{Var}(Z \mid S) = \frac{v}{\pi} + \left(\frac{1}{\pi} - 1\right)(m - p)^2.$$

Reducing the audit rate amplifies both irreducible variation and prediction error. For a paired contrast the same identity applies to the contrast and its predictor, preserving the within-case covariance. A better predictor or a tighter support can help; an overconfident biased predictor can make selective measurement inefficient even while correction preserves validity. This variance identity is elementary and is not an allocation-optimality result.

4.3 Simultaneous inference and finite-grid inversion

For each regime and target, maintain separate mixtures over positive and negative bets. Allocate $\alpha/(2HJ)$ to each signed stream. Ville’s inequality and a union bound imply that, with probability at least $1 - \alpha$, no true mean is rejected in any target, regime or time. Predictable interleaving of regime streams and data-dependent stopping preserve this event (Howard et al. 2021).

Proposition 2 (grid outer confidence set). In exact arithmetic, the implemented finite-grid interval contains the continuous acceptance set at every time. Together with Proposition 1 and the error allocation, it covers every target mean simultaneously over time with probability at least $1 - \alpha$.

Proof. For fixed realized data and propensities, every positive-bet factor is nonincreasing in θ ; every negative-bet factor is nondecreasing. Nonnegative products and fixed mixtures preserve those directions. The intersection of the two acceptance sets is therefore an interval. When grid nodes are

accepted, extend the smallest and largest accepted nodes outward by one grid cell, clipping to $[0, 1]$. No point of the continuous acceptance interval can extend beyond these neighbors without another grid node being accepted. When no node is accepted, return $[0, 1]$ and an unresolved grid status. This also contains the continuous acceptance set, including a nonempty interval narrower than a grid cell. The simultaneous claim follows from the previously stated probability event. $\square$

The proof concerns exact arithmetic. The Python and browser evidence engines evaluate log products and mixtures in floating point. Endpoint expansion addresses discretization; it is not a validated floating-point enclosure. Numerical regression and invariance tests detect specific failures but do not prove arbitrary-precision soundness. Empty grid acceptance forces abstention rather than a spurious conclusion that the population mean is impossible. The separate OCaml replay path below provides an outward integer-arithmetic enclosure for its restricted single-regime contract; it does not extend this claim to the existing general engines.

4.4 Feasibility and regret decisions

Convert normalized contrast intervals back to raw differences. Let L_{ha}, U_{ha} bound each risk and let $U_{h,ab}^\Delta$ bound $\mu_{ha} - \mu_{hb}$. Define

$$L_g(a) = \sup_{q \in \mathcal{Q}_\rho} \sum_h q_h L_{ha}, \quad U_g(a) = \sup_{q \in \mathcal{Q}_\rho} \sum_h q_h U_{ha}.$$

The certified and possible feasible sets are respectively

$$\hat{\mathcal{F}} = \{a : U_g(a) \leq \kappa\}, \quad \mathcal{P} = \{a : L_g(a) \leq \kappa\}.$$

For each candidate, compute

$$\bar{R}(a) = \max \left\{ 0, \max_{b \in \mathcal{P}} \sup_{q \in \mathcal{Q}_\rho} \left[\sum_h q_h U_{h,ab}^\Delta + \gamma(c_a - c_b) \right] \right\}.$$

Proposition 3 (supported selection). On the simultaneous coverage event, $\hat{\mathcal{F}} \subseteq \mathcal{F} \subseteq \mathcal{P}$. Any recommendation $a \in \hat{\mathcal{F}}$ with $\bar{R}(a) \leq \epsilon$ satisfies $g(a) \leq \kappa$ and $R(a) \leq \epsilon$.

Proof. For every q and action, interval coverage gives $q^\top L_a \leq q^\top \mu_a \leq q^\top U_a$. Taking suprema proves the set inclusions. Every truly feasible comparator lies in $\mathcal{P}$, and each of its contrasts is at most its displayed upper bound under every admissible mixture. Maximizing over the larger comparator set and mixtures proves the regret bound. $\square$

This is confidence-set projection, with direct precedents in constrained identification and selection (Cai and Kandasamy 2026; Li and Zhu 2026). Its purpose is to make the software’s recommendation semantics inspectable. A near-best candidate can still violate the absolute risk ceiling. If a required risk or comparison remains unresolved, the evaluator abstains. Boundary separation is not assumed merely to make the software return a winner.

5 Acquisition policies and costs

The engine separates statistical validity from acquisition policy. A requested audit rate can depend on past residuals and current cheap scores; the positivity rule may raise it. The policy must log the actual rate. Exact certificates may remove all need for reference measurement on some cases. Partial certificates may reduce the positivity floor for some targets without eliminating the shared audit needed by another target.

Proposition 4 (coupled exact substitution and cost). Fix a sequence of cases and a sample-count stopping rule based only on completed labels. Suppose a sound certificate either supplies exactly the label vector required by that rule or leaves it to the reference evaluator. Replacing those label evaluations by exact certificates leaves the completed evidence and stopping count unchanged. For a fixed n cases, if V_t is certificate acquisition/checking cost, D_t is the reference cost without substitution and S_t indicates that the certificate makes that reference measurement unnecessary, the coupled cost difference is

$$C_n^{\text{cert}} - C_n^{\text{full}} = \sum_{t=1}^n (V_t - S_t D_t).$$

Proof. Induction on cases shows that the completed labels and label-based stopping decisions agree. Per-case bookkeeping then gives the identity. Taking expectations gives a saving exactly when $\sum_t \mathbb{E}[S_t D_t] > \sum_t \mathbb{E}[V_t]$. $\square$

For shared reference measurements, D_t is the incremental cost actually avoided, not a sum of nominal per-target costs that would otherwise share one audit. The proposition does not apply to wall-clock stopping, changed acquisition policies, partial labels that alter the inferential update or selectively audited pseudo-observations. It establishes neither a universal speedup nor a cost-optimal evidence policy. Expensive or uninformative proof search can increase cost.

The experiment compares declared full, static, adaptive and certificate-assisted policies under a common implementation. The adaptive rules estimate useful audit effort from development information or prior residuals; their optimization is heuristic. Acquisition cost records include proof attempts, checking, cheap scores, reference measurement and orchestration. Generation cost for a prerecorded trace bank is reported separately when it is sunk or unavailable, rather than silently treated as free live inference. Unknown prices remain unknown.

6 Software and workflow benchmark

The release supplies four separable components: a typed workflow and trace contract; a source-grounded case runner; the audited evidence primitive; and the Frontier workbench. The trace-bank schema binds workflow definitions, source packets, case-family splits, attempts, reference outcomes and artifact digests. Workflow manifests include the model identifier, prompt, source-processing and tool policy, generation parameters, budget and escalation rule. Failures remain visible outcomes. Unknown provider, human or local-compute costs remain null rather than becoming zero.

The generated source-grounded suite contains 200 case families: 40 development, 40 calibration and 120 test families. Eight challenges cover current and superseded rules, future-effective sources, missing or conflicting facts, infeasibility, budget restrictions and boundary cases. Each packet defines a finite integer resource-addition problem. The reference resolves approved, effective, latest facts

and computes the least feasible addition. Correctness requires the prescribed decision, action and source identifiers. Primary loss counts any wrong decision, malformed output, citation failure or execution failure; unsupported action is reported separately. Three attempts produce one case-level mean.

The fixture adapters are deterministic rule programs, not language models. In particular, the typed-rules fixture shares the reference resolver. It is a privileged transparent-preprocessing check and cannot substantiate a claim that an LLM correctly compiled a source packet. These fixtures validate the pipeline and failure semantics. Their apparent accuracy is not a model benchmark result.

An optional HTTP adapter and an Inspect-compatible entry point permit later model execution ([UK AI Security Institute and Meridian Labs 2026](#)). The release keeps model generation separate from inference over a completed trace bank. Live execution requires a configured provider and explicit cost limits; no live provider request is part of this paper. Transport and mock-model tests check adapter behavior without becoming evidence of model capability. The retained runner receipt records 25 passing tests, actual loopback HTTP transport and six mock samples through installed Inspect 0.3.285 (three cases, two epochs). A schema-valid trace is also not a certificate that its historical measurements were honest or correct.

The statistical primitive rejects stale or foreign acquisition plans, preserves a pending audit coin across reference retries, requires known deployment costs when a positive cost weight is used, and quarantines an instance after an observed certificate contradiction. Unit and independent adversarial tests exercise exact audit expectations, support consistency, paired signs, grid abstention and cost accounting. These checks support implementation confidence within their tested cases; they do not replace the assumptions of the propositions.

A retained fixture run exercises 120 test families, four rule workflows and three attempts per workflow-case (1,440 attempt records). Case-level primary losses are 0.0000 for the version-aware fixture, 0.1417 for the first-revision fixture, 0.6583 for always-act and 0.8667 for always-abstain. The always-abstain fixture has zero unsupported-action loss and zero completion, illustrating why the risk metric alone is insufficient. These are deterministic pipeline checks with synthetic source packets; none is an independently measured language-model result. Costs remain unknown in that fixture receipt.

7 Independent OCaml verification core

7.1 Replay contract and trust boundary

The 0.35 extension keeps model execution, data preparation and exploratory evaluation in Python and the workbench in JavaScript. A separately compiled OCaml executable consumes a versioned request and recomputes the final archive inference. The supported contract has one regime, zero mixture-shift radius, zero deployment-cost weight, bounded workflow losses, full support intervals and a frozen finite bet set. It accepts no imported proof or support certificate. Multiple regimes, robust mixture optimization, partial supports and general certificate verification remain outside this core.

A request contains the ordered workflow menu, error budget, null-grid size, bets, risk ceiling, regret tolerance and contiguous observation sequence. Each observation records its case identifier, audit probability, audit flag, predictions, and reference losses only when audited. Decimal-looking model

and configuration values are exported as exact rational representations of their binary64 values; the verifier does not reinterpret them as an intended decimal ideal. The null grid is the exact rational grid $i/(m - 1)$. Consequently, disagreement with a floating-point engine at a boundary is possible and is not silently rounded into approval.

The parser rejects malformed or unsupported requests, unknown or duplicate fields, invalid rational encodings, inconsistent dimensions, noncontiguous sequences, hidden losses on unaudited cases and skipped probability-one audits. Before using the observed loss, it checks propensity sufficiency for both signs at every relevant outcome and null-domain endpoint. Abstract OCaml interfaces separate parsed data from validated requests and checked results. These interfaces reduce accidental misuse inside the program; runtime checks enforce properties of external JSON and recorded sequences.

The Python export route validates the complete trace bank, validates the report hash, and independently reruns its frozen configuration before producing a request. It compares every report field other than the volatile elapsed-time field and the separately validated content hash. The browser exports from its current locally computed result. Neither path elevates an imported report to a verified result by relabeling it. The native checker compares its recomputed status and selected workflow with the claim. A mismatch, unavailable executable, invalid request or execution timeout cannot fall back to the original recommendation. A wrapper receipt binds the exact request bytes and checker executable by SHA-256.

These bindings establish reproducible identity, not truth or authorship. The source identifiers echoed by the native core are unauthenticated metadata. A direct request can contain invented measurements with recomputed checksums. The trusted boundary includes the validated input interpretation, OCaml implementation, compiler and runtime, Yojson parser, Zarith/GMP arithmetic and execution environment. Reference correctness, honest provenance, eligibility completeness, randomization and the fixed-conditional-mean contract require separate evidence. The check never authorizes automatic deployment.

7.2 Conservative arithmetic and decision reconstruction

Let $D = 2^{128}$. Store a nonnegative product as integer endpoints $[L, U]$, denoting the real interval $[L/D, U/D]$. Start with $L = U = D$. For a nonnegative exact rational factor $g = n/d$, where $d > 0$, update

$$L' = \left\lfloor \frac{Ln}{d} \right\rfloor, \quad U' = \left\lceil \frac{Un}{d} \right\rceil.$$

All factors and integer divisions use exact arbitrary-size arithmetic. For a uniform mixture over K products, sum their lower integer endpoints and divide by KD to obtain a lower bound. A grid node is rejected only if that bound reaches the exact rational threshold $2J/\alpha$ for at least one sign. An uncertain threshold comparison retains the node. Products are not clipped at the rejection threshold, because a later factor can reduce them. Resource limits reject oversized requests rather than silently truncating the evidence stream.

Proposition 5 (outward replay enclosure). Suppose the validated request satisfies the nonnegativity contract and the stated integer operations are implemented correctly. At every update the stored interval contains the exact product represented by that request. Every grid node accepted by both exact mixtures is retained by the verifier. Expanding the hull of retained nodes to its neighboring exact grid points, or returning the full domain when none is retained, contains the

continuous exact acceptance interval. The reconstructed risk and regret decisions therefore inherit Proposition 3’s implication for this single-regime, zero-cost-weight contract on its simultaneous coverage event.

Proof. Initially the exact product is one. If $L/D \leq e \leq U/D$ and $g = n/d \geq 0$, multiplication preserves order, and floor and ceiling give $L'/D \leq eg \leq U'/D$. Induction establishes product containment. Summation preserves the lower bounds, so rejecting only when the lower mixture reaches the threshold cannot reject an exactly accepted grid point. The lower-bound recurrence is also monotone in θ : nonnegative multiplication and floor preserve the direction of the signed factors. Thus each lower mixture has the same monotonic direction as its exact counterpart. Their continuous joint accepted set is an interval containing the exact acceptance set, even if rounding creates extra retained nodes. Its grid hull expanded to neighboring grid points contains that interval by the argument of Proposition 2; if no node is retained, the full domain is conservative. Exact affine conversion of contrasts and rational comparisons preserve those outer bounds at the decision gate, where Proposition 3 applies. $\square$

This is an ordinary mathematical proof of the algorithm, conditional on the implementation and its dependencies. It is not a proof-assistant certificate about the executable. Fixed fractional precision may widen the enclosure and reduce decisiveness; it cannot by itself justify rejecting an uncertain null under this update rule. The verifier also checks the claimed final status, so an otherwise valid numerical replay that disagrees with the submitted claim is a refused claim rather than a successful verification. No speed advantage, universal equivalence with floating-point output or extension to unimplemented target classes is asserted.

7.3 Verification checks and observed boundary refusals

The native implementation compiled with OCaml 4.14.1, Dune 3.14.0, Zarith 1.13 and Yojson 2.1.2. Its local test executable passed 3,111 arithmetic and parser assertions. A separate Python standard-library `Fraction` oracle, without production statistical imports, checked 61 valid replays containing 127,834 exact factor products. The checker retained conservative bounds in these cases; one deliberately near-threshold case produced a conservative claim disagreement. It rejected all 35 malformed or unsupported requests and a deliberately false workflow-selection claim. A negative control confirmed that changed provenance hashes do not authenticate the observations. These are finite implementation checks, not a machine proof or a new coverage experiment.

Five end-to-end archive diagnostics exercised source validation, full report replay, exact request export and the compiled checker. Three claims were supported and two boundary claims were refused:

Archive and risk ceiling	Draws	Audits	Floating-point claim	Native result
Synthetic fixture, 0.30	224	123	Qualify fixture workflow	Supported
Public archive, 0.20	288	159	No feasible workflow	Supported
Public archive, 0.60	160	88	Qualify workflow A	Risk unresolved
Public archive, 0.59	192	106	Qualify workflow A	Risk unresolved
Public archive, 0.625	160	88	Qualify workflow A	Supported

At the 0.60 ceiling, the exact uniform grid produces an upper bound of $3/5$, while the binary64 input denoted by 0.6 is 5404319552844595/9007199254740992, slightly smaller. The floating-point engine displays both values as 0.60. The native check preserves their distinction and refuses qualification. The 0.59 case has the analogous boundary. The exactly representable ceiling $5/8$ lies between grid nodes and has a positive margin in the reported diagnostic. It is a different decision specification, not a repair or reinterpretation of either refused claim.

These diagnostics were iterative engineering checks: the initial three-scenario run exposed the 0.60 disagreement; the attempted 0.59 case exposed a second boundary; the 0.625 case was added to exercise a nonboundary qualification. Both earlier receipts and all refusals are retained. They were not preregistered comparative experiments and do not establish how often disagreement occurs. No old empirical result is relabeled as OCaml-verified. Local native runtimes for the final five scenarios ranged from approximately 0.32 to 1.02 seconds on this environment; no cross-language speed comparison is claimed.

The integrated regression suite ran 232 Python tests with zero failures or errors and one optional Inspect-dependency skip; all native verification integration checks ran. Six Node suites passed, including exact Python/JavaScript request transport and native replay of a browser-generated request. Browser receipt generation is tested separately from a successful browser file download. No new model-provider calls were made. Reproduction inputs and byte-bound native receipts are retained with `results/ocaml_archive_replay.json`, `results/ocaml_independent_review.json` and the source package.

8 Experiments

8.1 Frozen synthetic protocol

The first protocol was frozen after a separate exploratory timing pilot and before its confirmatory runs. Inspection of that completed version exposed a dominated proof-exploration expense. We retained every original result, amended the policy to skip proof calibration and exploration when one proof attempt costs at least a complete reference vector, and froze a second protocol before a fresh-seed run. The tables and main figures below report this second run. This is transparent sequential development with a new confirmation run, not a claim that the amendment preceded the original findings. It contains six conditions, six policies and 100 repetitions per condition-policy pair (3,600 trials). A rare-failure supplement adds 40 repetitions per policy (240 trials). Independent

pseudorandom streams separate pilot fitting, confirmatory case generation and policy audit coins; methods share the same case sequence within a repetition. The confirmatory seed differs from the timing seed. The retained protocol records the earlier pilot and the pre-confirmation grid change.

Three workflows yield six targets: three losses and three paired contrasts. Each synthetic case has a fair context bit and a fresh uniform latent variable; workflow losses are nested threshold indicators of that common variable. The first four conditions have context-specific risks $(.02, .08, .22)$ and $(.18, .52, .68)$, hence marginal risks $(.10, .30, .45)$, with $\kappa = .35$ and $\epsilon = .12$. Their certificate probability is .70, except that the no-certificate condition uses zero; a proof attempt costs .20 units, except that expensive proofs cost 5. The biased-judge condition reverses fitted scores. Near ties use marginal risks $(.25, .26, .27)$, $\kappa = .45$ and $\epsilon = .02$. The infeasible menu uses $(.50, .60, .75)$ and $\kappa = .30$. Rare failures use $(.002, .004, .006)$ with $\kappa = \epsilon = .01$.

These experiments have one inferential regime ($H = 1$), zero mixture-shift radius and zero deployment-cost weight; the context is cheap within-regime information. They do not empirically establish the benefit of robust regime allocation. The nested common-uniform construction makes pairing especially informative and does not represent arbitrary dependence.

All methods use the same signed fixed-bet mixture, magnitudes $(.05, .20, .40)$, total $\alpha = .05$ and outward grid inversion. Main runs use 201 grid nodes, checks every 32 cases and a 1,024-case cap; the supplement uses 401 nodes, checks every 128 and a 4,096-case cap. A run stops at a supported selection or exclusion of the whole menu; otherwise it is unresolved at the cap. The near-tie grid was increased from the exploratory pilot so discretization alone would not preclude its declared tolerance.

The six policies are full auditing, requested-probability-.5 auditing, a pilot-tuned static design, certificate attempts followed by full auditing, certificate attempts followed by requested-probability-.5 auditing, and adaptive acquisition. Actual audit probabilities are raised when positivity requires it. Fitted predictors use 64 independent pilot cases and Beta(1,1)-smoothed context means. Static tuning chooses from 36 per-context proof/audit designs by a pilot cost-times-normalized-variance objective. Adaptive acquisition uses past residual variance, a measured proof-success savings gate and one exploratory proof attempt every 64 cases. Neither allocation policy is claimed optimal.

Every joint audit costs 3 units and per-case context/orchestration costs .01. Fitted predictors pay all pilot labels; static and adaptive proof selection also pay 64 pilot proof attempts when proof acquisition is not ruled out by the known cost-dominance guard. The full-audit policies fit no predictor. This difference is intentional: pilot learning has a real acquisition burden and must earn its cost. Failed proof attempts are charged. Costs are declared relative units, not provider dollars or Python runtime.

A successful synthetic proof channel reveals the latent finite-world witness, which a checker uses to recompute the complete label vector. Thus certificates are sound by construction. Their availability and price are assumptions of the mechanism experiment, not measured performance of an LLM proving system. No case label is made available to a policy before its charged certificate or independent audit.

We report decision and unresolved counts, false decisions against exact law truth, joint coverage failures at scheduled checks, acquisition costs, sample counts and evidence purchases. Cost intervals are ordinary 95% normal Monte Carlo intervals; differences use matched repetitions. Wilson intervals quantify event-rate uncertainty. These descriptive intervals are not multiplicity-adjusted claims about every policy comparison. The empirical coverage scan is restricted to scheduled checks; the

anytime assertion comes from the conditional theorem, not from this finite scan.

8.2 Synthetic results and negative controls

Condition	Full	Cert. + full	Static tuned	Adaptive
Useful certificates	385.28	141.21	285.84	294.07
No certificates	387.21	412.93	416.23	407.58
Expensive proofs	380.46	748.97	395.41	392.16
Biased judge	390.10	144.88	349.32	294.34
Near ties	3,082.24	1,132.11	803.42	888.95
Infeasible menu	419.96	154.10	304.52	308.12
Rare failures	12,328.96	4,532.31	4,274.06	2,912.67

Table 1. Mean total acquisition units, including charged pilots. Primary rows use 100 repetitions per policy; rare failures use 40. Costs stop at a supported decision or the case cap. Near-tie and most rare-failure values are predominantly costs to an unresolved cap, not costs to successful identification. Figure 1 shows all six policies in the primary study; complete intervals and trial records accompany the release.

Exact substitution behaves as the coupled identity predicts. In the useful-certificate condition, full auditing costs 385.28 units, compared with 141.21 for certificate-plus-full auditing: a 63.35% reduction. The paired difference is -244.07 units, 95% CI [-250.97, -237.17]. Both policies use 128.00 cases on average, and every matched repetition has the same stopping count. Certificates replace 89.89 joint audits on average while every proof attempt remains charged. This result tests a specified information channel, not the practicality of obtaining such proofs in production.

The negative controls reverse the cost conclusion. With no useful certificates, certificate-plus-full auditing costs 412.93 versus 387.21. With expensive proofs it costs 748.97 versus 380.46; the paired excess is 368.51, CI [360.18, 376.84]. The amended static and adaptive policies suppress proof calibration and exploration here because the known attempt price exceeds the cost of the full vector it could replace. The fixed always-attempt policies deliberately retain the expensive-proof negative control.

Adaptive acquisition does not dominate the pilot-tuned static comparison. With useful certificates, adaptive costs 294.07 against static’s 285.84; the paired excess is 8.23, CI [3.83, 12.63]. Under reversed predictions, adaptive is cheaper than static by 54.97 units (adaptive-minus-static CI [-58.59, -51.36]). Fixed selective auditing without certificates also costs more than full auditing in the first four primary conditions and the infeasible-menu condition. Fewer paid audits can be offset by pilot expense and more cases needed to accumulate evidence.

Every policy selects a qualifying workflow in all 100 repetitions of each of the first four conditions. Every policy excludes the infeasible menu in all 100 repetitions. Of 600 near-tie runs, 599 remain unresolved at 1,024 cases; uniform auditing makes one qualifying selection. In the rare-failure supplement, between 0 and 3 of 40 runs per policy qualify; all remaining runs reach the 4,096-case cap unresolved. Cheaper unresolved evidence is not successful identification, and the present fixed-bet construction is not demonstrated as an efficient rare-event certifier.

We observe no false decisions in the 3,840 synthetic trials. One uniform-audit near-tie trial has a monitored joint-coverage failure. Policy trials share the same paired case stream, so their counts

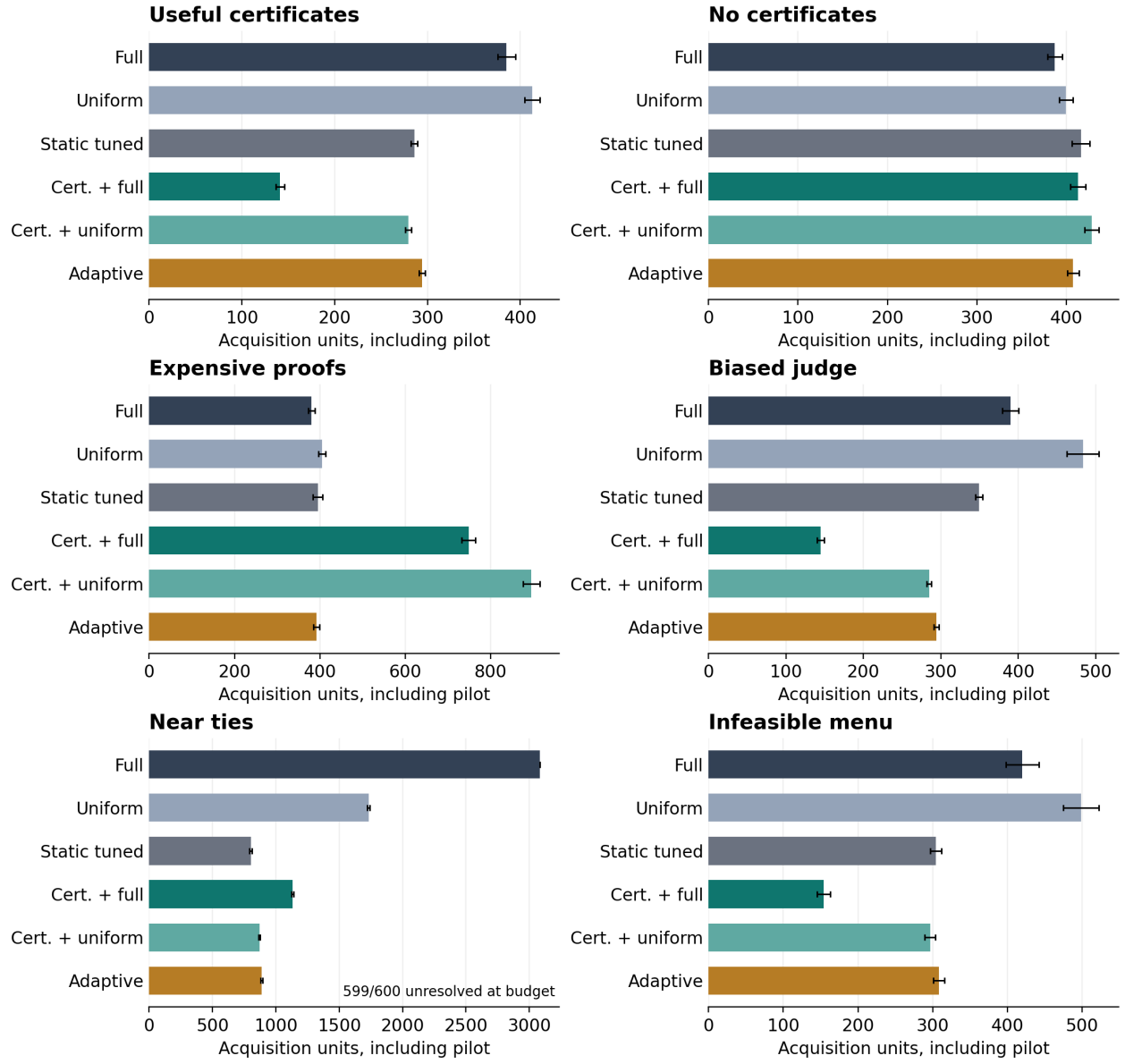


Figure 1. Synthetic acquisition cost with 95% Monte Carlo intervals. Each panel has its own horizontal scale. All proof attempts and fitted-predictor pilot costs are included. The near-tie panel contains one supported selection across 600 trials.

should not be pooled as independent operational replications. Within a 100-repetition condition-policy cell, zero events still has an approximately 3.70% Wilson upper endpoint; one event has an approximately 5.45% upper endpoint. The results neither prove perfect coverage nor estimate extremely small error rates. Recorded factor and propensity checks accompany the raw traces.

8.3 Retained first run and cost-dominance amendment

The original frozen run contained the same 3,840 synthetic trial configurations and 400 archive replay trials. Its expensive-proof static and adaptive means were 725.53 and 735.05 units, respectively. Both methods paid 320 units for 64 pilot proof attempts even though the known 5-unit attempt could only replace a 3-unit reference vector. This was an avoidable acquisition-policy defect, not an invalidity in the corrected evidence process.

The amended policy excludes that dominated action before calibration or exploration. The fresh-seed expensive-proof means become 395.41 and 392.16. These are separate-seed descriptive comparisons, not paired estimates of an amendment effect. The entire first protocol, source snapshots, trial records and replay results remain in `results/ai_v1_archive`; none of its losses or coverage failures was deleted. Its public replay also favored full auditing. The guard is specific to the declared one-case information and cost contract: it does not rule out a more expensive reusable proof that resolves other future work.

8.4 Retrospective public coding-workflow replay

We imported 1,500 published outcome records for the same 500 SWE-bench Verified cases from three mini-SWE-agent 1.7.0 workflows (Jimenez et al. 2024; SWE-bench contributors 2026). The official experiments repository is pinned to commit 40f164d5b8f1d249bf95a6df8b74b577fd8e519d; the selected run directories are dated August 7, 2025. Outcomes are the source’s executable-test resolution flags. We do not rerun benchmark containers, independently adjudicate these outcomes or claim that the archived model configurations are current.

Published model in mini-SWE-agent 1.7.0	Archive failure	After flag exclusion	Reported generation USD
GPT-5 mini	0.402	0.39957	17.73853
GPT-5 nano	0.652	0.63889	19.03759
gpt-oss-120b	0.740	0.73291	28.55953

Table 2. Descriptive failure means over the complete imported bank and sensitivity after removing all source-flagged case IDs. Historical generation costs are reported by the source, are sunk here and are unrelated to the counterfactual replay units.

The upstream git-inspection notes flag 2, 12 and 18 trajectories for the respective workflows. These are heuristic git-command flags, not proof of misconduct or of successful solution access. Their union contains 32 cases; excluding that union leaves 468 cases and the descriptive means in Table 2. The sensitivity does not establish clean-room benchmark validity. The experiment retains the flags, downloaded source hashes and complete matched case IDs.

Each replay draws independent uniform indices with replacement from the fixed archive. The statistical target is exactly its empirical mean; repeated draws are not new language-model executions. A separately sampled pilot trains the context predictor, using the source’s case-class partition,

without claiming an untouched holdout. The replay uses no certificates. It compares four policies at a risk ceiling of .55 and tolerance .12, with 100 repetitions each. The source reports a combined historical generation expense of \$65.33565194; this paper performs zero new provider calls and makes no dollar-savings claim.

Policy	Mean units [95% CI]	Mean draws	Qualified / 100
Full	561.55 [525.84, 597.25]	186.56	100
Uniform	702.37 [644.43, 760.30]	339.52	99
Static tuned	752.09 [710.11, 794.08]	222.40	100
Adaptive	730.29 [675.68, 784.91]	351.36	100

Table 3. Retrospective archive replay. Full, static and adaptive policies qualify the same lowest-failure workflow in 100/100 repetitions; uniform auditing qualifies it in 99/100 and is unresolved once. There are zero false decisions and three monitored joint-coverage failures against the empirical-bank target, one each in full, uniform and static auditing. The evidence-cost model charges 3 units per acquired label vector and includes independent pilot fitting.

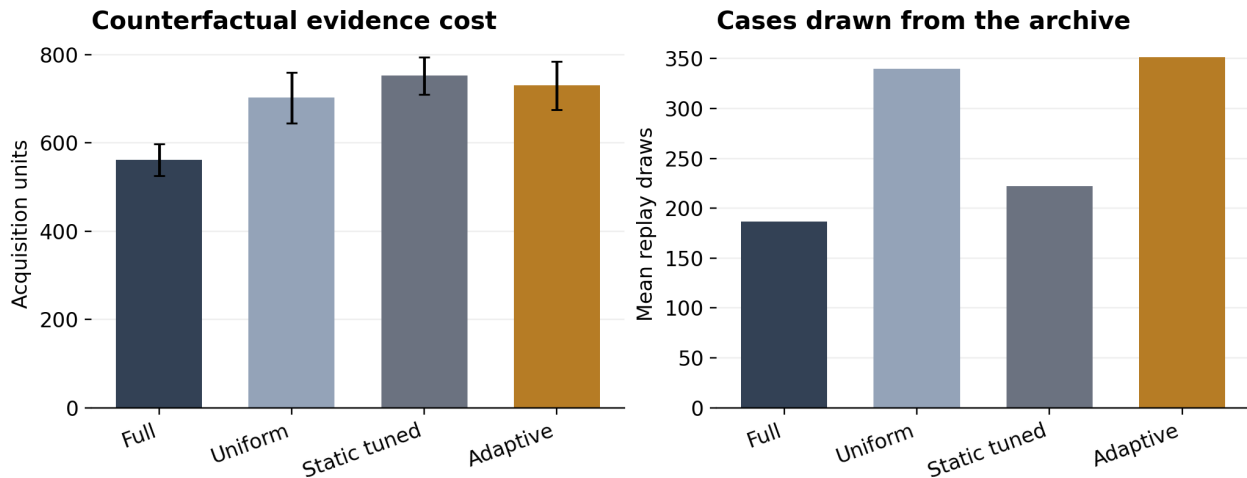


Figure 2. Public-outcome replay is a negative efficiency result under the declared acquisition model. Error bars are 95% Monte Carlo intervals for mean cost. Draw counts can revisit an archived case and do not measure new model generations.

Full auditing has the lowest mean cost, 561.55 units. Adaptive acquisition costs 730.29, uniform auditing 702.37, and static tuning 752.09. The respective mean numbers of distinct archive cases visited during confirmation are 153.42, 236.42, 229.95, and 170.73; these are smaller than replay draw counts. Selective policies buy fewer confirmation audits in some comparisons, yet pilot cost and slower evidence accumulation erase that saving. This result supports neither a new model ranking nor operational cost reduction.

8.5 Supplemental correction and dependence controls

A separate supplemental protocol was fixed locally before its draws. It is not externally preregistered and does not alter either main protocol. Two workflows have true risks (.15, .65), a failure ceiling of .40, and deliberately reversed judge predictions (.95, .05). In 100 repetitions with a 512-case cap,

the corrected procedure makes 98 correct selections, leaves 2 unresolved, and makes 0 incorrect decisions. A deliberately invalid control feeds those judge scores into inference as though they were reference labels: it selects the wrong workflow in 100/100 repetitions. The exact two-sided 95% upper error endpoint for the corrected 0/100 result is 3.62%; the invalid control’s lower endpoint is 96.38%. This demonstrates the necessity of the measurement contract under this constructed bias; the invalid control is not an admissible cheap competitor.

A dependence diagnostic holds marginal risks (.25, .45) and mean contrast -.20 fixed, and compares a shared uniform, independent uniforms and an antithetic construction. Each estimate averages 512 cases, with 100 repetitions. The single-case contrast variances follow directly from the joint Bernoulli laws; the estimator variance is that value divided by 512.

Dependence	Single-case variance	Estimator variance, theory	Estimator variance, measured
Common	0.160	0.00031250	0.00036305
Independent	0.435	0.00084961	0.00084692
Antithetic	0.660	0.00128906	0.00128521

Table 4. Pairing is advantageous for positively associated outcomes and disadvantageous for the antithetic law. Measured variances are across the 100 estimates. This diagnostic is not a complete ablation of the constrained-selection algorithm.

An independent implementation audit additionally enumerates 1,080 two-branch audit expectations; its maximum absolute conditional-mean discrepancy is 2.22e-16 in floating point. A stress test checks joint intervals after every case in 128 paths of 128 cases (16,384 checks), finding 0 paths with a coverage miss and 0 with a false decision. For the independent paths, balanced between 64 with and 64 without witnesses, a one-sided 95% zero-event upper bound for their average path-error probability is 2.31%. This bound follows from independence and the arithmetic–geometric mean inequality; it does not require identical path-error probabilities. This is a finite numerical stress test with biased proxies, antithetic outcomes and an assumed sound witness channel; it is not a formal numerical proof or live-model validation.

9 Limitations and research implications

The validity theorem and the implementation have different failure modes. A correct e-process cannot repair an invalid reference definition, a false support certificate, a contaminated holdout or a workflow changed during evaluation. The statistical support interface handles intervals and pairwise constraints; it does not certify arbitrary natural-language claims. The OCaml core is an independent inference replay checker and accepts no such imported supports in this release. A checksum detects a mismatch with a supplied record, not a maliciously fabricated record whose author recomputes the checksum.

The statistical engine is conservative in several ways. It divides an error budget across targets and regimes, uses a finite fixed mixture of bets, expands a numerical grid and may retain a large possible-feasible set. The resulting interval width can make selective auditing slower than full evaluation. A tighter construction may improve efficiency, but changing the bets or selection rule requires renewed validity and empirical checks.

The source-grounded benchmark is deliberately bounded. Programmatically generated rule packets make its reference outcomes reproducible; they do not reproduce expert interpretation of uncontrolled operational documents. Repeated variants of one source family must remain in the same split. The held-out benchmark measures performance on its declared cases, and a resampling experiment measures an explicitly empirical target. Neither automatically validates deployment on future real cases.

The empirical tracks answer different questions. The synthetic track has exact truth but assumes the proof channel. The generated source suite validates contracts using rule fixtures. The public track contains actual historical model outcomes but supports only retrospective resampling of a fixed archive, with no proof certificates, fresh model calls or clean-room guarantee. Their evidence must not be combined into a claim that live AI proof search produced the synthetic savings. The reported archive exclusion is a descriptive integrity sensitivity, not a repeated full replay on a newly established clean benchmark.

The current ablations vary certificates, their price, prediction bias, decision margins and menu feasibility. Supplemental diagnostics remove audit correction and compare matched marginal laws under different dependence. They do not constitute a complete workflow-selection ablation of pairing, test a learned proof generator, or compare full implementations of the closest published allocation algorithms. The modest 100-repetition main cells and 40-repetition supplement cannot verify rare-event reliability. Results use a fixed finite bet set and budget; alternative spending and proof-exploration policies might change the efficiency ordering. These missing comparisons limit the paper’s empirical novelty claim.

We do not establish a cost lower bound or a matching allocation theorem for shared certificates. A stronger theoretical contribution would identify a structured class where outcome-dependent joint support changes the optimal information per unit cost, and would give a computationally tractable procedure attaining an appropriate bound. Generic controlled-experiment and feedback-graph machinery already covers substantial parts of that program (Poiani et al. 2024; Russo, Song, and Pacchiano 2025). A broader novelty claim also requires full-text comparison with all close contemporary work; this release is not an exhaustive priority search.

The experiments should be read as evidence about mechanisms under their measured conditions. Useful certificates can replace work; biased predictors require correction; selective auditing can lose its advantage through variance or policy overhead; and constrained decisions can remain unresolved. A publication-level claim of operational benefit requires real workflow traces, an independently justified reference protocol, credible costs and demanding external baselines. These requirements are explicit release boundaries, not inferred from a polished interface.

10 Conclusion

CLARK Criterion provides a runnable route from frozen workflow definitions to bounded evidence, explicit costs and a supported selection or abstention. Shared certificates enter through a sound support contract; uncertain predictions enter through a separate audit-corrected process. The proofs establish the resulting conditional decision semantics. The measured study shows 63.3% synthetic savings for inexpensive exact substitution, meaningful losses when proof acquisition is expensive, persistent abstention for close or rare-event targets, and no efficiency benefit for selective auditing on the chosen public archive under the stated costs. The added OCaml core makes a restricted archive decision independently replayable with conservative numerical arithmetic and an explicit refusal on

claim mismatch. The work supports further study of the cost of reliable AI workflow decisions while retaining the distinction between demonstrated implementation behavior, conditional mathematics and unestablished deployment claims.

11 Reproducibility and availability

The release contains the inference module, benchmark and runner, experiment protocol, raw aggregate results, figure builder, unit and integration tests, and this manuscript in editable and typeset formats. A local command reproduces each reported study from its protocol. Hash receipts bind the protocol, implementation and result files used by the paper. Preparation of an archive is not evidence of a public repository publication, an arXiv submission or peer review.

From the repository root, install the declared Python dependencies and run the following commands. Fetching the public sources requires network access; the retained imported archive and byte hashes allow offline replay.

```
PYTHONPATH=src python experiments/run_ai_evaluation.py --fetch-public
PYTHONPATH=src OPENBLAS_NUM_THREADS=1 OMP_NUM_THREADS=1 \
  python experiments/run_ai_evaluation.py --workers 4
python paper_ai/refresh_results.py
python paper_ai/build.py
```

The protocol refuses silently changed inference or experiment sources. The amended protocol's canonical JSON SHA-256 is recorded in `paper_results_receipt.json`; the inference source begins 69d948a64baafec3. These canonical and source hashes are distinct from the pretty-printed protocol file's byte hash; the full values are retained in machine-readable receipts. The paper builder checks them before regenerating the figures and numeric text. `paper_results_receipt.json` binds the exact input files; `build_receipt.json` binds the PDF and native-equation Word export. Separate render receipts record all-page visual checks for the delivered hashes.

The author is responsible for the manuscript's claims, source verification and review. AI assistance was used in software development, literature synthesis and manuscript preparation. No automated drafting or internal review is represented as external peer review. Public posting, venue formatting and any venue-specific disclosure remain separate submission steps.

For the added verification path, build the independent native executable and validate an exported request:

```
(cd verification/ocaml && dune build && dune runtest)
clark-criterion verification-request examples/ai_tracebank_demo.json \
  --report decision.json --out verification-request.json
clark-criterion verify verification-request.json \
  --checker verification/ocaml/_build/default/bin/criterion_check.exe \
  --out verification-result.json
```

Here `decision.json` is a Python report produced by `clark-criterion evaluate` for that same bank. The source distribution includes the native tests, Python integration tests and separate verification receipts. The original empirical protocol remains frozen; these software checks do not rerun or relabel its statistical trials. The executable and exact request bytes are bound in the verification wrapper receipt. See `docs/OCAML_VERIFICATION.md` for toolchain prerequisites and the supported request limits.

References

- Angelopoulos, Anastasios N., Stephen Bates, Clara Fannjiang, Michael I. Jordan, and Tijana Zrnica. 2023. “Prediction-Powered Inference.” *Science* 382: 669–74. <https://doi.org/10.1126/science.ad16000>.
- Ao, Ruicheng, Hongyu Chen, Siyang Gao, Hanwei Li, and David Simchi-Levi. 2026. “Best Arm Identification with LLM Judges and Limited Human Audits.” <https://arxiv.org/abs/2601.21471>.
- Berrada, Gabrielle, Jannik Kossen, Freddie Bickford Smith, Muhammed Razzak, Yarin Gal, and Tom Rainforth. 2025. “Scaling up Active Testing to Large Language Models.” <https://arxiv.org/abs/2508.09093v2>.
- Cai, Ting, and Kirthevasan Kandasamy. 2026. “Constrained Best Arm Identification with Tests for Feasibility.” <https://arxiv.org/abs/2511.09808>.
- Csillag, Daniel, Claudio Jose Struchiner, and Guilherme Tegoni Goedert. 2025. “Prediction-Powered e-Values.” In *International Conference on Machine Learning*. <https://arxiv.org/abs/2502.04294v2>.
- Durmazkeser, Yavuz, Patrik Okanovic, Andreas Kirsch, Torsten Hoeffler, and Nezihe Merve Gürel. 2025. “Large Language Model Selection with Limited Annotations.” <https://arxiv.org/abs/2510.09418v2>.
- Forster, Yannick, Matthieu Sozeau, and Nicolas Tabareau. 2024. “Verified Extraction from Coq to OCaml.” In *Programming Language Design and Implementation*. <https://rocq-prover.org/paper/s/verified-extraction-from-coq-to-ocaml>.
- Ham, Dae Woong, Xuejun Zhao, Stefanus Jasin, and Fenghua Yang. 2026. “Human–AI-Powered Hypothesis Testing: Cost-Aware Selective AI Scoring and Sequential Human Escalation.” <https://arxiv.org/html/2609.28859v1>.
- Hardy, Michael, Ruhana Azam, Anka Reuel, Mykel Kochenderfer, and Sanmi Koyejo. 2026. “Agent Evaluation Reliability: More Tasks Won’t (Always) Fix an Agent Leaderboard.” <https://arxiv.org/abs/2610.00651>.
- Howard, Steven R., Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. 2021. “Time-Uniform, Nonparametric, Nonasymptotic Confidence Sequences.” *The Annals of Statistics* 49 (2): 1055–80. <https://doi.org/10.1214/20-AOS1991>.
- Jimenez, Carlos E., John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. “SWE-Bench: Can Language Models Resolve Real-World GitHub Issues?” In *International Conference on Learning Representations*. <https://arxiv.org/abs/2310.06770>.
- Kapoor, Sayash, Benedikt Stroebel, Zachary S. Siegel, Nitya Nadgir, and Arvind Narayanan. 2024. “AI Agents That Matter.” <https://arxiv.org/abs/2407.01502>.
- Kossen, Jannik, Sebastian Farquhar, Yarin Gal, and Tom Rainforth. 2021. “Active Testing: Sample-Efficient Model Evaluation.” In *Proceedings of the 38th International Conference on Machine Learning*, 139:5753–63. Proceedings of Machine Learning Research. <https://proceedings.mlr.press/v139/kossen21a.html>.
- Li, Jing, and Haibin Zhu. 2026. “Sequential Confidence Sets for Coverage-Constrained Conformal Model Selection.” <https://arxiv.org/html/2609.28522v1>.
- OCaml contributors. 2026. “Modules.” <https://ocaml.org/docs/modules>.
- Poiani, Riccardo, Remy Degenne, Emilie Kaufmann, Alberto Maria Metelli, and Marcello Restelli. 2024. “Optimal Multi-Fidelity Best-Arm Identification.” <https://arxiv.org/abs/2406.03033v2>.
- Russo, Alessio, Yichen Song, and Aldo Pacchiano. 2025. “Pure Exploration with Feedback Graphs.” <https://arxiv.org/abs/2503.07824>.
- SWE-bench contributors. 2026. “Official SWE-Bench Experiments Archive.” <https://github.com/SWE-bench/experiments>.

UK AI Security Institute and Meridian Labs. 2026. “Inspect: An Open-Source Framework for Large Language Model Evaluations.” <https://inspect.aisi.org.uk/>.