When Objectives Agree but Rules Do Not: The Hidden Cost of Exposing Rule Errors in Optimization Models

Anonymous authors
Paper under double-blind review

Abstract

An optimization-program revision may attain the expected optimum while enforcing the wrong rule, because the plans that expose the error can be worse than every operationally plausible decision. Such a revision can pass a check that uses only the optimal objective value. We formalize this *value–information barrier*. The critical loss $\ell(i, j)$ is the least objective degradation at which two rule interpretations disagree; its maximum against all alternatives exactly characterizes target identifiability from complete-plan membership answers on a finite shared domain. It also bounds the blind spot of objective-value checks. In a post-hoc replay of 278,208 sealed assignment class pairs, 35,106 pairs share an optimal value; 12,496 of those (35.6%) require additional loss beyond that common optimum to expose their rule difference. An LP slack law gives marginal-value bounds, and 63 constructed public-LP bound changes check the mechanism. One post-hoc public complete-plan case has a model-derived 17.6% barrier; broad full-program LLM evidence remains unmeasured. Two sealed three-domain studies show a steep rise: 1.9–4.2% of update targets are identifiable using only optimal plans, compared with 64.7–67.8% at 10% loss. Factorial studies in two domains reproduce the frontier, and registered append targets are harder to expose than retract targets (67.8% versus 95.1% at 10% loss). Critical-loss-Aware Planning (CAP) is an exact finite-budget reference that separates witness availability from question allocation and shows that isolating targets and shrinking ambiguity are different objectives. On 24 public benchmark revisions, 18/23 executable errors from three local models need positive local loss to expose. The evidence establishes a replicated finite-domain phenomenon and a limited public-task bridge; it does not estimate natural-history prevalence, open-ended generated-program loss, or human clarification cost.

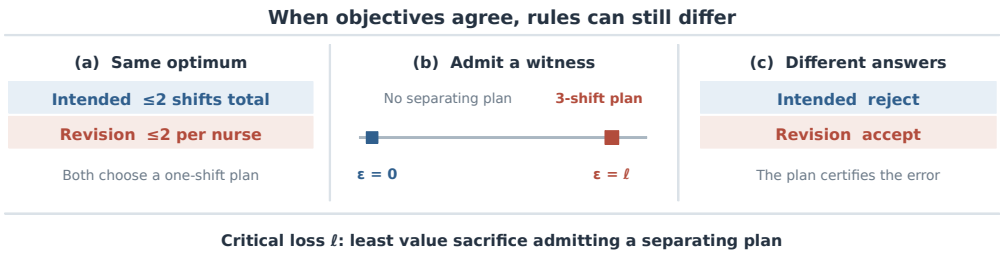


Figure 1: The paper's motivating failure mode (schematic, not an empirical result). The intended and mistaken overtime rules share an optimum, so objective-value checking misses the error. A three-shift plan separates their answers only when its loss is admissible. The critical loss $\ell(i, j)$ is the cheapest such witness; CAP allocates questions after witnesses become available.

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.

1 INTRODUCTION

Motivation. A program can return the right optimum while enforcing the wrong rule. **Core insight.** Under complete-plan membership queries, the error becomes testable only when the allowed value loss reaches a plan on which the two rules disagree; no such query policy can overcome a missing witness.

Optimization agents translate changing requirements into mathematical models and solver programs. A revision can achieve the expected objective while enforcing the wrong rule: the changed constraint may be slack at the optimum, duplicate rules may compensate, or append, overwrite, and retract may be confused. Suppose a planner requests at most two overtime shifts in total, but a revision allows two per nurse. Both select a one-shift optimum; only a three-shift roster reveals the wrong rule. An objective-value-only check would accept the revision, a blind spot highlighted by Constraint Injection (Luo et al., 2026); OptiBench exemplifies answer-based evaluation (Yang et al., 2025).

We ask exact binary membership questions about complete plans, limited by an explicit value tolerance. One use case is reviewing a small number of plausible rosters or orders before deploying a revised optimization model: the reviewer can say whether a concrete complete plan obeys the intended rules, while a large, implausible plan may not be a useful clarification artifact (Bessiere et al., 2017). The value tolerance formalizes this screening choice; it is an interface assumption, not an empirical estimate of human effort. Our primary question is *how much objective value must be sacrificed before a rule interpretation can be identified under this interface?* Directly asking for a rule or showing a local counterexample changes the interface and may remove the barrier. Once separating plans are admissible, a second question is how to use a small answer budget.

The answer is an exact *critical loss*: the least value degradation at which two interpretations disagree. It characterizes the availability barrier independently of any query policy and bounds the blind spot of objective-value checks. We then use an exact finite-pool policy, CAP, as a reference for question allocation after witnesses exist. Its Bellman recursion is prior-art machinery, not a new generic decision-tree framework. Empirically, composition deepens the barrier: in our assignment factorial, the median target threshold rises from 3.6% for one changed rule to 6.1% for three.

Our contributions are:

- **A value–information barrier.** Optimal and near-optimal plans can make distinct executable rule semantics observationally identical; pairwise critical loss and the target threshold $\epsilon_i^\star$ exactly characterize identification under finite complete-plan queries. A decomposition separates loss already implied by either interpretation's own optimum from extra loss needed to reveal a same-value rule error; LP sensitivity bounds price this latter effect through slack and marginal value.
- **A replicated structural pattern.** Sealed assignment and scheduling factorials show a similar frontier with different predicates, plans, objectives, and independently seeded scheduling cells.

  Appended and multi-rule update targets need more relaxation than retractions. A sealed-data replay identifies the subset of equal-optimum rule differences whose witness requires additional value loss, beyond the much larger class of pairs explained by optimum shifts.
- **A separate finite-budget objective.** CAP computes exact singleton coverage for a finite admissible query bank. It is an oracle reference rather than a proposed heuristic. It reveals that resolving some targets completely and shrinking ambiguity uniformly are different goals. Public benchmark revisions and an exact noise replay expose candidate-coverage and answer-reliability limits.

2 RELATED WORK

Optimization modeling and verification. OptiBench and ReSocratic evaluate LLM optimization modeling (Yang et al., 2025); MIPLIB-NL and OR-Space provide large modeling tasks and agent workspaces (Li et al., 2026; Zhou et al., 2026). Constraint Injection builds diagnostic instances for semantic errors (Luo et al., 2026), while VeriSimpl combines solver-generated checks with LLM reasoning (Rahman et al., 2026). Constraint Injection is closest: it shows that objective agreement can hide semantic errors. We quantify how deeply such errors hide, and identify when no plan-level question can expose them. Our object is the value required to make a semantic disagreement observable under sequential rule revisions.

Constraint acquisition and hypothesis identification. Membership queries and version-space filtering are classical in constraint acquisition (Bessiere et al., 2007; 2013; 2017; Tsouros et al., 2024).

Generalized binary search and adaptive policies study query allocation, and diagnosis determination optimizes testing cost in identification trees (Golovin & Krause, 2011; Cicalese et al., 2014). These works make balanced splitting and adaptive query trees prior art; our contribution is the value-admissibility characterization for optimization plans and the measured structure of its threshold. Formally, those methods optimize a policy after a query family is specified, whereas our first object is the equivalence relation induced when an objective-loss budget removes separating queries from that family. CAP addresses allocation only after this availability relation has been characterized.

Sensitivity analysis. Theorem 7 rests on the concavity of parametric linear-program value functions, a standard tool of sensitivity analysis (Bertsimas & Tsitsiklis, 1997). What is new is the reading: the value function prices the exposure of a rule error, and the slack of the intended rule sets that price.

Optimal decision trees. MurTree uses dynamic programming and search for globally optimal bounded-depth classification trees (Demirović et al., 2022). STreeD characterizes separable objectives and gives a general dynamic-programming framework for optimal trees (van der Linden et al., 2023). Accordingly, we do not present CAP's Bellman recursion as a new generic optimal-tree framework. CAP is an exact reference specialized to value-admissible optimization queries and singleton-leaf target coverage; it lets the experiments distinguish availability from finite-budget allocation.

Repair boundary. CEGIS and solver-guided LLM planning already alternate proposals and counterexamples (Solar-Lezama, 2018; Jha et al., 2023). Candidate construction is outside our certificate. If the intended interpretation is absent, filtering and CAP cannot recover it.

3 Critical loss characterizes identifiability

This section shows that one quantity, critical loss, decides both what exact plan questions can identify and what objective-value checks can miss. Let $u_{1:t}$ denote revisions through update t, and let candidate construction produce typed executable interpretations $\mathcal{H}_{t,0}$. For a complete decision x, interpretation h returns $\rho_h(x) = \mathbb{I}[x \in \mathcal{F}(h)]$. Exact answer $y_{t,j}$ filters the pool by

$$\mathcal{H}_{t,j+1} = \{h \in \mathcal{H}_{t,j} : \rho_h(x_{t,j}) = y_{t,j}\}. \tag{1}$$

Candidate construction at the next revision is separate and may lose coverage.

For a shared base feasible set $\mathcal{F}_0$ and maximization value v_0, define

$$L_0(x) = \frac{v_0^\star - v_0(x)}{\max\{|v_0^\star|, 1\}}, \qquad \mathcal{Q}_\epsilon = \{x \in \mathcal{F}_0 : L_0(x) \le \epsilon\}. \tag{2}$$

Minimization reverses the numerator. A query is a clarification artifact, not authorization to execute its plan. The denominator fixes an objective scale rather than an intrinsic rule property: adding a constant to v_0 changes these percentages without changing plans or responses. We therefore report raw objective differences for the public complete-plan witness and a translation-invariant base-range sensitivity analysis for the generated assignment results.

Proposition 1 (Conditional preservation)**.** *If the intended interpretation is initially present and evaluators and answers are correct, filtering preserves it. A singleton identifies it only within the initial pool; filtering cannot recover an absent interpretation.*

Proposition 2 (Admissible-query indistinguishability)**.** *If two interpretations agree on every $x \in \mathcal{Q}_\epsilon$, no adaptive policy restricted to $\mathcal{Q}_\epsilon$ can distinguish them using exact binary answers.*

Collapse candidates with identical response signatures on $\mathcal{F}_0$ into semantic classes. Raw programs with different source or future-revision behavior may collapse into one current-interface class. A *class target* is a class taken as the intended interpretation; an *update target* is the class that implements the intended append, overwrite, or retract. For classes $i \neq j$, define

$$\ell(i,j) = \begin{cases} \min\limits_{x \in \mathcal{F}_0 : \rho_i(x) \neq \rho_j(x)} L_0(x), & \exists x \in \mathcal{F}_0 : \rho_i(x) \neq \rho_j(x), \\ +\infty, & \text{otherwise}, \end{cases} \qquad \epsilon_i^\star = \max_{j \neq i} \ell(i,j). \tag{3}$$

In the overtime example of the introduction, $\ell(i,j)$ is the normalized extra cost of the cheapest roster with three overtime shifts.

Theorem 3 (Value-constrained identifiability)**.** *Assume a finite shared base domain, complete-plan membership queries, correct response evaluators, and exact answers. Classes i and j are distinguishable by queries in $\mathcal{Q}_\epsilon$ if and only if $\epsilon \geq \ell(i,j)$. Target i is distinguishable from every alternative in the supplied pool if and only if $\epsilon \geq \epsilon_i^\star$.*

We then call target i *identifiable* at tolerance ϵ; figures and tables use *certifiable* for the same event. This characterization separates two barriers. If $\epsilon < \epsilon_i^\star$, a separating witness is unavailable to every exact policy. If $\epsilon \geq \epsilon_i^\star$, witnesses exist, but a short answer budget can still prevent identification. The $+\infty$ branch applies to raw interpretations with identical responses on the entire interface; after full-domain deduplication, distinct finite-domain classes always have finite pairwise loss.

Critical loss also explains why such errors survive objective-value checks. Let $x_h^\star$ be an optimal plan of interpretation h over $\mathcal{F}_0$, assumed to exist.

Proposition 4 (Objective-value blind spot)**.** *If $\ell(i,j) > L_0(x_i^\star)$, then i and j have the same optimal value.*

Proposition 5 (Optimum-loss floor)**.** *For any two classes, $\ell(i,j) \geq \min\{L_0(x_i^\star), L_0(x_j^\star)\}$.*

Write $\delta_h = L_0(x_h^\star)$ and define the *extra exposure loss* $\Delta_{\mathrm{extra}}(i,j) = \ell(i,j) - \min\{\delta_i, \delta_j\} \geq 0$.

Proposition 6 (When the floor is exact)**.** *On a finite base domain, if $\delta_i \neq \delta_j$, then $\ell(i,j) = \min\{\delta_i, \delta_j\}$. If $\delta_i = \delta_j = \delta$, then $\Delta_{\mathrm{extra}}(i,j) > 0$ exactly when the two interpretations agree on every own-optimal plan; otherwise $\ell(i,j) = \delta$.*

Thus a high critical loss can reflect an ordinary optimum shift rather than an additional price to expose an objective-agreeing error. The last case in Proposition 6 is the paper's sharper hidden-error mechanism: even plans optimal for either interpretation fail to distinguish its rule semantics. This finite-domain result is proved in Appendix A.

The next result makes the depth of the blind spot explicit for linear programs. Let $g(t) = \max\{v_0(x) : x \in \mathcal{F}_0,\ a^\top x = t\}$ on feasible slices, and let t_0 be the largest value of $a^\top x$ over base-optimal plans.

Theorem 7 (Slack law)**.** *Let $\mathcal{F}_0$ be a polyhedron with finite, attained maximum of the linear objective v_0, and assume the optimal-face maximum t_0 exists and is finite. Let two interpretations differ in one bound: $\mathcal{F}(i) = \mathcal{F}_0 \cap \{a^\top x \leq b\}$ and $\mathcal{F}(j) = \mathcal{F}_0 \cap \{a^\top x \leqq b'\}$ with $b < b'$. If the tighter bound is slack at the optimum, $s = b - t_0 > 0$, and $b < \sup_{x \in \mathcal{F}_0} a^\top x$, then, reading the minimum in Equation 3 as an infimum,*

$$|g'_+(t_0)|\, s \ \leq\ \ell(i,j)\ \max\{|v_0^\star|, 1\}\ \leq\ |g'_-(b)|\, s.$$

The theorem gives endpoint-marginal bounds, not a fixed linear growth law: $g(t_0) - g(b)$ is piecewise linear in b, and equals a constant slope times s only where the marginal value stays constant. Alternative optima that reach a larger $a^\top x$ raise t_0 and shrink the slack, possibly to zero. The LP quantity is an *infimum*: a strict separating plan may not exist at exactly $\epsilon = \ell$, but exists for every $\epsilon > \ell$ when the stated bound change creates disagreement. The attained-threshold claim in Theorem 3 is only for finite domains. Our integer-domain tests do not empirically test this LP bound. The lower bound extends to errors in several bounds at once.

Proposition 8 (Several differing bounds)**.** *Let $\mathcal{F}(i) = \mathcal{F}_0 \cap \{a_k^\top x \leq b_k,\ k \in K\}$ and $\mathcal{F}(j) = \mathcal{F}_0 \cap \{a_k^\top x \leq b'_k,\ k \in K\}$, with $\mathcal{F}_0$ a polyhedron, a finite attained maximum of linear v_0, and finite optimal-face maxima $t_{0,k}$ for all differing bounds. Define g_k as above and $s_k = \min\{b_k, b'_k\} - t_{0,k}$. Set $m_k = |g'_{k,+}(t_{0,k})|$ when $\sup_{x \in \mathcal{F}_0} a_k^\top x > t_{0,k}$, and $m_k = 0$ otherwise. Interpret ℓ as an infimum. Then*

$$\ell(i,j)\ \max\{|v_0^\star|, 1\}\ \geq\ \min_{k:\, b_k \neq b'_k} m_k\ \max\{s_k, 0\}.$$

Thus, multiple changed bounds have a lower exposure bound given by the smallest of their per-bound lower bounds; a zero-slack bound makes this particular certificate vacuous.

4 When witnesses exist: exact answer allocation

Let S be the current semantic classes, $S_a(x) = \{h \in S : \rho_h(x) = a\}$, and $\mathcal{A}_\epsilon(S)$ contain admissible queries with two nonempty response buckets. Define

$$V_\epsilon(S,b) = \begin{cases} 1, & |S| = 1, \\ 0, & |S| > 1,\ (b = 0 \text{ or } \mathcal{A}_\epsilon(S) = \varnothing), \\ \max\limits_{x \in \mathcal{A}_\epsilon(S)} \sum\limits_{a \in \{0,1\}} V_\epsilon(S_a(x), b-1), & \text{otherwise.} \end{cases} \tag{4}$$

Proposition 9 (Budgeted coverage optimality)**.** *For a finite query bank, $V_\epsilon(S, b)$ equals the maximum number of target classes that any target-independent adaptive policy can place in singleton leaves with at most b answers per path. With precomputed partitions, the full subset dynamic program uses $O(Bm2^n)$ partition evaluations and $O(B2^n)$ memoized states for n classes and m admissible partitions.*

CAP instantiates Equation 4. In both structural implementations, equal-coverage actions are ordered by maximum branch size, the sum of squared branch sizes, current query loss, and a canonical query mask. Max-split uses the same order without the leading recursive-coverage objective. These implementation tie-breaks can change the selected tree and residual-size or noise results, although they do not change CAP's optimal singleton-coverage value. CAP is not a polynomial-time method for unbounded program spaces; on the 32-class factorial cells it uses a median of 11 admissible partitions and 155 visited states. A warmed single-process Python replay takes a median 0.13 ms per instance, excluding construction (Table 19); this is an implementation measurement, not an asymptotic claim. Proofs are in Appendix A.

The evaluation reports two distinct metrics. *Singleton coverage* is the fraction of possible targets placed alone at a leaf; Equation 4 optimizes its numerator. *Residual ambiguity* is the number of classes remaining after the observed path. A policy can improve one while worsening the other, so neither is presented as a universal operational objective.

5 Experimental design

Discovery and independent confirmation. A sealed study freezes and hashes its generators, seeds, endpoints, and analysis code before any outcome is computed; these are local seals, not externally timestamped preregistrations. Two sealed studies use 40 assignment, 40 scheduling, and 40 knapsack bases each, with disjoint seeds. Ten raw interpretations deduplicate to 8–10 semantic classes; append, overwrite, and retract yield 360 update targets per study. The independent study contains 1,188 semantic targets and evaluates six value tolerances, answer budgets 1–4, CAP, three deterministic baselines (max-split, loss scan, and gain per loss), and 32 fixed random trees. Instance-clustered bootstrap intervals keep all classes from one base together. Appendix B gives the generators and audit checks.

Controlled structural shift. A full-factorial assignment test varies rule count, semantic-class count, optimum multiplicity, update depth, and update family in an exhaustively enumerated six-task domain. Eight previously unused seeds are reused across 162 factor cells, giving 1296 rows and 24,192 uniformly weighted semantic targets. We therefore bootstrap the eight complete seed blocks rather than treat rows as independent. With only eight blocks, their percentile-bootstrap intervals are descriptive sensitivity ranges; nominal 95% coverage is not established. The equal-optimum pair decomposition and base-range normalization sensitivity were specified after seeing the original results; they replay sealed records without new sampling.

Independent scheduling confirmation. A sealed scheduling factorial uses all 720 permutations of six jobs and a distinct predicate bank. The same five factors define 162 cells, but each of two replicates per cell receives a unique seed: 324 instances and 6,048 semantic targets. Endpoints and CAP/max-split settings are fixed in advance; uncertainty resamples instances.

Actual-output bridge. A sealed bridge selects 24 OR-Space Revise texts with an unambiguous named numeric hard rule. A fixed prompt requests a JSON triple—operator, threshold, and aggregation scope—and a deterministic schema parser rejects malformed outputs. Qwen2.5-3B, Phi-4-mini, and Qwen3-8B each produce one greedy candidate per context; a fixed 12-plan bank measures local response differences. These are curated public benchmark revisions rather than naturally occurring histories, and the local loss proxy is not full-program objective loss.

Answer-noise sensitivity. On the 324 independently seeded scheduling instances, we exactly propagate symmetric answer flips of 0/1/5/10% through the selected two-question trees. One, three, or five independent answers per query are aggregated by majority vote. Outcomes distinguish correct singleton, false singleton, retained unresolved target, and excluded unresolved target. This is a mathematical sensitivity analysis, not a human-response model.

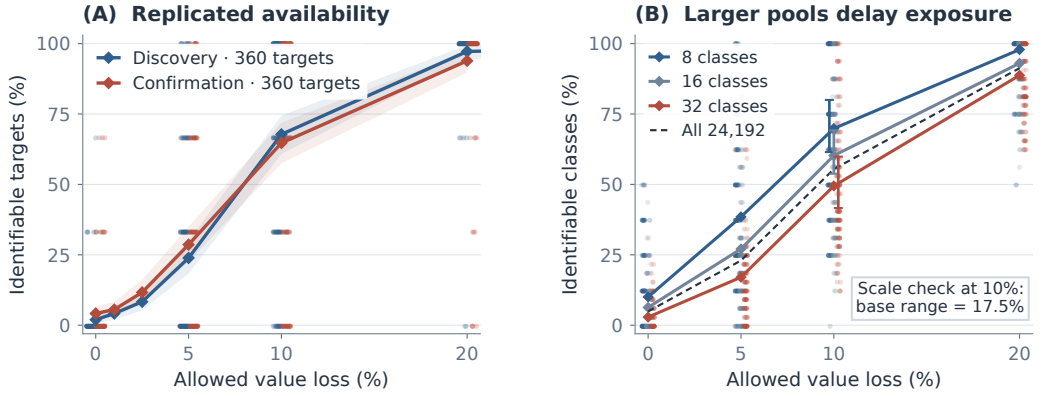


Figure 2: Finite-domain availability rises with allowed loss. (a) Two sealed three-domain tests use disjoint seeds; dots are instance rates, lines use all registered thresholds, and bands are 95% instance-clustered intervals. (b) Pool-size curves use all classes; dots are display subsamples and bars at 10% are eight-seed-block intervals. Jitter is display-only. Curves use the base-optimum scale; the box reports the alternative base-range scale at nominal 10% (Table 8).

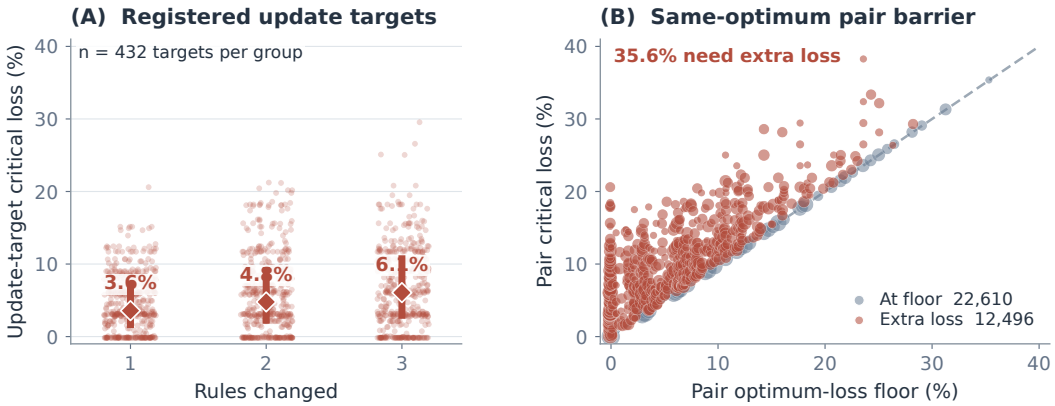


Figure 3: The measured barrier matches the claim's statistical units. (a) All 432 registered update targets per revision depth; diamonds mark medians, thick segments interquartile ranges. (b) All 35,106 same-optimum unordered class pairs at 620 distinct coordinates; the horizontal axis is the pairwise optimum-loss floor and the dashed diagonal is the exact floor. Red points need extra loss. Bubble area grows logarithmically with coordinate multiplicity. The pair decomposition is post-hoc.

6 Results

6.1 The availability transition replicates

In the generated three-domain study, optimal plans rarely separate rule interpretations; separation becomes more common after several percent of value is given up. The discovery study identifies 1.9% of update targets at strict optimality, 67.8% at 10% loss, and 97.2% at 20%, with median threshold 8.12%. Both studies count update targets, three per base; later sections count semantic classes, so their percentages are not directly comparable. The independent study reproduces the curve: 4.2% at zero loss, 64.7% at 10%, and 93.9% at 20%, with median 7.82%. Its median witnesses lose 12 assignment-utility points and reassign two of seven jobs, 14 scheduling-utility points and reslot two of eight jobs, or 20 knapsack-profit points and change three of twelve items. These units make the threshold concrete within each generated domain; they do not establish user acceptability.

Table 1: Post-hoc decomposition of sealed assignment data. Each observation is an unordered full-domain semantic-class pair; floor $= \min(\delta_i, \delta_j)$ and extra loss $\Delta = \ell -$ floor. The 35.6% positive-extra rate is conditional on equal optima; the corresponding all-pair rate is 4.5%. Medians for positive Δ condition on that subset; percentages use the base-optimum scale. Brackets give eight-seed-block 95% intervals or an interquartile range as labeled.

Pair stratum	Pairs	Med. floor / ℓ	Extra+ count; rate [CI]	Pos. extra med. [IQR]
All pairs	278,208	3.2% / 3.2%	12,496; 4.5% [3.8, 5.2]	3.2% [1.8, 6.1]
Same optimum	35,106	3.2% / 5.9%	12,496; 35.6% [26.9, 44.5]	3.2% [1.8, 6.1]
Different optimum	243,102	3.2% / 3.2%	0; 0.0%	–

6.2 Structural associations with critical loss

The factorial records associate the threshold with the number of competing interpretations and the update construction. Across the structural-shift test, the median class threshold is 9.2%; 55.5% of classes are certifiable at 10% loss and 91.3% at 20%. The transition therefore persists after changing the generator structure, but its location is not universal. With 8 semantic classes, 69.8% are certifiable at 10%; with 32 classes, 49.5% are. Increasing atomic rules from 4 to 8 changes the same endpoint from 58.3% to 50.1%. Unique and multiple optima have similar medians (9.4% and 9.1%) but different zero-loss mass.

The 1,296 rows share only eight underlying seeds across factor cells. A seed-block bootstrap leaves the 55.5% point estimate unchanged but widens its 95% interval to 48.2–65.2%; at 5% loss the interval is 16.0–33.6%. Thus the transition is visible, but the assignment-only uncertainty is materially larger than an instance-row bootstrap suggests. Unique versus multiple optima also change the objective coefficient distribution, so their marginal difference is an association rather than an isolated causal effect.

Registered update targets also vary by composition. Their median threshold is 3.6% for one changed rule and 6.1% for three; 10%-loss certifiability is 89.4% and 70.1%, respectively. Retract targets are easiest (95.1% at 10%), overwrite is intermediate (75.0%), and append is hardest (67.8%). These are 432 update targets per family, rather than uniformly sampled classes. The target's own-optimum loss correlates with its critical loss ($\rho = 0.823$ over 24,192 classes; eight-seed block interval $[0.799, 0.839]$), but this does not by itself establish an extra witness cost.

We therefore replayed all 278,208 sealed pair certificates and independently reconstructed both interpretations' own optima. Table 1 separates optimum shifts from the stronger phenomenon. Most pairs (243,102) have different optimal values and attain the floor of Proposition 5; the 35,106 equal-value pairs are the relevant objective-check blind spot. Of these, 12,496 (35.6%; eight-seed block interval 26.9–44.5%) require strictly more loss than their shared optimum to reveal the rule difference. Their median *additional* loss is 3.2 percentage points of the base optimum, or 2.62 utility units on this generated assignment scale. The remaining 22,610 equal-value pairs can already be separated by an own-optimal plan. This post-hoc decomposition is descriptive, not a prevalence estimate for naturally generated programs; Appendix J gives pool-size and update-family strata.

The replay checks all pair floors and stored witnesses with zero violations. At the target level, 11,252/24,192 class thresholds (46.5%) exceed the maximum pair-floor bound, as do 725/1,296 registered update thresholds (55.9%); these are different units from pair prevalence. Figure 3(b) shows the pairwise floor directly. On 21 selected public LP rows at the middle constructed threshold, the local marginal-price bound ranks measured exposure loss closely ($\rho = 0.988$; leave-one-source-LP-out range $[0.980, 0.993]$). This constructed, in-sample diagnostic is not natural-revision prediction (Appendix K).

6.3 Answer allocation and residual ambiguity are different objectives

On the independent 8–10-class test, CAP reaches 23.1% singleton coverage at 10% loss and two answers, versus 0.9% for max-split; at four answers both reach 62.9%. The gap at short budgets therefore measures allocation after queries become available, not a different availability ceiling. Because CAP is optimal for this objective by construction, the gap measures what a standard splitting heuristic leaves unused, not a method improvement. Under the tested uniform class prior and deterministic binary answers, generalized binary search, one-step information gain, and EC^2 edge

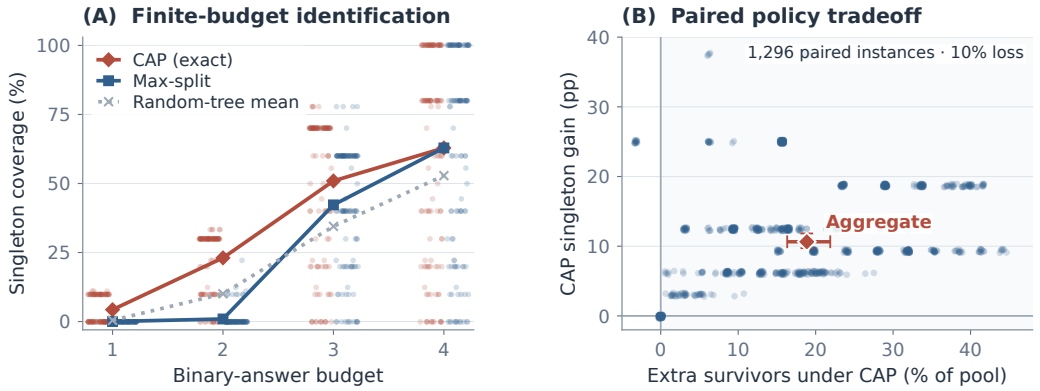


Figure 4: (a) Finite-budget singleton coverage over 1,188 independent-study semantic targets; dots show instances and lines class-weighted aggregates. CAP is exact for this objective; the balanced-split controls coincide here under uniform priors and shared tie-breaking. (b) Paired structural-shift outcomes at 10% loss: each dot is one of 1,296 instances, with CAP-minus-max-split residual survivors (percentage points of each pool) against singleton gain (percentage points). The diamond marks class-weighted aggregate differences on both axes; bars are eight-block percentile-bootstrap intervals. The direct-rule interface is separate (Table 18).

Table 2: Two-answer tradeoff on all 24,192 structural-shift class targets. "Available" is the unlimited-answer ceiling; survivors are shown in counts and as a percentage of the initial pool. Paired CAP-minus-max-split intervals resample eight complete seed blocks. CAP optimizes singleton coverage; max-split favors balanced branches.

Singleton coverage (% of all class targets)

Loss	Available	CAP	Max-split	Paired gain [block 95% CI]
10%	55.5	14.1	3.4	+10.6 [9.7, 11.8]
20%	91.3	15.0	1.6	+13.4 [12.8, 14.0]

Residual ambiguity (mean survivors; % of initial pool)

Loss	CAP	Max-split	Paired excess [block 95% CI]
10%	15.78 (62.8%)	11.05 (44.0%)	+4.73 [4.11, 5.48]
20%	16.24 (64.5%)	9.90 (39.0%)	+6.33 [5.93, 6.69]

cutting all order splits by the product of the two branch sizes. With the same tie break they coincide with max-split, so they are one control rather than three independent wins (Golovin & Krause, 2011). A separate exact direct-rule diagnostic can identify all ten raw typed candidates with four rule-membership answers, but it changes the question interface and target denominator; Table 18 reports it separately.

The structural shift makes the objective tradeoff explicit. At 10% loss and $B = 2$, CAP gains 10.6 percentage points of singleton coverage over max-split (paired eight-block 95% interval $[9.7, 11.8]$), but leaves 4.73 more candidates on average ($[4.11, 5.48]$). Normalized by each initial pool, the remaining sets are 62.8% versus 44.0%. CAP isolates more targets but leaves larger nonsingleton branches. Increasing tolerance can even make greedy singleton coverage non-monotone because the heuristic chooses from a larger query set, while the exact CAP objective remains monotone.

6.4 An independently seeded scheduling domain reproduces the frontier

The scheduling factorial contains 324 instances with no seed reused across cells. Across 6,048 classes, the median critical loss is 9.5%; certifiability is 54.9% at 10% loss (95% instance-bootstrap CI 52.2–57.7%) and 89.0% at 20%. These closely track the assignment point estimates of 55.5% and 91.3%, despite changing the plan space, objective generator, predicate bank, and randomization scheme. At 10% loss and two answers, CAP covers 14.5% of targets and max-split 3.7%. This is a second finite typed domain, not evidence that the same curve holds for arbitrary optimization software.

Table 3: Four post-hoc public complete-plan audits of archived single-rule outputs in independently reconstructed models. Objective directions and units vary; the witness is a complete plan. ∞ denotes a full-domain raw-rule alias. No row is a full model-generated program.

Case / objective	Archived model-derived rule patch	Ref./cand. optimum	Least witness	Barrier / ℓ
14 / max protein (g)	Daily sodium bound applied per meal	58/58	None in full domain	∞ (alias)
19 / min cost (USD)	**Global two-channel limit applied per supplier**	**\$4,150/\$4,150**	**\$4,880**	**\$730 / 17.6%**
26 / min labor cost (units)	Per-period part-time cap reversed	51.2/61.8	Base optimum	0 / 0%
53 / max score (USD)	Total-spend cap reversed	\$2,510/\$5,190	Base optimum	0 / 0%

6.5 Public model outputs expose coverage and local barriers

On 24 curated OR-Space Revise texts, three local models yield 72 valid typed rules, including 23 errors; 22/24 pools contain the reference. In fixed local microplans, 18/23 errors need positive loss (median 50.0%), which is not a full-program loss.

Table 3 audits four *post-hoc selected* complete-plan tasks by substituting an archived output rule into an independently reconstructed model. Case 19 retains its \$4,150 optimum, but its cheapest disagreement costs \$4,880 (\$730; 17.6%); exhaustive enumeration and CBC/HiGHS agree. Case 14 is a full-domain response alias; cases 26/53 differ at zero loss. Selection and hand-built unchanged rules give existence evidence only (Appendix I).

Five fresh frontier-alias calls per text extract equivalent local rules; five schema-conditioned case-19 IRs also match the reference. Two three-local-model pools omit it. Neither test estimates generated-program error prevalence; protocols are in Appendices F–I, and noise replay in Appendix E.

7 Discussion and limitations

The factorials remain procedural: assignment has eight shared seed blocks, scheduling only two replicates per cell, and the public bridge isolates one numeric rule. Human effort, correlated mistakes, and acceptability of value loss are unmeasured. CAP remains exponential in pool size. At nominal 10% loss, normalization changes assignment certifiability from 13,415/24,192 to 4,224/24,192; within-task witness ordering is unchanged. LP guarantees cover continuous bound changes; integer and structural changes are empirical. OptiBench CLI answer agreement (1623/2186 fields; 404/605 tuples) does not verify programs. Four selected complete-plan cases and one IR task do not estimate error rates. The 50-task reference queue, natural histories and provenance-only errors remain outside scope (Appendix L).

8 Conclusion

Critical loss certifies witness availability for finite complete-plan queries. In generated equal-optimum assignment pairs, 35.6% need extra loss; one selected public case needs \$730. CAP captures answer-budget limits. Program-error prevalence and human cost remain unknown.

References

Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1997.

Christian Bessiere, Remi Coletta, Barry O'Sullivan, and Mathias Paulin. Query-driven constraint acquisition. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, pp. 50–55, 2007. URL https://www.ijcai.org/Proceedings/07/Papers/006.pdf.

Christian Bessiere, Remi Coletta, Emmanuel Hebrard, George Katsirelos, Nadjib Lazaar, Nina Narodytska, Claude-Guy Quimper, and Toby Walsh. Constraint acquisition via partial queries. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, pp. 475–481, 2013. URL https://www.ijcai.org/Proceedings/13/Papers/078.pdf.

Christian Bessiere, Frédéric Koriche, Nadjib Lazaar, and Barry O'Sullivan. Constraint acquisition. *Artificial Intelligence*, 244:315–342, 2017. doi: 10.1016/j.artint.2015.08.001.

Ferdinando Cicalese, Eduardo Laber, and Aline Medeiros Saettler. Diagnosis determination: Decision trees optimizing simultaneously worst and expected testing cost. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 414–422, 2014.

Emir Demirović, Anna Lukina, Emmanuel Hebrard, Jeffrey Chan, James Bailey, Christopher Leckie, Kotagiri Ramamohanarao, and Peter J. Stuckey. Murtree: Optimal decision trees via dynamic programming and search. *Journal of Machine Learning Research*, 23(26):1–47, 2022.

Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011. doi: 10.1613/jair.3278. URL https://arxiv.org/abs/1003.3967.

Sumit Kumar Jha, Susmit Jha, Patrick Lincoln, Nathaniel D. Bastian, Alvaro Velasquez, Rickard Ewetz, and Sandeep Neema. Neuro symbolic reasoning for planning: Counterexample guided inductive synthesis using large language models and satisfiability solving, 2023. URL https://arxiv.org/abs/2309.16436.

Zhong Li, Hongliang Lu, Tao Wei, Yuxuan Chen, Wenyu Liu, Yuan Lan, Fan Zhang, and Zaiwen Wen. Constructing industrial-scale optimization modeling benchmark. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=0Q7adIsv6p.

Xizi Luo, Changhong He, Dongdong Geng, Chenggong Shi, and Yu Mei. Beyond objective equivalence: Constraint injection for LLM-based optimization modeling on vehicle routing problems, 2026. URL https://arxiv.org/abs/2606.04816.

Netlib. LP/DATA: Linear programming test problems. Netlib repository, n.d. URL https://www.netlib.org/lp/data/. Accessed 2026-09-23.

Sumaya Abdul Rahman, Seckhen Ariel Andrade Cuellar, Ghani Raissov, and Mohammad Raza. VeriSimpl: Robust optimization modeling from natural language using simplification-based verification, 2026. URL https://arxiv.org/abs/2607.20474.

Armando Solar-Lezama. Introduction to program synthesis: Lecture 17, introduction to verified functional synthesis. MIT course notes, 2018. URL https://people.csail.mit.edu/asolar/SynthesisCourse/Lecture17.htm.

Dimosthenis Tsouros, Senne Berden, and Tias Guns. Learning to learn in interactive constraint acquisition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 8154–8162, 2024. doi: 10.1609/aaai.v38i8.28655.

Jacobus G. M. van der Linden, Mathijs M. de Weerdt, and Emir Demirović. Necessary and sufficient conditions for optimal decision trees using dynamic programming. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

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. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=fsDZwS49uY.

Chenyu Zhou, Xinyun Lu, Jiangyue Zhao, Jianghao Lin, Dongdong Ge, and Yinyu Ye. OR-Space: A full-lifecycle workspace benchmark for industrial optimization agents, 2026. URL https://arxiv.org/abs/2605.28158.

Reproducibility statement

The supplement preserves generators or source snapshots, frozen registries, raw model outputs, executable analyses, independent checks, and hashes for every reported study, including the post-hoc complete-plan and one-task IR pilots. The invalidated assignment v1 run is retained. Appendix A states proof assumptions; Appendix I gives the evidence boundary.

AI USE STATEMENT

Generative AI tools assisted with hypothesis refinement, experimental design, software implementation, data formatting, quantitative analysis and interpretation, figure and table generation, and manuscript drafting and editing. Reported values were recomputed from sealed artifacts with automated source, hash, and source-to-claim checks. The authors are responsible for the final content.

A PROOFS AND SCOPE

Proposition 1.

Proof. The intended interpretation satisfies every equality in Equation 1. Each filtered set is a subset of its predecessor, so an interpretation absent from the initial pool cannot be added. □

Proposition 2.

Proof. The two interpretations give the same answer to every admissible first query. Inductively, they induce the same history, next policy action, and answer, including for any fixed realization of policy randomness. □

Theorem 3.

Proof. If $\ell(i,j) < +\infty$ and $\epsilon \geq \ell(i,j)$, the stored minimizer lies in $\mathcal{Q}_\epsilon$ and has different responses, so one exact answer distinguishes the pair. If $\epsilon < \ell(i,j)$, no admissible plan has different responses by definition of the minimum, and Proposition 2 applies. A raw pair with $\ell(i,j) = +\infty$ has one full-domain response signature and is one class after deduplication. Target i is distinguishable from every other class exactly when all pairwise inequalities hold, equivalently when $\epsilon \geq \max_{j \neq i} \ell(i,j) = \epsilon_i^\star$. The corresponding witnesses form a finite separating set. □

Proposition 4.

Proof. Let $\epsilon = L_0(x_i^\star)$. Since $\ell(i,j) > \epsilon$, classes i and j agree on every plan in $\mathcal{Q}_\epsilon$. The plan $x_i^\star$ lies in $\mathcal{Q}_\epsilon$, so it is feasible for j. Suppose some j-feasible plan y has $v_0(y) > v_0(x_i^\star)$. Then $L_0(y) < \epsilon$, so $y \in \mathcal{Q}_\epsilon$ and y is also feasible for i, contradicting the optimality of $x_i^\star$. Hence the optimal value of j equals that of i. □

Proposition 5.

Proof. If $\ell(i,j) = +\infty$ the bound holds. Otherwise a minimizing plan x has $\rho_i(x) \neq \rho_j(x)$, so it is feasible under exactly one interpretation, say h. Then $v_0(x) \leq v_0(x_h^\star)$, so $\ell(i,j) = L_0(x) \geq L_0(x_h^\star) \geq \min\{L_0(x_i^\star), L_0(x_j^\star)\}$. □

Proposition 6.

Proof. Suppose $\delta_i < \delta_j$. Any i-optimal plan has greater base objective value than every j-feasible plan; it is therefore infeasible for j and separates the pair at loss δ_i. Proposition 5 gives the reverse inequality. The other strict ordering is symmetric. If $\delta_i = \delta_j = \delta$, a separating own-optimal plan gives $\ell = \delta$ by the same lower bound. If every plan in the union of the two own-optimal sets has equal responses, then no separating plan has loss δ; finiteness of the base domain makes the minimum separating loss strictly larger. Distinct full-domain classes guarantee such a plan exists at some finite loss. □

Theorem 7.

Proof. The plans on which the two interpretations disagree are exactly $x \in \mathcal{F}_0$ with $b < a^\top x \leq b'$, so $\ell(i,j) \max\{|v_0^\star|, 1\} = v_0^\star - \sup_{b < t \leq b'} g(t)$. The function g is concave as the value function of a parametric linear program. The finite, attained t_0 gives $g(t_0) = v_0^\star$, and because t_0 is its largest maximizer, g is nonincreasing to its right. Since $t_0 < b < \sup_{x \in \mathcal{F}_0} a^\top x$, the point b is interior to the projected domain, where g is continuous, so the supremum equals $g(b)$. Concavity gives $g(b) \leq g(t_0) + g'_+(t_0)(b - t_0)$ and $g(t_0) \leq g(b) + g'_-(b)(t_0 - b)$. Both derivatives are nonpositive; substituting $b - t_0 = s$ and $g(t_0) = v_0^\star$ gives the two bounds. □

Proposition 8.

Proof. If there is no separating plan, the left side is infinite and the claim is immediate. Otherwise let $x \in \mathcal{F}(i) \setminus \mathcal{F}(j)$. It violates some bound of j, $a_k^\top x > b'_k$, while satisfying $a_k^\top x \leq b_k$; hence $b'_k < b_k$ and $a_k^\top x > \min\{b_k, b'_k\}$. If $s_k \leq 0$, the kth lower bound is zero. Otherwise $a_k^\top x > t_{0,k} + s_k$, so the projected domain extends right of $t_{0,k}$ and m_k is defined by its right derivative. Concavity then gives

$v_0(x) \le g_k(a_k^\top x) \le v_0^\star - m_k(a_k^\top x - t_{0,k}) \le v_0^\star - m_k s_k$. The case $x \in \mathcal{F}(j) \setminus \mathcal{F}(i)$ is symmetric. Taking the minimum over k gives the claim. □

Proposition 9.

Proof. Proceed by induction on b. A singleton identifies its sole target. A nonsingleton identifies none when its budget is zero or no admissible query gives two nonempty branches; this includes empty query banks. Otherwise any adaptive policy can choose an informative admissible root query x. Its answer branches contain disjoint target sets and have budget $b-1$, so the induction hypothesis bounds their combined coverage by the sum in Equation 4. Maximizing over x attains the bound by joining the two optimal subtrees. There are $O(B2^n)$ budgeted subset states and at most $O(Bm2^n)$ precomputed-partition evaluations; actual bitset runtime also depends on the number of machine words per subset. □

The propositions assume a finite shared base domain, correct candidate evaluators, and exact answers. They certify identification only within the supplied candidate pool. They do not certify candidate construction, natural-language parsing, user acceptability, or correctness beyond the query interface.

Table 4: Evidence map. Each row has a distinct sampling unit and claim boundary.

Study	Unit	Established	Outside scope
Three-domain frontier	240 bases across two sealed studies	Critical-loss transition and task-unit impact	Natural errors; human cost
Assignment factorial	1,296 rows in eight shared seed blocks	Structural associations and policy tradeoff	Independent-row uncertainty; language
Post-hoc pair replay	278,208 class pairs within those 1,296 rows	Optimum-floor and extra-loss decomposition	Natural pair prevalence; independent pair sampling
Scheduling factorial	324 unique-seed bases	Cross-domain frontier and exact coverage	Arbitrary programs; human cost
Public microprobe	24 revisions and 72 actual outputs	Typed-pool coverage and local barriers	Full-program loss; natural histories
Complete-plan bridge	4 post-hoc public revisions	One hidden model-derived rule error at 17.6% loss; one alias; two zero-loss errors	Population rates; generated full programs
Full-model IR pilot	1 public revision, 5 frontier calls	All five complete schema-conditioned IRs match reference	Open-ended programs; cross-task rates
Frontier CLI rule probe	24 revisions, five fresh sessions each	Bounded typed-rule equivalence for 120 outputs	Fixed snapshot; full programs
Public LP bound audit	21 source rows and 63 constructed rule pairs	Continuous-LP slack mechanism on source models	Natural revisions; generated programs
OptiBench CLI pass	605 questions, one response each	Released-answer agreement with archived raw outputs	Program semantics; fixed snapshot; repeated draws
Noise replay	324 scheduling bases	Exact sensitivity to independent symmetric flips	Human or correlated errors

B THREE-DOMAIN FRONTIER AND CAP AUDIT

Assignment instances enumerate seven jobs on three machines with capacity three (1,050 base-feasible assignments). Scheduling instances enumerate eight jobs in four exactly filled two-job slots (2,520 schedules). Knapsack instances enumerate twelve binary items under a generated capacity. Positive integer utilities, profits, and weights make normalized losses lie in $[0, 1]$. Atomic rules cover assignment and forbidden pairs, equality and load bounds; scheduling windows, precedence and nonadjacency; and knapsack inclusion, exclusion, implication, xor, and group coverage.

Each instance samples an old rule, a new rule, and two shifted alternatives. Ten raw candidates encode empty or retracted, stale, overwrite, append, shifted, conservative-union, and swapped combinations.

The construction filter requires at least six distinct full-domain response signatures and distinct append, overwrite, and retract targets. Selected instances contain 8–10 semantic classes (median 10). For each nonconstant partition, the artifact stores its minimum-loss complete plan; these representatives reconstruct every pairwise critical loss and policy replay.

The independent policy study contains 1,188 semantic targets. QA recomputes all 2,880 instance–tolerance–budget CAP values and the corresponding baselines. CAP never returns lower singleton coverage than any stored deterministic tree and is strictly higher than all three in 612 cells. This is a recurrence check, not a statistical comparison over independent policy cells.

Table 5: Concrete impact of the hardest critical witness for 360 targets in the independent study. Values are domain-specific medians.

Domain	Critical loss	Objective units	Decision changes
Assignment	6.84%	12.0	2.0
Scheduling	6.60%	14.0	2.0
Knapsack	13.07%	20.0	3.0

Decision changes are Hamming distance from the nearest base-optimal complete decision, so alternative optima do not inflate the count. Objective units are native generated utility or profit units and are not comparable across domains. Registered QA regenerates every base, matches full digests, replays 5,285 pair witnesses, checks semantic deduplication, and verifies monotonic pair coverage and certifiability.

C Assignment factorial and shared-seed audit

The grid has $3 \times 3 \times 2 \times 3 \times 3 = 162$ cells: atomic-rule count 4/6/8, requested semantic-class count 8/16/32, unique/multiple base optimum, update depth 1/2/3, and append/overwrite/retract. Eight seeds per cell give 1,296 base instances. Plans are all 3^6 task–machine assignments filtered by capacity. Candidate conjunctions assign each typed predicate an ignore, require, or forbid state, and full-domain signatures define semantic classes. Classes, rather than syntactic variants, receive uniform weight. Table 6 labels its separate class and registered-update denominators; its n values are target counts, not independent base samples.

The analysis contains 24,192 class targets. Certifiability rises from 4.9% under strict optimality to 23.0%, 55.5%, and 91.3% at 5%, 10%, and 20% loss. Every instance digest and 5,184 policy cells is regenerated in QA. CAP weakly dominates each stored tree for singleton coverage and is strictly better in 3,323 cells; Table 2 shows why this does not imply uniformly smaller residual sets.

The registered rows reuse the same eight seeds across factor cells. The original class-within-row bootstrap understated uncertainty; the final audit resamples eight complete seed blocks. Table 7 reports percentile-bootstrap ranges. With only eight blocks, these ranges describe sensitivity to the observed seed blocks and do not establish nominal 95% coverage. Factor marginals are descriptive because the unique/multiple condition also changes objective coefficient distributions.

The percentage-loss frontier also depends on its denominator. Table 8 replays every stored class threshold using either the original base-optimum magnitude or the translation-invariant range of feasible base objective values. A nominal 10% has different raw-unit budgets under these conventions; the comparison is a sensitivity analysis, not a controlled change in the query algorithm.

The first sealed preflight stopped at instance 55 because one three-rule append produced an infeasible target. Its exposed seeds were used only for regression. The repaired generator resamples deterministically until base and target are executable; the formal test uses previously unused seeds. No preflight outcome enters the estimates.

D Independent scheduling confirmation

The scheduling registry contains $3 \times 3 \times 2 \times 3 \times 3 \times 2 = 324$ instances with unique formal seeds. All 720 six-job permutations are enumerated. Predicates cover precedence, first/last half, adjacency, and separation; semantic signatures are deduplicated over the full plan domain. QA checks source and raw hashes, seed uniqueness, optimum multiplicity, class counts, and all 67,824 pair certificates.

Table 6: Marginal controls in the registered assignment factorial. "Class" rows uniformly weight all semantic classes; "Update" rows use one prespecified intended update per instance. n is the row denominator, so the two panels are not directly pooled. At 0/10% means certifiable within that value loss. Other factors are balanced; these descriptive marginals do not isolate optimum multiplicity from its coefficient generator.

Target unit / control	Level	n	Median $\epsilon^\star$	At 0%	At 10%
Class: atomic rules	4	8,064	9.1%	3.6%	58.3%
	6	8,064	9.1%	4.9%	57.9%
	8	8,064	9.9%	6.2%	50.1%
Class: pool size	8	3,456	7.1%	10.0%	69.8%
	16	6,912	8.8%	6.3%	60.2%
	32	13,824	10.1%	2.9%	49.5%
Class: base optimum	unique	12,096	9.4%	1.2%	53.7%
	multiple	12,096	9.1%	8.6%	57.2%
Update: rules changed	1	432	3.6%	18.3%	89.4%
	2	432	4.8%	16.2%	78.5%
	3	432	6.1%	15.3%	70.1%
Update: operation	append	432	6.5%	5.6%	67.8%
	overwrite	432	6.1%	15.7%	75.0%
	retract	432	2.9%	28.5%	95.1%

Table 7: Shared-seed sensitivity analysis. Point estimates are unchanged; intervals resample the eight complete seed blocks.

Endpoint	Estimate (%)	Seed-block 95% CI (%)
Certifiable at 0% loss	4.9	[2.9, 7.0]
Certifiable at 5% loss	23.0	[16.0, 33.6]
Certifiable at 10% loss	55.5	[48.2, 65.2]
Certifiable at 20% loss	91.3	[88.7, 94.3]
CAP, 10% loss	14.1	[13.6, 14.6]
Max-split, 10% loss	3.4	[2.6, 4.1]
CAP, 20% loss	15.0	[14.6, 15.3]
Max-split, 20% loss	1.6	[1.2, 2.0]

Table 8: Assignment-factorial normalization sensitivity over 24,192 class targets. The base-optimum and base-range columns use different raw-unit budgets at the same nominal percentage. Intervals resample eight complete seed blocks.

Nominal loss	Base-optimum count	Rate [block CI]	Base-range count	Rate [block CI]
0%	1,184/24,192	4.9 [2.9, 6.9]	1,184/24,192	4.9 [2.9, 6.9]
5%	5,558/24,192	23.0 [15.9, 33.7]	1,848/24,192	7.6 [4.0, 11.8]
10%	13,415/24,192	55.5 [48.1, 65.4]	4,224/24,192	17.5 [14.8, 20.6]
20%	22,082/24,192	91.3 [88.7, 94.3]	11,355/24,192	46.9 [42.9, 51.4]

Table 9: Second-domain scheduling confirmation. Intervals resample 324 independent instances and retain all classes within an instance.

Scheduling endpoint	Estimate (%)	95% CI (%)
Certifiable at 0% loss	6.6	[5.4, 7.9]
Certifiable at 5% loss	27.5	[25.2, 29.9]
Certifiable at 10% loss	54.9	[52.2, 57.7]
Certifiable at 20% loss	89.0	[87.3, 90.7]

The median critical loss is 9.5% and the 10%-loss estimate is 54.9% (95% CI 52.2–57.7%). Unique and multiple conditions yield 50.1% and 59.7% certifiability at 10%, but their coefficient distributions differ, so this marginal contrast is not causal. CAP breaks coverage ties by maximum branch size, squared branch sizes, current loss, and canonical mask. Max-split uses the same order without recursive coverage.

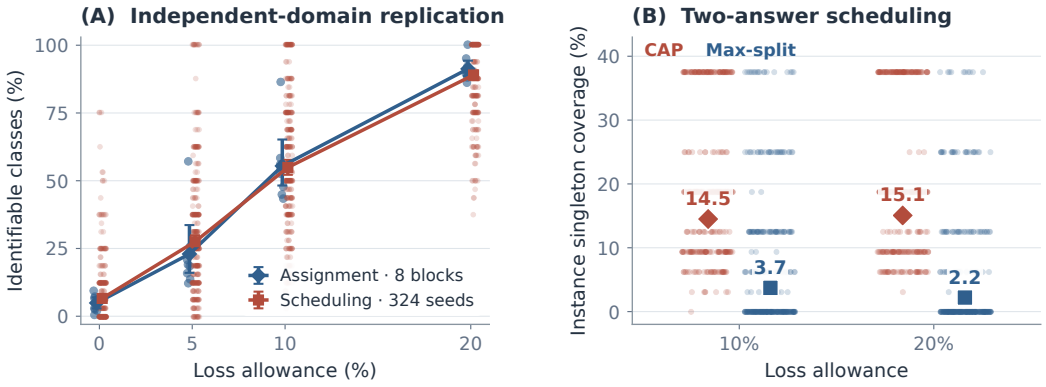


Figure 5: Independent scheduling confirmation. (a) Assignment block rates and scheduling instance rates at each tolerance, with 95% intervals resampling eight assignment blocks or 324 independent scheduling instances. (b) Instance-level two-answer scheduling singleton coverage and class-weighted aggregates. Model-output and answer-noise results are tabulated separately in Appendices F and E.

E Answer-noise replay

For flip probability p and odd repeat count $r \in \{1, 3, 5\}$, majority error has exact probability

$$p_r = \sum_{k=(r+1)/2}^{r} \binom{r}{k} p^k (1-p)^{r-k}.$$

Dynamic propagation keeps the true target identity after a wrong branch, separating correct singletons, false singletons, retained unresolved targets, and excluded unresolved targets. The model assumes independent symmetric flips; intervals resample instances rather than answer trajectories. With exact answers, scheduling CAP has 14.5% correct singletons and no false singleton at 10% loss. Under independent 5% flips, one answer per query yields 13.1% correct and 8.0% false singletons. Three-answer majority vote gives 14.3% correct and 1.2% false singletons, increasing expected raw answers from 1.99 to 5.98. At 10% flips, five-answer voting still leaves 1.4% false singletons. Max-split produces fewer false singletons at the same settings but also much lower correct identification (Table 20); the replay does not model correlated or human errors.

F Public revision microprobes

The bridge fixes 24 OR-Space Revise documents whose new hard rule has an explicit numeric parameter, direction, and aggregation scope. Selection and references were sealed before model calls but followed source inspection, so the set is curated rather than a blind benchmark split. Qwen2.5-3B, Phi-4-mini, and Qwen3-8B each contribute one greedy typed output per context. All 72 parse; 49 match the reference and 23 are executable errors, producing 22/24 covered three-model pools.

The prompt requests one JSON object with an inequality operator, numeric threshold, and aggregation scope. A deterministic parser admits only schema-valid triples; it does not repair a model output. Table 11 reports model-specific outcomes. The 23 errors contain 17 scope errors, five direction errors, and one joint threshold-and-scope error.

Each bank contains 12 two-component counterfactual plans. Global rules act on the sum; per-period or per-entity rules act componentwise. For upper bounds, local value increases with activity; for lower bounds, it decreases with activity. Of the 23 wrong candidates, 18/23 require positive local loss to expose and their median threshold is 50.0%.

Table 10: Exact CAP sensitivity at 10% loss and two logical questions. Raw answers include majority-vote repetition.

Flip	Repeats	Correct singleton	False singleton	Raw answers
0%	1	14.5%	0.00%	1.99
0%	3	14.5%	0.00%	5.98
0%	5	14.5%	0.00%	9.96
1%	1	14.2%	1.64%	1.99
1%	3	14.5%	0.05%	5.98
1%	5	14.5%	0.00%	9.96
5%	1	13.1%	8.04%	1.99
5%	3	14.3%	1.19%	5.98
5%	5	14.5%	0.19%	9.96
10%	1	11.8%	15.68%	1.99
10%	3	13.7%	4.55%	5.98
10%	5	14.3%	1.40%	9.96

Table 11: Per-model typed-candidate outcomes on the same 24 curated public revision contexts. Counts are descriptive and paired by context.

Model	Parsed	Exact rule	Executable wrong
Qwen2.5-3B	24/24	13	11
Phi-4-mini	24/24	16	8
Qwen3-8B	24/24	20	4
All outputs	72/72	49	23

Table 12: Public benchmark revision bridge. Pool coverage means that at least one of three actual outputs matches the typed reference rule.

Public microprobe endpoint	Result
Revision contexts	24
Actual model outputs	72
Pools containing exact rule	22/24
Executable wrong candidates	23
Wrong candidates with positive local barrier	18
Median local $\epsilon^\star$	50.0%

The evaluator exposes objective values rather than a rule-level reference program. Consequently, this study measures candidate coverage and constraint-local disagreement only. It does not establish full-program $\epsilon_i^\star$, natural-history prevalence, or end-to-end repair success.

The two uncovered contexts make the coverage boundary concrete. For `max_pairs`, the reference is a global-total upper bound of two, while all three models emit a per-component upper bound of two. For `max_distinct_patterns`, all three again replace a global-total rule by a per-component rule, and Phi-4-mini also changes the threshold from four to five. These are typed, executable interpretations; filtering cannot recover the missing reference from either three-model pool.

G SEEDED LOCAL-MODEL REPEAT SAMPLING AND RULE CERTIFICATES

To test whether simple resampling repairs candidate coverage, we froze the same 24 documents and JSON prompt, then drew five responses per document from Qwen3-8B with distinct seeds, temperature 0.7, top-p 0.9, the checkpoint's top-k 20, and a 96-token cap. All 120 outputs parse. The five full response strings are identical within each document, so no new typed candidate enters any pool and coverage stays 20/24 for every prefix (Table 13). A post-hoc open-ended control at the same sampling settings produced four distinct strings in five draws, confirming that sampling was active. This negative finding is conditional on the model, prompt, and decoding setup; it is not a claim about frontier-scale models or all stochastic decoders.

Table 13: Exploratory Qwen3-8B repeat sampling on the 24 frozen public revision contexts. "Equivalent" means the one extracted rule agrees with its typed reference throughout the bounded continuous two-component microdomain, not that complete programs agree.

Draws per context	Outputs	Parsed	Equivalent	Pools covered
1	24	24	20	20/24
2	48	48	40	20/24
3	72	72	60	20/24
4	96	96	80	20/24
5	120	120	100	20/24

For each parsed rule we compared reference and candidate in both directions over $0 \le a, b \le 2T$, where T is the reference threshold. Each rule is a conjunction of one or two rational halfspaces. Enumerating all vertices of the bounded reference-feasible and candidate-feasible polygons finds a counterexample whenever an opposing inequality has strictly positive maximum violation. Across the 120 sampled outputs, 100 are equivalent on this declared microdomain and 20 have a rational counterexample. The same audit gives 49 equivalent and 23 non-equivalent among the original 72 greedy outputs. We independently re-evaluated all 48 directional witnesses with the rule-response function; no 12-plan-grid alias hid a bounded-domain disagreement in these records. This strengthens the local rule check but supplies neither a full-program reference nor variable alignment. Raw generations, model snapshot and seeds, certificates, and QA are preserved in `REPEAT_SAMPLING_RAW.jsonl`, `REPEAT_SAMPLING_PROTOCOL.json`, `REPEAT_SAMPLING_SEMANTIC_CERTIFICATES.json`, and `REPEAT_SAMPLING_SAMPLER_CONTROL.json`.

H FRESH-SESSION FRONTIER-ALIAS RULE EXTRACTION

We used the logged-in Codex CLI with the `gpt-6-astra` model alias for the same 24 frozen OR-Space Revise documents. Each of five calls per document was a fresh ephemeral session containing only that document, the original three-field rule-extraction prompt, and a JSON output schema; the reference operator and aggregation label were withheld. Calls ran on 23 September 2026. The CLI reported version `0.155.0-alpha.16`, but it did not expose an immutable model snapshot, temperature, or sampling seed. We therefore describe these as five fresh calls, not five controlled stochastic samples. The five calls for a document ran concurrently, and every completed session has its own raw JSONL event stream, stderr, final message, and SHA-256 record in `PUBLIC_REVISION_FRONTIER_CLI_v1.zip`.

All 120 outputs parse as the requested triple; all 120 match the frozen typed reference and are equivalent in both directions over $0 \le a, b \le 2T$ under the exact rational halfspace check. Every

context is covered by its first call; all five complete output strings are identical within each context. Table 14 compares this descriptive endpoint with the separately seeded local-model diagnostic. Their decoding protocols and model access differ, so the table is not a controlled model-ranking test. The bounded rule check says nothing about the rest of each optimization program, naturally occurring revision histories, or full-program $\epsilon^\star$.

Table 14: Five-call typed-rule diagnostics on the same 24 curated texts. "Equivalent" is exact on the declared bounded two-component rule domain only. Pool columns count contexts containing an equivalent rule among the first k calls. †Contexts with more than one distinct typed output across five calls.

Model / five-call setup	Valid	Equivalent	Wrong	Pool $k{=}1$	Pool $k{=}5$	Varied†
Qwen3-8B, seeded	120/120	100/120	20/120	20/24	20/24	0/24
GPT-6 Astra, CLI alias	120/120	120/120	0/120	24/24	24/24	0/24

I POST-HOC COMPLETE-PLAN BRIDGE AND FULL-MODEL IR PILOT

We asked whether archived *actual* local-model rule outputs separate interpretations on complete plans, rather than on the earlier two-component microdomain. After inspecting three tractable cases (14, 26, and 53), we added case 19 to test for a positive complete-plan barrier. This adaptive four-case selection cannot estimate an error rate or a barrier distribution. We independently reconstructed the unchanged rules and objective from the public OR-Space Revise documents and CSVs; the benchmark release provides no executable revised reference program. Each archived output replaces only its named rule in that hand-built model. Thus these are model-derived rule patches in complete task models, *not* complete programs generated by the models.

Table 15: Complete-plan checks for four post-hoc selected public revisions. "Ref./cand." gives each interpretation's optimum on its own feasible plans; values have task-specific units and are not comparable across rows. Critical loss is computed on the shared revision-context carrier domain after removing only the tested rule. An infinite value means both responses agree on every complete plan. These four tasks are not a random sample.

OR-Space	Archived local-model rule output	Complete-plan relation	Optimum ref./cand.	ℓ
14	daily sodium bound applied per meal	Equivalent: all six units contain at most 474 mg, below 1,500 mg	58/58 g protein	∞
19	weekly two-channel budget applied per supplier	Candidate admits three-supplier plans; cheapest witness costs \$4,880	\$4,150/\$4,150	17.6%
26	per-period part-time cap reversed	Different optima and a separating base optimum	51.2/61.8	0
53	total-spend cap reversed	Different optima and a separating base optimum	\$2,510/\$5,190	0

Case 19 supplies an optimum-preserving error on the full task. The source requires at most two active manufacturer–channel combinations across all suppliers, while the archived Qwen2.5-3B output applies the bound per supplier. Because each supplier already chooses at most one channel, the latter admits all 105,694 complete plans satisfying the other rules, versus 4,838 under the reference. Both optima cost \$4,150, achieved by ten regular-channel orders from supplier C. The least-cost disagreement orders two batches from A, one from B, and six from C, all through regular channels: 100 chairs for \$4,880. Its exact normalized loss is $730/4150 = 73/415$ (17.6%). A separate integer-program encoding solved by both CBC and HiGHS matches the exhaustive optimum and witness cost; all reported witnesses were re-evaluated against the source-derived constraints.

The negative and zero-loss cases are equally informative. In case 14, any two 300-gram meals contain at most $6 \cdot 79 = 474$ mg of sodium, below the 1,500 mg daily limit; therefore the archived per-meal mistake and the global rule are response-equivalent on every complete plan. Their $\ell = \infty$ is a property of the two *raw* rule representations; after full-domain deduplication they are one semantic class. In cases 26 and 53, the reversed bound produces complete-plan semantic differences, but the cheapest witness is already a base optimum and the candidate's optimum differs from the reference. These results show why the positive microdomain counts cannot simply be relabeled full-program losses. The enumeration, two-solver checks, original model output rows, source hashes, and explicit post-hoc selection record are archived in `FULL_PROGRAM_EVIDENCE_v1.zip`.

Separately, we asked a frontier CLI alias to generate the *whole* case-19 model in a fixed executable, task-specific IR: all total-chair bounds, supplier dependencies, budget direction/scope, cost components, and objective direction. The plan schema itself enforces one channel per active supplier, and numerical prices come from the CSV. Five fresh `gpt-6-astra` sessions produced valid IRs; all five were identical to the independently written reference IR and matched its feasibility response and optimum on the complete shared plan domain. The alias again exposes no immutable snapshot, temperature, or seed. Three local models' initial free-form outputs failed the specified IR schema; after an explicitly recorded schema-visible prompt adaptation, only Qwen3-8B returned a valid IR, which was also equivalent. We retain both local prompt versions and all failures, without treating the adapted calls as repeats of the first pass. This one-task schema-conditioned diagnostic supplies actual generated-model records and semantic verification, but no positive generated-program error, no open-ended Python-program audit, and no cross-task full-program error prevalence.

J ADDITIONAL AUDITED ANALYSES

These analyses replay sealed records; they add no benchmark instances or model generations. Source hashes and code are given in `FOLLOWUP_ANALYSIS.json` and `run_stored_followups.py` in the local research package. The full original inline review notes are preserved in `ORIGINAL_INLINE_COMMENTS.txt` and `ORIGINAL_APPENDIX_COMMENTS.txt` in this Overleaf project.

The v24 pair decomposition is a separate post-hoc replay of the same sealed assignment data, not a prospective experiment. The script `audit_pair_decomposition_v24.py` verifies the raw SHA-256 seal, reconstructs every class's own optimum from the stored rules and full assignment domain, and checks that every recorded witness truly separates its pair at the recorded loss. It finds 0 floor violations and 0 witness failures across 278,208 unordered pairs. Equal optimum means $|\delta_i - \delta_j| \le 10^{-9}$; positive extra loss means $\Delta_{\text{extra}} > 10^{-9}$. A 2,000-draw bootstrap resamples all 162 factor rows in each of the eight shared seed blocks. The pair is the descriptive counting unit, but the seed block is the uncertainty unit; the interval should not be read as 278,208 independent observations.

Table 16: Post-hoc pairwise mechanism strata from the assignment factorial. Rows in the upper and lower panels each partition the same 278,208 pairs, so they must not be added across panels. "Extra-positive" is $\ell(i,j) > \min(\delta_i, \delta_j)$; the final column conditions on equal own-optimum values. The update family denotes the instance cell, not a claim that both members of each pair implement that update.

Assignment stratum	All pairs	Same-value pairs	Extra-positive	Extra-positive / same
Pool size 8	12,096	1,819	752	41.3%
Pool size 16	51,840	7,006	2,493	35.6%
Pool size 32	214,272	26,281	9,251	35.2%
Append cell	92,736	11,756	4,282	36.4%
Overwrite cell	92,736	11,565	3,873	33.5%
Retract cell	92,736	11,785	4,341	36.8%

For each target class i, the strongest floor implied by the other classes is $F_i = \max_{j \neq i} \min(\delta_i, \delta_j)$. The target threshold exceeds F_i for 11,252/24,192 classes (46.5%) and 725/1,296 registered update targets (55.9%). These target-level rates are not interchangeable with the 12,496/35,106 positive-extra rate among equal-value *pairs*. For positive-extra pairs, the median additional loss is 3.19% of the base optimum, 2.62 raw assignment-utility units, or 7.36% of the base objective range. The original normalization gives 13,415/24,192 classes identifiable at 10% of $\max(|v_0^\star|, 1)$; using 10% of the attainable base range $v_0^\star - \min_{x \in \mathcal{F}_0} v_0(x)$ gives 4,224/24,192. The latter scale is translation invariant but asks a different operational question; zero/positive extra classifications and within-instance witness ordering remain unchanged. All counts and subgroup medians are in `PAIR_DECOMPOSITION_v24.json`.

Table 17: Stored assignment-factorial mechanism audit. The 95% intervals resample eight complete seed blocks. Correlation is descriptive; all 278,208 pair certificates satisfy Proposition 5's optimum-loss floor. Only 12 changed numeric-bound cases meet the LP theorem's positive-slack premise, and all have slack one, so a slack correlation is not estimable.

Update family	Classes	$\rho(\epsilon^\star, L_0(x_i^\star))$ [95% CI]
Append	8,064	0.821 [0.797,0.839]
Overwrite	8,064	0.827 [0.804,0.844]
Retract	8,064	0.821 [0.801,0.838]
All	24,192	0.823 [0.799,0.839]

Table 18: Singleton coverage (%) on 1,188 independent-study semantic targets at 10% loss. Survivors are class-weighted means at two answers. GBS, one-step information gain, and EC^2 edge cutting coincide with max-split for uniform classes and deterministic binary responses under the same tie break. †The direct-rule oracle instead queries rule membership over 1,200 raw typed candidates (ten per base), so its row is a different interface and denominator.

Policy	B=1	B=2	B=3	B=4	Survivors B=2
CAP (exact)	4.3	23.1	50.9	62.9	4.59
Max-split / GBS / IG / EC^2	0.0	0.9	42.3	62.9	2.78
Gain per loss	0.4	7.4	35.4	57.2	3.26
Loss scan	0.5	9.0	34.6	55.7	3.44
Random mean (32)	0.4	10.1	34.5	52.9	3.47
Direct-rule oracle†	0.0	10.0	60.0	100.0	2.80

Table 19: Exact CAP replay cost on the assignment factorial at 10% loss and two answers. Times are per-instance medians of three warmed single-process Python wall-time runs on a 13th Gen Intel Core i9-13980HX; they exclude instance construction and are descriptive of this implementation. Every recomputed singleton count and visited-state count matches the sealed record.

Classes	Cases	Partitions med/max	States med	Time med/max (ms)
8	432	6/14	37	0.027/0.10
16	432	10/26	80	0.074/0.39
32	432	11/41	155	0.122/1.11

Table 20: Max-split on the same 324 scheduling instances, 6,048 class targets, two logical questions, and independent symmetric flip model as Table 10. Probabilities are exact expected fractions, not Monte Carlo frequencies; three or five independent answers are combined by majority vote.

Flip	Repeats	Correct singleton	False singleton	Raw answers
0%	1	3.72%	0.00%	1.98
5%	1	3.36%	1.11%	1.98
5%	3	3.67%	0.16%	5.95
10%	1	3.02%	2.20%	1.98
10%	3	3.52%	0.62%	5.95
10%	5	3.66%	0.19%	9.92

K SOURCE-DERIVED CONTINUOUS-LP SLACK AUDIT

The original LP models come from the public Netlib collection (Netlib, n.d.). We fixed eight named SIF inputs before the audit and archived their SHA-256 digests and the source-mirror commit in `NETLIB_SLACK_REGISTRY.json`. Seven models parse and solve with our MPS reader; `BLEND` is a recorded parse failure. One of the seven, `SC50B`, has no inequality meeting the prespecified headroom threshold. For each inequality $a^\top x \le B$ in the remaining models, we optimize $a^\top x$ over the original optimum face and over the full original feasible set $\mathcal{F}_0$, obtaining t_0 and $t_{\max}$. Among rows with positive numerical headroom $t_{\max} - t_0$, a fixed hash order selects at most four per model, or 21 rows total. We then set a tighter bound $b = t_0 + q(t_{\max} - t_0)$ for $q \in \{0.25, 0.50, 0.75\}$, keeping the original model as the looser interpretation. These are constructed changes to real LP rows, not natural rule revisions or model-generated errors.

All 63 pairs keep the original optimum while admitting a strict feasible-set disagreement. The infimum separating loss is obtained by solving the LP on $a^\top x = b$; a second LP over $a^\top x \ge b$ gives the same value. All 21 paired rows have nondecreasing loss over the three q values. Across rows, the median normalized loss is 0.35%, 0.76%, and 1.25% respectively; these are descriptive paired medians, not an estimate of a task population. For every case, finite-difference secants near t_0 and b bracket the measured loss as implied by the concavity argument. HiGHS dual-simplex and interior-point solves agree at the declared tolerance in 63/63 cases; independent CBC re-solves agree within 10^{-7} relative objective error in 63/63. We also save an explicitly strict separating witness for each case; the largest base-domain residual is 9.31×10^{-10}. Numerical agreement is a mechanism check, not a proof of exact equivalence or of natural-error prevalence.

Table 21 replays those 63 cases. At each q, ranks compare the 21 selected source rows using normalized exposure loss and the normalized local bound $m_0 s / \max(|v_0^\star|, 1)$, where $m_0 = |g'_+(t_0)|$ is estimated by the stored near-endpoint secant. Because raw slack is measured in differently scaled row units, we do not use a cross-row slack-only comparison. Leave-one-source-LP-out ranges show whether a single source model dominates the rank association; the lower/actual quantiles show bound tightness beyond correlation. These are *constructed, in-sample* diagnostics, not prospective prioritization. Solver tolerances can place the numerical "lower" estimate up to 6.6×10^{-5} above the measured value in relative terms.

Table 21: Post-hoc ranking and bound tightness on the Netlib construction. q is the fixed fraction of each row's feasible headroom; each line contains the same 21 selected rows from six LPs. ρ is Spearman rank correlation, with the range after omitting each source LP in turn. Lower/actual reports median and tenth percentile ratios of the local secant bound to measured exposure loss. Rows recur across thresholds and are not independent task samples.

q	Rows/LPs	Median ℓ	$\rho(m_0 s, \ell)$	Leave-one-LP ρ range	Median lower/actual	10th pct. lower/actual
0.25	21/6	0.35%	0.999	[0.998, 1.000]	1.000	0.975
0.50	21/6	0.76%	0.988	[0.980, 0.993]	1.000	0.393
0.75	21/6	1.25%	0.973	[0.958, 0.985]	0.837	0.401

L OPTIBENCH ANSWER AUDIT AND FULL-PROGRAM GATE

The official OptiBench JSON contains 605 records, each with exactly `index`, `question`, `type`, and `results`; `results` is an answer dictionary, not an executable formulation. None supplies reference variables, objective or constraint code, or a semantic-equivalence certificate. Table 22 gives the released type counts. The source SHA-256, repository commit, schema audit, and a deterministic pre-annotation selection of 40 linear-no-table and 10 linear-table tasks are stored in `OPTIBENCH_SCHEMA_AUDIT.json`. Selection does not make these 50 tasks reference-eligible: independently authored and cross-checked formulations still need to be built. Because 183 records are labeled nonlinear, a two-MILP counterexample check cannot cover the whole release.

L.1 SINGLE-PASS FRONTIER-ALIAS ANSWER AUDIT

We queried all 605 released questions once on 23 September 2026 (UTC) through a logged-in Codex CLI session with model alias `gpt-6-astra`, without a separate API key. Contiguous groups of 20 questions shared one model context, so responses within a group are dependent. Each

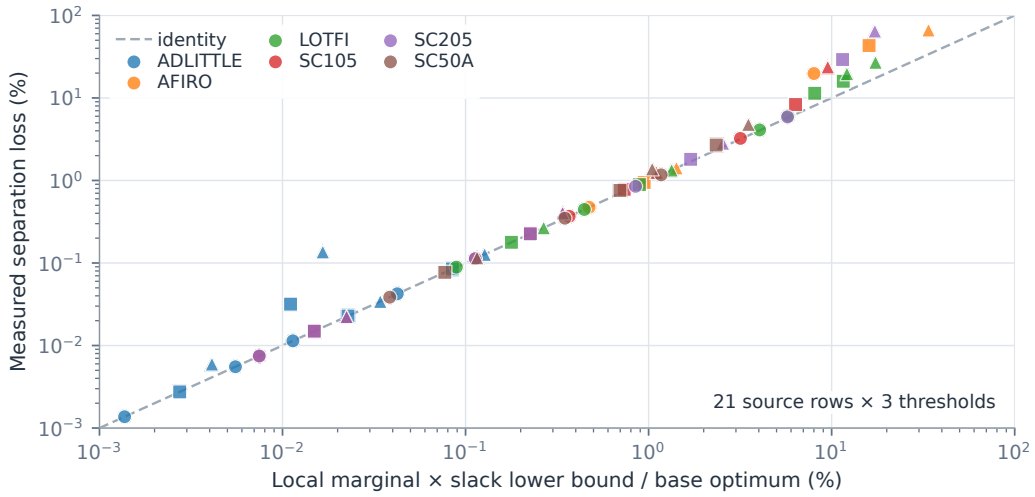


Figure 6: Measured exposure loss versus its local marginal-price bound for 63 constructed row–threshold pairs from six source LPs. Colors identify source models and symbols the three prespecified headroom fractions. The dashed diagonal marks equality; vertical gaps above it quantify bound looseness. Logarithmic axes use the same base-optimum normalization. No language-model output or historical revision is used.

Table 22: Schema audit of the 605 released OptiBench records. All four categories provide question text and answer dictionaries but zero executable reference programs or variable schemas. "Linear" and "nonlinear" are the release labels, not a verifier-certified classification.

Released type	Records	Reference programs
Linear, no table	342	0
Linear, with table	80	0
Nonlinear, no table	133	0
Nonlinear, with table	50	0
All	605	0

retained prompt contains question text and requested answer-field names, but withholds released answer values; prompt withholding does not rule out pretraining exposure. The structured response includes a formulation *string* and answer fields. These strings were not executed or independently verified. The CLI did not expose a dated backend snapshot, and this is neither five independent draws per question nor the preregistered fixed-snapshot full-program study. The archive `OPTIBENCH_FRONTIER_CLI_v1.zip` contains the exact prompts, JSONL events, responses, model-alias and timing metadata, scoring code, and archive QA; accepted event streams have no tool calls.

Table 23: Single-pass answer agreement with released OptiBench values by release category. "Cautions" counts model responses marked ambiguous or insufficient. Every released field and every task remains in the denominator, including missing or nonnumeric outputs. This table measures answer-tuple matching, not formulation correctness.

Release category	Tasks	Cautions	Fields agree	Tuples agree
Linear, prose	342	35	896/1117	266/342
Linear, table	80	13	326/435	50/80
Nonlinear, prose	133	29	279/437	63/133
Nonlinear, table	50	16	122/197	25/50
All	605	93	1623/2186	404/605

A field matches when its parseable numeric answer differs from the released value by at most $\max(10^{-4}, 10^{-6}|y_{\text{release}}|)$; strict tuple agreement requires all fields to match. Of 2186 released

numeric fields, 1895 model fields are parseable and 1623 match; 404 of 605 complete task tuples match. The model itself marks 76 tasks ambiguous and 17 insufficient. These are descriptive comparisons to a single released answer dictionary, not estimates of semantic correctness.

To check the meaning of a mismatch, we independently re-solved five *post-hoc selected* disagreements (indices 16, 18, 82, 111, and 169; code and exact records in `RELEASE_ANSWER_CHECKS.json`). For index 16, exhaustive integer enumeration gives the model's 33.5-hour transport schedule, while the released feasible schedule takes 34.5 hours. For index 82, the released 22-worker vector misses two daily coverage constraints, whereas the model vector is feasible and CBC confirms 22 as the optimum. For indices 18, 111, and 169, distinct model and released variable tuples are both feasible global optima under independent enumeration or direct re-evaluation. These targeted checks demonstrate that tuple disagreement can coexist with valid model answers or a defective release label; their post-hoc selection gives no population rate. Agreement likewise cannot certify the generated formulation, which has not been aligned with a reference variable schema or checked by bidirectional optimization.

Two cases in the frozen 50-task reference queue make the gate concrete (`OPTIBENCH_PILOT_GATE_EXAMPLES.json`). In index 186, the literal requirement that wages be *below* 150,000 is infeasible: if $n = s + j \geq 100$ and $s \geq j/3$, then $W = 1000n + 2000s \geq 1500n \geq 150{,}000$, while the released answer attains equality. An annotator must adjudicate whether "below" means strictly less or at most. In index 590, a *synthetic* deletion of the wrapping cap preserves the released zero optimum but admits $(30, 0)$ meals, which the stated cap rejects. The ablation is not an LLM output; it demonstrates why objective agreement cannot supply a semantic gold label.

The proposed *OptiBench* objective-correct-program table and full-program loss figure therefore remain unmeasured. The one-task schema-conditioned OR-Space IR pilot in Appendix I does not supply this denominator. A valid linear-task pilot must first produce independently authored and cross-checked reference formulations on a fixed shared variable schema, document disputed labels, then freeze a dated model snapshot where available, prompts, independent request counts, execution sandbox, solver tolerances, and failure categories. A current model alias is reported with its run window rather than treated as an immutable snapshot. Only after generating and aligning candidate programs may bidirectional optimization checks test feasible-set equality; unresolved numerical cases remain unresolved, and nonlinear tasks require a separate verifier. OptiBench has no natural versioned base domain, so a full-program critical loss comparable to Equation 3 also requires a separately fixed $\mathcal{F}_0$ and v_0. The 200-task, three-model, five-sample expansion is conditional on this gate. Neither typed-rule microprobe estimates full-program error prevalence or critical loss.

M Reproducibility and artifact map

Every study contains its registry or source snapshot, raw records, analysis, QA, and hash manifest. The package maps the two three-domain studies to `value_identifiability_frontier_v1` and `value_aware_policy_v1`; the assignment factorial and block correction to `structural_shift_v2` and `structural_shift_blocked_audit_v1`; and the scheduling, noise, and public studies to `structural_shift_scheduling_v1`, `noisy_oracle_sensitivity_v1`, and `public_revision_microprobe_v1`. The 120-call rule audit is archived in `PUBLIC_REVISION_FRONTIER_CLI_v1.zip`; the OptiBench answer audit is in `OPTIBENCH_FRONTIER_CLI_v1.zip`, with summary QA and score files beside the manuscript. Recompilation reuses static artifacts and is not an independent replication. No learned selector checkpoint exists because no selector is trained or claimed.