ADMISSION WITHOUT ANSWERS: LABEL-FREE CERTIFICATION AND EXPERIENCE LEARNING FOR LLM-BASED OPTIMIZATION MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Agents that learn from experience improve at optimization modeling by storing solved trajectories and reusing them as skills. A wrong trajectory that enters the library can be retrieved again and again, and on a stream of new problems there is no ground-truth answer to decide with. Existing learners admit trajectories by matching known optima or labels, and label-free substitutes such as execution success or agreement at one instance can admit wrong models. We introduce ADMITOR, a label-free admission gate. It generates models from three model families, runs each on the stated problem and on instances with resampled parameters, keeps the largest group of models whose optimal values agree on every instance across families, and applies a threshold fitted on solver-verified problems to accept, abstain, or escalate, with a finite-sample bound on the false-discovery rate among accepted values. Inside a state-of-the-art skill learner, ADMITOR raises candidate-level admission precision to 0.927, against 0.871 for majority vote over the host's own samples and 0.726 for execution success, and its library, the smallest of the four, reaches the highest macro accuracy over five public benchmarks, 58.4 against 54.8 for majority vote. An ablation on the same records shows that the gain comes from the accepted value being external to the learner and unanimous across families; on this stream, resampling never changed an accepted value and only reduced coverage. The false-discovery bound holds on the calibration set but not on the benchmark stream: an audit of every false certificate traces most of them to benchmark texts that omit or round the numbers needed to reproduce the labeled answer, and a label-free check of the extracted numbers against the text flags most of these cases. Code and benchmarks will be released upon acceptance.

1 INTRODUCTION

Large language models can translate natural-language descriptions of operational problems into executable optimization models, and recent agents improve further by learning from experience: solved trajectories are distilled into reusable insights, skills, or exemplars (Kong et al., 2025; Yang et al., 2026; Liang et al., 2026). Once stored, however, a wrong trajectory is no longer a single wrong answer: it can be retrieved repeatedly and affect many later decisions. Existing learners avoid this risk by admitting only trajectories that match known optima (Kong et al., 2025), are labeled against ground truth (Yang et al., 2026), or are curated by experts (Liang et al., 2026). A stream of real problems provides no such answers, and we refer to this lack of reliable answers as the label wall.

The natural label-free substitutes are unsafe. Execution success shows only that a program runs, not that it encodes the right objective, constraints, and data. Self-assessment is unreliable: models miss their own errors and prefer their own outputs (Huang et al., 2024a; Gou et al., 2024; Kamoi et al., 2024; Panickssery et al., 2024), and at the system level Xiu et al. (2026) admit memories by self-assessment and execution success and report that naive retrieval from the resulting store can reduce accuracy. On our 300-problem stream, an execution-only rule admits 878 candidate models, 241 of which disagree with the withheld answers. A useful admission rule therefore has to test how a model behaves, not whether it runs or how confident it reports itself to be.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

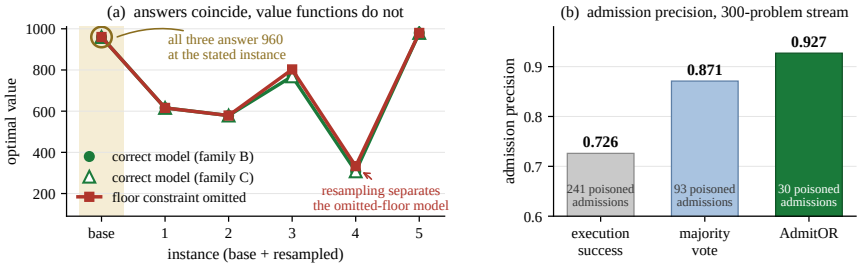


Figure 1: **(a)** Why one-point agreement is insufficient. A model omitting the floor constraint returns 960 at the stated instance, exactly as the correct models do, but its optimal value separates on resampled instances, and the cross-family agreement excludes it. **(b)** Candidate-level admission precision of three label-free judges on the same 300-problem stream, with poisoned-admission counts.

Behavior at one instance is not enough either. Consider allocating up to 150 crates across three stores with capacities of 60, unit profits of 8, 6, and 4, and a contractual floor of 20 crates per store. At the stated instance the floor is inactive: the correct model and a model that omits the floor both ship 60, 60, and 30 crates and both return 960. Comparing final objective values, whether by majority vote or against an answer key, cannot distinguish the two formulations here; their agreement is accidental and breaks once the parameters change. On a resampled instance where the third store's unit profit is negative, the floor becomes binding: the correct model still ships 20 crates to that store, the other model ships none, and the two optimal values differ (Figure 1a). The evidence we need is therefore about how a model's optimal value changes with the parameters, not about one answer.

ADMITOR (read *admitter*) builds its decision on this kind of evidence. It generates candidate models from three model families with different prompting strategies and solver stacks and runs every candidate on the stated problem and on instances resampled from a parameter domain anchored at the stated values. Candidates that describe the same problem should return the same optimal value on every instance; because not every candidate is expected to agree, the gate keeps the largest group whose values agree on every instance and requires that group to span more than one family. Proposition 1 shows that two models with different value functions differ on a region of positive measure, so independent resampling misses that region with probability that decreases geometrically in the number of usable instances. A threshold fitted on solver-verified problems then turns the strength of the agreement into ACCEPT, ABSTAIN, or ESCALATE, with the false-discovery rate (FDR) among accepted values as the quantity under control. Only accepted values are used for learning.

We evaluate the gate inside a running skill learner (Yang et al., 2026) with a collect-once, replay-many protocol: candidate models and solver logs are generated once on a 300-problem stream whose labels are withheld, and four judges replay the same records to build separate libraries, using sealed ground truth, majority vote over the host's own sampled trajectories, execution success, or ADMITOR. Candidate-level admission precision rises from 0.726 for execution success to 0.871 for majority vote and 0.927 for ADMITOR, and the ADMITOR library is the smallest and reaches the highest macro accuracy across five public benchmarks. Two further analyses use the same records: an ablation over intermediate judges shows which part of the gate produced the gain, and the preregistered false-discovery criterion, which holds on solver-verified data, fails on the benchmark stream for a reason that an audit of every false certificate finds in the benchmark texts themselves. The paper thus reports where the method performs well, which of its parts did the work, and where its assumptions fail.

Contributions. **(i) Problem.** Admission into a persistent library is the step that current experience learners leave to ground-truth labels. We argue that the right safety target for this step is the false-discovery rate among admitted items, not certainty about each one. **(ii) Method.** We design an admission procedure that compares independently generated models from several model families and tests whether their optimal values stay consistent under parameter changes, with an identifiability result and a finite-sample calibration whose transfer assumption is stated explicitly. **(iii) Protocol.** A collect-once, replay-many design holds candidate generations and solver logs fixed while changing only the judge, so that judges are compared on identical inputs. **(iv) Evidence.** ADMITOR produces 8× fewer poisoned admissions than execution success and reaches the highest downstream accu-

racy with the fewest library items, with a paired-bootstrap gain over majority vote; an ablation on the same records attributes the gain to an external, unanimous certificate and shows that resampling changed no accepted value on this stream. **(v) Measurement.** Auditing every false certificate shows that many failures come from missing or rounded numbers in the benchmark text rather than from the generated models: at least 10.9% of admitted problems cannot be answered from the printed text alone, and a check of the extracted numbers against the text flags most of these cases without labels. Code, verdicts, run ledger, and complete case packets are released.

2 Related Work

Experience learning with labeled admission. The systems closest to our setting learn reusable knowledge from solved problems, but all rely on answers for admission: insight libraries matched against known optima (Kong et al., 2025), skills distilled from ground-truth-labeled trajectories (Yang et al., 2026), and expert exemplar banks (Liang et al., 2026). AdmitOR replaces this supervision signal; we evaluate it inside one of these systems with the native ground-truth oracle as the reference (Section 4.2).

Self-verification and its limits. Intrinsic self-correction can degrade reasoning (Huang et al., 2024a), effective critique requires external tools (Gou et al., 2024), and models miss their own errors (Kamoi et al., 2024) while favoring their own outputs (Panickssery et al., 2024); Xiu et al. (2026) show the failure at the system level. Li & Hai (2026) prove that no sound and nontrivial fixed-threshold perturbation tester exists; they establish sound tests for individual models, whereas we calibrate admission across several models.

Consensus, voting, and juries. Self-consistency and its variants select the answer that recurs most often across sampled reasoning paths (Wang et al., 2023; Chen et al., 2023; Aggarwal et al., 2023). These methods, like jury-style aggregation (Liu, 2026), annotation-free routing (Niu et al., 2026; Guha et al., 2024), and cross-backbone debate (Lin et al., 2026), aggregate agreement at a single problem instance, and they select an answer rather than bound the error rate of what is accepted. Our majority-vote arm is self-consistency as the host runs it: the modal answer of the three trajectories the host samples for a problem, all from one backbone (Section 4.2). AdmitOR differs in both: agreement is measured over optimal values on resampled instances, and the output is an admission decision with a calibrated error budget. The shared-error floor of Liu (2026) motivates the use of several model families, and Zhou (2026) identifies the failure it protects against: a judge conditioned on a shown candidate scores plausibility rather than correctness, and a strict three-judge ensemble still admits 55% of the resulting false positives unless each judge first commits an answer of its own. Every family in our panel commits its own executed answer before any comparison; the extractor is the one shared stage (Section 4.3).

Benchmark reliability. The benchmarks used here (Xiao et al., 2024; Tang et al., 2024; Huang et al., 2024b; Lu et al., 2025; Yang et al., 2024) are known to contain errors. An expert audit with each case cross-validated by at least three reviewers reports rates of at least 54.0% on IndustryOR, 24.3% on ComplexOR and 23.7% on Mamo ComplexLP, attributes them to inconsistent statements, underdetermined parameters and incorrect answers, and releases cleaned versions (Xiao et al., 2025). Those audits review datasets by hand; Section 4.3 reaches the same object from the other direction: the disagreements of a label-free judge locate one specific error, a printed text from which the reference answer cannot be recovered, and quantify it as a floor on the false-discovery rate that any text-faithful certifier can reach.

Verifiers for optimization modeling. Modeling agents such as OptiMUS (AhmadiTeshnizi et al., 2024) check their own formulations by re-solving; recent verifiers, semantic checkers, and behavioral test batteries (Liu et al., 2026; Rahman et al., 2026; Zhang et al., 2025; Li et al., 2026; Lian et al., 2026) filter individual generations, and the test batteries of Lian et al. (2026) and Li & Hai (2026) perturb the instance of a single model and check its response. These methods verify or select single outputs without deciding admission into a persistent library or bounding the false-discovery rate of what is admitted, so Table 1 lists single-model perturbation testing as its own row.

Table 1: Properties targeted by each admission judge. *Executed value*: compares executed objective values rather than run status, text, or self-report. *Beyond one point*: the comparison spans resampled instances. *Budget*: targets a calibrated false-discovery rate.

Judge	Label-free	Executed value	Beyond one point	Budget
Ground-truth labels	×	✓	×	×
Execution success	✓	×	×	×
Majority vote	✓	✓	×	×
Single-model perturbation test	✓	✓	✓	×
ADMITOR	✓	✓	✓	✓

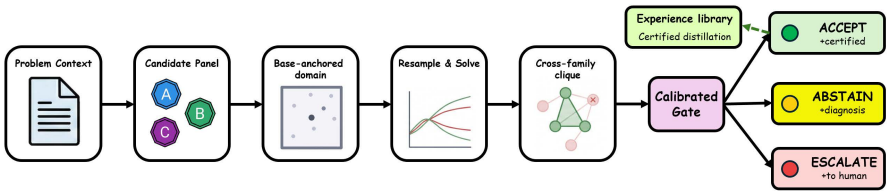


Figure 2: The ADMITOR gate. Candidates from three model families are run on instances resampled from a domain anchored at the stated problem, the largest group of candidates whose optimal values agree on every instance across families is found, and a calibrated threshold turns that agreement into a three-state decision toward $\alpha = 5\%$ with a controlled budget of 2α (Proposition 2). Accepted values relabel the host's trajectories, which are then distilled into the library.

Conformal selection. Our admission layer is finite-sample calibrated selection toward an FDR target, with split-conformal Benjamini–Hochberg (Jin & Candès, 2023) as its large-sample version, applied to a new score, cross-family agreement over resampled optimal values, and combined with policy-matched calibration because escalation makes admission adaptive (Lemma 1). Table 1 summarizes four properties of an admission judge; to our knowledge, ADMITOR is the first label-free method designed for all four, and Section 4.3 shows where the calibrated budget stops transferring.

3 THE ADMITOR GATE

Setup. The idea is that two models of the same problem should return the same optimal value whenever the parameters change. A *ticket* is a natural-language optimization problem x with stated parameter values θ_0 and a perturbation domain Θ anchored at θ_0. A candidate M is an executable program that builds and solves a model for any $\theta \in \Theta$; its optimal objective defines the *value function* $V_M(\theta)$. Two candidates are behaviorally equivalent when their value functions agree on Θ, whatever their code. The gate receives a panel $\mathcal{C} = \{M_1, \ldots, M_k\}$ with family labels $f(M_i)$ and returns ACCEPT with an accepted value $\hat{z} = V(\theta_0)$, ABSTAIN, or ESCALATE. We call an ACCEPT that carries a value at θ_0 a *certificate*; the word refers to the agreement that supports the value, not to a proof of correctness. An accept whose agreeing group formed only on resampled instances certifies nothing and admits nothing. Insufficient comparable evidence is logged as UNINFORMATIVE and maps to escalation in deployment. Section 4.2 evaluates admitted candidate models; Section 4.3 evaluates problem-level certificates, the units of the FDR target. Figure 2 and Algorithm 1 (Appendix B) summarize the five steps.

Step 1: generate a decorrelated panel. We generate candidates from three model families under different prompting strategies and solver stacks, each on a separate backend. Every consensus method has a shared-error floor, the probability that all panel members make the *same* mistake, and the floor is higher within one family because related models share training data and failure modes (Liu, 2026). Diversity across families and solver APIs lowers several sources of correlated failure but cannot detect a misinterpretation of the input text shared by all of them (Section 5).

Step 2: extract a base-anchored domain and resample. An extractor LLM reads x and returns the stated base value and a relative or absolute perturbation domain for each parameter. The domain

is anchored so that instance 0 is exactly the stated problem, and guardrails keep structural size parameters fixed so that resampled instances preserve their dimensions (Appendix B). The pilot runs a single extractor; Appendix B.1 specifies the check on the extracted numbers that Section 4.3 motivates. The harness then draws $\theta_1, \ldots, \theta_m$ independently from the domain and runs every candidate on every instance, recording the objective value and solver status of each run. An instance is *informative* when at least two candidates return a finite value; infeasible and failed runs make an instance uninformative, and the gate requires at least three informative instances before issuing a verdict.

Step 3: compare value-function traces. Two candidates agree on an instance when their objective values a and b satisfy $|a - b| \leq 10^{-4} \max(1, |a|, |b|)$, and they are consistent when they agree on every informative instance, including the stated one. Pairwise consistency defines a graph over $\mathcal{C}$, and the gate takes a maximum clique of that graph, the largest group of candidates that are all consistent with one another. If the clique spans at least two model families, the gate returns ACCEPT with the common value at θ_0; otherwise it returns ABSTAIN. With one candidate per family, as in the pilot, clique size and family coverage coincide. Each candidate outside the clique is rejected with a diagnosis naming a resampled instance on which it differs, which localizes the modeling disagreement: in the crate example, a negative unit profit activates the omitted floor constraint. Proposition 1 states when random perturbations detect such a difference.

Proposition 1. *For linear and mixed-integer models over a full-dimensional perturbation domain, the candidate value functions are piecewise polynomial in θ, and affine when only right-hand sides or only costs vary. If two candidates are not equivalent on the domain, the disagreement region contains an open set. Under continuous resampling, the probability that m independent instances all miss this region decreases geometrically with m, at a rate set by the probability measure of the disagreement region. With an agreement tolerance ε, the same bound holds for the region where the two value functions differ by more than ε whenever that region has positive measure; a modeling error whose effect on the optimal value stays within ε over the whole domain is not detectable by resampling. A formal statement, the proof, and the scope of the result for integer-valued parameters appear in Appendix A.*

Step 4: calibrate the admission threshold. Agreement is evidence rather than proof, so its strength is calibrated rather than thresholded by hand. The deployed score orders the number of families in the clique first, then the clique size, then the number of informative instances, encoded as the scalar $10 \cdot (\text{families}) + |Q| + (\text{informative})/10$ (Algorithm 1). It takes finitely many values, four of which occur in the pilot; pairwise agreement margins are recorded but do not enter the score. We fit the acceptance threshold τ on problems whose ground truth is verified *by construction*: the unchanged gate runs on a stratified sample of solver-verified synthetic instances, so every accepted certificate can be labeled true or false without benchmark data. A fixed threshold is not a valid substitute, because no sound and nontrivial fixed-threshold perturbation tester exists (Li & Hai, 2026), and the following pair licenses the selection.

Assumption 1 (Transfer)**.** *(i)* Faithful encoding*: the problem text determines the labeled instance, so the extracted base specification identifies it up to scoring tolerance. (ii)* Certificate exchangeability*: accepted deployment certificates with score at least τ are exchangeable with calibration certificates at the same score with respect to the true/false label.*

Proposition 2 (Calibrated admission)**.** *Let T be the attainable score grid ($|T| = 4$ in the pilot), p_τ the false-certificate probability among accepted certificates with score at least τ, and U_τ the level-$(1-\delta)$ Clopper–Pearson upper bound from the calibration counts. The fit selects $\tau^* = \min\{\tau \in T : \hat{p}_\tau \leq \alpha,\ U_\tau \leq 2\alpha\}$. Then (i) for each fixed τ, $\Pr(p_\tau \leq U_\tau) \geq 1 - \delta$ exactly; (ii) over the data-dependent selection, $p_{\tau^*} \leq 2\alpha$ holds simultaneously with probability at least $1 - |T|\delta$; (iii) under Assumption 1, the same bounds apply to the false-certification rate among deployment admissions at τ^*; and (iv) as the calibration null count grows, Benjamini–Hochberg over split-conformal p-values replaces the grid bound and controls the false-discovery rate, the expectation of the false-discovery proportion, at level α in finite samples under clause (ii) alone (Jin & Candès, 2023). Proof in Appendix A.*

Throughout, $\alpha = 5\%$ is the nominal selection target applied to $\hat{p}_\tau$ and $2\alpha = 10\%$ is the finite-sample *controlled* budget; the two figures should not be confused. The pilot uses $\delta = 0.05$ per threshold and also reports the 95%-simultaneous choice $\delta = 0.0125$ (Table 8). The judge swap of Section 4.2 ran with the admission rule of Step 3; the threshold was fitted afterwards on the calibration set and

applied to the same stored verdicts in Section 4.3, which also measures how far Assumption 1 is violated on the benchmark stream. Deployment adds one requirement:

Lemma 1 (Policy-matched calibration, informal)**.** *If admission is adaptive, borderline cases enter an escalation ladder that adds instances or candidates before a new decision. The guarantee then holds only when calibration uses the same ladder policy as deployment; calibration on single-pass decisions does not remain valid under escalated deployment (formal statement and proof in Appendix A).*

Step 5: certified distillation. Only certificates contribute to learning. The host's own trajectories for the problem are relabeled with the accepted value: a trajectory whose answer matches it is admitted and passed to the unmodified distillation procedure of the host, with the accepted value in place of the missing answer; the panel candidates themselves are not stored. Admission is therefore a value test against an external certificate, and Section 4.2 measures what follows from this choice.

4 EXPERIMENTS

Setup and pins. The host experience learner (Yang et al., 2026) runs the released pipeline with a single fixed backbone. The panel of the gate uses the three model families of Section 3, one candidate per family, with prompting strategies and solver stacks divided across them; the extractor is the first family's backbone. Each certification includes the mandatory stated instance and $m = 5$ resampled instances drawn under a fixed seed. The 300-problem stream consists of the first 300 problems, by index, of the training split of OptMATH (Lu et al., 2025), with labels withheld from every judge except the ground-truth reference; it shares no item with the 1,100 evaluation items at the exact, normalized, or near-duplicate level (Appendix C). Every remaining configuration choice is pinned in the release(Appendix B).

4.1 HOST REPRODUCTION AND THE MEASURING STICK

Before replacing the admission judge, we reproduce the complete host on five public benchmarks (Xiao et al., 2024; Tang et al., 2024; Huang et al., 2024b; Lu et al., 2025; Yang et al., 2024) using the released skill library and a uniform scorer, because the repository provides no scoring code. Appendices B and C give the scoring rule and per-benchmark differences. The reproduced macro accuracy is within 2.6 points of the reported result, with two findings that shape what follows. First, the evaluation labels are imperfect: on ComplexOR our pipeline is penalized for a correct answer, and correcting that single label recovers the reported score exactly (Appendix C). The protocol was fixed before any judge was evaluated, so the main results retain the published labels, and label errors count against every method, including ADMITOR. Second, the OptMATH reproduction has a known deficit of 3 to 5 points, attributed by partial reruns to solver contention and a capability gap (Appendix C); we base no central claim on its absolute scores.

4.2 THE JUDGE SWAP

For the 300-problem stream, we collect candidate generations and solver logs once. Four judges then replay the same logs, assign labels, and pass their admitted trajectories to the unmodified distillation procedure of the host: sealed ground truth, majority vote over the three trajectories the host samples per problem, execution success, and ADMITOR. Certification returns 174 ACCEPT, 114 UNINFORMATIVE, 10 ABSTAIN, and 2 error outcomes, the errors from truncated extractor output. ABSTAIN indicates value-level disagreement; none of the 114 UNINFORMATIVE cases arises from conflicting evidence, most being due to infeasible resampled instances and to candidates that never return a value (Appendix C). Figure 1b reports admission precision against the sealed vault over admitted *candidate models*, the units on which each judge acts: 878 candidates for execution success, 721 for majority vote, and 413 for ADMITOR, whose admission recall, the proportion of correct executable candidates recovered, is 0.601 against 1.0 by construction for execution success. The ADMITOR library was built with the admission rule of Step 3; the calibrated threshold of Section 4.3, applied to the same admissions, would remove 27 problems and 69 candidates, 8 of the 30 poisoned ones, for a precision of 0.936. Library sizes fall from 163 files for execution success to 101 for ADMITOR (Appendix C).

Table 2: Downstream accuracy (%, round-aware scorer) of the host with the library produced by each judge; every arm replays the same collection logs. *Macro* averages the five benchmarks equally, *Micro* weights by item over all 1100 items; Mamo.C is Mamo ComplexLP. Bold marks the best result in each column.

Judge	ComplexOR	IndustryOR	Mamo.C	OptMATH	OptiBench	Macro	Micro
Ground truth	**66.67**	31.00	52.61	**56.02**	63.14	53.89	57.18
Execution success	**66.67**	35.00	53.08	55.42	**72.40**	56.51	62.64
Majority vote	61.11	33.00	53.55	54.22	72.23	54.82	62.18
ADMITOR	**66.67**	**39.00**	**57.82**	**56.02**	72.23	**58.35**	**63.91**

Table 3: Judge ablation on the stored records, without new model calls. Certificates are problems with an accepted value, scored round-aware against the vault; admitted candidates are host trajectories matching the accepted value, scored under the host's equality rule, with recall over the 637 vault-correct trajectories. Panel judges use the three family candidates at the stated instance only.

Judge	Certificates	Wrong	Admitted	Poisoned	Precision	Recall
Execution success	–	–	878	241	0.726	1.000
Majority vote over host samples	254	38	721	93	0.871	0.986
Panel: any two families agree at θ_0	245	38	610	45	0.926	0.887
Panel: all three families agree at θ_0	200	26	516	32	0.938	0.760
ADMITOR, admission rule of Step 3	170	29	413	30	0.927	0.601
ADMITOR, calibrated τ	138	22	344	22	0.936	0.505

Table 2 answers the judge-swap question. The clean comparison is among label-free judges on the same candidates: ADMITOR matches or exceeds majority vote on all five benchmarks and adds 3.5 macro points while using the smallest library. The ground-truth arm is not an upper bound, because its downstream result also depends on retrieval and skill selection. Intervals come from a paired stratified bootstrap on the macro scale (Appendix C). Against majority vote the gain is $+3.53$ points, with a 95% interval of $[+0.87, +6.68]$; repeating the comparison on the item set defined by the sensitivity analysis below leaves it at $+3.49$, $[+0.78, +6.73]$, so the preregistered criterion K2 is met on both bases. Against execution success the macro gain of $+1.84$ has interval $[-0.27, +3.96]$ and does not exclude zero, and three of the five benchmarks are at or near parity; what we claim over execution success is therefore equal accuracy with roughly one-third fewer items and an audit trail, not higher accuracy.

Two qualifications are necessary. First, on OptiBench the ground-truth arm fails at *skill selection* rather than modeling on 85 of 605 items; the protocol counts these as errors in every arm, and excluding them raises ground-truth macro accuracy to 55.95, still below ADMITOR, with the corrected intervals above using that item set (Appendix C). Second, downstream accuracy also depends on retrieval: on OptiBench, which holds 55% of the items, the label-free arms differ by 0.2 points while their libraries differ by 62 files, and the host selects from 19 distinct files in the ADMITOR arm against 30 for execution success, one file receiving 45% of the ADMITOR selections (Appendix C). Library size is therefore not what separates the arms there; the differences arise on IndustryOR and Mamo ComplexLP.

What produced the gain. The judge swap shows that the gate improves on majority vote, but not which of its parts is responsible. Three further judges that separate the parts are computable from the stored verdicts: agreement of any two panel families at the stated instance, unanimity of all three at the stated instance, and the calibrated rule of Section 4.3. Table 3 reports all six judges on the same records, and two facts follow. First, the gain over majority vote comes from the accepted value being external to the learner: panel unanimity at the stated instance alone reaches precision 0.938. The host's own modal answer admits at least two trajectories whenever it is wrong, whereas an external value admits a wrong trajectory only when host and panel err on the same value; the 30 poisoned admissions of the deployed arm arise on only 15 problems. Second, resampling changed no accepted value. The calibrated gate admits a strict subset of the problems on which the panel is unanimous at the stated instance, with the same accepted value on all 138, and the 62 problems it withholds carry 4 wrong values and 58 correct ones: 57 are withheld for want of three informative instances, four because a candidate failed on perturbed instances, and one because the values diverged, with

a correct stated-instance value in all five. Under the pilot's admission rule this is the expected shape. A model whose omitted constraint is inactive at the stated instance returns the correct value there, so admission by value cannot register the model-level detections that resampling provides, and the judge swap measures the gate as a label-free value certifier. The lost coverage comes from the extracted domains, not from unlucky draws: 27% of all resampled candidate-instance pairs are infeasible, and re-drawing the 57 withheld problems with solvability screening recovers 8, each with the same accepted value (Appendix C).

The family-by-outcome matrix (K4). The remaining preregistered check concerns the panel itself: if the three arms failed in the same way, several families would add little over one. Their failure profiles differ ($\chi^2 = 30.96$, $\mathrm{df} = 8$, $p = 1.4 \times 10^{-4}$), and the difference survives restriction to candidates that reached value comparison ($\chi^2 = 8.75$, $\mathrm{df} = 2$, $p = 0.013$), although execution failures, which track the solver stack, carry most of the full statistic. The matrix and the limits of its reading appear in Appendix C.

4.3 Calibrated admission: the criterion that fails

We fit the calibrated admission layer as Section 3 specifies: the unchanged gate runs on a stratified sample of 150 solver-verified NANO-CO instances (Yang et al., 2026) under the single-pass policy that deployment uses, so the requirement of Lemma 1 is met by construction. The gate returns 103 accepts, 64 of them three-family cliques. The deployed score takes only two bands, $22.x$ for two-family and $33.x$ for three-family cliques, so the selected $\tau = 33.3$ means exactly that all three families agree and at least three instances are informative, and the four attainable thresholds differ only in the informative count. Among the 63 value-bearing three-family certificates one is false: $\hat{p} = 1.6\%$, with upper bound 7.3% at $\delta = 0.05$ and 9.7% at $\delta = 0.0125$, the 95%-simultaneous choice over the grid, so the rule of Proposition 2 holds under both choices and $\tau = 33.3$ is its minimum (Table 8). Replaying this rule on the 170 value-bearing verdicts of Section 4.2 admits 138 cases, which we score out of sample against the sealed vault.

The preregistered criterion fails, and we report it without modification (Table 8 and Appendix C). Under the uniform scorer, 22 of 138 admitted certificates disagree with the sealed vault, a realized proportion of 15.9% with a 95% upper bound of 22.0%; the host's stricter equality rule finds 26, and the two sets overlap in 21 cases. The realized proportion exceeds the 5% target and its upper bound exceeds the $2\alpha = 10\%$ budget, so the criterion fails under both readings. Tightening as the protocol permits does not recover the target: across the four thresholds the realized proportion stays between 14.7% and 16.5%, and sixteen of the twenty-two carry *full* evidence, all three families agreeing on every sampled instance, so no threshold on the amount of agreement separates them (Figure 4a in Appendix C). More resampling from the same specification cannot show whether that specification represents the source data, and the audit below locates the defect there.

We audit all twenty-two under a protocol that assumes neither value correct and requires every verdict to cite the text against a code line or an explicit derivation (Appendix C); the audit is the authors' own, and the released packets and per-case attribution file allow re-audit. It changes our initial taxonomy (Table 6 and Figure 4b). We expected gate defects, in which the agreeing models omitted a stated constraint, and legitimate alternative readings of ambiguous text; the audit finds no case in either category. Two cases are label errors, both re-derived independently of the gate, and in both the published label is *below* the true minimum of the stated minimization problem and cannot be attained by any feasible solution.

The remaining twenty cases share a root cause that was not preregistered: *the problem text is not a faithful encoding of the labeled instance*. In fifteen cases, the printed text omits decisive data, typically a cost or demand matrix. In five cases, parameters are printed at insufficient precision, so the optimum of the stated text differs from the label by more than the scoring tolerance. One case is diagnosable from the sign of the objective alone: the accepted value is *negative*, which is impossible under any nonnegative cost matrix, so the extractor must have supplied entries that the text never printed, while the magnitude of the label is consistent with a real instance of the stated form (Appendix C). We call this category (d): the text does not determine the label.

Category (d) is a property of the benchmark, not a solver failure. At least fifteen of the 138 admitted problems, 10.9%, cannot be answered from the printed text by any method. The rate is measured

on the admitted subset, where a false-discovery rate is defined, and it lower-bounds the measurable FDR of any system that, like ours, conditions its candidates on a single extraction of the printed text and admits a similar set. The estimate is conservative: only the disagreements were audited, and the five truncated-precision cases are excluded. Expert audits report aggregate error rates for these benchmarks without separating the errors behind them (Xiao et al., 2025); to our knowledge this category has not previously been isolated, nor quantified as a floor on the attainable false-discovery rate.

These failures reveal a limitation of consensus-based verification. All three families solve a common base specification produced by one extractor, so when the text omits a matrix the extractor fills it and the three independent families agree on the same fabricated instance; cross-family agreement certifies the extracted instance, not the one the benchmark author intended. NANO-CO texts are generated from their instances, so clause (i) of Assumption 1 holds there by construction, and the benchmark stream violates exactly that clause: the failure is a measured violation of a stated assumption, not a defect in Proposition 2. The remedy therefore has to act on the extracted numbers before any model is generated. Agreement between independent extractors, the obvious candidate, does not work at this granularity: on the 22 audited failures and 22 concordant controls, three extractor families never once agreed on parameter names, and after aligning parameters by shape and value, disagreement was as common on faithful texts, 21 of 22, as on unfaithful ones, 18 of 22. A simpler check does separate them: every base value in the extracted specification should be printed in the text. Applied after the fact to the pilot's stored specifications, this numeric-coverage check flags 9 of the 15 missing-data cases, none of the 5 truncated-precision cases or the 2 label errors, none of the 116 concordant admissions, and none of the 148 calibration specifications (Appendix B.1); it is the check we specify for deployment.

The cost of the gate is predictable: relative to majority vote over the same panel, it adds one extraction call per problem and six solver runs per candidate, and on problems of this size the solver runs cost far less than one model call; complete call-level logs accompany the release.

5 Limitations

First, the gate certifies agreement across independently derived behaviors, not the intended meaning of the problem: a misinterpretation shared by every family survives any amount of resampling, which is why the target is the false-discovery rate over admitted items rather than a guarantee for each one.

Second, higher admission precision reduces coverage: the gate withholds a verdict on a substantial fraction of the stream, mostly for want of informative instances, and the withheld values were mostly correct; the cause lies in the extracted domains, which yield infeasible instances for most draws. Third, the pilot admits host trajectories by matching the accepted value, so the model-level evidence of resampling is not used at admission; applying the behavioral test to the host's own trajectories would require parameterized code, which the host does not produce. Fourth, the guarantee of Proposition 2 is conditional on Assumption 1, whose violation Section 4.3 measures, and an escalation policy other than the calibrated one weakens it further (Lemma 1). Fifth, the host uses one generation backbone, and the intermediate judges of Table 3 were evaluated at admission but not downstream, because the host backbone was withdrawn by its providers during the study, and we do not evaluate a single-model selection judge, the configuration whose false-positive rate Zhou (2026) measures at 0.719.

6 Conclusion

AdmitOR decides whether a solved trajectory should enter a persistent library before the agent reuses it. In the controlled judge swap it reaches the highest candidate-level admission precision and downstream macro accuracy with the smallest library, and the ablation on the same records attributes the gain to an accepted value that is external to the learner and unanimous across model families. The benchmark-stream experiment clarifies when the method succeeds and when it fails: calibrated FDR does not transfer when benchmark texts omit or round the numbers needed to reproduce the labeled answer, and a check of the extracted numbers against the text, which flags most of these cases without labels, must come before any model-level evidence.

AI USE STATEMENT

We used generative AI tools for the following tasks. In the required-disclosure categories: implementing analysis code used to compute reported statistics, providing feedback on experimental design, and assisting in the interpretation of experimental results. In the recommended-disclosure categories: editing the manuscript for readability, and editing, refactoring, and debugging research code. We did not use generative AI tools to generate data or to alter any recorded experimental output; all reported quantities are computed by the released code from the released logs, and the analysis scripts were validated against reconciliation checks that reproduce previously published values before any new number was accepted. All AI-assisted code was executed and verified by the authors, and all AI-assisted text was reviewed by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam. Let's sample step by step: Adaptive-consistency for efficient reasoning and coding with LLMs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Ali AhmadiTeshnizi, Wenzhi Gao, and Madeleine Udell. OptiMUS: Scalable optimization modeling with (MI)LP solvers and large language models. In *International Conference on Machine Learning (ICML)*, 2024.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. Universal self-consistency for large language model generation. *arXiv preprint arXiv:2311.17311*, 2023.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations (ICLR)*, 2024.

Neel Guha, Mayee F. Chen, Trevor Chow, Ishan S. Khare, and Christopher Ré. Smoothie: Label free language model routing. *Advances in Neural Information Processing Systems*, 37:127645–127672, 2024.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *International Conference on Learning Representations (ICLR)*, 2024a.

Xuhan Huang, Qingning Shen, Yan Hu, Anningzhe Gao, and Benyou Wang. Mamo: A mathematical modeling benchmark with solvers. *arXiv preprint arXiv:2405.13144*, 2024b.

Ying Jin and Emmanuel J. Candès. Selection by prediction with conformal p-values. *Journal of Machine Learning Research*, 24(244):1–41, 2023.

Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. When can LLMs actually correct their own mistakes? A critical survey of self-correction of LLMs. *Transactions of the Association for Computational Linguistics*, 12, 2024.

Minwei Kong, Ao Qu, Xiaotong Guo, Wenbin Ouyang, Chonghe Jiang, Han Zheng, Yining Ma, et al. AlphaOPT: Formulating optimization programs with self-improving LLM experience library. *arXiv preprint arXiv:2510.18428*, 2025.

Haifeng Li and Mo Hai. Falsification-based verification of LLM-generated optimization models: Sound test batteries and their detection limits. *arXiv preprint arXiv:2607.16646*, 2026.

Zhong Li, Zihan Guo, Xiaohan Lu, Juntao Wang, Jie Song, Chao Shen, Jiageng Wu, and Mingyang Sun. OptArgus: A multi-agent system to detect hallucinations in LLM-based optimization modeling. *arXiv preprint arXiv:2605.11738*, 2026.

Junbo Jacob Lian, Yujun Sun, Huiling Chen, Chaoyu Zhang, Hanzhang Qin, and Chung-Piaw Teo. ReLoop: Structured modeling and behavioral verification for reliable LLM-based optimization. *arXiv preprint arXiv:2602.15983*, 2026.

Kuo Liang, Yuhang Lu, Jianming Mao, Shuyi Sun, Chunwei Yang, Congcong Zeng, Xiao Jin, Hanzhang Qin, Ruihao Zhu, and Chung-Piaw Teo. Large-scale optimization model auto-formulation: Harnessing LLM flexibility via structured workflow. *arXiv preprint arXiv:2601.09635*, 2026.

Jianghao Lin, Zi Ling, Chenyu Zhou, Tianyi Xu, Ruoqing Jiang, Zizhuo Wang, and Dongdong Ge. From soliloquy to agora: Memory-enhanced LLM agents with decentralized debate for optimization modeling. *arXiv preprint arXiv:2604.25847*, 2026.

Haoyang Liu, Jie Wang, Boxuan Niu, Xiongwei Han, Yian Xu, Mingxuan Ye, Zijie Geng, et al. Opt-Verifier: Unleashing the power of LLMs for optimization modeling via dual-side verification. *arXiv preprint arXiv:2605.29556*, 2026.

Ning Liu. LLMs as a jury: Cross-model consensus can outperform process reward models for LLM reasoning. *arXiv preprint arXiv:2607.10139*, 2026.

Hongliang Lu, Zhonglin Xie, Yaoyu Wu, Can Ren, Yuxuan Chen, and Zaiwen Wen. OptMATH: A scalable bidirectional data synthesis framework for optimization modeling. *arXiv preprint arXiv:2502.11102*, 2025.

Tianyi Niu, Justin Chen, Genta Indra Winata, Shi-Xiong Zhang, Supriyo Chakraborty, Sambit Sahu, Yue Zhang, Elias Stengel-Eskin, and Mohit Bansal. Routing with generated data: Annotation-free LLM skill estimation and expert selection. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32441–32466, 2026.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. LLM evaluators recognize and favor their own generations. In *Advances in Neural Information Processing Systems*, 2024.

Sumaya Abdul Rahman, Seckhen Ariel Andrade Cuellar, Ghani Raissov, and Mohammad Raza. VeriSimpl: Robust optimization modeling from natural language using simplification-based verification. *arXiv preprint arXiv:2607.20474*, 2026.

Zhengyang Tang, Chenyu Huang, Xin Zheng, Shixi Hu, Zizhuo Wang, Dongdong Ge, and Benyou Wang. ORLM: Training large language models for optimization modeling. *arXiv preprint arXiv:2405.17743*, 2024.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Ziyang Xiao, Dongxiang Zhang, Yangjun Wu, Lilin Xu, Yuan Jessica Wang, Xiongwei Han, Xiaojin Fu, Tao Zhong, Jia Zeng, Mingli Song, and Gang Chen. Chain-of-experts: When LLMs meet complex operations research problems. In *International Conference on Learning Representations (ICLR)*, 2024.

Ziyang Xiao, Jingrong Xie, Lilin Xu, Shisi Guan, Jingyan Zhu, Xiongwei Han, Xiaojin Fu, WingYin Yu, Han Wu, Wei Shi, Qingcan Kang, Jiahui Duan, Tao Zhong, Mingxuan Yuan, Jia Zeng, Yuan Wang, Gang Chen, and Dongxiang Zhang. A survey of optimization modeling meets LLMs: Progress and future directions. *arXiv preprint arXiv:2508.10047*, 2025.

Xianchao Xiu, Jianhao Li, Huangyue Chen, and Wanquan Liu. OptGraph: Large language models enhanced evolutionary optimization via graph retrieval-augmented generation. *arXiv preprint arXiv:2607.27918*, 2026.

Haochen Yang, Ke Zhao, Mengyuan Ma, Xingyu Lu, Xiangfeng Wang, and Hong Qian. OptSkills: Learning generalizable optimization skills from problem archetypes via cluster-based distillation. *arXiv preprint arXiv:2605.29829*, 2026.

Zhicheng Yang, Yiwei Wang, Yinya Huang, Zhijiang Guo, Wei Shi, Xiongwei Han, Liang Feng, Linqi Song, Xiaodan Liang, and Jing Tang. OptiBench meets ReSocratic: Measure and improve LLMs for optimization modeling. *arXiv preprint arXiv:2407.09887*, 2024.

Yansen Zhang, Qingcan Kang, Yujie Chen, Yufei Wang, Xiongwei Han, Tao Zhong, Mingxuan Yuan, and Chen Ma. SAC-Opt: Semantic anchors for iterative correction in optimization modeling. *arXiv preprint arXiv:2510.05115*, 2025.

Chenyu Zhou. More convincing, not more correct: Self-play reward hacking of reference-free LLM judges. *arXiv preprint arXiv:2607.05904*, 2026.

A PROOFS

A.1 IDENTIFIABILITY BY RESAMPLING

Assumptions. **(A1)** The perturbation domain $\Theta \subset \mathbb{R}^d$ is compact and full-dimensional, and the resampling law P has a density bounded below by $\rho > 0$ times Lebesgue measure λ on Θ. **(A2)** Each candidate M is solver-exact for the model its code constructs. On input θ, it returns $V_M(\theta) = \min_{x \in X_M} c_M(\theta)^\top x$ subject to $A_M x \le b_M(\theta)$, where $c_M(\cdot)$ and $b_M(\cdot)$ are affine in θ, the constraint matrix A_M does not depend on θ, and X_M imposes integrality on a subset of coordinates. **(A3)** Over Θ, each candidate has finitely many optimal integer patterns. Bounded integer variables and rational data guarantee this condition. In addition, V_M is finite on Θ. The harness discards infeasible or unbounded draws as uninformative, so each verdict is conditional on finiteness.

Lemma 2 (Piecewise-polynomial structure)**.** *Under (A2) and (A3) there is a finite partition of Θ into relatively open, full-dimensional cells $C_1, \dots, C_K$ (plus a Lebesgue-null boundary set) such that on each cell V_M coincides with a polynomial of degree at most two in θ, affine in θ whenever the perturbation touches only b_M or only c_M.*

Proof. Fix an integer pattern z and consider the LP obtained by fixing the integer coordinates at z. Parametric linear programming partitions the parameter space into finitely many polyhedral critical regions. Within each region, an optimal basis B remains fixed, and the LP value is $c_M(\theta)^\top x_B(\theta)$. Because $x_B(\theta)$ is affine in $b_M(\theta)$, it is also affine in θ. Since $c_M(\theta)$ is affine in θ, their product is a polynomial of degree at most two. It reduces to an affine function when only one block varies. By (A3), the MILP value is the pointwise minimum of finitely many such functions, one for each integer pattern feasible at θ. Pattern feasibility is a polyhedral condition because it is affine in $b_M(\theta)$. A common refinement of the critical regions and the loci at which the minimizing pattern changes yields the finite cell partition after excluding the null set of cell boundaries. □

Proposition 1 (formal). *Let M_1, M_2 satisfy (A1) through (A3), write $\Delta(\theta) = V_{M_1}(\theta) - V_{M_2}(\theta)$, and let $D = \{\theta \in \Theta : \Delta(\theta) \neq 0\}$. Then exactly one of the following holds. (i) $\lambda(D) = 0$: the two candidates agree P-almost surely, deliver the same certified value on almost every instance, and any disagreement is confined to a Lebesgue-null set. (ii) $\lambda(D) > 0$: then $p := P(D) \ge \rho\,\lambda(D) > 0$, and for i.i.d. draws $\theta_1, \dots, \theta_m \sim P$,*

$$\Pr\big[\Delta(\theta_j) = 0 \text{ for all } j \le m\big] = (1-p)^m \le e^{-pm}.$$

Moreover, with an agreement tolerance $\varepsilon > 0$ the same bound holds with D replaced by $D_\varepsilon = \{\theta : |\Delta(\theta)| > \varepsilon\}$ whenever $\lambda(D_\varepsilon) > 0$; if $\lambda(D_\varepsilon) = 0$, the two candidates are indistinguishable at tolerance ε and no amount of resampling separates them.

Proof. Apply Lemma 2 to both candidates and take a common refinement. The restriction of Δ to each full-dimensional cell C_k is then a polynomial q_k. If every q_k is identically zero, Δ vanishes outside the null boundary set, and case (i) holds. Otherwise, some q_k is a nonzero polynomial on the full-dimensional cell C_k. The zero set of a nonzero polynomial within C_k has Lebesgue measure zero. Therefore, $\lambda(D \cap C_k) = \lambda(C_k) > 0$, which establishes the premise of case (ii). Independence gives the probability bound directly. By (A1), each draw belongs to D with probability $P(D) \ge \rho\,\lambda(D)$, so all m draws miss D with probability $(1-p)^m$. For the tolerance statement, D_ε differs from $\bigcup_k (\{|q_k| > \varepsilon\} \cap C_k)$ by a null set, and each piece of this union is open because q_k is continuous on the open cell C_k. Hence $\lambda(D_\varepsilon) > 0$ exactly when some piece is nonempty, in which case $P(D_\varepsilon) \ge \rho\,\lambda(D_\varepsilon) > 0$ and the product bound applies with p replaced by $P(D_\varepsilon)$. The zero-set argument says nothing about $\lambda(D_\varepsilon)$: a nonzero polynomial can stay within ε on all of Θ, and then $\lambda(D_\varepsilon) = 0$ and the detection probability is zero. □

Remarks. (1) The dichotomy defines the certification semantics. Candidates that differ only on a null set are behaviorally indistinguishable and certify the same value almost surely. Thus, equivalence under the gate is almost-everywhere equivalence. (2) The base instance θ_0 is not sampled. Instance 0 is mandatory and evaluated deterministically, so a disagreement at the stated instance is detected with probability one, independently of the bound. (3) The detection power is $1 - (1-p)^m \ge 1 - e^{-pm}$. Increasing m through escalation increases this quantity. (4) If perturbations enter the constraint matrix, the cell-wise values become rational functions. The zero-set

argument still applies to nonzero real-analytic functions. The dichotomy is therefore not restricted to affine perturbations, although our analysis does not rely on this extension. (5) The result assumes a full-dimensional continuous domain. In the pilot, 173 of the 298 extracted specifications contain at least one integer-valued parameter, and the sampler draws such parameters continuously and rounds them. The guarantee therefore applies to the continuous coordinates, conditional on the rounded ones; two inequivalent models can agree on every lattice point of an integer coordinate, and we make no claim there.

A.2 POLICY-MATCHED CALIBRATION

Setup. A *policy* π is a measurable map from the complete record ω of a problem to a final admission score. The record includes the text, panel randomness, and instance randomness, all of which are internal to that problem. We write the score as $S = g_\pi(\omega) \in \mathbb{R}$. An escalation ladder is a policy in which borderline intermediate scores trigger additional instances or candidates before the final score is emitted, with a cap of K rounds. Calibration draws $\omega_1, \ldots, \omega_n$ i.i.d. from a by-construction law Q that includes *null* records observed with certainty. These null records are accepted certificates for solver-verified calibration problems whose certified values contradict the verified answers. Deployment evaluates test problems whose null records are exchangeable with the calibration nulls. The gate accepts a test problem when the conformal p-value computed from the calibration scores passes the selection procedure at level α (Jin & Candès, 2023).

Proof of Proposition 2. (i) is the defining exactness of the Clopper–Pearson interval for a binomial proportion: for fixed τ the selected calibration certificates are, under the sampling model, independent Bernoulli trials with parameter p_τ, and U_τ is constructed so that $\Pr(p_\tau > U_\tau) \le \delta$ with no asymptotic step. (ii) The selection τ^* is measurable with respect to the calibration data, so the event $\{p_{\tau^*} > 2\alpha\}$ is contained in $\bigcup_{\tau \in T}\{U_\tau \le 2\alpha \text{ selected while } p_\tau > 2\alpha\} \subseteq \bigcup_{\tau \in T}\{p_\tau > U_\tau\}$, whose probability is at most $|T|\delta$ by (i) and the union bound. The factor $|T|$ is the entire price of the data-dependent selection; the pilot grid has four points. (iii) Under clause (ii) of Assumption 1, deployment certificates at score $\ge \tau^*$ and calibration certificates at the same score are exchangeable with respect to the true/false label, so their false rates share the parameter bounded in (ii); clause (i) is the mechanism premise under which the extracted specification, and hence the score, is a function of the intended instance. (iv) With a growing null pool, the conformal p-value of a test certificate is uniformly valid under exchangeability alone, and Benjamini–Hochberg over these p-values controls the false-discovery rate, the expectation of the false-discovery proportion, at level α in finite samples (Jin & Candès, 2023); the grid certificate is the small-null surrogate the pilot deploys because the p-value floor $1/(n_{\text{null}} + 1)$ with ten nulls cannot clear $\alpha = 0.05$. The policy-matching requirement of the formal lemma below applies to both instruments; the pilot's grid certificate inherits it exactly as the large-sample conformal form does. □

Lemma 1 (formal). *(i) If calibration and deployment apply the* same *policy π, the calibration scores $g_\pi(\omega_1), \ldots, g_\pi(\omega_n)$ and each null test score $g_\pi(\omega')$ are i.i.d., the conformal p-values are valid, and the selection procedure controls FDR at level α. (ii) If calibration uses a single-pass policy π_0 while deployment uses a ladder $\pi \ne \pi_0$, the guarantee can fail: for every K there exist score laws and ladders for which the null-acceptance probability is $1 - (1-\alpha)^K$ against a nominal α, a factor that grows with K and is bounded by $1/\alpha$.*

Proof. (i) The function g_π is a fixed measurable map applied to i.i.d. records. The calibration-null and test-null scores are therefore exchangeable. The validity of the conformal p-values and FDR control of the selection procedure follow from the standard split-conformal argument (Jin & Candès, 2023). Two conditions are essential: π uses only the record of the current problem, with no cross-problem adaptivity, and the same map is applied during calibration and testing. (ii) Take null intermediate scores i.i.d. $\text{Unif}[0,1]$ across ladder rounds, let τ be the $(1-\alpha)$-quantile calibrated under π_0 (one round), and let the deployed ladder re-score any below-threshold problem up to K times, accepting on the first exceedance. A null problem is then accepted with probability $1 - (1-\alpha)^K$, which approaches $K\alpha$ for small α and 1 as K increases, while the nominal budget remains α. This is a statement about the null-acceptance probability, the type I rate; the false-discovery rate among admissions rises with it unless the acceptance probability of correct certificates rises in the same proportion, which a ladder that re-scores only borderline cases does not guarantee.

Generating calibration trajectories under the deployed ladder and recalibrating τ_π restores case (i) because the ladder is again a fixed map applied identically during calibration and deployment. □

Remark. The counterexample reflects a common implementation pattern. Retry-until-agreement escalation takes a maximum over dependent scores near the threshold. Operationally, the same ladder used in deployment must also generate the calibration trajectories.

B PROTOCOL REGISTRATION AND SCORING

Extraction guardrails. The validator applied to every extracted specification reverts conflicting ranges to relative perturbations, holds structural size parameters fixed so that resampled instances preserve their dimensions, and widens degenerate domains with a warning. Under the data-layer stage of Appendix B.1, a specification whose base values are not printed in the text is emitted as a data-layer diagnosis rather than a verdict.

Model and solver pins. The host runs with a single fixed backbone, DeepSeek-V3.2 in non-thinking mode at temperature 0. The only modifications to the host are failure handling and call logging, both documented in the release. The panel of the gate uses DeepSeek-V3.2, GPT-5.4, and Claude Sonnet 4.6, one candidate per family at temperature 0. The first and third families use direct modeling, whereas the second uses a structured strategy. Solver stacks are divided between Pyomo with HiGHS and `gurobipy`. The extractor is the first family's backbone at temperature 0 with an 8,192-token output limit; candidate generation uses a 3,000-token limit. The second and third family names were set through run-time environment variables, and the code default for the second family was aligned with the run afterwards. A response-side audit of all 60,000 host calls finds two model identifiers, `deepseek-v3.2` (97.6%) and `deepseek-v3-2-251201` (2.4%), the second being a hosting provider's dated tag for the same December 2025 release; no host call was served by another model, and a per-item join shows no association between the dated tag and either selection failures or accuracy. Response identifiers were not logged for the extractor and candidate calls, which are pinned at request time.

Preregistered decision criteria. The following criteria were fixed before collecting admission or evaluation data, and we report the outcomes without modification. K1: if the best label-free arm achieves less than 70% of the macro accuracy of the ground-truth arm, we withdraw the mechanism claim and reposition the work as selective prediction. K2: if the gate does not outperform majority vote downstream, we withdraw the consensus-mechanism claim and retain only the efficiency contribution. K3: calibration fails if the realized false-discovery rate exceeds 5% in point estimate or 10% at the 95% upper bound. One rerun with a stricter threshold is allowed and must be reported. K4: a nearly uniform family-by-error matrix weakens the decorrelation argument and must be reported as such. Sections 4.2 and 4.3 report K1–K3. Section 4.2 reports K4, and Table 5 provides the full matrix extracted from the released certification verdicts.

Certification pseudocode. Algorithm 1 specifies the complete certification procedure for one problem. Section 3 describes each step. In the judge swap of Section 4.2 the threshold test on line 9 is absent: every clique spanning two families with a value at θ_0 is accepted, and τ is applied to the stored verdicts in Section 4.3.

Scoring rule. One released scorer evaluates every benchmark and experimental arm. A prediction is correct if it matches the published label within relative tolerance 10^{-4} or if rounding the prediction to the number of decimal places printed in the label reproduces that label exactly. Labels are frequently rounded. For example, an exact solver optimum of 10.3333 is correct when the printed label is 10.33. Predictions that cannot be parsed count as errors. We also release stricter (10^{-6}) and looser (10^{-2}) scoring tiers for sensitivity analysis.

Candidate-generation rules. We developed the panel prompt once on a five-problem development split and fixed it before data collection. The Pyomo model object is named `model` and is never reused as a loop variable. The prompt avoids reserved component names and blanket exception handling. A canonical HiGHS solve-and-extract template defines success by the availability

Algorithm 1 ADMITOR certification of one problem P

1: extract base-anchored parameter spec θ_0 from P; validate and freeze structural sizes
2: draw perturbed instances $\theta_1, \ldots, \theta_m$ around θ_0 (seeded); instance 0 is P itself
3: generate candidate programs $C_1, \ldots, C_k$ from distinct model families and strategies; probe each on θ_0, one repair round on failure
4: **for** each candidate C_i and instance θ_j **do**
5: solve; record objective $V_i(\theta_j)$ and solver status
6: **end for**
7: keep instances on which at least two candidates return a finite value; if fewer than three remain, return UNINFORMATIVE (escalation in deployment)
8: form the agreement graph on value-function agreement across the informative instances; find a maximum clique Q; score $s = 10 \cdot |\{f(M) : M \in Q\}| + |Q| + n_{\text{inf}}/10$
9: **if** Q spans at least two families and $s \geq \tau$ **then**
10: ACCEPT: certify $V_Q(\theta_0)$; relabel the host's trajectories with it
11: **else**
12: ABSTAIN with an instance-level diagnosis for each excluded candidate
13: **end if**

of an objective value rather than by a status-enum comparison. Extraction returns base-anchored parameter ranges and never perturbs structural size parameters. If a candidate fails the base-instance probe, it receives one repair round with the error context included. The following two prompts are complete. Line breaks are adjusted for layout, and {...} denotes fill-in fields. Instructions for the solver stack are inserted for each family. The released repository contains the complete scripts.

Candidate-generation prompt (verbatim).

```
You are an expert in optimization modeling. Write a COMPLETE Python
script that defines a function solve(params) which builds and solves
the optimization problem below.

STRICT RULES:
1. Use {stack}.
2. Every numeric quantity listed in PARAMS must be read from the
   `params` dict (params["<name>"]); do NOT hardcode their values.
   Structural constants (set sizes, index structure) may be inline.
3. solve(params) must return a dict:
   {"objective": <float>, "status": "optimal"} on success;
   {"objective": None, "status": "<short reason>"} if infeasible/failed.
4. No printing, no file I/O, no network. Output ONLY one python code
   block.
5. Name the model object `model`; NEVER reuse `model` or `m` as a loop
   index or comprehension variable. Avoid Pyomo reserved attribute
   names for components (e.g. activate, deactivate, name, clone,
   index, display).
6. Do not wrap the whole body in a blanket try/except; let unexpected
   exceptions propagate (the harness captures them).
[structured strategy only]
7. Before the code, include as a Python comment a compact IR in JSON
   (sets, params, decision variables with types, constraints one line
   each, objective), then implement EXACTLY that IR.

PARAMS (name: meaning):
{param_desc}

PROBLEM:
{question}
```

Extraction prompt (verbatim).

```
You are an optimization-structure extractor. From the problem below,
list the NUMERIC PARAMETERS (costs, prices, capacities, demands,
budgets, coefficients) together with their ACTUAL VALUES as given in
the problem data.

For each parameter also propose a perturbation domain for sensitivity
testing, anchored at the true values: keep it physically meaningful
but STRESS-TEST constraint activation (allow sign changes only where
economically plausible, e.g. a net profit that may turn negative;
widen enough that different constraints become binding). The domain
must remain feasible-plausible: do not propose ranges that contradict
the parameter's role (e.g. capacities near zero while demands stay
large).

Index-set sizes and cardinalities (number of employees, projects,
cars, ...) must NOT be perturbed: either OMIT them from params
entirely, or freeze them with a degenerate abs range (lo == hi ==
base). Changing a size without resizing every dependent array
produces structurally invalid instances.

Output ONLY a JSON object, no prose, exactly this schema:
{
  "params": {
    "<name>": {
      "meaning": "<short>",
      "base": <number or list, the TRUE values from the problem>,
      "perturb": {"mode": "rel", "r": <0..1>}
                 or {"mode": "abs", "lo": <num>, "hi": <num>,
                     "integer": <bool>}
      (in "abs" mode, lo and hi are ABSOLUTE values on the same scale
       as base, NOT offsets or deltas around it)
    }
  },
  "objective_sense": "max" or "min"
}

PROBLEM:
{question}
```

A certified skill (excerpt). The following file is one of the 101 items in the gate-built library. It was distilled from admitted trajectories by the unmodified procedure of the host, with no manual editing. Workflow 2 and some implementation details are omitted for space, and typography is adjusted slightly. The complete file is included in the released library.

```
name: Bin Packing with Resource Activation
description: Model and solve resource minimization problems with
  capacity constraints using binary assignment and activation
  variables, producing exact or feasible solutions with clear
  verification steps.

# Workflow 1 (CP-SAT with Explicit Linkage)

## Modeling stage

### Strategy Overview
This workflow uses Google's OR-Tools CP-SAT solver, designed for
discrete optimization with Boolean logic. It models the problem with
separate binary variables for assignment and resource usage, linking
them via simple linear constraints.

### Step 1 - Define Core Variables
```

```
- Binary assignment variables x[i][j] for each item i and resource j.
- Binary usage variables y[j] for each resource j to track activation.

### Step 2 - Enforce Assignment and Capacity
- Exclusive assignment for each item i:
  sum(x[i][j] for j in resources) == 1.
- Capacity for each resource j:
  sum(weight[i] * x[i][j] for i in items) <= capacity.

### Step 3 - Link Assignment to Usage
- For each i, j add the implication x[i][j] <= y[j], so a resource is
  marked used if any item is assigned to it.

### Step 4 - Formulate Objective
- Minimize the number of used resources:
  minimize sum(y[j] for j in resources).

### Common Pitfalls
- Forgetting to link assignment to usage, which lets unused resources
  escape the objective.
- Setting an insufficient number of resources, causing infeasibility;
  initialize with a safe upper bound like the number of items.

## Solving stage

### Step 1 - Configure Solver
- max_time_in_seconds, num_search_workers, random_seed; for exact
  solutions set relative_gap_limit = 0.0.

### Step 2 - Solve and Check Status
- Accept only OPTIMAL or FEASIBLE; on infeasibility check the lower
  bound ceil(total_weight / capacity).

### Step 3 - Extract and Verify Solution
- Read the objective, per-resource usage, and per-item assignment;
  recompute per-resource total weight against capacity.

### Common Pitfalls
- Not checking both OPTIMAL and FEASIBLE statuses.
- Failing to provide a time limit for large instances.
```

B.1 THE DATA-LAYER STAGE

The stage runs before Step 1. Its instrument is a numeric-coverage check on the extracted specification: every base value, scalar or array entry, must appear among the numbers printed in the text, matched within relative tolerance 10^{-6} after normalizing separators, percentages, and units. A specification fails when any array with at least four entries has fewer than half of its entries printed, or when fewer than 80% of all base values are printed; a failing ticket routes to ESCALATE with the unmatched keys, and no candidate is generated. The check was designed after the audit of Section 4.3 and evaluated post hoc on the pilot's stored specifications: it flags 9 of the 15 missing-data cases, 0 of the 5 truncated-precision cases, 0 of the 2 label errors, 0 of the 116 concordant admissions, and 0 of the 148 calibration specifications, whose texts are generated from their instances (Fisher exact $p < 10^{-3}$ on the 138 admitted problems). It cannot flag truncated precision, where the printed values are present but rounded, so that sub-class remains outside the reach of any text-only check.

Agreement between independent extractors, the redundancy design one would try first, was tested as a preliminary on the 22 audited disagreements and 22 concordant admissions matched on admission score: the stored extraction of the pilot family plus one fresh extraction each from the second and third families, with the unchanged prompt at temperature 0 (88 calls). Table 4 reports pairwise disagreement at four levels. Independent extractors never agreed on parameter names, so a rule requiring identical key sets is unusable. After aligning parameters across specifications by shape and by base values within scoring tolerance, disagreement remained as common on faithful texts as

on unfaithful ones, because the families differ in which quantities they list as parameters far more often than in the values of shared parameters (across all pairs, 1,058 unmatched parameters against 48 value conflicts). The truncated-precision cases behaved as expected, the extractors agreeing on the values the text prints. Whether all three families fabricated identical values for unprinted data could not be measured, since a full three-way alignment succeeded in only 5 of the 44 cases.

Table 4: Two-extractor preliminary: cases with at least one pairwise disagreement among three extractor families ($E = 3$), by group, with Fisher's exact test between the 22 audited disagreements and the 22 concordant controls. The five truncated-precision cases are a subset of the first column.

Agreement level	Audited	Controls	Fisher p	Truncated
Raw parameter names	22 / 22	22 / 22	1.00	5 / 5
Parameter sets after alignment by shape and value	18 / 22	21 / 22	0.34	1 / 5
Elementwise values after alignment	18 / 22	21 / 22	0.34	1 / 5
Perturbation domains after guardrails	22 / 22	22 / 22	1.00	5 / 5

C HOST REPRODUCTION DETAILS AND SUPPLEMENTARY TABLES

The comparison against the ground-truth arm. On the full item set the gain of ADMITOR over the ground-truth-labeled library is $+4.46$ points, with a 95% interval of $[+2.34, +6.65]$. Restricted to the sensitivity item set, with the 85 anomalous items removed from every arm, it narrows to $+2.14$ points, $[+0.04, +4.29]$. This comparison is reported as an observation and no claim of the paper rests on it. On ComplexOR, the smallest plate, a single item is worth 1.11 macro points.

Bootstrap and library-size bookkeeping. Intervals are drawn from 10,000 paired resamples within each benchmark (seed 42, released script). On the item-weighted scale the gain over majority vote is $+1.73$, $[-0.27, +3.73]$, and over execution success $+1.27$, $[-0.73, +3.27]$; neither excludes zero. The benchmarks differ in size by a factor of thirty, and the item-weighted column of Table 2 preserves the ordering of the macro column with smaller margins. The four library sizes are 163, 145, 130 and 101 files for execution success, majority vote, ground truth and ADMITOR respectively. On the sensitivity item set the ground-truth arm reaches 73.5 on OptiBench.

The ComplexOR label error at evaluation. On ComplexOR the pipeline returns 250 for an instance labeled 200, and manual verification establishes 250 as the true optimum, so the model is penalized for a correct answer. ComplexOR scores 66.7 under the published labels and matches the reported 72.2 exactly once that single label is corrected. The main results retain the published labels, because the protocol was fixed before any judge was evaluated.

Calibration-set counts. On the 150 stratified NANO-CO instances the gate returns 103 ACCEPT, 36 UNINFORMATIVE, 9 ABSTAIN, and 2 error outcomes. Of the 103 accepts, 39 are two-family cliques (score $22.x$) and 64 are three-family cliques (score $33.x$). Comparison with the verified answers finds 10 false cliques: 8 among the two-family accepts, which the threshold excludes, and 2 among the three-family accepts, one of which carries no value at θ_0 and therefore certifies nothing. Table 8 reports the per-threshold counts. The replayed rule admits 138 of the 170 value-bearing non-dev verdicts, or 81.2%; the 170 are the 174 accepts less three dev-split problems and one accept without a value at θ_0. The 26 disagreements under the host's equality rule and the 22 under the

Table 5: The preregistered K4 check: candidate arm by terminal outcome, mined from the 298 stored certification runs; the two error outcomes have no verdict (zero additional model calls). The profiles differ significantly ($\chi^2 = 30.96$, $\mathrm{df} = 8$, $p = 1.4 \times 10^{-4}$).

Candidate arm	Clique	Value excl.	Infeasible	Partial crash	All crash
DeepSeek + Pyomo (direct)	164	17	71	5	41
GPT + Pyomo (structured)	154	9	75	9	51
Claude + Gurobi (direct)	172	30	73	1	22

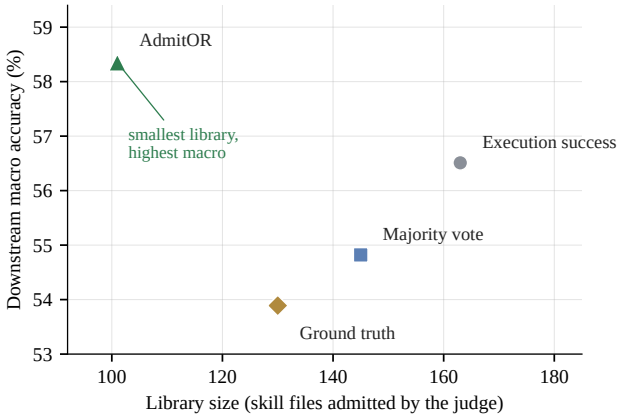


Figure 3: Library size and downstream macro accuracy for the four judges. Less selective judges admit more items and build larger libraries. The certified library is the smallest of the four and scores highest. Marker positions are the measured values of Table 2 and the library file counts of Section 4.2.

Table 6: Human-audited attribution of all 22 disagreements. No case is a gate model defect or a legitimate alternative interpretation. The dominant category (d) is a property of the benchmark, not of a solver.

Audited root cause	Cases
(a) gate defect: text states a constraint the clique drops	0
(c) legitimate alternative reading of ambiguous text	0
(b) label error (one confirmed by exact re-derivation)	2
(d) decisive data absent from the printed text	15
(d) parameters printed at truncated precision	5

round-aware scorer are not nested: 21 are common, 5 are wrong only under the equality rule, and 1 only under the round-aware scorer.

Decomposition of the uninformative verdicts. Of the 114 UNINFORMATIVE certifications in Section 4.2, 65 are dominated by infeasible resampled instances, 45 contain at least two candidates that never return a value, and 4 result from execution failure. None arises from conflicting evidence.

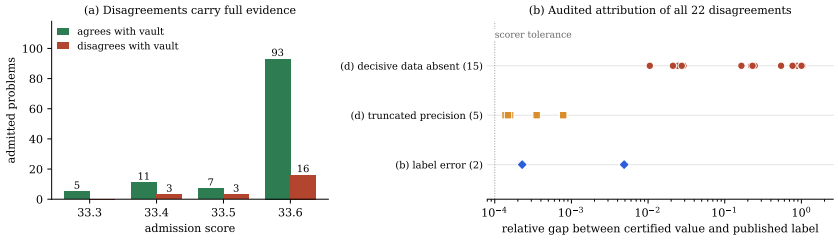


Figure 4: The anatomy of the K3 failure, from the released replay report. (a) Admitted problems by admission score: sixteen of the twenty-two disagreements sit at the maximal score, indistinguishable there from the 93 correct admissions, so no threshold on the amount of agreement separates them. (b) Every disagreement placed by its relative gap to the published label and colored by audited root cause: the truncated-precision cases and the two label errors hug the scorer tolerance, while the cases whose decisive data never appear in the text spread across two orders of magnitude.

The audit protocol and its packets. Each of the twenty-two disagreements was reviewed from a case-specific packet containing the problem text, the code of every clique member, both values, the relative gap, and the agreement pattern. The protocol assumes neither value correct and requires each verdict to cite a span of the text together with a code line or an explicit derivation. Three of the twenty-two verdicts were settled by exact re-derivation rather than judgment, and the per-case attribution is released as a table alongside the packets. The two label errors were settled by re-derivation independent of the gate: for a six-month capacitated lot-sizing problem in which the capacities never bind, exact dynamic programming reproduces the certified optimum; for a contract-allocation problem whose complete instance is printed, exact solution of the integer program produces the same result. The sign-checkable category-(d) exemplar is a six-producer allocation problem that prints two cost entries and states that the remaining entries have similar structures; the certified objective is negative, which no nonnegative cost matrix admits. The post-hoc screen referenced in Section 4.3 derives its bounds from the text alone, using the sign of the optimum and cost lower bounds implied by total demand, and suggests an inexpensive pre-consensus stage. It flags nine of the twenty-two disagreements without using labels, was applied after the fact, and affects no reported result.

Certification cost budget. Certification spends a fixed budget per problem: one extraction call, one generation call per family, at most one repair call per candidate, and $(m + 1)k$ solver runs ($m = 5$, $k = 3$ here). Majority vote over the same panel consumes the generation calls alone, so the gate adds one extraction call per problem and six solver runs per candidate, plus any repair calls.

The threshold sweep and the confidence bookkeeping. The four attainable thresholds differ only in the number of informative instances demanded, since family coverage and clique size are saturated on every admitted certificate. The realized false-discovery rates across them are 15.9%, 16.5%, 16.0% and 14.7%; the strictest, which requires three families, a three-member clique and six informative instances, still admits 109 cases with 16 disagreements. On the confidence side, $\delta = 0.05$ per threshold makes the selected-threshold certificate a 95% statement read per threshold and the simultaneous statement of Proposition 2(ii) a $1 - |T|\delta = 80\%$ statement over the four-point grid; the 95%-simultaneous reading corresponds to $\delta = 0.0125$, under which the value-bearing certificate at $\tau = 33.3$ has upper bound 9.7% (Table 8).

Reading the family-by-outcome matrix. The profiles of Table 5 separate along three interpretable dimensions. Execution failures track the solver stack: the two Pyomo arms fail on 15 to 20% of problems against 8% for Gurobi, reproducing at scale a pattern first seen during development. Instance infeasibility is flat across arms at 71 to 75, as expected for a property of the perturbed instance rather than of the candidate. The arm with the fewest execution failures reaches value comparison most often, producing 30 value-level disagreements against 9. Because all arms are run on shared instances, the profiles are more similar under the null of identical mechanisms than independently drawn profiles would be, so the observed separation is conservative. Two bounds on the reading: model family, prompting strategy, and solver stack co-vary by design, so the check establishes that the arms are not interchangeable rather than isolating family as the source of decorrelation; and the evidence concerns decorrelation conditional on a shared extracted specification, not at extraction itself. Restricted to candidates that reached value comparison, the three-by-two table of clique membership against value exclusion still rejects uniformity ($\chi^2 = 8.75$, $\mathrm{df} = 2$, $p = 0.013$; permutation $p = 0.012$), and adding the infeasible column gives $\chi^2 = 10.30$, $\mathrm{df} = 4$, $p = 0.036$.

Stream provenance and disjointness. The 300-problem stream is the first 300 problems, by index, of the OptMATH training split; the 150 calibration instances are a stratified sample of NANO-CO. Compared with all 1,100 evaluation items, no stream or calibration item matches exactly, after normalization, or as a near duplicate (5-gram token Jaccard at least 0.8); the maximum Jaccard is 0.21 for the stream and 0.02 for the calibration set, with OptMATH-Bench included in the comparison. File hashes are in the release.

Judge ablation details. The panel judges of Table 3 use the stored stated-instance values of the three family candidates with the agreement rule of Step 3; the calibrated judge applies $\tau = 33.3$ to the stored scores. At the two-family level, the 245 problems on which any two families agree at θ_0 split into 170 accepted by the admission-time rule (29 wrong certificates), 68 uninformative (6 wrong), and 7 abstentions (3 wrong). Against the host's vote, the two rules certify 224 problems in

common and agree on 206 of them; where they disagree, the host's certificate is wrong in 14 cases and the panel's in 6. On the 30 problems only the host certifies, its certificate is wrong 12 times; on the 21 problems only the panel certifies, its certificate is wrong 20 times, the host's three samples having disagreed among themselves. Of the five base-unanimous problems on which resampling excluded a candidate, four are execution failures on perturbed instances (an invalid value drawn into a 0/1 matrix, a non-integer demand, a negative weight, a dimension error) and one is a disagreement in the values themselves; the stated-instance value is correct in all five. Applying $\tau = 33.3$ to the deployed library would remove 16 of its 99 clusters outright and thin 6 more; 11 files hold all 30 poisoned admissions, one of them 9. Re-drawing the 57 withheld base-unanimous problems with draws rejected when no candidate solves them, at most ten attempts per draw, brings 8 to three informative instances; all 8 certify the base-unanimous value, 7 of them correct against the vault.

Table 7: Retrieval concentration per arm on the two largest benchmarks: distinct library files selected by the host, and the share of items on which the single most-selected file was chosen. Selection-failure rows per arm over all five benchmarks: ground truth 95 (85 on OptiBench), majority vote 10, execution success 10, ADMITOR 2; the selector never returned an identifier absent from its library.

	Library files	OptiBench (605)		Mamo ComplexLP (211)	
Arm		distinct	top share	distinct	top share
Ground truth	130	31	20.3%	20	20.9%
Majority vote	145	38	32.2%	22	20.9%
Execution success	163	30	39.5%	19	19.9%
ADMITOR	101	19	44.6%	18	22.3%

The OptiBench selection anomaly. In the 85 affected items of the ground-truth arm, the selector of the host repeatedly generates a skill identifier that is not present in the library. Each affected row is identified in the released logs. The anomaly shows that library *contents* can affect the reliability of the selection stage: admission policy determines both what a library contains and how reliably that library can be searched.

Case study: a label convicts the innocent. One instance in a widely used cleaned benchmark carries a published label of 200 although its optimum verifies by hand as 250 (full derivation in the release). This case touches every stage of our evaluation. In reproduction (Section 4.1) the standard host returns 250 and is scored as incorrect. At admission, the judges diverge exactly as designed. The ground-truth judge compares against the published answer and rejects the correct trajectories, while the three families in ADMITOR agree on the sampled value-function trace and certify 250. The behavioral evidence therefore recovers a correct model that the erroneous label discards. At evaluation all four arms return 250 and are penalized, because the preregistered protocol retains the published labels. Label error can therefore move a small benchmark materially. The behavior of the gate follows from the design rather than from a favorable scoring choice, since it never reads the label.

The calibration replay supplies a complementary example. For one audited instance the certified and reference models differ by an entire constraint family, yet their objective values differ by 0.023%, and an exact solution of the fully printed instance reproduces the certified value. The reference therefore errs in this case. Similar objective values do not establish model equivalence, just as different values do not by themselves identify which model is wrong. The audit trail resolves this distinction.

Table 8 jointly summarizes the calibration and replay results from Section 4.3.

Table 9 reports the end-to-end reproduction of the host system with the released skill library. The repository does not provide scoring code, so the uniform round-aware scorer defined in Appendix B is applied to every benchmark and every downstream judge arm.

Annotations. ComplexOR. With $n = 18$, each item changes accuracy by 5.56 points. Therefore, falling within a ± 3-point band requires an exact match in the number of correct answers. The single difference is the instance discussed in Section 4.1. Its published label is 200, manual verification gives 250, and our pipeline returns 250. After correcting the label, the score becomes $13/18 =$

Table 8: Calibration and replay of the calibrated admission rule. Left: for each attainable threshold, the value-bearing three-family certificates on the 150 solver-verified NANO-CO instances, the false ones among them, and the exact one-sided Clopper–Pearson upper bounds at $\delta = 0.05$ and $\delta = 0.0125$; the rule of Proposition 2 holds at every row under $\delta = 0.05$ and at $\tau \in \{33.3, 33.4\}$ under $\delta = 0.0125$. Counting the one three-family accept without a value at θ_0 gives 64 certificates and 2 false at $\tau = 33.3$ ($\hat{p} = 3.1\%$, $U_{0.05} = 9.5\%$, $U_{0.0125} = 12.1\%$). Right: replay on the benchmark stream, scored against the sealed vault with the uniform scorer (the host's equality rule in parentheses).

Calibration (NANO-CO)						**Replay (benchmark stream)**	
τ	n	false	$\hat{p}$	$U_{0.05}$	$U_{0.0125}$		
33.3	63	1	1.59%	7.31%	9.71%	accepts replayed	170
33.4	62	1	1.61%	7.42%	9.86%	admitted at τ	138
33.5	61	1	1.64%	7.54%	10.01%	disagreements	22 (26)
33.6	58	1	1.72%	7.92%	10.50%	realized proportion	15.9% (18.8%)
						95% upper bound	22.0% (25.2%)

Table 9: Host reproduction on five public benchmarks (accuracy, %). Published labels as-is; annotations below.

Benchmark	n	Reported	Ours	Δ
IndustryOR	100	36.00	36.00	0.00
Mamo.Complex	211	63.51	62.09	−1.42
OptiBench	605	77.02	75.87	−1.15
ComplexOR	18	72.22	66.67	−5.55
OptMATH-Bench	166	61.45	56.63	−4.82
Macro		62.04	59.45	−2.59

72.22, which exactly matches the reported value. Following the preregistered protocol, all main tables retain the published label. **OptMATH.** We rerun a cluster of 17 empty predictions in isolation with one worker. Only 3 change to correct answers, which is below the preregistered materiality threshold of 5. We therefore retain the original score. Solver contention explains at most 1.8 points of the deficit. The remaining difference reflects a capability gap in the reproduced pipeline on this benchmark. No central claim depends on the absolute OptMATH scores. An audit of the model identifier in each response confirms that every call used the fixed backbone. **IndustryOR.** Three labels contain the sentinel value −99999, which provides additional evidence of answer-key errors in widely used benchmarks. The release includes the complete run ledger, including call-level logs, segment-level resume records, and environment snapshots.