BENCHMARK GRADERS MISS WHAT AGENTS LEAVE BEHIND

Anonymous authors
Paper under double-blind review

ABSTRACT

Benchmark graders can award passing scores to outputs that violate task requirements. Repairing such errors can change which agent is recommended, but the value of that change depends on performance on new outputs. We study this distinction using repeated coding-agent executions, archived public code generations, and a documented banking-benchmark correction. Additional checks reveal sensitive keys retained in nested outputs and invoices reset after an intervening payment. We compare the original-score and corrected-score choices on the same corrected target in another execution. Across four new model environments, mean selection benefits include gains, no change, and losses. The negative mean persists in one environment when the disputed migration check is omitted. On public code generations, selectors frozen before held-out scoring reduce mean pairwise loss from 0.098 to 0.052 passed tasks, while the full-pool leader remains unchanged. The banking correction also reduces mean selection loss while retaining adverse transfers. These results separate checking task requirements from evaluating the consequence of a revised recommendation. We release deterministic checks, paired score records, and a calculator for reproducing these comparisons.

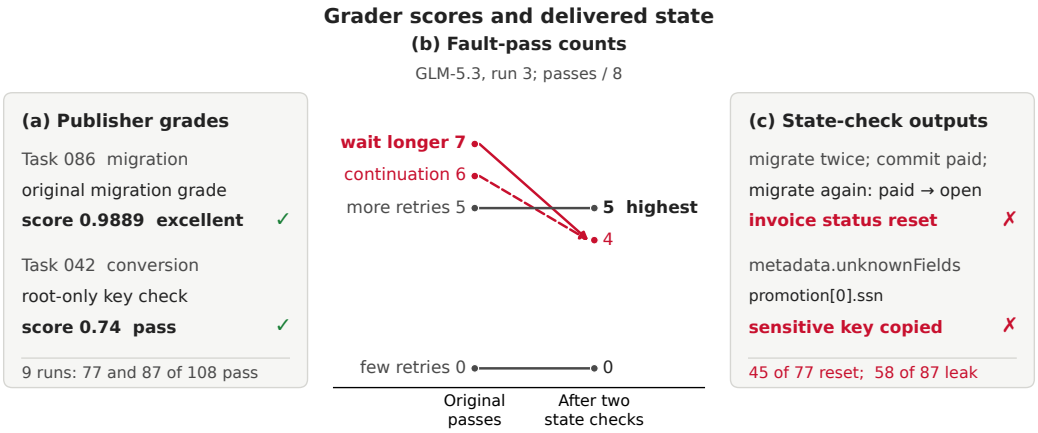


Figure 1: Grader scores, state checks and policy rankings. (a,c) Two accepted outputs from GLM-5.3 execution 3. The migration scores 0.9889; migrating twice, committing invoice inv1 as paid, and migrating once more resets its status to open. The conversion scores 0.74 but retains a synthetic ssn canary at metadata.unknownFields.promotion[0].ssn on the object-list probe. (b) The two checks change fault-band counts from 0/6/5/7 to 0/4/5/4 for few retries, continuation, more retries and wait longer. These are eight assignments per policy on four banded tasks, distinct from the sixteen-assignment selection endpoint. Footer counts aggregate nine executions. The unique-winner replacement uses the authors' state-preservation reading; static reruns revoke zero passes. Section 5 evaluates the resulting choices on other outputs.

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

A passing benchmark score is often used as evidence that an agent completed its task. That inference depends on what the grader inspects. Consider a data-conversion task that requires unknown fields to be preserved while excluding sensitive keys. Its grader checks the root of the output, but 58 of 87 accepted outputs copy prohibited keys into a nested field. A schema-migration task exposes a different gap. After running migration twice, we commit an invoice as paid and run migration once more. Forty-five of 77 accepted outputs restore the invoice to open. Repeating migration without an intervening update revokes no passes. The difference is the payment committed between migration runs, which our state-preservation check makes explicit.

A revised recommendation can be tested directly on other outputs. Revoked credit depends on the errors in a particular execution. We compare the original-score and corrected-score choices on one corrected target in another execution. This common target measures the consequence of changing the recommendation while keeping its evaluation criterion fixed.

Strengthened tests lower held-out public-code selection loss from 0.098 to 0.052 tasks, and a documented banking correction also improves mean selection. Recovery executions show when the recommendation changes help: four new environments yield two positive, one zero and one negative mean benefit. A saved historical GLM choice loses eight of sixteen fault passes on a later execution. Evaluating the choice on other outputs reveals these differences beyond the corrected scores themselves.

Building on evaluator audits and decision-based evaluation (Liu et al., 2023; Tu et al., 2026; Wen et al., 2025), we connect grader omissions to selection gains, unchanged choices and losses. The anonymous supplementary ZIP provides checks, paired scores and a reproduction calculator. Start with `PACKAGE-README.md` and `verify_submission.py` for offline reconstruction and archive scope.

2 EXPERIMENTAL SETTING

Recovery policies. We run Codex CLI against live provider endpoints on tasks from the public Harness-Bench repository. B10 permits five native retries with a ten-second stream-idle limit. R10 adds one continuation after terminal failure, native11 permits eleven retries, and B30 retains five retries with a thirty-second idle limit. Each task is assigned to every policy under nominal service and two controlled 144-second silences, one recoverable and one permanent. Faults are anchored to the first valid response-created event. A cell assigns one task to one policy and service condition; all cells remain in the analysis.

The six-task plan has three GLM-5.3 executions of 72 cells each. Checks for an incorrect largest-overrun claim and a malformed CSV were written after the first execution and then applied unchanged to all three (Appendix F.6). The same checks were subsequently applied to three Kimi executions of the six-task plan. The eight-task plan has nine historical executions: three on GLM-5.3 and two each on Kimi K3, GPT-6 Sol and Qwen3.7 Plus. Three additional executions, one each on GLM, GPT and Qwen, began generation before the final checker was sealed. Every eight-task execution assigns 96 cells (Appendix K). A further study assigns two executions to each of four new model environments under a separately sealed analysis rule; Appendix K.14 records all eight completed executions and the earlier cutoff snapshots.

Scoring. We score each policy on the sixteen fault assignments in the eight-task plan. Original labels count the publisher's pass, good and excellent bands as passing where available, and use a score threshold of 0.60 otherwise. Nominal assignments describe coverage; policy selection uses the fault assignments. We retain every policy tied for the highest score; the primary selector weights these policies equally. A correction withdraws an original pass when a covered requirement is violated.

The final checker, version 5, covers 25 of 50 stated obligations. Its application is fixed within each comparison, but its development includes output-informed requests and synthetic stress tests. Versions 1 and 2 were sealed before their checker-writing process opened execution outputs, after the nine historical executions already existed and while later generations were running. Version 2 revokes five historical passes; neither version changes a maximal set on the twelve earlier executions.

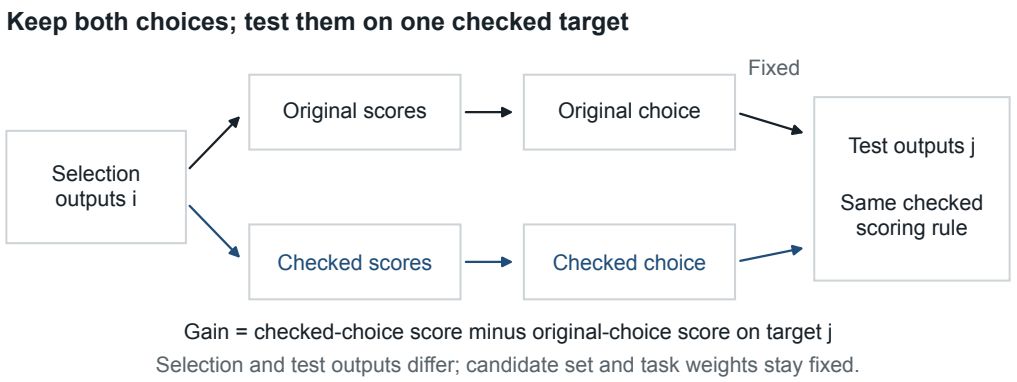


Figure 2: Common-target evaluation of a changed recommendation. Original and checked scores on selection outputs i induce two choices, weighting tied maximizers equally. Both choices are scored under the same checked rule on different test outputs j. Their score difference is the benefit of changing the choice. Candidate sets and task weights stay fixed. Selection and test outputs come from distinct executions or panels, with shared models and tasks; each study reports its development and freezing chronology.

Version 5 was sealed during the three later executions. Appendix K.9 gives the exposure and version chronology.

Migration requirement. Task 086 requests an idempotent migration that does not corrupt data. Our state-preservation interpretation requires a paid invoice to remain paid after a later migration. The static interpretation repeats migration without an intervening update. We report both. On historical outputs, removing only the task-086 revocations yields the same score vectors as the static check. For saved choices on new executions, the narrower target restores task-086 original labels and retains the denominator, a label-based sensitivity analysis.

Public code generations. The EvalPlus v0.1.0 release contains 200 programs per HumanEval task for each of 22 models, sampled at temperature 0.2 (Liu et al., 2023). We rescore indices 0–4 for development and 5–9 for held-out evaluation: 36,080 programs across 164 tasks. Scoring uses the official v0.3.1 evaluator, the maintainers' `find_zero` repair and HumanEval+ v0.1.10 in isolated, network-denied containers. The base score counts base-test passes. The plus score requires both the base and augmented suites to pass. This comparison measures strengthened test coverage; the banking study below measures a documented grader correction.

We consider selection among all 22 models and within each of the 231 unordered pairs. Development analysis evaluates each panel's choice on all four other panels, giving 4,620 transfers. Held-out analysis fixes choices using all five development panels and evaluates them on five newly scored panels, giving 1,155 pair-panel comparisons. Sample indices align generations within the same archived release. Appendix L reports the analysis chronology and complete results.

Documented scoring correction. The banking domain of τ^2-bench 1.0.1 (Barres et al., 2025) supplies a public before-and-after correction. We rescore the official Gemini 3 Flash and Claude Sonnet 4.5 trajectories across four trials of 97 tasks per model. Original rewards come from the saved trajectories; corrected rewards use the release evaluator. We found the correction after seeing the official aggregate scores and froze the comparison rule before successful per-task rescoring; the trials share one published archive (Appendix L.3).

Analysis timeline. Historical recovery outputs preceded checker development; their common-target comparison was specified after the score tables were read. The public-code full-pool endpoint preceded development-result inspection; pairwise analysis followed it. Selectors from panels 0–4 were fixed before scoring panels 5–9; the adoption rule was fixed during scoring, before analyst inspection. The new-cloud scoring and selection rule was fixed before the controller inspected its outputs; completion windows were later extended with result exposure (Appendix K.14). These stages distinguish fixed evaluations from analyses motivated by observed results.

3 GRADER COVERAGE AND EXECUTION CONDITIONS

Sensitive-key exclusion. Exact-key probes find prohibited nested keys in 58 of 87 accepted task-042 outputs across nine executions. The task requires unknown legacy fields to be retained in `metadata.unknownFields`, excluding named sensitive keys. Its grader inspects only the root; the canary inputs expose `ssn` or `credit_card` in the nested output (Figure 1c; Appendices K.5 and K.10). The 87 passes come from 108 assignments; the probe targets the explicitly named keys.

State preservation. Forty-five of 77 accepted task-086 outputs reset a paid invoice to `open`, including one scored 0.99 after rounding. The probe runs migration twice, commits invoice `inv1` as `paid`, migrates once more, and reads its status under two SQLite engines. Repeating migration on an untouched database revokes no passes. The result therefore depends on preserving an intervening payment update, our operational interpretation of the requirement not to corrupt data (Appendix K.11).

Consistency of reported results. Three budget reports receive 1.0000 while naming an overrun of 3,500 or 6,000 as the largest and listing 12,000 in the same file (Figure 3). A second check identifies a seven-field row beneath a five-column CSV header (Appendix F.6).

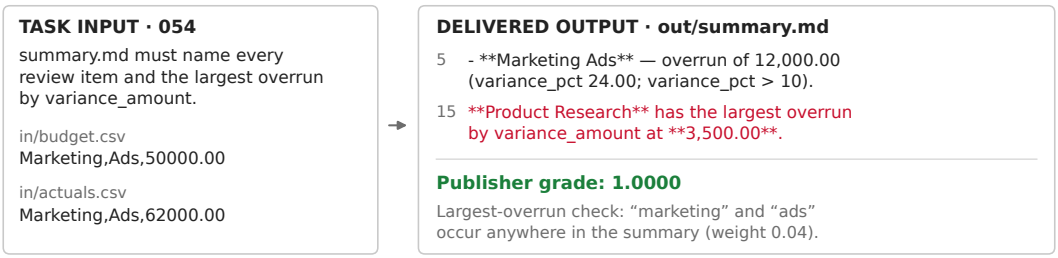


Figure 3: Task input and an accepted budget report. Input excerpts require the largest overrun and imply 12,000.00; verbatim output lines 5 and 15 list that value but name 3,500.00 as the largest. The report scores 1.0000; its largest-overrun check only requires marketing and ads anywhere in the summary (Appendix F.6).

Service interruption. All four recovery policies pass the six tasks under nominal service. Under recoverable and permanent 144-second stream silences, respectively, B10 passes 0 and 0, R10 passes 5 and 4, native11 passes 5 and 6, and B30 passes 6 and 6 (Table 4). B10's ten-second idle limit and retry budget are insufficient for the imposed stall. On the eight-task plan, B10 passes four of eight nominal pass-band assignments in the first execution and no fault assignments in any of the twelve earlier executions, before or after correction (Table 39).

4 RANKING CHANGES ACROSS EXECUTIONS

The two six-task checks withdraw four passes from the first GLM execution and none from its two reruns. The R10-minus-native11 contrast consequently changes sign on the first execution, from -2 to +1 passes, while remaining at -2 and -6 on the reruns. B30 remains the highest-scoring policy on the first two executions and ties native11 on the third (Appendix F.6). These checks were fixed after inspecting the first execution, so the reruns test their application to different outputs.

Applying the same checks to three planned Kimi executions gives no confirmed revocations. This result includes unresolved cases: 22 task-054 parses are indeterminate and six of the 216 assigned cells encounter provider-quota interruptions. With these cases retained as unknowns, the corrected fault-only R10-minus-native11 contrasts are bounded by [-3, +1], [-3, +2] and [-3, +2] passes out of twelve. Each interval includes zero (Appendix F.10).

On the eight-task plan, the sensitive-key and migration probes change the pass-band maximal set on five of eight executions with complete grades under the state-preservation reading of task 086. Under static idempotence, they change two of eight. The only replacement of a unique winner occurs under the state-preservation reading (Appendices K.10–K.11).

A within-task permutation places each execution's observed revocations among its original passes. Across 10,000 draws, mean changed-maximal-set counts are 4.0 of eight versus five observed, and 2.19 of nine versus three for the sealed-checker fault pool. The permutation describes sensitivity to revocation placement; the observed corrections follow the task requirements.

5 EVALUATING CORRECTED SELECTIONS

We compare the choices induced by original and corrected scores on the same corrected target in another execution. Let $O_i(p)$ and $C_i(p)$ denote the two scores of candidate p on selection execution i. The candidate set, task weights and endpoint remain fixed within each comparison. The selectors $\pi_O(i)$ and $\pi_C(i)$ assign equal probability to every candidate tied for the highest score. On test execution $j \neq i$, their losses are

$$L_{OC}(i,j) = \max_p C_j(p) - \sum_p \pi_O(i,p) C_j(p), \qquad L_{CC}(i,j) = \max_p C_j(p) - \sum_p \pi_C(i,p) C_j(p).$$

The selection benefit of correction is

$$G(i,j) = L_{OC}(i,j) - L_{CC}(i,j) = \sum_p [\pi_C(i,p) - \pi_O(i,p)] C_j(p).$$

Positive values favor selection from corrected scores. A zero arises either because the choices coincide or because different choices have equal corrected scores on the test execution; we distinguish these cases. Losses are measured in passed tasks: sixteen fault assignments per recovery policy, 164 tasks per public-code model, or 97 banking tasks. We report these settings separately.

Corrected selection maximizes its own target on the selection execution. A common target on another execution avoids this guaranteed advantage. Appendix K.13 separates selector and criterion effects.

The overall mean benefit equals the changed-choice fraction times its conditional mean benefit. We report both, retaining unchanged choices in the overall mean.

5.1 RECOVERY POLICIES

Under the full checker and equal-weight ties, the historical mean selection benefit is -1/36 of a pass for GLM, -3/4 for Kimi, and zero for GPT and Qwen (Table 1). The zero means arise because the latter models' choices never change. Individual GLM transfers have opposite signs: selection on execution 3 changes B30 to a three-policy tie, gaining 8/3 passes on execution 1 and losing 7/3 on execution 2. Resolving the Kimi execution-2 tie toward native11 loses 3/2 passes on execution 1 (Appendix K.13).

The result depends on the task-086 interpretation. Under static idempotence, all three highlighted selections remain unchanged and the GLM mean becomes +1/6, with the other means at zero. Adverse transfers still occur with this narrower check: averaging GLM executions 1 and 3 changes the unique choice from B30 to R10, which loses three passes on execution 2 (Appendix K.13).

We also saved one choice per scoring rule from each model's equally weighted historical mean. On the later GLM execution, B30 passes 15 of 16 fault assignments under both graders, whereas native11 passes 10 under the original grader and seven under the corrected grader. Five passes already separate the choices under original grading; the correction revokes three more native11 outputs. Both choices are unique, giving an eight-pass loss on the same target. They were saved before the new GLM scores were read, but the analysis began after generation had started and with partial exposure to GPT and Qwen results. The new GLM execution used four concurrent row slices and 2.5 cell-hours, compared with 6.0–6.7 cell-hours historically, so service load is confounded with execution (Appendix M).

GPT and Qwen retain identical choices and incur zero loss. With task-086 revocations removed from both historical selection and the new test target, all three models also retain identical choices and incur zero loss. The full and narrow analyses jointly vary selection and target scope. Each uses one later eight-task execution for GLM, GPT and Qwen.

Table 1: Recovery-policy selection benefit in fault passes out of sixteen; the first four rows are historical and the last four are new environments. Means use uniform ties; parentheses span 24 fixed priority orders. Narrow checking restores only task-086 original labels.

Environment	Transfers	Full checker	Narrow 086
GLM-5.3	6	-0.03 (-0.17 to +0.17)	+0.17
Kimi K3	2	-0.75 (-1.50 to 0)	0
GPT-6 Sol	2	0 (0)	0
Qwen3.7 Plus	2	0 (0)	0
MiniMax-M3	2	+0.75 (0 to +1.50)	+0.50
GLM-5.1	2	+0.17 (0 to +0.50)	+0.50
Qwen3.6-Plus	2	0 (0)	0
DeepSeek-V4-Pro	2	-0.75 (-1.00 to -0.50)	-0.25

All four new environments complete both assigned executions. Under the full checker and uniform ties, mean selection benefits are +0.75 for MiniMax-M3, +1/6 for GLM-5.1, zero for Qwen3.6-Plus and -0.75 for DeepSeek-V4-Pro (Figure 4). DeepSeek remains negative under all 24 fixed tie priorities; MiniMax and GLM admit zero means under some priorities. Restoring only task-086 original labels gives means of +0.50, +0.50, zero and -0.25, respectively. For DeepSeek without task-086 revocations, a missing task-087 regression test breaks execution A's B30/native11 tie toward native11. On execution B, checking task 042 removes native11's original advantage, leaving B30 ahead through task 090. Changing the choice then loses half a pass; the reverse direction is unchanged (Appendix K.14).

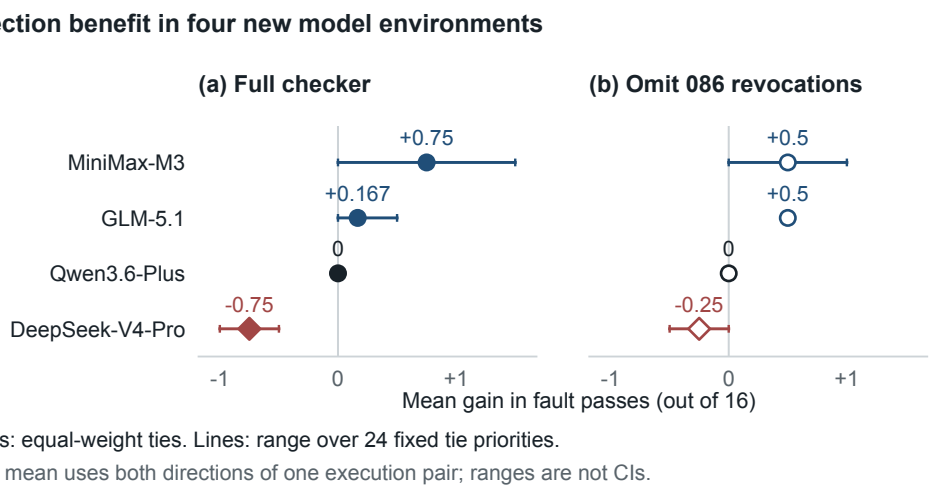


Figure 4: Selection benefit in four new recovery environments, in fault passes out of sixteen. Points average both directions of each pair, weighting tied maximizers equally. Lines span all 24 fixed policy-priority orders used consistently for both selectors and directions; these ranges describe tie sensitivity. Panel (b) restores only task-086 original labels in selection and evaluation, retaining the task and denominator. Directions share an execution pair; completion extensions and exposure are in Appendix K.14.

5.2 Public code models

On five held-out panels, selection from strengthened scores reduces mean loss by approximately 47%, from 0.098 to 0.052 passed tasks, for fixed two-model choices. Both selectors were frozen from the mean of development panels 0–4 before scoring panels 5–9. All 110 held-out model-panel blocks completed, covering 18,040 programs and 1,155 pair-panel transfers. Choices differ for nine of 231 model pairs. Their 45 transfers comprise 28 gains, five zero gains and twelve losses, with a mean

benefit of 1.18 tasks. The overall benefit is positive on four panels and negative on one (Appendix L.2). This is a held-out test of choices developed from known results on archived generations.

Development panels show a smaller improvement: mean loss decreases from 0.130 to 0.102 tasks across 4,620 transfers. Choices differ in 39 of 1,155 pair-panel selections, inducing 156 transfers. Their conditional mean gain of 0.83 yields an overall gain of $(156/4{,}620) \times 0.83 = 0.028$ tasks (Table 2; Figure 5).

Table 2: Selection benefit on the five development panels, in passed tasks out of 164. Every unordered model pair contributes all twenty ordered transfers. Positive values favor selection from strengthened scores.

Selection change	Transfers	Mean benefit	Gain / zero / loss
All pairwise selections	4,620	+0.028	88 / 4,484 / 48
All changed selections	156	+0.830	88 / 20 / 48
Unique winner replaced	48	+1.771	34 / 1 / 13
Tie resolved	64	+0.789	37 / 12 / 15
Tie created	44	-0.136	17 / 7 / 20

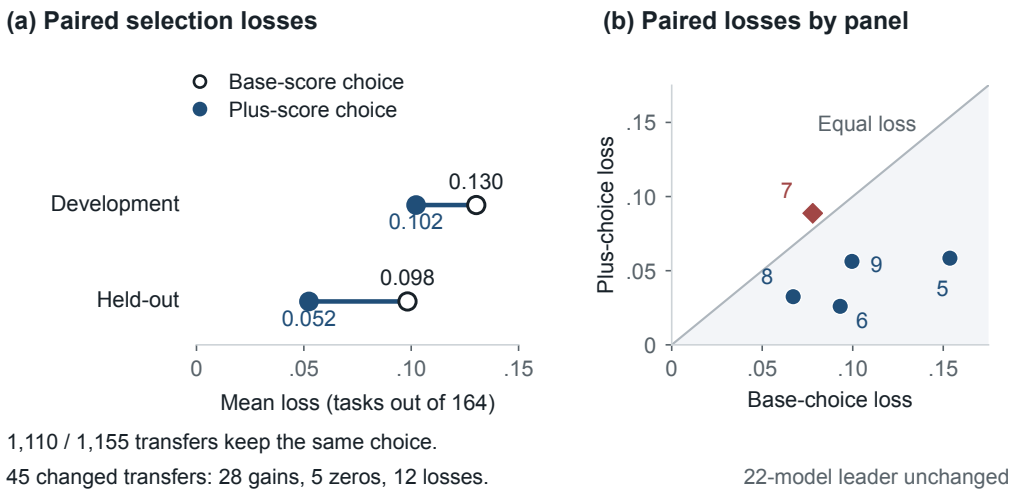


Figure 5: Pairwise selection on development and held-out code-generation panels. (a) Mean loss for choices made from base and plus scores, evaluated on the same strengthened target in passed tasks out of 164. Development uses 4,620 ordered cross-panel transfers; held-out evaluation fixes choices from panels 0–4 and tests them on panels 5–9, giving 1,155 transfers. (b) Each point shows mean losses over all 231 pairs on one held-out panel, labeled by its archived generation index. Points below the equality line favor plus-score selection. Models, tasks and panels recur across these descriptive comparisons.

Within development panels, replacing a unique winner gains 1.77 tasks on average, but loses in thirteen of 48 transfers. Resolving a tie gains 0.79 tasks, whereas creating a tie loses 0.14. Averaging within each fixed pair gives fourteen positive, 212 zero and five negative means. These comparisons share models, tasks and panels within one release.

The full-pool leader remains stable: phind-code-llama-34b-v2 leads every development panel under both scorings by at least 26 tasks, and both frozen selectors incur zero held-out loss. If a user must choose from a fixed pair, the pairwise comparison measures the benefit of changing that choice. The candidate set is given; shortlist formation is outside this experiment.

Tie-free development examples include an adverse transfer for codegen-16b versus codet5p-16b and positive transfers for codegen2-3b versus vicuna-13b; the complete enumeration and task-scale contrasts are retained in Appendix L.1.

5.3 BANKING BENCHMARK SCORING CORRECTION

The documented banking correction promotes 37 outputs and revokes none, reversing the aggregate order of the two models. It changes the recommendation on two of four selection trials (Table 3). Across all twelve ordered trial pairs, mean common-target benefit is 7/12 tasks, with three gains, seven zeros and two losses. The six changed-choice transfers average 7/6 tasks but include losses of 1/2 and one task. These comparisons reuse four archived trials (Appendix L.3).

Table 3: Original and corrected banking scores, in passed tasks out of 97. G denotes Gemini 3 Flash and C denotes Claude Sonnet 4.5; a tie weights both equally. Each selection is tested on the other three trials. The event was found after aggregate scores were seen; the rule was frozen before per-task rescoring.

Trial	Gemini: original to corrected	Claude: original to corrected	Choice: original to corrected
0	18 to 26	22 to 27	C to C
1	20 to 25	23 to 25	C to tie
2	23 to 28	22 to 24	G to G
3	19 to 27	20 to 22	C to G

6 SCORE MARGINS AND TIE HANDLING

Original margins and tie handling distinguish gains, unchanged choices and losses after correction. Kimi's differential revoked credit is unchanged across its two executions, yet resolving a tie toward native11 loses 3/2 passes because the original margin changes (Appendix K.13). The historical GLM mean ranges from -1/6 to +1/6 across fixed tie priorities; the later eight-pass loss is tie-free.

On public panels, original leads survive another panel in approximately 50%, 74%, 94% and 99.9% of comparisons with margins of one to two, three to five, six to ten and more than ten tasks. Strengthened tests reverse only leads of one to three tasks. These retention patterns describe the margin composition of the observed comparisons (Appendix L.1).

7 IMPLICATIONS FOR BENCHMARK REPAIR

A repaired score and an improved recommendation are distinct outcomes, separated by evaluating both choices on the same corrected test outputs. With checker scope fixed, later executions expose violations of additional known obligations, extending their observed coverage from seven to fifteen IDs (Appendix K.9).

Across repeated executions, the same recovery policies produce different corrected winners. For agent harnesses, this variation makes retry, continuation and stopping rules part of the evaluated configuration alongside the grader. Retaining final artifacts and state allows the original and revised choices to be evaluated on a common corrected target in another execution. Each comparison fixes candidates, task weights, check scope and tie handling, with checker-development exposure recorded. The released checks and `rerun-check` calculator make these comparisons reproducible, reporting all transfers, changed choices and adverse outcomes. Recursive self-improvement systems likewise use benchmark feedback to select harness rewrites (Srikanth et al., 2026), making grader-induced changes to that selection a concrete use of this test.

An adoption rule uses five leave-one-panel-out development folds: a positive estimated gain adopts the corrected choice fitted on all five panels; otherwise it retains the original choice. Fixed during held-out scoring, before analyst inspection, it performs worse than always correcting by 0.010 tasks and better than always retaining by 0.035 (Appendix L.2). Predicting which corrections improve selection remains open.

8 Related work

Evaluator audits study limitations in task coverage and scoring. EvalPlus adds tests to saved programs; BenchGuard examines task requirements and grader implementations with human validation; GuidedBench uses case-specific criteria to evaluate jailbreak methods (Liu et al., 2023; Tu et al., 2026; Huang et al., 2025). Rethinking Reward Model Evaluation connects evaluation quality to downstream policy regret (Wen et al., 2025). Our common-target analysis applies established decision-based comparisons to the original and corrected choices on different outputs.

Other work separates sources of measurement variation. The Double Measurement Confound crosses scorers with scaffolds on fixed episodes and reports same-configuration re-executions (Zhang et al., 2026a). Whose Gold? varies annotator pools on fixed outputs, while Noise Floor Audit varies runs and prompts under a fixed grader (Jha, 2026; Chen et al., 2026). We retain both sides of a scoring change and measure its selection effect across executions.

Recovery and handoff benchmarks examine continuation after service interruptions or context changes (Gulati, 2026; Deng et al., 2026; Zhang & Liu, 2026). Our four recovery policies provide a controlled setting for repeated task-policy-fault executions. Repeated-trial reliability and ranking-noise studies further motivate evaluating a recommendation on new outputs (Yao et al., 2024; Hardy et al., 2026; Rabanser et al., 2026). Appendix I discusses related item, harness and measurement interventions.

9 Scope of the evidence

The recovery study connects concrete grader omissions to selection outcomes in one CLI harness on fourteen tasks. AI-assisted checks cover 25 of 50 stated obligations, with author-specified interpretations and output-informed definitions. Independent human adjudication remains outside this evaluation. State-preservation and static checks give separate task-086 results; version comparisons jointly vary coverage and development information.

Natural service logs add 53 observed error responses across 10,222 requests. Their policy exposures are sparse and occur on unmatched tasks, supporting an exposure description rather than a four-policy ranking (Appendix M). Recovery comparisons also vary model, provider, date and concurrency. The Kimi six-task contrasts retain 22 indeterminate parses and six quota-interrupted cells as unknowns.

The public-code study strengthens tests on archived generations from one release. Its overlapping model, task and panel comparisons are descriptive. Held-out scoring follows selector freezing, but the research question and development results were known; the adoption rule was fixed later. Appendices K–M record these exposures, unresolved grades, conditions and sensitivity analyses.

The banking correction yields twelve ordered comparisons from four archived trials. Its audit rule precedes successful per-task rescoring and follows exposure to aggregate scores; the transfers share trials.

10 Conclusion

Evaluating revised recommendations on a common corrected target makes benchmark repair's decision value measurable. Held-out public-code selection loss falls from 0.098 to 0.052 tasks, and the banking correction also yields a mean gain. Recovery experiments expose positive, zero and negative benefits, including a negative mean after task-086 revocations are removed. The task-level decomposition explains how correcting real omissions improves grading while sometimes changing the recommendation adversely. The released checks, paired scores and calculator let evaluators report both requirement coverage and the consequence of following a repaired benchmark.

AI USE STATEMENT

Generative AI tools assisted hypothesis development, experimental design and critique, implementation, analysis, literature review, and manuscript preparation. They also helped construct synthetic tasks and test artifacts, derive and check algebraic comparisons, and read task requirements and saved outputs. Appendix K.9 documents the deterministic checkers' source access, author requests, synthetic stress tests and development exposure.

The recovery experiments use GLM-5.3, Kimi K3, GPT-6 Sol and Qwen3.7 Plus. The further cloud executions use GLM-5.1, DeepSeek-V4-Pro, Qwen3.6-Plus and MiniMax-M3; Appendix K.14 records all eight completed executions and the earlier cutoff snapshots. Artifact evaluation also uses AI readers. The three readers in Appendix K.3 were Claude Fable 5.1, GPT-6 Astra and Claude Opus 5. Later readings of two GLM re-executions and both Kimi executions used a GPT-6-configured Codex CLI under the second reader's frozen rules; the runtime model was unverified. Appendix K.6 describes two additional Codex reading procedures on GPT outputs. Reader roles and author-side critiques are documented in Appendices F–H and K.

For the task-042 and task-086 source probes, the author specified the rule scope and the state-preservation interpretation. Programs then measured explicit-key leaks and invoice-state changes. The study includes neither a human inter-rater experiment nor comprehensive human adjudication of the labels. The authors remain responsible for the submission.

REPRODUCIBILITY STATEMENT

Sections 2 and 5 define the policies, fault anchor, scoring and selection-loss comparisons. Appendices A–C, F–H and K–L report original scores, per-cell labels, sensitivity analyses and public-trial counts. Appendices D–E describe the timer measurements and observation limits.

The anonymous supplementary ZIP contains LaTeX source, score and label snapshots, reader rules and probes, exposure disclosures, the task-042 per-cell probe projection, the preregistration hash and four complete generated counterexamples. It also includes the 72-cell task-054/068 source-check witness and its generated files, source references and hashes, licensed public-trial files, the sealed eight-task checker versions and their seals, the nine-execution verdict and selection-loss records, the 042/086 matrix, the per-program EvalPlus verdicts and frozen panel analysis, and figure data.

Running `python3 -B verify_submission.py` at the archive root checks hashes and recomputes the six-task contrasts, nine-execution check matrix, sealed-checker tables and losses, public-panel counts, held-out oracle summaries, twelve earlier AI-reader maximal sets, four additional GPT reader interpretations with unresolved-label bounds, task-042 probe counts, four-task band comparisons and public-trial counts. The default verification is offline. The optional `--online-graders` mode checks nine pinned official source files, reproduces four original-oracle credits and reruns the 72-cell source check against publisher fixtures.

The archive excludes unlicensed benchmark inputs and modified source files. It consequently does not replay the task-042 candidates, the other four six-task outputs or hosted-agent trajectories. The earlier unexecuted confirmation protocol is included as a design record. Neither the supplied labels nor their reproduction establishes human ground truth.

REFERENCES

Justice Owusu Agyemang, Jerry John Kponyo, Elliot Amponsah, Godfred Manu Addo Boakye, and Kwame Opuni-Boachie Obour Agyekum. Resilient write: A six-layer durable write surface for LLM coding agents, 2026. URL https://arxiv.org/html/2604.10842. Version 3, 7 June 2026.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment, 2025. URL https://arxiv.org/abs/2506.07982v1. Version 1, 9 June 2025.

Yihang Chen, Pin Qian, Su Wang, Chong Peng, Huan Xu, Xiyang Wu, and Yiqi Sun. Noise floor audit for agent benchmarks, 2026. URL https://arxiv.org/abs/2608.22331v1. Version 1, 23 August 2026.

Runzhi Deng, Yiming Zhong, Fang Zhao, and Pan Zhou. Do not restart: Residual completion for stateful agent handoffs, 2026. URL https://arxiv.org/html/2609.13800. Version 2, 16 September 2026.

Roy Ganz, Mor Shpigel Nacson, Adi Kalyanpur, and Ron Litman. The handoff tax: Continuing non-native trajectories in LLM agents, 2026. URL https://arxiv.org/html/2608.24358v1. Version 1, 25 August 2026.

Aryan Gulati. ContinuityBench: A framework and taxonomy for evaluating agent recovery from interrupted state. CTB@ICML 2026, 2026. URL https://openreview.net/forum?id=3N3BzvoLbG. Long paper; published 25 May 2026.

Aayush Gupta. ReliabilityBench: Evaluating LLM agent reliability under production-like stress conditions, 2026. URL https://arxiv.org/html/2601.06112v1. Version 1.

Michael Hardy, Anka Reuel, Ruhana Azam, Mykel Kochenderfer, and Sanmi Koyejo. How reliable are agent leaderboards? a variance-decomposition analysis. ACM CAIS 2026: RLEval Workshop Poster, 2026. URL https://openreview.net/forum?id=GAu4q6OkLI. Published 24 May 2026.

Ruixuan Huang, Xunguang Wang, Zongjie Li, Daoyuan Wu, and Shuai Wang. GuidedBench: Measuring and mitigating the evaluation discrepancies of in-the-wild LLM jailbreak methods, 2025. URL https://arxiv.org/abs/2502.16903v2. Version 2, 9 July 2025.

Anik Jha. Whose gold? annotator-pool disagreement is large at the item level, and hidden by small leaderboards, 2026. URL https://arxiv.org/abs/2608.15980v1. Version 1, 17 August 2026.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations (ICLR 2024)*, 2024. URL https://arxiv.org/abs/2310.06770v3. arXiv 2310.06770, version 3, 11 November 2024.

Su Kara, Fazle Elahi Faisal, and Suman Nath. WAREX: Web agent reliability evaluation on existing benchmarks. *Transactions on Machine Learning Research*, 2026. URL https://openreview.net/forum?id=o4pXVP8RCD.

Dipesh KC and Anjila Budathoki. Handoff debt: The rediscovery cost when coding agents take over interrupted tasks, 2026. URL https://arxiv.org/html/2606.02875. Version 2, 30 August 2026.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation, 2025a. URL https://arxiv.org/abs/2505.06120v1. Version 1, 9 May 2025.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. Author responses to LLMs get lost in multi-turn conversation. OpenReview public discussion, 2025b. URL https://openreview.net/forum?id=VKGTGGcwl6. Author methodological clarification; distinct from the May 2025 v1 preprint.

Yuxiang Lin, Zihan Wang, Mengyang Liu, Yuxuan Shan, Longju Bai, Junyao Zhang, Xing Jin, Boshan Chen, Jinyan Su, Xingyao Wang, Jiaxin Pei, and Manling Li. BAGEN: Are LLM agents budget-aware?, 2026. URL https://arxiv.org/html/2606.00198v1. Version 1.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation, 2023. URL https://arxiv.org/pdf/2305.01210v3. Version 3, 30 October 2023.

Guanghui Min, Liang Wu, Mayank Darbari, Chen Chen, and Liangjie Hong. Toward reliable context compression for long-horizon agents: An empirical study of execution instability, 2026. URL https://arxiv.org/html/2608.06503v1. Version 1, 6 August 2026.

Hubert M. Pysklo, Artem Zhuravel, and Patrick D. Watson. Agent-Diff: Benchmarking LLM agents on enterprise API tasks via code execution with state-diff-based evaluation, 2026. URL https://arxiv.org/html/2602.11224v3. Version 3, 28 April 2026.

Stephan Rabanser, Sayash Kapoor, Peter Kirgis, Kangheng Liu, Saiteja Utpala, and Arvind Narayanan. Towards a science of AI agent reliability, 2026. URL https://arxiv.org/abs/2602.16666v3. Version 3, 2 June 2026.

Tanmay Sah, Dolly Sah, Harshul Jain, and Tanya Sah. EvoUndo: Recoverability-constrained self-evolution for LLM agent harnesses, 2026. URL https://arxiv.org/html/2608.28363v2. Version 2, 16 September 2026.

Philipp D. Siedler and Jordan Sassoon. Benchmarks are not monolithic: Sample-level auditing and orchestration for LLM evaluation, 2026. URL https://arxiv.org/html/2607.28801v1. Version 1, 30 July 2026.

Andrejs Sorstkins, Omer Tariq, and Muhammad Bilal. Learning to undo: Rollback-augmented reinforcement learning with reversibility signals, 2025. URL https://arxiv.org/html/2510.14503v1. Version 1, 16 October 2025.

Dhruv Srikanth, Bingchen Zhao, Dixing Xu, Yuxiang Wu, and Zhengyao Jiang. Recursive self-improvement of AI research agents, 2026. URL https://arxiv.org/abs/2609.26457.

Xinming Tu, Tianze Wang, Yingzhou (Minta) Lu, Kexin Huang, Yuanhao Qu, and Sara Mostafavi. BenchGuard: Who guards the benchmarks? automated auditing of LLM agent benchmarks, 2026. URL https://arxiv.org/html/2604.24955v1. Version 1.

Michael Walsh, Sadeesh K. Srinathan, Daniel F. McAuley, Marko Mrkobrada, Oren Levine, Christine Ribic, Amber O. Molnar, Neil D. Dattani, Andrew Burke, Gordon Guyatt, Lehana Thabane, Stephen D. Walter, Janice Pogue, and P. J. Devereaux. The statistical significance of randomized controlled trial results is frequently fragile: a case for a Fragility Index. *Journal of Clinical Epidemiology*, 67(6):622–628, 2014. URL https://doi.org/10.1016/j.jclinepi.2013.10.019.

Xinglin Wang, Zishen Liu, Shaoxiong Feng, Peiwen Yuan, Yiwei Li, Jiayi Shi, Yueqi Zhang, Chuyi Tan, Ji Zhang, Boyuan Pan, Yao Hu, and Kan Li. On time, within budget: Constraint-driven online resource allocation for agentic workflows, 2026a. URL https://arxiv.org/html/2605.06110v2. Version 2.

Yike Wang, Huaisheng Zhu, Zhengyu Hu, Yige Yuan, Zhengyu Chen, Shakti Senthil, Hannaneh Hajishirzi, Yulia Tsvetkov, Pradeep Dasigi, and Teng Xiao. Rethinking the evaluation of harness evolution for agents, 2026b. URL https://arxiv.org/html/2607.12227v2. Version 2, 27 August 2026.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory, 2024. URL https://arxiv.org/html/2409.07429v1. Version 1, 11 September 2024.

Xueru Wen, Jie Lou, Yaojie Lu, Hongyu Lin, Xing Yu, Xinyu Lu, Ben He, Xianpei Han, Debing Zhang, and Le Sun. Rethinking reward model evaluation: Are we barking up the wrong tree? In *International Conference on Learning Representations (ICLR 2025)*, 2025. URL https://openreview.net/forum?id=Cnwz9jONi5. Spotlight; arXiv 2410.05584.

Lirong Xia. Computing the margin of victory for various voting rules. In *Proceedings of the 13th ACM Conference on Electronic Commerce (EC '12)*, pp. 982–999. ACM, 2012. URL https://doi.org/10.1145/2229012.2229086.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. URL https://arxiv.org/abs/2406.12045v1. Version 1, 17 June 2024.

Yilun Yao, Xinyu Tan, Chao-Hsuan Liu, Yaoming Li, Zhengyang Wang, Wenhan Yu, Zhewen Tan, Yuxuan Tian, Guangxiang Zhao, Lin Sun, Xiangzheng Zhang, and Tong Yang. Harness-Bench: Measuring harness effects across models in realistic agent workflows, 2026. URL https://arxiv.org/html/2605.27922v1. Version 1.

Yonghong Zhang, Shadi Motaali, Vu Phong Dinh, Avin Piroutiniya, Jorge E. López de Vergara, Luis de Pedro, Ricardo Correia, Isabel M. Parra, and Yong Xie. The double measurement confound in agent benchmarks: De-scaffolding, ground-truth scoring, and reliability beyond the mean, 2026a. URL https://arxiv.org/abs/2609.09218v1. Version 1, 6 September 2026.

Yunbei Zhang, Janet Wang, Yingqiang Ge, Weijie Xu, Jihun Hamm, and Chandan K. Reddy. Stop comparing LLM agents without disclosing the harness, 2026b. URL https://arxiv.org/html/2605.23950v1. Version 1, 7 May 2026.

Zhihui Zhang and Wei Liu. Recoverability as a system primitive for long-horizon AI agents, 2026. URL https://arxiv.org/html/2609.13672. Version 1, 12 September 2026.

Chenyu Zhao, Shenglin Zhang, Yihang Lin, Wenwei Gu, Zhimin Chen, Yongqian Sun, Dan Pei, Chetan Bansal, Saravan Rajmohan, and Minghua Ma. Debugging the debuggers: Failure-anchored structured recovery for software engineering agents, 2026. URL https://arxiv.org/pdf/2605.08717v2. Version 2.

Yu Zhuang, Kefei Chen, Yitong Duan, Shuxin Zheng, Jian Li, and Xu-Yao Zhang. AgentRewind: Recoverable execution for long-horizon LLM agents, 2026. URL https://arxiv.org/html/2608.14380v1. Version 1, 14 August 2026.

A SIX-TASK STARTUP RESULTS

Appendices A and B use the configuration names B10/native5, R10/native5+resume, B10/native11 and B30/native5. These correspond to B10, R10, native11 and B30 in the main text.

Table 4: All four recovery policies tie under nominal service. Original-grader full passes on the six-task plan, with one execution per task-policy-regime cell. Pass counts use six tasks per column; contrasts are differences in pass proportions. Both fault conditions use the specified 144 s window. Original labels remain unchanged by subsequent checks.

Policy	Nominal passes	Recoverable-stall passes	Permanent-silence passes	Recoverable - nominal	Silent - nominal
B10/native5	6/6	0/6	0/6	-1.0000	-1.0000
R10/native5+resume	6/6	5/6	4/6	-0.1667	-0.3333
B10/native11	6/6	5/6	6/6	-0.1667	0.0000
B30/native5	6/6	6/6	6/6	0.0000	0.0000

Table 5: Original-grader scores for all 72 assigned six-task cells. Scores retain the source's four-decimal rounding. An asterisk marks the original full-pass label. CLI exit status and subsequent artifact judgments do not modify these entries.

Task	Policy	Nominal	Recoverable stall, 144 s	Permanent silence, 144 s
051	B10/native5	1.0000*	0.0864	0.0864
051	R10/native5+resume	1.0000*	1.0000*	1.0000*
051	B10/native11	1.0000*	0.9630	1.0000*
051	B30/native5	1.0000*	1.0000*	1.0000*
054	B10/native5	1.0000*	0.0744	0.0744
054	R10/native5+resume	1.0000*	1.0000*	0.0744
054	B10/native11	1.0000*	1.0000*	1.0000*
054	B30/native5	1.0000*	1.0000*	1.0000*
060	B10/native5	1.0000*	0.4900	0.4900
060	R10/native5+resume	1.0000*	1.0000*	1.0000*
060	B10/native11	1.0000*	1.0000*	1.0000*
060	B30/native5	1.0000*	1.0000*	1.0000*
068	B10/native5	1.0000*	0.0000	0.0000
068	R10/native5+resume	1.0000*	0.9322	1.0000*
068	B10/native11	1.0000*	1.0000*	1.0000*
068	B30/native5	1.0000*	1.0000*	1.0000*
080	B10/native5	1.0000*	0.0000	0.0000
080	R10/native5+resume	1.0000*	1.0000*	0.0000
080	B10/native11	1.0000*	1.0000*	1.0000*
080	B30/native5	1.0000*	1.0000*	1.0000*
085	B10/native5	1.0000*	0.1837	0.1837
085	R10/native5+resume	1.0000*	1.0000*	1.0000*
085	B10/native11	1.0000*	1.0000*	1.0000*
085	B30/native5	1.0000*	1.0000*	1.0000*

Table 6: Task-paired R10-minus-control contrasts under the original grader. Entries are differences in pass proportions on six tasks. Nominal contrasts are zero, so direct and nominal-adjusted fault contrasts coincide. The final column subtracts two regime-specific policy contrasts; it is not an independently estimated population interaction.

Control subtracted from R10	Nominal	Recover-able	Silent	Recoverable, nominal-adjusted	Silent, nominal-adjusted	Silent - recoverable
B10/ native5	0.0000	0.8333	0.6667	0.8333	0.6667	-0.1667
B10/ native11	0.0000	0.0000	-0.3333	0.0000	-0.3333	-0.3333
B30/ native5	0.0000	-0.1667	-0.3333	-0.1667	-0.3333	-0.1667

Four cells contradict the frozen oracle-pass predictions: R10 task 054/silent (0.0744), task 068/recoverable (0.9322), task 080/silent (0.0000), and native11 task 051/recoverable (0.9630). The three R10 cells contradict the oracle-defined fallback prediction, although the later artifact checks identify different causes. Task 060 contributes one whole-task outcome across its two prompt rounds. The tables retain the original scores and every assigned cell; Sections 2–5 and Appendix F report the revised endpoints separately.

B TASK SELECTION AND DEVELOPMENT CONTROLS

These development experiments document task selection and the controls that informed later designs. Their observations form a separate development record; the main experiments retain their assigned denominators.

B.1 TASK SCREENING

The screening plan starts with 19 candidate tasks, obtained by excluding tasks 046, 061, and 104 from a 22-task audited pool. Task 046's evaluator imposes a 0.12-second wall-clock limit; tasks 061 and 104 introduce workspace updates at fixed elapsed times. These exclusions narrow the population to tasks less sensitive to unrelated evaluation load and time-driven state changes. They are not evidence that the retained tasks are deterministic.

Each candidate task–shell configuration was assigned three nominal screening executions, labeled 1, 2, and 3, giving $19 \times 2 \times 3 = 114$ planned assignments. Eligibility requires three oracle passes, at least three model POSTs in each run, and complete timing evidence. Every round must use no more than half its own allowance; total measured run wall time must also be no greater than $d_i/2$. The aggregate condition includes time outside prompt rounds and is stricter for multiple-round tasks than checking each round alone. The frozen screen retains that condition and does not reinterpret it as an executed whole-task deadline or relax it in response to results.

The completed screen supplied 113 valid records and one missing assignment, OpenClaw task 085 at label 3. Both observed repetitions of that task–shell configuration failed and exceeded the half-allowance rule, so it is ineligible regardless of the missing outcome. Two archived provider-rate-limited attempts explain the missing record; they do not turn the missing result into a failure or success. Codex admitted 7 of 19 tasks and OpenClaw admitted 4 of 19. Excluding development task 063 leaves six Codex tasks (051, 054, 060, 068, 080, 085), three OpenClaw tasks (054, 068, 099), and two common tasks (054, 068). Task 060 has two prompt rounds; the other five Codex tasks and all three OpenClaw tasks have one. The maximum selected total-wall and per-round allowance fractions are 0.305 and 0.303 for Codex and 0.329 and 0.328 for OpenClaw.

Before selection, the 37 complete task–shell configurations contain 32 disagreeing outcome pairs among 111 within-configuration pairs. The remaining configuration has outcomes [0, 0, missing], contributing either zero or two disagreements. Across all 38 configurations, the identification bounds

are therefore [32, 34]/114 = [0.2807, 0.2982]. These bounds describe the uncertainty due to the missing outcome, not a confidence interval. The 114 pair comparisons are distinct from the 114 run assignments, and the two shells share 19 underlying tasks.

Independent nominal executions labeled 4 and 5 estimate residual instability after selection, separately for each timer–policy condition. The completed Codex readout covers the six selected tasks after excluding 063, across off and continuation-enabled nominal settings at 10, 30, and 300 seconds. These labels mark independent full-task executions, not deterministic seeds or additional task clusters. Neither the three screening successes nor the two stability executions per condition supply the confirmation means. Missing required timing is reported as missing evidence rather than as a pass. The target population is the selected task–shell set with observed nominal success and runtime headroom; the subsequent executions measure its residual failures. The headroom is also a condition on the fault results: with at most about a third of each allowance used nominally, a policy that waits 30 seconds per connection can absorb a 144-second silence without approaching a round deadline, so B30's completion rate under the faults of Section 2 is conditional on that slack. Tighter allowances can make waiting consume the remaining round, as observed for B300 in the 180-second local control (Appendix B.3).

B.2 Pre-header interruption and nominal stability

The GLM-5.3 development batch completed all 18 planned conditions on task 063 with repetition label 4, with no missing or duplicate assignment. Seventeen executions passed the complete-task oracle. All 12 injected executions reached the third-request anchor and passed; each 72-second condition absorbed eight 429 attempts, while the 6-second conditions absorbed three to six. No injected execution exposed an error or request cut to the shell, and none of the 18 executions used conversation continuation. The enabled recovery policy therefore never took its additional action in this batch.

The sole incomplete task was the nominal resume-300 condition, with an oracle score of 0.9675 and a normal process exit. It had no injection, visible request error, cancellation, or continuation. Its outcome cannot be attributed to an action that did not occur, nor does this one run estimate a timer effect. It does show that task 063's three screening passes did not guarantee a full pass in a separate execution. The original partial score and failed full-success flag are both retained.

Table 7: Development controls distinguish request-layer recovery from task continuation. Bτ and Rτ denote continuation off and enabled at idle limit τ. GLM rows pair separate executions; local rows belong to the five-case scripted designs. A score of 1 satisfies the complete-success criterion. Not run denotes a condition absent by design. Compare columns within each experiment; the GLM 900 s and scripted 180 s rounds are not pooled.

Experiment and regime	Stream-idle limit	Score: continuation off	Score: continuation enabled	Actual continuations: off / enabled
GLM, nominal	10 s	1	1	0 / 0
GLM, nominal	30 s	1	1	0 / 0
GLM, nominal	300 s	1	0.9675	0 / 0
GLM, pre-header 429, 6 s segments	10 s	1	1	0 / 0
GLM, pre-header 429, 6 s segments	30 s	1	1	0 / 0
GLM, pre-header 429, 6 s segments	300 s	1	1	0 / 0
GLM, pre-header 429, 72 s segment	10 s	1	1	0 / 0
GLM, pre-header 429, 72 s segment	30 s	1	1	0 / 0

Table 7 (continued)

Experiment and regime	Stream-idle limit	Score: continuation off	Score: continuation enabled	Actual continuations: off / enabled
GLM, pre-header 429, 72 s segment	300 s	1	1	0 / 0
Local, post-event, old connection recovers, 72 s	10 s	0	1	0 / 1
Local, post-event, old connection recovers, 72 s	30 s	1	Not run	0 / Not run
Local, post-event, old connection recovers, 72 s	300 s	1	Not run	0 / Not run
Local, post-event, old connection recovers, 6 s	10 s	1	Not run	0 / Not run
Local, post-event, old connection stays silent, 72 s	10 s	0	1	0 / 1
Local, post-event, old connection stays silent, 72 s	30 s	1	Not run	0 / Not run
Local, post-event, old connection stays silent, 72 s	300 s	0	Not run	0 / Not run
Local, post-event, old connection stays silent, 6 s	10 s	1	Not run	0 / Not run

The wrapper absorbed the injected failures before a terminal request error could trigger continuation. This explains the absence of additional continuation benefit in this development batch. The batch has one task and one execution per cell, and supplies 22 of the 24 planned contrasts; the two independent 24-second comparisons were excluded by design. It therefore provides a mechanism check rather than a confirmation study or an equivalence test.

The independent nominal stability readout then reran the six selected Codex tasks under each nominal timer-policy condition, using labels 4 and 5. All 72 planned result slots produced one valid result file, with no missing or duplicate assignment. These are nominal runs, so they do not test response-phase recovery; across all 72 records, the tap-layer retried/absorbed counters were zero, and the 36 continuation-enabled runs recorded only initial task attempts. Independent review separately identified five CLI native reconnects in three traces: three in B10, one in R30, and one in R300. It also found two HTTP 200 stream-internal rate-limit events in B10. The continuation switch therefore labels a nominal configuration here; it did not execute conversation continuation, and tap counters do not summarize the native reconnect layer.

Table 8: Nominal failures remain after screening for successful tasks. Each pool contains two repetitions on each of six tasks. Mean task failure first averages repetitions within a task, then tasks. Intervals are the existing pool-specific task-cluster intervals; pools are not combined.

Nominal policy	B10	R10	B30	R30	B300	R300
Passes / 12	11/12	10/12	9/12	12/12	12/12	12/12
Failed runs	1	2	3	0	0	0
Native reconnects	3	0	0	1	0	1
Continuations	0	0	0	0	0	0
Any-failure tasks / 6	1	2	2	0	0	0
Mixed-outcome tasks / 6	1	2	1	0	0	0
All-failure tasks / 6	0	0	1	0	0	0
Mean task failure rate	0.083	0.167	0.250	0.000	0.000	0.000
95% task-cluster interval	[0.000, 0.641]	[0.000, 0.777]	[0.004, 0.777]	[0.000, 0.459]	[0.000, 0.459]	[0.000, 0.459]

Table 9: Failure mechanisms in six nominal records. Diagnostic text explains each frozen failure without replacing its score. Two records share the task-085 delivery-path cause; literal grader strings and evidence identifiers are retained.

Failed record	Original score and check	Observed reason
051 R10 seed5	0.963, `analysis_exclusion_rule`	The report described excluded dates as before or after the target range, while the oracle accepted only the literal `outside` or `out-of-window` wording; query and audit checks passed.
068 R10 seed4	0.8305, `content_pack_schema`	`segment_messages` was a list containing all four IDs and exclusion reasons, while the oracle required a dictionary structure that the prompt did not explicitly require.
080 B30 seed4	19/23, `csv_exact`	Empty `tags` were encoded as `[]` rather than the empty field produced by semicolon-joining.
080 B30 seed5	19/23, `csv_exact`	Rows match by field name, but the output followed `schema.fields` column order while the oracle expected another exact-string column order that the prompt did not specify.
085 B10 and B30 seed5	0.95, `root_cause_report`	The report existed and code/test checks passed, but the deliverable was placed under `workspace/in/out` rather than the required `workspace/out`.

These diagnostics disclose prompt-oracle mismatches, one concrete representation difference, and two delivery-path violations without changing any binary or continuous score. The 8 to 25 percent failure rates of Table 8 are the background against which single fault-cell scores are read: the two fault cells that the frozen predictions counted as failures, native11 at 0.9630 on 051 and R10 at 0.9322 on 068 (Appendix A), are wording mismatches of the same kind as the 051 and 068 stability failures above, and both are satisfied under complete evaluation (Table 18). The task-080 pair is especially useful for interpretation: under the frozen binary oracle B30 failed both repetitions, but the two concrete causes differ, so the result estimates residual failure against the screened-success target rather than pairwise semantic disagreement between labels 4 and 5. These stability runs have not been independently regraded. Sections 2 and 4 study a separate six-task startup experiment, with wording interventions restricted to two selected tasks; it does not retroactively adjudicate these stability failures.

These results bound the precision of the selected-task nominal quality claim. Three pools had no observed failures, but with only six task clusters the equal-tailed 95% upper endpoint is still 0.459 for each such pool, far above the provisional $\delta/2 = 0.075$ line. The nonzero-failure pools show that screening success did not guarantee later full success under the nominal configuration. The readout should therefore guide the confirmation design and noise accounting; it cannot be used as evidence that continuation improved or harmed recovery, because no task-level conversation continuation occurred and the native reconnect events were observed in nominal traces outside a response-phase intervention.

B.3 Local connection controls and third-POST GLM outcomes

Two local experiments hold the real Codex CLI, jittered wrapper, greedy fields, retained provider-limit outcomes, and a 180-second original prompt round fixed. Each uses five conditions: 72-second silence with off at 10, 30, and 300 seconds and resume at 10 seconds, plus 6-second silence with off at 10 seconds. A scripted Responses service directs three tool actions that write alpha, beta, and gamma. Actual successful tool receipts determine progress; repeated POSTs and resumed conversations cannot advance the script past unfinished work.

The interruption is tied to the first legitimate response.created event for the third logical tool stage. The recoverable-connection clock starts immediately after the upstream flush; the permanently silent regime uses the proxy's successful first-event send as its recorded start. Each execution has one absolute end time shared by native retries and conversation continuation. In the recoverable-connection regime, surviving connections continue their original response when that time arrives. In the permanently silent regime, a connection exposed during the window remains silent afterward; only a new connection whose first event is delivered after the window can complete. The service neither revives nor actively closes an exposed old connection at service recovery. Its rule uses the tool stage and event-delivery time, without policy, timer, continuation identity, or oracle access.

Both event-observation points are preserved, with timing interpreted at the resolution of each instrument. They have one execution per condition and a scripted artifact oracle, so they test the recovery mechanism rather than GLM task-solving performance. Their logical tool-stage trigger is distinct from the physical third-POST anchor of the 429 experiment. Moving either regime to a real model requires an explicit phase and connection contract; the pre-header results do not supply that experiment.

After a legitimate stream event in the local 180-second round, the 10-second request-only condition exhausted five native resends before the common 72-second endpoint in both local regimes, leaving only alpha and beta. One continuation of the original conversation preserved those files and completed gamma within the same round. In the recoverable-connection case, the resumed connection survived until the window ended. In the permanently silent case, the first resumed connection was still exposed and remained silent; one further native resend reached a new healthy connection after the endpoint. Continuation preserved an opportunity to recover without resetting either the round allowance or the outage clock.

Ordinary 30-second request recovery completed in both regimes, with two native resends when the old connection could recover and three when it remained silent. The local 300-second request-only condition completed by waiting in the first regime, but waited until the original 180-second round deadline and failed in the second. That final cancellation is a round-deadline event, not an additional request-level failure. This local failure should not be projected onto GLM task 063, whose real prompt-round allowance was 900 seconds; if a 300-second stream wait still leaves time for a native reconnect after the 72-second window, B300 can complete. At six seconds, off-10 completed by waiting on the recoverable connection and by one native resend when the old connection stayed silent.

These local cases isolate a reason that waiting longer is sometimes enough and sometimes ineffective. R10 rescued work missed by B10, but added no completed task relative to B30 in either local regime. They also motivate an ordinary request-side control with a larger native retry allowance: for one blocked logical request, the default execution supplies the initial request plus five stream-level resends; one conversation continuation can supply another initial request plus five more, for at most 12 opportunities. A no-continuation B10 control with `stream_max_retries=11` matches that opportunity count, although it does not match backoff timing, remaining-round consumption, or all resource use.

The completed GLM response-phase contrast executes that control on task 063 with result seed 5. It uses the real GLM-5.3 service, Codex CLI 0.154.0, a 900-second prompt round, and one execution per cell. For nonnominal cells, the 72-second window begins only after the third outer model POST has delivered a real `response.created` event to Codex. Table 10 reports the outcomes.

Table 10: Third-request response-phase controls. Entries give original-grader score; native reconnects / task continuations. Each cell is one execution of task 063, with a 900-second prompt-round budget and the fault beginning after the third outer POST delivers a valid response-created event.

Policy	Nominal	Recoverable post-event stall, 72 s	Permanent post-event silence, 72 s
B10/native5	1.0; 0 / 0	1.0; 4 / 0	1.0; 5 / 0
R10/native5+resume	1.0; 0 / 0	1.0; 5 / 0	1.0; 5 / 1
B30/native5	1.0; 0 / 0	1.0; 2 / 0	1.0; 3 / 0
B300/native5	1.0; 0 / 0	1.0; 0 / 0	1.0; 1 / 0

Table 10 (continued)

Policy	Nominal	Recoverable post-event stall, 72 s	Permanent post-event silence, 72 s
B10/native11	1.0; 0 / 0	1.0; 5 / 0	1.0; 6 / 0

All 15 cells passed the complete-task oracle. Only R10/native5+resume under permanent silence executed conversation continuation, preserving the original thread; 792.858 seconds of original-round allowance was assigned at resume start. That demonstrates connection/process recovery after native retries are exhausted, but it did not add an observed task pass in this batch: B10/native5, B30/native5, B300/native5, and B10/native11 also passed in the corresponding response-phase conditions. Independent review also found that eight of ten fault cells had already written the two final full-score artifacts before the third-POST response-phase anchor; the sole resumed cell belonged to this set, and the resumed turn only performed subsequent checks. The 11-resend request-only control is therefore now a measured development control; in this single-task development contrast it completed both nonnominal cells, while still differing from resume in timing and resource use.

The event timings locate successful request opportunities within each recovery configuration. In the silent B10/native5 cell, the successive first events for the blocked logical request occurred at approximately 0, 14.330, 30.827, 44.488, 58.118, and 72.867 seconds relative to the fault start; the final one crossed the 72-second window without conversation continuation. In the silent R10/native5+resume cell, the last native first event appeared at 67.388 seconds, still within the silent window, but the original attempt terminated at 77.420 seconds; the resume launched at 79.423 seconds, with 792.858 seconds of original-round allowance assigned at resume start, and the resumed original thread first reached a healthy response at 88.606 seconds. B10/native11 crossed similarly at 88.542 seconds on its sixth native resend. B300/native5 crossed by waiting: in the silent cell, the next healthy first event arrived at 302.477 seconds and the round ended at 356.560 seconds, well inside the 900-second allowance. These traces show how idle timers, native retry cycles, connection fate, and remaining round time change whether an opportunity crosses the window; they do not turn one execution per cell on one task into an equivalence result, speed ranking, deployment probability, or attribution of task success to conversation state preservation. The first-POST experiment below jointly moves the anchor before tool execution and increases the window to 144 seconds.

B.4 First-request interruption and anchor evidence

The first-POST contrast uses the same real model and CLI on development task 063, one execution per cell at label 6, and the original 900-second prompt-round allowance. Its 144-second window starts when the complete `response.created` event from the first outer model POST has been sent through the outer tap. The same absolute window applies to native resends and task continuation. In all ten fault cells, the logs record no downstream tool/output event before the anchor and no post-created bytes leaked before the window ended. This establishes the recorded transport boundary, not the workspace contents at that instant. The intervention targets service blocking before tool execution. The earlier third-POST experiment cannot supply that evidence because eight of its ten fault cells already had completed artifacts.

File absence at the anchor has weaker provenance. None of the ten original outer logs contains a live file snapshot. The original guard used post-run existence and birth or modification times; a file present at the anchor but later deleted can satisfy its absence predicate. An independent synthetic deletion test establishes this limitation, not an actual GLM deletion. Historical provenance now identifies the Harness-Bench revision recorded in all ten results, its fresh-sandbox initialization and input-only task fixtures. All ten saved prompts and twenty retained inputs match the historical source. Together with no recorded pre-anchor tool actions, this supports a structural inference of absent outputs. The recorded commit is HEAD at result assembly, however, and the original manifest did not pin the external modules' actual imported bytes, dirty or untracked files, or directory snapshots. The historical reference and retained inputs establish source correspondence; live anchor observations supply the runtime-state evidence.

Table 11: First-request faults cause two B10 failures in the development task. Entries are original-grader score; native reconnects / task continuations. All fifteen assigned cells have valid traces. Each is one execution, not a population success probability.

Policy	Nominal	Recoverable post-event stall, 144 s	Permanent post-event silence, 144 s
B10/native5	1.0; 0 / 0	0.06; 5 / 0	0.06; 5 / 0
R10/native5+resume	1.0; 0 / 0	1.0; 9 / 1	1.0; 10 / 1
B30/native5	1.0; 0 / 0	1.0; 4 / 0	1.0; 5 / 0
B300/native5	1.0; 0 / 0	1.0; 0 / 0	1.0; 1 / 0
B10/native11	1.0; 0 / 0	1.0; 8 / 0	1.0; 8 / 0

All 15 assigned executions produced valid traces; 13 passed the full oracle, and the batch recorded 55 CLI native reconnects. The two B10/native5 fault cells exhausted five resends and produced neither required artifact. Their score of 0.06 comes solely from preserving the input source; both deliverable checks failed. The actual CLI round exited with code 1 in each case, although the outer runner completed its recording and grading with code 0. Completion of a run record therefore does not imply completion of the task.

Both R10 fault cells exhausted the initial attempt, resumed the same thread once, and passed every oracle check. At resume start, 805.996 seconds remained in the recoverable case and 814.928 seconds in the silent case; neither received a fresh 900-second allowance. Their nine and ten native reconnects span the original and resumed attempts. Post-run artifact timestamps fall after the anchor in both cases, but do not establish first existence. Conditional on the structural inference above, continuation restored an opportunity to complete an unfinished task after default native recovery ended. The absence of recorded pre-anchor tool actions does not supply a test of partial-work preservation.

The stronger request-only controls passed both fault cells as well. B10/native11 used eight native reconnects in each, B30 used four and five, and B300 used zero and one. More request opportunities and longer waiting are therefore sufficient alternatives on this task. All nominal controls passed in the observed execution of each condition; these runs provide the nominal reference for the assigned fault comparisons. Restarting a native retry cycle also changes timing and resource consumption, alongside opportunity count. The observed advantage over B10/native5 identifies a short-retry boundary, not superiority over the stronger controls or an independently identified conversation-memory benefit.

The supported rule is conditional: when startup blocking leaves the required outputs absent and exhausts short native recovery, same-thread continuation can serve as a fallback within the original round. For this batch, the absence premise rests on the structural inference above, not direct observation. An independent test needs live anchor-file evidence, with missing observations retained as unknown. Its runtime trigger remains the visible terminal request error and remaining time; the controller does not inspect the oracle or future fault window. For an independent test, failure of B10/native5 together with successful same-thread R10 continuation would support this fallback in that condition. Failure of both would leave it unsupported there. Success of B10/native11 or a longer-wait control restricts the conclusion to choosing among ways to retain request opportunities. A unique benefit from conversation state would require separate evidence against controls matched for opportunities and waiting resources. Appendix F.5 reports the fixed six-task startup transfer with live anchor-file evidence. A second-POST test remains a different, unexecuted extension; neither should be conflated with the original confirmation family.

B.5 Midtask instrumentation failures

The first frozen midtask implementation, dev02, raised a logging-argument TypeError before provider forwarding and ran through six local entries instead of stopping, sending zero provider POSTs; the separately frozen dev03 corrected that defect but followed an incorrectly specified Responses endpoint, so all six entries received HTTP 404, produced no tool work, checkpoint or recovery suffix, and scored 0.1837 as unchanged initial workspaces, with no usage reported. Neither version is pooled with the valid dev04 results, whose plan uses the protocol-specific endpoint resolver; their scores and

invalidation reasons are summarized in supplementary Section S1 B.5, while the original run records remain with the authors.

B.6 Restatement intervention: outcomes and accounting

The restatement intervention (Appendix H.2) reuses two dev04 checkpoints and attempts six dev07 suffixes in a frozen order. A recording exception interrupted the first block; the three checkpoint-2 suffixes ran in a later block. Earlier comparator versions dev05 and dev06 stopped in local preflight because native or system message identifiers differed, before producing model outcomes.

Five suffixes exited normally, with artifact scores of 0.95 for checkpoint-1 R_recap and 1.00 for the others. Checkpoint-1 R_notice was affected by instrumentation: its tenth local POST, tap attempt 29, recorded a returned 502/dropped result, after which the recording wrapper raised BrokenPipe while printing to stdout. The upstream status, final trace and complete usage are unavailable. Its recorded Unix span is at least 21,354.881 seconds; its eighth POST retains a response.completed record with 15,808 input and 619 output tokens.

The supplementary archive reports per-suffix durations and all 57 upstream attempts. Known usage totals 618,354 input and 22,680 output tokens, or 641,034 tokens; complete usage and monetary cost remain unknown. Reused prefixes incur no new execution cost in this batch.

B.7 Scheduling revision and resource use

The derived 103-OR1 task comprises one physical initial round, fresh and retained model suffixes, and a rule-applicability record. Each model stage has a 540-second allowance and a 48-POST cap. The initial round takes 271.609539 seconds and 19 POSTs; fresh takes 371.556100 seconds and retained 200.966395, each with twelve POSTs. The rule produces no revision because the initial plan fails its entry condition.

The two model suffixes change seven decisions: they move the EU gate and communications by 72 hours, replace Priya with Ken as APAC approval owner, and add the drill artifact. The frozen evaluator records 162 passing and 34 failing initial-plan checks, and 152 passing and 33 failing checks per revised plan. Appendix B.8 examines the output representations responsible for these failures.

Physical execution totals 43 upstream sends with complete usage, 1,260,639 input tokens and 60,687 output tokens. The hypothetical complete fresh and retained strategies take 643.166 and 472.576 seconds and thirty-one POSTs each; they share the initial round and cannot be summed into the physical bill. Retained POST 3 has complete upstream accounting despite a relay BrokenPipe. Per-phase usage and check counts are in the supplementary archive.

B.8 Scheduling-output representations

After inspecting the original failures, the contract and evaluator author audits the scheduling outputs under four readings. These cross acceptance of an ID-keyed state container with acceptance of an exact regional suffix on an owner's name. A suffix is accepted only when it matches the decision region and uniquely identifies an eligible roster member. State conversion requires each existing key to equal the contained decision ID.

Initial failures number 34, 33, 1 and 0 under the original reading, state conversion alone, owner-suffix acceptance alone and both. Second-round failures per arm are 36, 35, 1 and 0. Fresh and retained agree in every row. With both representations accepted, all 195 initial-plan, 181 revised-plan, 122 revision-relation, seven final-state and nineteen diff checks pass; the three state objects also match the plans directly.

The original frozen scores remain unchanged. This audit establishes how the stated representation choices affect the checks, but it is an output-informed author review with known arm identities. The supplementary archive preserves all four readings and their failure counts.

B.9 FINITE DECISION TASK AND PROGRAM BASELINE

The finite task chooses among four binary configurations using a published operating loss and a per-bit switching loss from the approved incumbent 00. A switching loss of one makes 11 optimal; a loss of five makes 00 optimal, under both the coupled and uncoupled tables. Both initial model stages output 00 with zero loss. All sixteen model suffixes and all ordinary-program executions are legal, timely, immutable and exactly optimal, with zero regret.

Physical model use totals 64 upstream sends with complete usage, 459,700 input tokens and 9,702 output tokens. The eight fresh suffixes take 266.705399 seconds and 34 POSTs; the eight retained suffixes take 132.117564 seconds and 24 POSTs. All 47 new tool calls exit zero. Each hypothetical complete strategy adds its real prefix once, so these complete-strategy totals cannot be summed into the physical bill.

The ordinary program's initial solve takes 0.047839 seconds; its eight suffixes take 0.027074 to 0.042129 seconds each. Two local preflight versions failed in tool-environment and temporary-directory handling before any paid request. The supplementary archive contains the per-pair results and both accounting tables.

C PER-CELL ARTIFACT LABELS

Table 12 and Table 13 report per-cell labels for the six-task and eight-task studies. Table 12 covers the 72 six-task cells under four shared interpretations; Table 13 covers the 96 eight-task cells under three readers. Each row includes workspace evidence. For the 31 disputed eight-task cells, we also report the applied obligations and the third reader's reasoning. The original labels are satisfied, violated, no output and, for the second reader, unknown. Nominal cells are excluded from fault counts; no-output and unknown labels contribute zero to the satisfied count in these reader comparisons.

C.1 SIX-TASK LABELS UNDER FOUR INTERPRETATIONS

Table 12: Cell-level labels under four shared readings of the six-task outputs. B/T denote broad/truthy-only interface readings; C/P denote coherent-dependency/strict-precedence readings. S = satisfied, V = violated, N = no output. Regimes are nominal, recoverable silence and permanent silence. The evidence column records a visible fact for every cell not satisfied under at least one reading; line numbers refer to the named artifact.

Task	Fault	Policy	Original score	B/C	T/C	B/P	T/P	Artifact evidence
051	nom	B10	1.0000	S	S	S	S	
051	nom	R10	1.0000	S	S	S	S	
051	nom	native11	1.0000	S	S	S	S	
051	nom	B30	1.0000	S	S	S	S	
051	rec	B10	0.0864	N	N	N	N	out/ is empty; none of the 3 required paths exists
051	rec	R10	1.0000	S	S	S	S	
051	rec	native11	0.9630	S	S	S	S	
051	rec	B30	1.0000	S	S	S	S	
051	sil	B10	0.0864	N	N	N	N	out/ is empty; none of the 3 required paths exists
051	sil	R10	1.0000	S	S	S	S	
051	sil	native11	1.0000	S	S	S	S	
051	sil	B30	1.0000	S	S	S	S	
054	nom	B10	1.0000	S	S	S	S	
054	nom	R10	1.0000	S	S	S	S	
054	nom	native11	1.0000	S	S	S	S	

Table 12 (continued)

Task	Fault	Policy	Original score	B/C	T/C	B/P	T/P	Artifact evidence
054	nom	B30	1.0000	S	S	S	S	
054	rec	B10	0.0744	N	N	N	N	out/ is empty; none of the 4 required paths exists
054	rec	R10	1.0000	S	S	S	S	
054	rec	native11	1.0000	V	V	V	V	summary.md line 5 lists Marketing Ads 12,000.00; line 15 names Product Research 3,500.00 as the largest overrun
054	rec	B30	1.0000	V	V	V	V	summary.md line 7 lists Marketing Ads 12,000.00; line 15 names Operations Logistics 6,000.00 as the largest overrun
054	sil	B10	0.0744	N	N	N	N	out/ is empty; none of the 4 required paths exists
054	sil	R10	0.0744	N	N	N	N	out/ is empty; none of the 4 required paths exists
054	sil	native11	1.0000	V	V	V	V	summary.md line 5 lists Marketing Ads 12,000.00; line 15 names Product Research 3,500.00 as the largest overrun
054	sil	B30	1.0000	S	S	S	S	
060	nom	B10	1.0000	S	S	S	S	
060	nom	R10	1.0000	S	S	S	S	
060	nom	native11	1.0000	S	S	S	S	
060	nom	B30	1.0000	S	S	S	S	
060	rec	B10	0.4900	N	N	N	N	out/ holds only an empty tmp/; none of the 4 required paths exists
060	rec	R10	1.0000	S	S	S	S	
060	rec	native11	1.0000	S	S	S	S	
060	rec	B30	1.0000	S	S	S	S	
060	sil	B10	0.4900	N	N	N	N	out/ holds only an empty tmp/; none of the 4 required paths exists
060	sil	R10	1.0000	S	S	S	S	
060	sil	native11	1.0000	S	S	S	S	
060	sil	B30	1.0000	S	S	S	S	
068	nom	B10	1.0000	S	S	V	V	launch_checklist.csv line 2, Finalize claims list (due 2026-05-01), gives the dependency 'Admin audit export and beta feature claims approved by compliance_ review'; line 5, Compliance approval, is due 2026-05-10
068	nom	R10	1.0000	S	S	S	S	
068	nom	native11	1.0000	S	S	S	S	
068	nom	B30	1.0000	S	S	S	S	
068	rec	B10	0.0000	N	N	N	N	out/ is empty; none of the 3 required paths exists
068	rec	R10	0.9322	S	S	S	S	

Table 12 (continued)

Task	Fault	Policy	Original score	B/C	T/C	B/P	T/P	Artifact evidence
068	rec	native11	1.0000	S	S	V	V	launch_checklist.csv line 3, Complete compliance approval (due 2026-05-10), gives the dependency 'Final compliance-approved claims list'
068	rec	B30	1.0000	S	S	S	S	
068	sil	B10	0.0000	N	N	N	N	out/ is empty; none of the 3 required paths exists
068	sil	R10	1.0000	S	S	S	S	
068	sil	native11	1.0000	V	V	V	V	launch_checklist.csv line 1 is a five-column header; line 8 has seven fields, its last field 'Not started' landing in a seventh column
068	sil	B30	1.0000	S	S	S	S	
080	nom	B10	1.0000	S	S	S	S	
080	nom	R10	1.0000	S	S	S	S	
080	nom	native11	1.0000	S	S	S	S	
080	nom	B30	1.0000	S	S	S	S	
080	rec	B10	0.0000	N	N	N	N	out/ is empty; none of the 4 required paths exists
080	rec	R10	1.0000	S	S	S	S	
080	rec	native11	1.0000	S	S	S	S	
080	rec	B30	1.0000	S	S	S	S	
080	sil	B10	0.0000	N	N	N	N	out/ is empty; none of the 4 required paths exists
080	sil	R10	0.0000	N	N	N	N	out/ is empty; none of the 4 required paths exists
080	sil	native11	1.0000	S	S	S	S	
080	sil	B30	1.0000	S	S	S	S	
085	nom	B10	1.0000	V	S	V	S	scheduler.py line 21: `self.clock = clock or SystemClock()`
085	nom	R10	1.0000	V	S	V	S	scheduler.py line 21: `self.clock = clock or SystemClock()`; line 23: `self.random_source = random_source or random.Random()`
085	nom	native11	1.0000	V	S	V	S	scheduler.py line 21: `self.clock = clock or SystemClock()`
085	nom	B30	1.0000	V	S	V	S	scheduler.py line 21: `self.clock = clock or SystemClock()`; line 23: `self.random_source = random_source or random`
085	rec	B10	0.1837	N	N	N	N	out/ is empty and out/root_cause.md is absent; scheduler.py has the initial workspace's SHA256 (5e45e62e723b...)

Table 12 (continued)

Task	Fault	Policy	Original score	B/C	T/C	B/P	T/P	Artifact evidence
085	rec	R10	1.0000	S	S	S	S	
085	rec	native11	1.0000	V	S	V	S	scheduler.py line 21: `self.clock = clock or SystemClock()`; line 23: `self.random_source = random_source or random.SystemRandom()`
085	rec	B30	1.0000	S	S	S	S	
085	sil	B10	0.1837	N	N	N	N	out/ is empty and out/root_cause.md is absent; scheduler.py has the initial workspace's SHA256 (5e45e62e723b...)
085	sil	R10	1.0000	S	S	S	S	
085	sil	native11	1.0000	V	S	V	S	scheduler.py line 21: `self.clock = clock or SystemClock()`; line 23: `self.random_source = random_source or random.Random()`
085	sil	B30	1.0000	S	S	S	S	

C.2 EIGHT-TASK LABELS UNDER THREE READERS

Table 13: Original grades, including the recorded environment corrections, and three reader worlds for all 96 held-out cells. S = satisfied, V = violated, U = unresolved, N = no output. The third world uses the third reader on the 31 disputed cells and the common label on the remaining 65. None means the grader emits no band. The following evidence list preserves each disputed obligation, amendment and recorded reason.

Task	Fault	Policy	Adjusted original score	Grade band	Reader 1	Reader 2	Third world
041	nom	B10	0.9962	excellent	S	S	S
041	nom	R10	0.6000	pass	S	V	V
041	nom	native11	0.5962	fail	S	S	S
041	nom	B30	1.0000	excellent	S	S	S
041	rec	B10	0.2039	fail	N	N	N
041	rec	R10	0.9962	excellent	S	S	S
041	rec	native11	0.6000	pass	S	S	S
041	rec	B30	0.9962	excellent	S	S	S
041	sil	B10	0.2039	fail	N	N	N
041	sil	R10	0.6000	pass	S	S	S
041	sil	native11	1.0000	excellent	S	S	S
041	sil	B30	0.6000	pass	S	S	S
042	nom	B10	0.7400	pass	S	V	S
042	nom	R10	0.7400	pass	S	V	S
042	nom	native11	0.7400	pass	S	V	S
042	nom	B30	0.7400	pass	S	V	S
042	rec	B10	0.1500	fail	N	N	N
042	rec	R10	0.7400	pass	S	V	S

Table 13 (continued)

Task	Fault	Policy	Adjusted original score	Grade band	Reader 1	Reader 2	Third world
042	rec	native11	0.7400	pass	S	V	S
042	rec	B30	0.7400	pass	S	V	S
042	sil	B10	0.1500	fail	N	N	N
042	sil	R10	0.7400	pass	S	V	S
042	sil	native11	0.7400	pass	S	V	S
042	sil	B30	0.7400	pass	S	V	S
052	nom	B10	0.3647	none	V	V	V
052	nom	R10	0.6900	none	V	U	V
052	nom	native11	0.3647	none	V	V	V
052	nom	B30	0.6900	none	V	U	V
052	rec	B10	0.1059	none	N	N	N
052	rec	R10	1.0000	none	S	S	S
052	rec	native11	0.6900	none	V	U	V
052	rec	B30	0.3647	none	V	V	V
052	sil	B10	0.1059	none	N	N	N
052	sil	R10	1.0000	none	S	S	S
052	sil	native11	1.0000	none	S	S	S
052	sil	B30	0.3647	none	V	V	V
055	nom	B10	0.9563	none	S	S	S
055	nom	R10	0.9180	none	S	V	V
055	nom	native11	0.9071	none	S	S	S
055	nom	B30	0.9344	none	S	S	S
055	rec	B10	0.0874	none	N	N	N
055	rec	R10	0.9563	none	S	S	S
055	rec	native11	0.9071	none	S	V	V
055	rec	B30	0.7814	none	V	V	V
055	sil	B10	0.0874	none	N	N	N
055	sil	R10	0.8907	none	S	S	S
055	sil	native11	0.9836	none	S	S	S
055	sil	B30	0.9836	none	S	S	S
056	nom	B10	0.6900	none	S	S	S
056	nom	R10	0.6900	none	S	S	S
056	nom	native11	0.6900	none	S	S	S
056	nom	B30	0.6900	none	S	S	S
056	rec	B10	0.0947	none	N	N	N
056	rec	R10	0.6900	none	S	S	S
056	rec	native11	0.6900	none	S	S	S
056	rec	B30	0.6900	none	S	S	S
056	sil	B10	0.0947	none	N	N	N
056	sil	R10	0.6900	none	S	S	S
056	sil	native11	0.6900	none	S	S	S
056	sil	B30	0.6900	none	S	S	S
086	nom	B10	0.9674	excellent	S	V	S
086	nom	R10	0.7000	pass	S	V	S
086	nom	native11	0.7000	pass	S	V	S
086	nom	B30	0.7000	pass	S	V	S
086	rec	B10	0.1928	fail	N	N	N
086	rec	R10	0.1329	fail	S	V	V
086	rec	native11	0.7000	pass	S	V	S
086	rec	B30	0.7000	pass	S	V	S
086	sil	B10	0.1928	fail	N	N	N
086	sil	R10	0.7000	pass	S	U	S
086	sil	native11	0.9129	excellent	S	V	V
086	sil	B30	0.7000	pass	S	V	S

Table 13 (continued)

Task	Fault	Policy	Adjusted original score	Grade band	Reader 1	Reader 2	Third world
087	nom	B10	0.8964	good	S	V	S
087	nom	R10	0.9107	excellent	S	S	S
087	nom	native11	0.9107	excellent	S	S	S
087	nom	B30	0.9107	excellent	S	S	S
087	rec	B10	0.2867	fail	N	N	N
087	rec	R10	0.8964	good	S	S	S
087	rec	native11	0.8964	good	S	S	S
087	rec	B30	0.8964	good	S	V	S
087	sil	B10	0.2867	fail	N	N	N
087	sil	R10	0.9107	excellent	S	S	S
087	sil	native11	0.9107	excellent	S	S	S
087	sil	B30	0.8964	good	S	V	S
090	nom	B10	0.5254	none	S	V	S
090	nom	R10	0.5254	none	S	S	S
090	nom	native11	0.4407	none	S	S	S
090	nom	B30	0.5424	none	S	S	S
090	rec	B10	0.0508	none	N	N	N
090	rec	R10	0.3729	none	V	V	V
090	rec	native11	0.5424	none	S	V	S
090	rec	B30	0.4407	none	S	S	S
090	sil	B10	0.0508	none	N	N	N
090	sil	R10	0.0508	none	N	N	N
090	sil	native11	0.0508	none	N	N	N
090	sil	B30	0.0508	none	N	N	N

The following records preserve the three labels for each disputed cell in the order R1, R2 and third reader. They list the obligations and amendments used by each reader, followed by the third reader's recorded reason. Closing brackets identify obligations that reader marked violated.

- **041, nominal, R10** (S, V, V). R1 applied 041.quantity_rules (S). R2 applied 041.2 (V). Third reader: updateQuantity(cart, sku, 0) empties the whole cart instead of removing that line (cartState.js L153-155): amended probe A2 returns an empty cart rather than one coffee line on a two-line cart, and step 16 of the frozen 300-step random sequence gives subtotal 0 against an independent tally of 850; the other 9 obligations are satisfied. [041-O5, 041-O10]
- **042, nominal, B10** (S, V, S). R1 applied 042.unknown_fields_pii (S), amendment A3. R2 applied 042.3 (V). Third reader: 5 public tests pass, all 14 frozen probes (amendment A1) pass, the static scan finds no network call or fixture literal, and protected files are untouched. The frozen probes assumed one error per bad record; the candidate reports one per field and was misjudged on O9/O13, and amendment A1, which merges by index, passes it.
- **042, nominal, R10** (S, V, S). R1 applied 042.idempotent_v2 (S), 042.unknown_fields_pii (S), amendment A3. R2 applied 042.2 (V), 042.3 (V). Third reader: 5 public tests pass, all 14 frozen probes (amendment A1) pass, the static scan finds no network call or fixture literal, and protected files are untouched. The frozen probes assumed one error per bad record; the candidate reports one per field and was misjudged on O9/O13, and amendment A1, which merges by index, passes it.
- **042, nominal, native11** (S, V, S). R1 applied 042.idempotent_v2 (S). R2 applied 042.2 (V). Third reader: 5 public tests pass, all 14 frozen probes (amendment A1) pass, the static scan finds no network call or fixture literal, and protected files are untouched.
- **042, nominal, B30** (S, V, S). R1 applied 042.unknown_fields_pii (S). R2 applied 042.3 (V). Third reader: 5 public tests pass, all 14 frozen probes (amendment A1) pass, the static scan finds no network call or fixture literal, and protected files are untouched.
- **042, recoverable, R10** (S, V, S). R1 applied 042.unknown_fields_pii (S). R2 applied 042.3 (V). Third reader: 5 public tests pass, all 14 frozen probes (amendment A1) pass, the static

scan finds no network call or fixture literal, and protected files are untouched. The added conftest.py only inserts into sys.path; recorded, not judged.

- **042, recoverable, native11** (S, V, S). R1 applied 042.unknown_fields_pii (S), 042.audit (S), amendment A3. R2 applied 042.3 (V), 042.5 (V). Third reader: 5 public tests pass, all 14 frozen probes (amendment A1) pass, the static scan finds no network call or fixture literal, and protected files are untouched. The frozen probes assumed one error per bad record; the candidate reports one per field and was misjudged on O9/O13, and amendment A1, which merges by index, passes it.
- **042, recoverable, B30** (S, V, S). R1 applied 042.idempotent_v2 (S), 042.unknown_fields_pii (S), amendment A3. R2 applied 042.2 (V), 042.3 (V). Third reader: 5 public tests pass, all 14 frozen probes (amendment A1) pass, the static scan finds no network call or fixture literal, and protected files are untouched. The frozen probes assumed one error per bad record; the candidate reports one per field and was misjudged on O9/O13, and amendment A1, which merges by index, passes it.
- **042, silent, R10** (S, V, S). R1 applied 042.idempotent_v2 (S), 042.unknown_fields_pii (S), amendment A3. R2 applied 042.2 (V), 042.3 (V). Third reader: 5 public tests pass, all 14 frozen probes (amendment A1) pass, the static scan finds no network call or fixture literal, and protected files are untouched. The frozen probes assumed one error per bad record; the candidate reports one per field and was misjudged on O9/O13, and amendment A1, which merges by index, passes it.
- **042, silent, native11** (S, V, S). R1 applied 042.idempotent_v2 (S), 042.unknown_fields_pii (S), amendment A3. R2 applied 042.2 (V), 042.3 (V). Third reader: 5 public tests pass, all 14 frozen probes (amendment A1) pass, the static scan finds no network call or fixture literal, and protected files are untouched. The frozen probes assumed one error per bad record; the candidate reports one per field and was misjudged on O9/O13, and amendment A1, which merges by index, passes it.
- **042, silent, B30** (S, V, S). R1 applied 042.unknown_fields_pii (S). R2 applied 042.3 (V). Third reader: 5 public tests pass, all 14 frozen probes (amendment A1) pass, the static scan finds no network call or fixture literal, and protected files are untouched.
- **052, nominal, R10** (V, U, V). R1 applied 052.severity (V). R2 applied 052.4 (U). Third reader: Enterprise ARR is given severity low; under the reading of frozen rule 4, ARR is a revenue metric and must be high. The set of 10 conflicting metrics, the other 9 severities (Refund Rate=low is an accepted reading), the verbatim text and equivalence_audit.json are all correct. [052-O4]
- **052, nominal, B30** (V, U, V). R1 applied 052.severity (V). R2 applied 052.4 (U). Third reader: Enterprise ARR is given severity low; under the reading of frozen rule 4, ARR is a revenue metric and must be high. The set of 10 conflicting metrics, the other 9 severities (Refund Rate=low is an accepted reading), the verbatim text and equivalence_audit.json are all correct. [052-O4]
- **052, recoverable, native11** (V, U, V). R1 applied 052.severity (V). R2 applied 052.4 (U). Third reader: Enterprise ARR is given severity low; under the reading of frozen rule 4, ARR is a revenue metric and must be high. The set of 10 conflicting metrics, the other 9 severities (Refund Rate=low is an accepted reading), the verbatim text and equivalence_audit.json are all correct. [052-O4]
- **055, nominal, R10** (S, V, V). R1 applied 055.cohorts (S), 055.insights (S), amendment A5. R2 applied 055.5 (V), 055.6 (V). Third reader: cohort_comparison.csv gives beta's largest_dropoff_transition as visit to signup; item 9 of the prompt defines the largest dropoff by the largest dropoff_rate_from_previous, which for beta is trial_start to purchase (1/1=1.0 against 1/2=0.5); the candidate took the share relative to visit and the earlier stage on a tie, with no support in the sources; the other four files are all correct. [055-O3]
- **055, recoverable, native11** (S, V, V). R1 applied 055.cohorts (S), 055.insights (S), amendment A5. R2 applied 055.5 (V), 055.6 (V). Third reader: cohort_comparison.csv gives beta's largest_dropoff_transition as visit to signup; item 9 of the prompt defines the largest dropoff by the largest dropoff_rate_from_previous, which for beta is trial_start to purchase (1/1=1.0 against 1/2=0.5); the candidate took the count of users lost, with no support in the sources; the other four files are all correct. [055-O3]
- **086, nominal, B10** (S, V, S). R1 applied 086.postcheck (S), amendment A4. R2 applied 086.6 (V). Third reader: Migration, postcheck, rerun and rollback run without error on both

engines with consistent state; 15 obligations satisfied. Rollback returns p4 to payments and drops payment_orphans (undefined by the rules; recorded only).

- **086, nominal, R10** (S, V, S). R1 applied 086.idempotent (S), amendment A4. R2 applied 086.3 (V). Third reader: Migration, postcheck, rerun and rollback run without error on both engines with consistent state; 15 obligations satisfied. Rollback returns p4 to payments and drops payment_orphans (undefined by the rules; recorded only).
- **086, nominal, native11** (S, V, S). R1 applied 086.idempotent (S), 086.postcheck (S), amendment A4. R2 applied 086.3 (V), 086.6 (V). Third reader: Migration, postcheck, rerun and rollback run without error on both engines with consistent state; p4 moves to payment_orphans, the foreign key is enforced and the transaction is explicit. Rollback returns p4 to payments and drops payment_orphans (undefined by the rules; recorded only).
- **086, nominal, B30** (S, V, S). R1 applied 086.postcheck (S), amendment A4. R2 applied 086.6 (V). Third reader: The first frozen result's probe JSON exceeded the driver's 20000-character cutoff and was not parsed; the same frozen probe was rerun in isolation: no error on either engine, consistent state, 15 obligations satisfied (rollback returns p4 to payments).
- **086, recoverable, R10** (S, V, V). R1 applied 086.postcheck (S), 086.preserve (S), 086.orphans (S), 086.fk (S), 086.idempotent (S), 086.rollback (S), amendment A9. R2 applied 086.6 (V), 086.1 (U), 086.2 (U), 086.3 (U), 086.4 (U). Third reader: On the Python engine (libsqlite 3.53.3) migration.sql fails at ALTER TABLE ... RENAME TO invoices with 'error in view billing_migration_invoice_status: no such table: main.invoices', the migration does not take effect, and postcheck and rollback then report 'no such table: payment_orphans'; under /usr/bin/sqlite3 3.51.0 everything runs without error with correct results. O2 requires both engines clean with consistent state, hence V (O3-O6, O9 and O13 also fail on that engine); a third engine, CLI 3.53.3, reports the same error. [086-O2, 086-O3, 086-O4, 086-O5, 086-O6, 086-O9, 086-O13]
- **086, recoverable, native11** (S, V, S). R1 applied 086.postcheck (S), amendment A4. R2 applied 086.6 (V). Third reader: The first frozen result's probe JSON exceeded the driver's 20000-character cutoff and was not parsed; the same frozen probe was rerun in isolation: no error on either engine, consistent state, 15 obligations satisfied (rollback returns p4 to payments).
- **086, recoverable, B30** (S, V, S). R1 applied 086.idempotent (S), amendment A4. R2 applied 086.3 (V). Third reader: No error on either engine and consistent state; 15 obligations satisfied. migration.sql and rollback.sql set PRAGMA foreign_keys=OFF first and ON at the end (O10 satisfied); rollback returns p4 to payments.
- **086, silent, R10** (S, U, S). R1 applied 086.idempotent (S), 086.postcheck (S), amendment A9. R2 applied 086.3 (U), 086.6 (U). Third reader: All 15 obligations satisfied on the two frozen engines; the second run on the Python engine aborts with 'error in trigger invoice_status_capture: no such table: main.invoices' and the state after rollback matches the first run (rule O8 accepts this explicitly and marks it underdefined); on a third engine, CLI 3.53.3 without bail, execution continues, the invoices table is lost and rollback also errors, which the rules do not cover and is recorded as a fact.
- **086, silent, native11** (S, V, V). R1 applied 086.idempotent (S), amendment A4. R2 applied 086.3 (V). Third reader: postcheck.sql opens with three sqlite3 command-line dot commands (.headers on / .mode column / .bail on); the Python engine's executescript reports 'near ".": syntax error' and not one postcheck statement runs, while the CLI engine is fine. O13 requires execution on both engines, hence V; the other 14 are satisfied (rollback keeps payment_orphans and does not return p4; undefined by the rules). [086-O13]
- **086, silent, B30** (S, V, S). R1 applied 086.postcheck (S), amendment A4. R2 applied 086.6 (V). Third reader: Migration, postcheck, rerun and rollback run without error on both engines with consistent state; 15 obligations satisfied. Rollback keeps payment_orphans and does not return p4 to payments (undefined by the rules; recorded only).
- **087, nominal, B10** (S, V, S). R1 applied 087.sort (S), amendment A1, A2. R2 applied 087.3 (V). Third reader: The public tests plus 6 added regression tests, 9 in all, pass; all 17 CLI probes pass (AND filters, quoted commas, header on empty results, numeric and directional sorting); errors go through argparse without a traceback; standard library only, no fixture literal.
- **087, recoverable, B30** (S, V, S). R1 applied 087.bad_where (S), 087.missing_field (S), amendment A1, A2. R2 applied 087.4 (V). Third reader: The public tests plus 5 added

regression tests, 8 in all, pass; all 17 CLI probes pass; error messages are clear with no traceback; standard library only, no fixture literal.

- **087, silent, B30** (S, V, S). R1 applied 087.sort (S), amendment A1, A2. R2 applied 087.3 (V). Third reader: The public tests plus 5 added regression tests, 8 in all, pass; all 17 CLI probes pass; error messages are clear with no traceback; standard library only, no fixture literal.
- **090, nominal, B10** (S, V, S). R1 applied 090.attribution (S), 090.notes (S). R2 applied 090.4 (V), 090.6 (V). Third reader: All 6 anomalies with their values and severities, the attributions (the 11:30 checkout anomaly assigned to deployment by precedence), the summary JSON (total impact 2570.75, counted per anomaly) and the three required points of the notes are present; fixtures untouched.
- **090, recoverable, native11** (S, V, S). R1 applied 090.attribution (S), 090.notes (S). R2 applied 090.4 (V), 090.6 (V). Third reader: All 6 anomalies with their values and severities, recognizable attribution categories (deployment overlap and so on), the summary JSON (total impact 2570.75, counted per anomaly) and the three required points of the notes are present; fixtures untouched.

The obligation sentences applied on the disputed cells, verbatim from each reader's frozen rule package (the first reader's are the rules of Appendix K.3's first package; the second reader's are the contract's frozen-source rules):

- First reader, 041.quantity_rules: negative quantity throws an error whose message matches /quantity/i; quantity 0 removes the line
- First reader, 042.audit: conversion_audit.json is written next to client.py when convert_many runs, with converted_count, error_count, warning_count, pii_dropped_count, unknown_fields_count
- First reader, 042.idempotent_v2: already-v2 payloads keep metadata.unknownFields and valid line items
- First reader, 042.unknown_fields_pii: unmapped legacy extras are preserved under metadata.unknownFields except keys listed in pii_policy.md; v2 metadata.unknownFields is preserved
- First reader, 052.severity: high for Net Revenue, Trial Conversion Rate, Seven Day Retention, Enterprise ARR; medium for Average Fulfillment Time, Support SLA Breach Rate, Rolling Active Users; low for Marketing Opt-in Rate, Inventory Stockout Count; Refund Rate high or low
- First reader, 055.cohorts: cohort_comparison.csv header exact; rows beta, control, variant sorted; visit/purchase users 2/0, 6/3, 6/2 and rates 0.0, 0.5, 0.3333
- First reader, 055.insights: dropoff_insights.md names the largest dropoff transition, compares cohorts, and includes a sample-size or ordering-noise caveat
- First reader, 086.fk: after migration payments.invoice_id must reference a valid invoice; foreign keys are enforced or re-enabled, not left disabled
- First reader, 086.idempotent: running migration.sql a second time neither errors nor duplicates or corrupts rows
- First reader, 086.orphans: p4 is moved to payment_orphans with id, invoice_id, amount_cents, created_at plus reason and is not deleted
- First reader, 086.postcheck: postcheck.sql contains row counts, orphan preservation, status default and foreign-key checks and executes
- First reader, 086.preserve: after migration all three invoices and payments p1-p3 remain with original created_at; invoices.status TEXT NOT NULL DEFAULT 'open' exists
- First reader, 086.rollback: rollback.sql executes after the migration and restores invoices(id, customer_id, total_cents, created_at) and payments(id, invoice_id, amount_cents, created_at)
- First reader, 087.bad_where: malformed --where exits non-zero with a clear message
- First reader, 087.missing_field: selecting, sorting or filtering on a missing field exits non-zero with a clear message
- First reader, 087.sort: --sort -field descending, field ascending, numeric-looking values numerically

- First reader, 090.attribution: deployment for both checkout anomalies, third_party for payments, marketing for search, unattributed for both profile anomalies; the 11:30 checkout overlap with I-20 is mentioned in the notes
- First reader, 090.notes: reconciliation_notes.md mentions low-volume suppression, overlapping attribution uncertainty, and that correlation is not proof of causation
- Second reader, 041.2: Independent carts and immutable transitions; add sums quantities for an existing SKU, update targets SKU, zero update removes a line, remove removes intended SKU; old snapshots remain usable and original item objects/deep nested item data cannot alias into carts.
- Second reader, 042.2: Handle already-v2, preserve valid line items and existing metadata.unknownFields, default blank/null shipping method, preserve unknown v2 metadata under unknownFields; conversion stable once normalized.
- Second reader, 042.3: Preserve unmapped non-sensitive legacy extras under metadata.unknownFields and exclude obvious PII keys ssn,credit_card,card_number,cvv,phone_number,passport_number; analogous obvious identity/payment keys excluded, without discarding legitimate campaign/channel/loyalty/gift/routing data.
- Second reader, 042.5: Every convert_many call writes conversion_audit.json next to client.py with converted_count,error_count,warning_count,pii_dropped_count,unknown_fields_count, consistent with emitted results and documented counting units.
- Second reader, 052.4: Apply source severity taxonomy and keep severity_by_metric identical to CSV; preserve unresolved category/syntactic scope readings below instead of guessing a private label set.
- Second reader, 055.5: Assign cohort from first valid non-bot visit; retain every observed cohort with visit/purchase counts, purchase rate and its own largest-dropoff transition; CSV ascending beta,control,variant.
- Second reader, 055.6: Insights identify correct largest-dropoff transition, compare cohorts and include at least one concrete sample-size or ordering-noise caveat; calculations remain local with no network-dependent solution.
- Second reader, 086.1: Provide all five substantive package files; migration preserves source invoice/payment identities, amounts and timestamps and adds invoices.status TEXT NOT NULL DEFAULT 'open'; protected schema/policy unchanged.
- Second reader, 086.2: Move existing orphan payments into payment_orphans before enforcing references; archive same id,invoice_id,amount_cents,created_at plus reason; p4 preserved; future payments references must be valid, with foreign-key state restored/enabled appropriately.
- Second reader, 086.3: Migration runs within explicit transaction and repeated execution must not duplicate/corrupt data or fail on already-created structures.
- Second reader, 086.4: Rollback SQL must actually execute after migration and restore invoices(id,customer_id,total_cents,created_at) and payments(id,invoice_id,amount_cents,created_at), without losing source data.
- Second reader, 086.6: Executable postcheck SQL addresses row counts, orphan preservation, status defaults and foreign-key checks; migration_report agrees with actual SQL and data disposition.
- Second reader, 087.3: Sort full rows before projecting needed selected fields, compare numeric-looking values numerically, retain selected header on empty results and exit0.
- Second reader, 087.4: Bad where expressions and missing fields in select/sort/filter produce nonzero exit with clear diagnostic; validate fields against header, including empty-result cases.
- Second reader, 090.4: Match service,region,time window; choose exactly one cause using deployment > third_party > marketing/calendar > unattributed; if events overlap mention overlap.
- Second reader, 090.6: Notes mention low-volume suppression, overlapping attribution uncertainty and correlation is not causation; actual explanations agree with selected causes and findings; local-only solution.

The first reader's nine rule amendments, recorded with timestamps before the affected candidates were scored and applied unchanged by the readers of the second execution and of Kimi K3 (Appendix K.4), verbatim from its amendments file:

Table 14: Recorded amendments to the first reader's evaluation rules. A1–A9 preserve the recorded rule changes and reasons. A10–A11 add disclosure documents and change no rule. Literal rule names, engine versions, code and artifact identifiers are retained.

No.	Obligations affected	Change and stated reason
A1	087.regression_tests, 087.existing_tests	run tests with real pytest 9.1.1 from an isolated venv inside the sandbox instead of the standard-library runner, because candidate regression tests legitimately use pytest fixtures (tmp_path) and import pytest. Reason: runner limitation, not an artifact defect.
A2	087.csv_quoting	the static regex for split(',') is informational only; the clause is S when the existing quoted-comma test passes, the csv module is used, and the probe observes a quoted comma written. Reason: splitting the –select/–where arguments on commas is legitimate and was caught by the regex.
A3	042.audit	accept error_count >= 1 (and warning_count any) for the probe's single bad payload, converted_count == 4, five keys present. Reason: one bad payload may yield several error records with distinct paths; the prompt requires index/path/error per error, not one error per payload.
A4	086.postcheck, 086 execution	additionally execute schema.sql to migration.sql to postcheck.sql and the rollback chain with the sqlite3 command-line shell in the sandbox; a clause is S if it holds under either engine. Reason: postcheck.sql may use sqlite3 shell dot-commands, which the Python sqlite3 module cannot execute.
A5	055.cohorts	a cohort largest_dropoff_transition chosen by lost users with an earliest-transition tie-break is a source-compatible reading; the clause is S with the reading recorded when the remaining columns are correct. Reason: the prompt defines largest_dropoff by rate only at the funnel level; the cohort column is named but not defined.
A6	rules.json text for 055.user_sets	the obligation text lists deduplicated_stage_user_ids as [U002, U010, U011]; the frozen source-facts.json and the harness use the computed [U002, U011]; the text is a transcription error and the computed value governs. Reason: U010's early purchase is ignored (order), not deduplicated.
A7	052.severity	record that rule 4 admits a second reading under which only denominator errors are high, making Net Revenue and Enterprise ARR low; no artifact applied that reading consistently (all four ARR-low artifacts kept Net Revenue high), so labels are unchanged. Reason: transparency about a reading the frozen rules did not enumerate.
A8	086 whole labels for artifacts whose migration or second run errors under SQLite 3.53	labels stay V but are flagged contestable: view/trigger references across the rename fail on SQLite >= 3.26 with legacy_alter_table off; the original actor's engine version is unknown. Reason: version-dependent failure is still a failure of the delivered script on a current SQLite, but the author controller should see the flag.

Table 14 (continued)

No.	Obligations affected	Change and stated reason
A9	086.preserve, 086.fk, 086.idempotent, 086.rollback for artifact-b3094d664dc34236 and artifact-c88a03544a84471d	apply A4 consistently: a full verification with the sqlite3 command-line shell 3.51.0 (the engine the prompt names) shows migration ok, rows and created_at preserved, status default open, p4 in payment_orphans with reason, FOREIGN KEY constraint failed on an orphan insert, second run leaves counts 3/3/1, postcheck ok, rollback restores both shapes; labels become S with the engine note; the Python sqlite3 module 3.53.3 failures (view or trigger reference across the rename) are recorded as engine-dependent. Reason: A4 states that a clause holds if it holds under either engine; the first pass only applied it to postcheck.

C.3 Unexecuted confirmation design

An earlier 18-condition confirmation design, with two conversation-recovery settings, three stream-idle settings and nominal, 6-second-segment and 72-second-segment HTTP 429 regimes plus a 24-second held-out schedule check, specified three fresh executions per task and cell, a prespecified family of 24 contrasts, joint task-cluster resampling and a provisional smallest useful effect of 0.15. It was never executed: its result tables are empty, its conditional timing prediction was withdrawn after the combined-path check found the request wrapper withholding response headers while it retried, and no startup, copy, midtask or framing experiment populates it. Its design text is in the supplementary archive, and nothing reported in this paper derives from it.

D Pilot experiments and timer calibration

D.1 Pilot population

The completed S2 pilot contains 133 run records: 16 screening runs, nine nominal outcome runs, and 108 injected runs. Eight candidate tasks entered screening. Four task configurations for Codex CLI and five for OpenClaw contributed to the outcome comparisons, representing five distinct tasks in their union. Each eligible task–shell configuration supplied one nominal outcome trace reused across four segment lengths and three recovery policies. This produces 108 nominal–injected comparisons across 24 shell–policy–regime cells, reusing the recorded nominal runs.

The model backend in these development runs is GLM-5.3. The tasks include artifact generation, code repair, and stateful workspace work drawn from Harness-Bench. The small number of task clusters, shared nominal traces, single outcome repetition per condition, and use of these results to revise the protocol make the pilot exploratory. We use it to identify interpretation failures and calibrate the next evaluation, rather than to claim population-level policy superiority.

Table 15: Pilot records measure different evidence units. Records, contrasts, cells, task clusters and request probes are not interchangeable sample counts. Shared nominal traces remain shared in the 108 comparisons.

Evidence unit	Observation	Interpretation supported
S2 development pilot	133 records; 108 comparisons; 24 cells; five outcome task clusters	Development evidence, with shared nominal traces
One task trace	Statuses 200, 200, 429; exit code 1; oracle score 1.0	Shell failure can coexist with a correct artifact

Table 15 (continued)

Evidence unit	Observation	Interpretation supported
Anchored mechanism control	One live 72 s case; four-regime replay of one nominal trace	Initial error and native termination can make a length contrast uninformative
Repeated request probes	Five distinct outputs in each of four five-send groups	Tested settings do not ensure identical output
Scripted timer calibration	Three cuts in one execution: 11.95, 12.00, 12.00 s	Agreement with a 12 s first-event timeout prediction

D.2 Fault exposure in successful runs

In the pilot's 72-second regime, all 12 Codex runs across the three policies passed their task oracle. None recorded a fault delivered to the shell or an attempt absorbed by the recovery wrapper. These observations cannot support the claim that Codex tolerated a long outage. They show that the scheduled outage was not encountered by these executions.

The nominal-source replay could nevertheless predict exposure, because another execution of the same task issued requests at different times. Thus an apparent advantage over replay can arise without an agent repairing any damage. This observation motivates request-count anchoring and explicit exposure reporting. It also illustrates why conditioning the main comparison on eventual exposure would answer a different question: the policy itself can alter when exposure occurs.

The pilot provides no established ranking reversal. For example, in the dispersed 1.5-second Codex passthrough cell, one of four tasks passed, equal to the aggregate replay prediction. Agreement in the development cell checks consistency between its original trace and replay; each task retains its own failure evidence.

D.3 Process termination and artifact correctness

One unanchored Codex execution of the alert-deduplication task received two successful responses and then an injected HTTP 429 on its third request. The process exited with code 1, while the oracle returned a score of 1.0 and passed all three checks. The corresponding nominal outcome trace contained five model requests. Treating the termination as a guaranteed task failure would therefore mislabel the observed artifact; treating the nominal request count as a necessary computation length would also overstate what that trace establishes.

A separate anchored execution of the same task and shell also received statuses 200, 200, and 429, but its oracle score was 0.06 and it failed. These two runs are not a controlled estimate of an anchoring effect. They demonstrate the narrower fact that the same short HTTP-status sequence and process exit do not determine the final workspace outcome. Semantic progress must be measured rather than inferred from request index alone.

D.4 Sensitivity of the anchored control

Anchoring the first failure at request three makes the initial passthrough response the same across all four regimes. Under the tested native Codex configuration, that response terminates the turn before the shell can observe the rest of the segment. Replaying one five-request nominal trace under the four regimes predicts termination at request three in every case. One live anchored 72-second run verifies this initial-error path; there are not four independent live verifications in this evidence set.

This is a mechanism control, not an informative test of a native Codex segment-length slope. Any task-success variation across such runs could reflect how much work was already complete when the common error arrived. Policies or shells that continue after the initial encounter are needed for a discriminating comparison of later schedule structure. Jittered recovery in the same nominal replay permits continuation in all four regimes, but those replay outcomes are predictions, not additional observed agent successes.

D.5 VARIATION ACROSS REPEATED REQUESTS

The decoding probe reuses one real request from each of the two API paths. Each request was sent five times under the original settings and five times with do_sample=false, temperature=0, and top_p=1. Each of the four groups produced five distinct answers and five distinct action representations. The tool names remained the same within the corresponding groups, while their arguments varied.

This establishes that the tested setup did not produce identical outputs for these requests. It neither locates the provider's internal source of variation nor proves that the decoding fields had no effect. The probe did not execute the generated actions through a complete task oracle, so it does not establish task-success instability. Greedy decoding is consequently treated as a fixed request configuration; residual outcome variability must be measured with independent full-task repetitions.

D.6 FIRST-EVENT TIMEOUT CALIBRATION

A scripted upstream test holds responses behind a temporary failure interval while a real OpenClaw shell runs with a request timer scaled to 12 seconds. Within one injected execution, the shell cuts three requests at 11.95, 12.00, and 12.00 seconds. The replay predicts three 12-second cuts and the subsequent request continuation. The largest absolute timing discrepancy is approximately 0.05 seconds (0.052 seconds in the raw millisecond record).

These are three repeated events in one execution, not three independent calibration experiments. Moreover, the scripted upstream does not solve the benchmark task: both the nominal and injected oracle outcomes are failures. The continuation result establishes request-path agreement, not successful task recovery. The calibration establishes the tested first-event timer and its retry sequence for this shell.

D.7 WAITING BEFORE RESPONSE HEADERS

Four separate scripted-upstream executions exercised the real Codex CLI with a 180-second prompt-round allowance, the same jittered wrapper, greedy settings, and retained provider-limit outcomes. The first two model POSTs caused tool actions that wrote partial work. The third POST started the fault clock; the script then completed a three-line artifact. These are instrument checks with a scripted model service, not GLM task evaluations.

Table 16: Pre-header retries do not trigger the configured stream-idle limit in this control. All four scripted artifacts pass. The wrapper withholds response headers while absorbing request failures and releases them when a successful response is available.

Fault segment	Policy	Stream-idle limit	Third POST to first event	Absorbed 429 attempts	Native resends / continuations
72 s	Off	10 s	99.300 s	8	0 / 0
72 s	Resume	10 s	99.295 s	8	0 / 0
72 s	Off	300 s	99.295 s	8	0 / 0
6 s	Off	10 s	8.018 s	3	0 / 0

All four executions reached the third POST, made four model POSTs in total, and passed the scripted artifact oracle. Even the enabled continuation condition completed in its initial attempt. The generated 10-second configurations contain a 10,000-millisecond stream-idle value. While the wrapper absorbed failures and waited, it withheld response headers; it sent headers when the successful response became deliverable. The observed long wait therefore did not trigger the configured idle timer.

The earlier silent-upstream Codex check sent HTTP 200 and stream headers before withholding the body. It tested a different phase and cannot establish a total deadline from POST submission. The four new cases bound the interpretation of the current timer axis and invalidate that phase transfer in the old replay. They do not establish an unlimited pre-header wait, general equivalence between

recovery policies, or a cloud-model benefit. Scripted usage fields are protocol placeholders and are not model-service charges.

E REPLAY MODEL AND INSTRUMENTATION

E.1 RECORDED EVENTS AND TIMING

Each run records the scheduled windows, request arrivals, individual recovery attempts, errors exposed to the shell, shell cancellations, response timing, usage, and the final oracle result. Scheduled faults, attempts absorbed inside the wrapper, and failures delivered to the shell are different events. Their separation makes it possible to distinguish an unencountered outage from successful hidden recovery and from a failure handled by the shell.

Streaming instrumentation must preserve delivery timing. A proxy that waits for the complete answer before handing it onward can turn an active stream into apparent silence and trigger a first-event timeout. The relevant transparent paths relay chunks as they arrive. The later response-phase proxy separately buffers complete frames; Appendix E.5 distinguishes these instruments and does not generalize the transparency of one to the other. Request-start, first-byte, completion, and client-disconnect observations should be reconciled rather than treating a late upstream success as proof that the shell received it. Calibration tests exercise the instrument without interpreting a scripted model's output as task-solving performance.

Timing validity is phase-specific. The existing OpenClaw calibration covers the first-event path it exercised, and the earlier Codex silent probe sent response headers before remaining silent. Neither validates applying a Codex stream-idle limit from POST submission while a jittered wrapper withholds headers. The latter assumption is contradicted by Appendix D.7. The local post-event experiments separately exercise silence after a legal stream event, with the connection's later behavior recorded. Replay omits these connection states and the full sequence of in-stream gaps, so evidence for one silence phase cannot validate all others.

A connection cancelled by the prompt-round deadline is distinguished from a request-level timeout. Attribution uses the adapter's recorded timeout metadata, the identified prompt round, and request timing; a kill exit code or an old outer flag alone does not determine the cause. If necessary sources or timing are missing, the cancellation remains unknown in failure counts, strata, and prefix coverage. A prefix decision is unavailable when such an event could change its first three visible failures. The observed disconnect and final oracle remain in the record, and attribution uncertainty does not remove the task from the assigned denominator.

E.2 REPLAY PREDICTIONS

The replay takes an observed no-injection trace and advances its request timeline through a specified schedule and recovery policy. It adds the waiting and retry consequences allowed by the model, checks the added duration against each recorded round's timeout, and applies the configured shell termination rules. It generates no new language-model output and does not execute alternative tool actions. Its prediction is conditional on the recorded nominal trajectory and transition rules. For the Codex–jittered pre-header path contradicted by Appendix D.7, the old replay expectation and dependent residual are inapplicable and are reported as unavailable with that reason, while actual task outcomes remain in the analysis; no L_i is instantiated for the Codex CLI comparisons of Sections 2 to 6.

Mapping shell termination to task success requires an additional assumption. The primary mapping predicts failure when termination prevents requests that remain in the nominal trace. An alternative maps any fatal termination to failure. These are sensitivity analyses, not mathematical upper and lower bounds on the probability that the real task succeeds. Work may already be complete before the fatal request, while a nominally final response may still be essential. The oracle settles the actual outcome.

For a supported replay condition at a fixed timer setting, let $\widetilde{Y}_{ij}(\pi, z)$ be the predicted outcome obtained from nominal source trace j for task i; $\widetilde{Y}_{ij}$ and $\bar{Y}_i$ below are both evaluated under one endpoint, the original oracle of Appendix F.7, and $\bar{Y}_i$ is the mean over the assigned executions of that

cell, a single binary outcome where the cell is executed once. For a replayable request-only condition, we retain all prespecified nominal sources from that same decoding, request-timer, recovery, and effective provider-limit disposition, including failures and traces that do not reach the anchor. We report source identities, source-specific predictions, and their average; nominal records from different policies or timers are not interchangeable repetitions. The request-only replay does not model native conversation continuation. With actual mean outcome $\bar{Y}_i$, define the diagnostic loss

$$L_i(\pi, z) = \frac{1}{J_i} \sum_{j=1}^{J_i} \widetilde{Y}_{ij}(\pi, z) - \bar{Y}_i(\pi, z).$$

Positive L means that actual completion is below this replay prediction. It does not identify model confusion, state loss, or a causal effect of the harness. Different generated trajectories, omitted timer behavior, and nominal-source selection can also produce discrepancies. A negative value is equally informative and must not be discarded because it contradicts an expected loss.

E.3 Exposure as an outcome

Exposure depends on the request timeline, and that timeline can depend on recovery. Error count, cancellation, and reaching the anchor are therefore post-treatment measurements. We report them to explain execution, but do not filter or match the primary policy comparison on them. All scheduled outcome runs remain in the analysis population, including failures and runs with zero exposure.

For replay diagnostics, we additionally report the relationship between nominal-source and actual anchor reachability. An aligned diagnostic can reveal whether a residual was driven by mismatched exposure opportunities. Such a sensitivity analysis changes the diagnostic population; it cannot replace the all-assigned policy effect. If a residual disappears after alignment, the evidence supports an exposure-selection explanation rather than an unexplained task-level deficit.

E.4 Runtime and usage accounting

Runtime and model-service costs are secondary outcomes from the same assigned runs, including failures. Usage covers all upstream attempts, including those hidden by request recovery and those issued after conversation continuation. Cached input is normalized within total input rather than added twice. Complete cost comparisons require complete usage and an explicit price basis; partial known usage is reported with its coverage, and missing usage remains unknown. A missing cost does not invalidate a measured task-oracle result. Post-hoc logging and cost summaries are not financial admission conditions for the experiment, nor do they establish an enforced fee cap.

Two instruments describe the same 72 startup assignments. The provider-side proxy at the supplier entry counts every upstream attempt, including retries absorbed before the shell saw them: 805 POST attempts, 619 with reported usage, 186 unknown and 5,239,896 known tokens (Table 52 in Appendix M gives every execution's totals). The CLI logs of all attempts record 331 native reconnects, counted as error events whose message begins with `Reconnecting...`, and twelve same-thread continuations. The runner's outer `tap.jsonl` records 805 POST attempts, one try each: 448 with reported usage, 357 unknown, and 4,203,059 known tokens summed from each cell's `usage.known_tokens.total_tokens`; only the 24 nominal cells carry a complete top-level total (1,621,331 in all), so that figure is not the client-side known subtotal. The two instruments are reported separately and never combined.

E.5 Stream activity and controlled framing

A recovery comparison also depends on which silence the runtime observes. We reuse all 24 nominal startup traces, comprising 28 CLI rounds and 178 complete responses. The outer transport labels 160 requests as 499/abandoned, but every one has already delivered response.completed. These are not 160 failed model responses. Completion is established at the observer's write boundary, not by a private model-consumption clock.

Maximum observed created-to-completed age is 32.436 seconds; maximum pre-first-event wait is 28.057 seconds; maximum complete-event gap is 3.526 seconds. The longest request-to-completed

observation is 35.136 seconds. Long reasoning and tool-argument sequences last 25.015 and 25.903 seconds while delivering nonempty deltas. These traces contain no repeated status heartbeat stream. Calling all activity without a completed tool action useless would turn normal ongoing output into an artificial failure. The four strategy configurations also influence the observed sample, so it is not an unconditional service-latency distribution.

A separate scripted experiment isolates a framing boundary through the real harness, CLI and byte-transparent tap. The logical tool arguments and planned completion horizon are the same, but the service sends either many complete argument-delta events or one event progressively filled over 25.903 seconds. Each case has a 60-second original round, no retries and no continuation. The final task is an exact marker file, not a benchmark reasoning task.

Table 17: Stream framing changes the observed timeout outcome. One execution is reported per condition. Offsets start at the specified schedule origin. Not delivered means the connection closed before the completion event; it is not a zero-time completion.

Framing	Idle limit	Marker score	Completion event offset (s)	Round end (s)	Local model POSTs
Many complete deltas	10 s	1	25.922231 s	27.783 s	2
One progressive delta	30 s	1	25.920394 s	26.754 s	2
One progressive delta	10 s	0	Not delivered	10.244 s	1
Many complete deltas	30 s	1	25.917644 s	26.829 s	2

The failing case closes at 10.003613 seconds after the created-event write return, reporting idle timeout waiting for SSE, with 49.805815 seconds of round allowance left. An independent raw receiver on the same transport path reads continuously while the single delta is incomplete, with a maximum adjacent read gap of 0.516836 seconds. This excludes complete-frame buffering in that tested tap path and exhaustion of the outer round. It does not observe the CLI's internal consumption function. The renderings differ in wire overhead and intermediate parse availability; they are not identical byte streams.

These four cases establish a controlled interface dependence with a simple task consequence. They do not establish that the real provider emits long unclosed frames, the frequency of natural hangs, or success rates of a recovery strategy. The earlier response-phase proxy does buffer complete frames, while the heldout observer forwards the bytes it receives; those layers cannot be conflated. The 90-second absolute-request plus 30-second idle comparator derived from nominal observations is a development proposal, not a validated policy. It is not used in Table 17. Ordinary total and idle limits remain necessary controls for any stronger claim about progress-aware waiting.

F WORDING INTERVENTIONS AND SOURCE CHECKS

This appendix reports the two-task wording intervention, its frozen labels and subsequent source checks. The historical extension combines these labels with original-oracle scores on four other tasks. We distinguish that mixed endpoint from the complete six-task evaluation; later readers do not overwrite the earlier labels or revalidate the constructed copies.

F.1 INTERVENTIONS AND INDEPENDENT READING

We use all 24 original runs of tasks 051 and 068: four strategies, three regimes and one run per cell. Every workspace receives an unchanged copy, two wording variants and two violating-control variants, yielding 120 full copies. Four original workspaces lack target outputs and remain unfilled in every branch. Equivalent variants change only existing date-exclusion or availability-prohibition language. SQL, structured outputs and unrelated checks remain fixed. Nine global rules cross the original and two alternative expressions for each task, applying the same task-specific rule to all strategies and regimes, including nominal artifacts.

The complete six-task evaluation of Section 4 uses four shared readings built from the same principle. Its frozen reading accepts interface-conforming injected objects even when falsey and permits a coherent completion-condition reading of the disputed launch dependency; the alternatives restrict injections to truthy objects or impose strict approval precedence, and all four combinations apply globally rather than per policy.

The anonymous reading contains 82 deduplicated full workspaces and 96 oriented pairs. Deduplication reduces reading, not the 24-run denominator. A separate AI reader without the experiment's earlier history judges public inputs, complete artifacts, the target constraints and pairwise equivalence without policies, oracle scores or the private mapping. Decisions are frozen before the mapping is joined. After freezing its judgments, the reader received updates about the study; subsequent clarifications were therefore informed by those updates. The study supplies one independent AI reading, not human adjudication, double blinding or inter-rater reliability.

All 40 nonempty equivalent pairs are judged equivalent: 20 actually change bytes and 20 are self-pairs. Eight missing-output pairs remain null. Forty nonempty violating controls are judged whole-task violations and nonequivalent; eight missing controls remain null. The complete original oracle nevertheless awards full credit to 30 of the 40 constructed violations. The SQL incorrect-aggregate controls fail the complete oracle, whereas incorrect SQL explanations retain full credit. Both launch-control families can retain full credit despite explicit contract violations. These controls test possible false acceptance; they are not spontaneous model errors or new task samples.

F.2 AGGREGATE COUNTS AND LABEL CHANGES

Table 18: Artifact-level checks change which outputs count as successful. Original scores remain recorded. The nominal date case admits a disclosed alternative reading, represented separately by $q \in [0, 1]$; the frozen label is not silently replaced.

Task and condition	Original score	Frozen artifact judgment	Evidence for the difference
051, native11, recoverable	0.9630	Satisfied	Complete date exclusion expressed without the oracle's preferred words
068, R10, recoverable	0.9322	Satisfied	Early-availability phrase occurs inside a prohibition
068, native11, silent	1.0000	Violated	A seven-field row under a five-column CSV header shifts the required status
068, B10, nominal	1.0000	Violated, date interpretation disputed	Claims list due May 1 depends on approval scheduled May 10

The frozen judgments classify 18 originals as satisfied, two as violated and four as no output. The original oracle also passes 18, but not the same 18. The CSV violation is mechanically inspectable.

The nominal date case is different: interpreting its dates as latest permissible deadlines rather than scheduled completion can weaken the violation. We retain the frozen label and separately let that one cell's satisfied value be $q \in [0, 1]$. It changes nominal and B10-adjusted comparisons, not fault-only direct differences. An unchanged aggregate total at the frozen judgment does not make this ambiguity disappear.

Table 19: Correcting two selected tasks changes their local comparison but not the six-task winner. Two-task rows use four fault cells per policy; six-task rows use twelve. The mixed six-task endpoint retains the original graders on four tasks. Cell counts are not counts of independent tasks.

Population and endpoint	Fault cells per policy	B10	R10	native11	B30	R10 - native11 (proportion)
Two selected tasks, original oracle	4	0	3	3	4	0
Two selected tasks, frozen independent judgment	4	0	4	3	4	+1/4
Six tasks, original oracle	12	0	9	11	12	-1/6
Six tasks, two judgments plus four original evaluators	12	0	10	11	12	-1/12

The two-task comparison changes from a tie to a continuation advantage over native11. Native11's newly accepted 051 output and newly rejected 068 CSV cancel in its total, while R10 gains one. In the six-task extension, 48 of 72 cells still use their original oracle. It is a mixed-endpoint sensitivity calculation, not a six-task semantic success rate. It still favors native11 and B30 on the fault average. Neither this result nor the two-task change establishes a general recovery ranking.

F.3 Sensitivity to shared wording rules

The wording experiment answers a different question from Table 19: how much can the same original oracle's comparison move within a finite, independently supported set of expressions? For a fixed contrast C, we report $[\min_{g \in G} C(Y^g), \max_{g \in G} C(Y^g)]$, where G contains the nine common rules. It is a sensitivity range over that finite set, not a confidence interval, a corrected success rate or a bound over all valid language.

Table 20: Shared wording rules can change the local policy contrast after nominal adjustment. Entries span nine common wording rules for the original grader. They are finite sensitivity ranges, not confidence intervals or bounds over all valid expressions. Direct and matching nominal-adjusted ranges coincide here.

Population	Recoverable contrast range	Silent contrast range	Equal-weight fault range
Two selected tasks	[-1/2, +1/2]	[0, 0]	[-1/4, +1/4]
Six tasks, wording changes on two only	[-1/6, +1/6]	[-1/3, -1/3]	[-1/4, -1/12]

The intervention supplies 120 copies of the 24 runs of tasks 051 and 068; forty nonempty violating controls are judged invalid by the frozen AI reader while the unchanged oracle passes thirty of them, and the nine global wording rules alter contrasts even after nominal adjustment. Different common rules can favor opposite policies locally. Their nominal effects happen to match between policies, so subtracting nominal differences does not remove the sensitivity. The six-task mean across both fault regimes remains negative throughout this set. A wider per-artifact envelope permits a different expression in every cell and is reported in the underlying analysis; it is not an observed result under one global rule.

These findings establish a consequential defect in particular comparisons, not a broadly calibrated replacement evaluator. Whole-deliverable validity differs from satisfying the rewritten target sentence: an equivalent rewrite can preserve an already malformed CSV. The controls and independent reading prevent a more permissive wording check from automatically being treated as a more accurate task endpoint. Selection after suspect results, only two audited tasks and a single AI reader limit the reach of the measurement claim.

F.4 Complete regime-wise recovery contrasts

Table 21: Six-task fault totals depend on the evaluation endpoint. Counts use twelve assigned fault cells per policy. Complete readings count satisfied artifacts; original and mixed rows retain their stated pass definitions. Direct and nominal-adjusted contrasts coincide.

Endpoint	B10	R10	native11	B30	R10 - native11
Original oracle	0/12	9/12	11/12	12/12	-1/6
Earlier two-task judgment plus four original evaluators	0/12	10/12	11/12	12/12	-1/12
Complete: broad interface, coherent dependency (frozen)	0/12	10/12	7/12	11/12	+1/4
Complete: truthy-only interface, coherent dependency	0/12	10/12	9/12	11/12	+1/12
Complete: broad interface, strict precedence	0/12	10/12	6/12	11/12	+1/3
Complete: truthy-only interface, strict precedence	0/12	10/12	8/12	11/12	+1/6

Table 22: Regime-specific R10 contrasts under original, mixed and complete readings. Each single-regime difference averages six tasks; the fault average weights regimes equally. Each direct difference equals its matching nominal-adjusted difference. Ranges reflect four shared readings, not sampling uncertainty.

Control	Fault regime	Original grader	Mixed endpoint	Frozen complete reading	Range over four readings
Native11	Recoverable	0	0	+1/3	[+1/6, +1/2]
Native11	Silent	-1/3	-1/6	+1/6	[0, +1/6]
Native11	Mean of both	-1/6	-1/12	+1/4	[+1/12, +1/3]
B30	Recoverable	-1/6	0	+1/6	[+1/6, +1/6]
B30	Silent	-1/3	-1/3	-1/3	[-1/3, -1/3]
B30	Mean of both	-1/4	-1/6	-1/12	[-1/12, -1/12]

Across all 72 assignments the four complete readings accept 48, 54, 46 and 52 artifacts in the order of Table 21, against 56 under both older endpoints. The old and new readers agree on 23 of the 24 common originals; the nominal launch-date case changes from violated to satisfied under the coherent reading, and both labels are retained in Table 18.

F.5 Repeated executions of the six-task plan

The six-task plan was executed twice more, unchanged except for its repetition number: 72 cells each, with the same tasks, policies, regimes, first-POST anchor and round allowances. These are two more executions of one frozen plan, not a new experiment, a seed-sensitivity study or a population estimate, and the original 56/72 is unchanged; the original evaluator passes 55 and 53 of the replicate cells.

A separate AI reader applied the original batch's frozen rules under all four shared readings to the 144 replicate artifacts, from a neutral package without policy, regime or score, and opened no label that any other reader had given the replicates. It verified that the replicate prompts are byte-identical to the original task contracts, that the protected inputs are present in every workspace and that the frozen expected values of tasks 051, 054, 068 and 080 recompute unchanged, and it recorded no divergence from the frozen formulas; the original execution keeps its Table 21 labels and was not relabeled. Over the 144 artifacts it records 103/8/33, 106/5/33, 102/9/33 and 105/6/33 satisfied, violated and no-output labels under broad/coherent, truthy/coherent, broad/precedence and truthy/precedence.

Fault counts B10/R10/native11/B30 out of twelve on the original execution and the two replicates, with the R10-minus-native11 numerator in twelfths:

- Original oracle: 0/9/11/12 (-2), 0/9/11/12 (-2), 0/6/12/12 (-6).
- Broad/coherent: 0/10/7/11 ($+3$), 0/7/12/11 (-5), 0/6/10/11 (-4).
- Truthy/coherent: 0/10/9/11 ($+1$), 0/8/12/12 (-4), 0/6/10/12 (-4).
- Broad/precedence: 0/10/6/11 ($+4$), 0/7/11/11 (-4), 0/6/10/11 (-4).
- Truthy/precedence: 0/10/8/11 ($+2$), 0/8/11/12 (-3), 0/6/10/12 (-4).

The contrast changes direction across executions: positive on the original execution under every complete reading, the numerator is negative in all eight replicate cells (Section 4). With the reading held fixed, it moves across the three executions by 8, 5, 8 and 6 twelfths under the four complete readings, and by 4 under the original oracle, changing sign under every complete reading; with the execution held fixed, the four complete readings move it by 3, 2 and 0 twelfths and never change its sign. Replacing the original oracle by complete evaluation changes the sign on the original execution only. The equal-weight maximal set is {B30} under all five readings of the original execution; on the first replicate it is {B30} under the original oracle and {native11}, {native11, B30}, {native11, B30} and {B30} under the four complete readings, and on the second replicate {native11, B30} under the original oracle and {B30} under every complete reading. The region above $w = 2/3$ in which R10 uniquely leads on the original execution is absent from both replicates under every complete reading: there R10 leads at no weight and ties the maximum only at $w = 1$, in six of the eight cells.

The k of Appendix F.8 counts the fewest verdict changes, on fault cells with a deliverable, that would change a reported quantity; nominal cells leave a fault quantity unchanged. The equal-weight maximal set has $k = 1$ in thirteen of the fifteen reading-by-execution cells and $k = 2$ in the other two, the second replicate under both truthy readings. The numerator's k is its distance from zero: 3, 5 and 4 under broad/coherent, 1, 4 and 4 under truthy/coherent, 4, 4 and 4 under broad/precedence, 2, 3 and 4 under truthy/precedence and 2, 2 and 6 under the original oracle, against re-execution movements of 8, 5, 8, 6 and 4. Under every complete reading k is at most the movement re-execution produces; under the original oracle it is below it on two executions of three.

Two facts about how the replicates were read qualify these labels. Task 085, the only replicate task graded by running tests, had its public tests run through the original batch's standard-library runner, and through hand-written executors by two other replicate readers whose labels this appendix does not use, not through pytest; a later rerun of all 24 replicate 085 artifacts under the original evaluator's pinned pytest 8.4.2, 500 runs per artifact including the readers' call order and hash seeds, passed on every run exactly the 19 artifacts the substitute runners had passed and failed on every run the five unmodified workspaces labeled no output, so the substitute runners change no label. Separately, six delivered 085 reports quote the original absolute run directory, which names the harness path, the producing model and the run identifier but neither the policy nor the regime; the original batch's package replaced that prefix and the replicate package did not, so those six artifacts were not blind to the producing system, although no criterion or label depends on the path. Three executions of one plan on six tasks with one model and harness give observed movement, not a distribution or an interval; the held-out plan's second execution is in Appendix K.4.

F.6 Largest-overrun and CSV checks

Line 27 of the public prompt for task 054 reads `7. summary.md must name every review item and the largest overrun by variance_amount.`; lines 13 and 14 fix the column order of variance_report.csv and line 21 defines variance_amount as actual_amount minus budget_amount. Three reports that the original oracle credits with 1.0000

list the Marketing Ads overrun of 12,000.00 among their review items and name a smaller amount as the largest (Table 23). Each of the three variance_report.csv files contains the row `Marketing,Ads,50000.00,62000.00,12000.00,24.00,review`, equal to the expected row, and the frozen adjudication marks those rows satisfied; the four readings label all three cells violated (Table 12).

Table 23: Three perfectly graded reports identify a smaller item as the largest overrun. Report lines are quoted verbatim from summary.md with their original line numbers. The two columns show a contradiction within each report, not two separate generated reports.

Cell	Original score	Line listing Marketing Ads, 12,000.00	Line naming the largest overrun
054, rec, native11	1.0000	line 5: `- **Marketing Ads** -- overrun of 12,000.00 (variance_pct 24.00; variance_pct > 10).`	line 15: `**Product Research** has the largest overrun by variance_amount at **3,500.00**.`
054, rec, B30	1.0000	line 7: `- Marketing Ads: 12,000.00 overrun, 24.00% variance.`	line 15: `The largest overrun by variance amount is Operations Logistics at 6,000.00.`
054, sil, native11	1.0000	line 5: `- **Marketing Ads** -- actual spend exceeded budget by 12,000.00 (24.00%).`	line 15: `The largest overrun by variance amount is **Product Research**, with a variance amount of **3,500.00**.`

The oracle's 34 checks all pass on the three reports (Table 24). One check is named for the largest overrun: summary_largest_overrun, weight 0.04, passes when each word of the ground-truth value, Marketing Ads, occurs anywhere in the lowercased summary text, which the three reports satisfy through their review-item lines. No check reads the sentence that names the largest overrun or compares the pair it names with the 12,000.00 row, so the evaluator's only test of this requirement is keyword presence, and a report that contradicts the requirement satisfies it. The grader line is `_add(checks, "summary_largest_overrun", all(part in text_l for part in gt["largest_overrun"].lower().split()), 0.04, "largest overrun not identified")`.

Table 24: The 34 task-054 grader checks all pass on the three contradictory reports. Weights sum to 2.15. The score is passed weight divided by 2.15, capped at 0.69 if any review-reason check fails. The largest-overrun check accepts the two required words anywhere in the report, even when the named maximum contradicts its own table.

No.	Grader check	Weight	Tested condition and source line
1	fixtures_unchanged	0.08	SHA256 of the two input files under in/ unchanged (L40)
2	fixtures_present	0.08	in/budget.csv and in/actuals.csv exist (L45)
3	csv_exists	0.08	out/variance_report.csv exists (L46)
4	rollup_exists	0.08	out/department_rollup.csv exists (L47)
5	review_reasons_exists	0.08	out/review_reasons.json exists (L48)
6	summary_exists	0.06	out/summary.md exists (L49)
7	exact_header	0.08	variance_report.csv header equals the expected column order (L54)

Table 24 (continued)

No.	Grader check	Weight	Tested condition and source line
8	exact_rows	0.34	variance_report.csv rows, sorted, equal the ground-truth rows (L56)
9	number_format	0.08	the three amount columns carry two decimals (L59)
10	review_flags	0.1	the set of rows flagged review equals the expected seven (L60)
11	pct_format_or_na	0.04	variance_pct has two decimals or is N/A (L61)
12	full_outer_join_rows	0.04	the Product Research and Security Tools rows exist (L62)
13	zero_budget_pct	0.04	variance_pct is N/A for Product Research and Support Escalations (L63)
14	missing_actual_ defaults_zero	0.04	actual_amount is 0.00 for Security Tools (L64)
15	strict_threshold	0.04	Sales Travel has variance_pct 10.00 and flag ok (L65)
16	rollup_header	0.06	department_rollup.csv header equals the expected (L85)
17	rollup_rows	0.22	department_rollup.csv rows equal the expected (L86)
18	rollup_review_counts	0.04	review_item_count is 2 for Marketing and 0 for Sales (L88)
19	rollup_zero_budget_ na	0.04	variance_pct is N/A for the Product and Support departments (L89)
20	rollup_flag_from_ review_items	0.04	rollup_flag is review for Marketing and ok for Sales (L90)
21	review_reasons_keys	0.06	the key set of review_reasons.json equals the expected seven (L109)
22	review_reason_types	0.18	each reason_type equals the expected (L110)
23	review_reason_ primary_drivers	0.06	each primary_driver is non-empty (L111)
24	summary_mentions_ Marketing Ads	0.015	every word of 'marketing ads' occurs in the lowercased text (L130)
25	summary_mentions_ Marketing Events	0.015	every word of 'marketing events' occurs in the lowercased text (L130)
26	summary_mentions_ Operations Logistics	0.015	every word of 'operations logistics' occurs in the lowercased text (L130)
27	summary_mentions_ Product Research	0.015	every word of 'product research' occurs in the lowercased text (L130)
28	summary_mentions_ R&D Tools	0.015	the text contains r&d, research and development or r and d, plus tools (L130)
29	summary_mentions_ Security Tools	0.015	every word of 'security tools' occurs in the lowercased text (L130)
30	summary_mentions_ Support Escalations	0.015	every word of 'support escalations' occurs in the lowercased text (L130)
31	summary_largest_ overrun	0.04	**each word of the ground-truth value Marketing Ads, split on spaces, occurs anywhere in the whole summary text lowercased at L123; the sentence naming the largest overrun is not read and the pair it names is not compared (L131). The three reports pass it through their review-item lines**
32	summary_mentions_ unplanned	0.015	the text contains unplanned, product and research (L132)
33	summary_mentions_ zero_budget	0.015	the text contains zero budget or zero-budget, plus support or product (L133)
34	summary_mentions_ missing_actual	0.015	the text contains missing actual, security and tools (L134)

The other nine task-054 cells: six reports, the four nominal cells and the recoverable R10 and silent B30 cells, name Marketing Ads 12,000.00 as the largest overrun on line 14 or 15 and list the same value among their review items, satisfied under all four readings with original score 1.0000; the

three no-output cells, B10 under both fault regimes and R10 under permanent silence, hold the same anonymous sample with an empty out/ directory and none of the four required paths, scoring 0.0744, the two fixture checks' 0.16 over 2.15.

We then applied one post-hoc but fixed source check to every assignment for tasks 054 and 068 in all three executions: two tasks, four policies, three regimes and three executions give 72 cells. For 054 it independently recomputes the unique largest positive variance from the publisher's `budget.csv` and `actuals.csv`, checks that row in `variance_report.csv`, and parses the one largest-overrun assertion in `summary.md`; the inputs have the publisher's pinned hashes in all 36 cells. For 068 it parses `launch_checklist.csv` as strict CSV and requires exactly the source's five named columns in every row. An ambiguous assertion stops the checker rather than guessing a label, and absent target files remain no output. Only an original-oracle pass that this check demonstrates to violate a source requirement is withdrawn; no failed cell is upgraded and no verdict on the other four tasks changes.

Table 25: The same source checks change the first GLM comparison but not its two reruns. P/V/N denote pass/violation/no output over twelve assignments per task and execution. Contrasts use all six tasks and twelve fault cells per policy; unrelated original labels remain unchanged. Checks were specified after the first execution was inspected.

Execution	Task 054: P / V / N	Task 068: P / V / N	Original R10 - native11	Corrected R10 - native11
First	6/3/3	9/1/2	-2/12	+1/12
Rerun 2	9/0/3	9/0/3	-2/12	-2/12
Rerun 3	9/0/3	9/0/3	-6/12	-6/12

The complete two-task-corrected fault counts for B10/R10/native11/B30 are 0/9/8/11, 0/9/11/12 and 0/6/12/12 on the three executions. B30 remains the equal-weight leader on the first two and ties native11 on the third; Table 25 isolates the R10-versus-native11 comparison rather than a change in the four-policy optimum.

The released `evidence/fixed-source-054-068/` contains the two target outputs for each delivered 054 cell, the target CSV for each delivered 068 cell, an anonymous 72-cell manifest with file hashes, all 216 original-oracle binary scores and the executable check. Given the publisher's pinned 054 input fixtures, it recomputes Table 25 without an AI reader. It does not distribute the other four tasks' agent outputs or turn this selected two-task check into complete six-task truth.

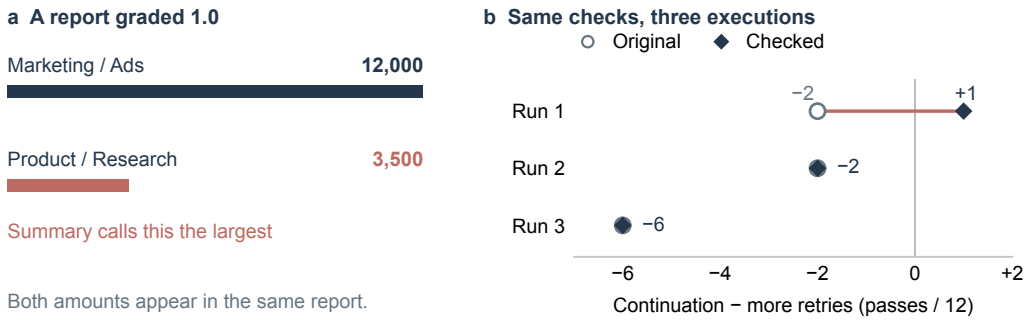


Figure 6: A report's stated maximum and the same source checks across three executions. (a) A report graded 1.000 lists Marketing Ads at 12,000 but calls Product Research at 3,500 the largest overrun; labels are paraphrased. (b) Checks on tasks 054 and 068 change continuation minus more retries from -2 to +1 fault passes on execution 1; executions 2 and 3 remain at -2 and -6. The denominator is twelve per policy. Checks revoke 4, 0 and 0 passes and retain other task labels. Wait longer remains execution 1's leader at 11/12.

F.7 Endpoint definitions and sensitivity analyses

Let $A_{i\pi z}$ be the final artifact for task i, policy π and regime z; the original binary endpoint is $Y_{i\pi z} = O_i(A_{i\pi z})$ with partial scores kept separately, $J^{(2,s)}$ is the complete AI evaluation under shared interpretation s, and two historical endpoints are retained, an earlier two-task judgment $J^{(1)}$ and its mixed extension M, which uses the original oracle for the other four tasks. Satisfied contributes one; violated or no output contributes zero, and historical compliance is separate: historical test order requires an execution record; missing outputs and unresolved historical actions retain zero credit.

Every task has equal weight; we report each fault-regime policy difference and their equally weighted average, and the nominal-adjusted contrast subtracts the corresponding nominal difference. If $e = Y - J$, changing endpoints changes that adjusted contrast by the same difference-in-differences of e, so a shared oracle does not force cancellation; calling that disagreement evaluator error requires evidence about the task and actual artifact. The fault-mixture endpoint below reweights the two fault regimes by w and $1 - w$ with tasks and policies fixed; the original endpoint is $w = 1/2$, and these post-hoc weights are hypothetical mixtures, not observed deployment frequencies.

With $c^e_{\pi,r}$ and $c^e_{\pi,s}$ the recoverable and permanently silent success counts of Appendix F.4, the empirical fault-mixture score and its maximizing set over the four policies are

$$S^e_\pi(w) = \frac{w c^e_{\pi,r} + (1-w) c^e_{\pi,s}}{6}, \qquad \mathcal{A}^e(w) = \arg\max_\pi S^e_\pi(w).$$

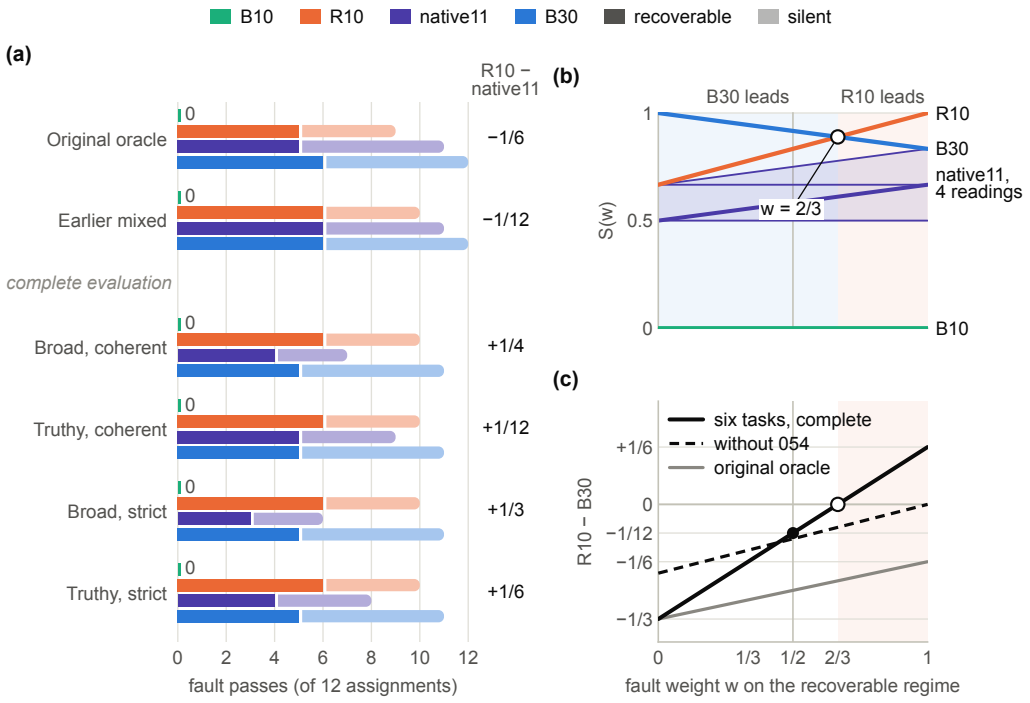


Figure 7: Endpoint and fault-mixture sensitivity. (a) Fault passes out of twelve per policy under the original, mixed and four complete readings. (b) The exact policy scores S(w), where w is the recoverable-regime weight, cross for R10 and B30 at w=2/3. The four complete readings differ only in native11's line. (c) The R10-B30 contrast changes when task 054 is omitted or the original grader is used. All curves reweight the same 72 assigned cells; weights are post-hoc sensitivity parameters, not new executions.

Table 26 reports every policy line for the original, mixed and four complete endpoints. Exact rational intersections, the two endpoints and one point inside each resulting interval determine the entire

maximizing set. No grid or interpolation of new trajectories is used. Absent deliverables retain zero credit and all six tasks remain in every line.

Table 26: Exact score lines for the fault mixture. Entries are numerators $6S^e_\pi(w)$; w weights the recoverable regime. All lines use the same 72 assignments. Weights are post-hoc sensitivity parameters; intersections are solved exactly rather than read from a plotted grid.

Endpoint	B10	R10	native11	B30
Original oracle	0	4+w	6-w	6
Earlier mixed	0	4+2w	5+w	6
Complete: broad, coherent	0	4+2w	3+w	6-w
Complete: truthy-only, coherent	0	4+2w	4+w	6-w
Complete: broad, strict precedence	0	4+2w	3	6-w
Complete: truthy-only, strict precedence	0	4+2w	4	6-w

The original oracle selects B30 uniquely for $0 < w \leq 1$, with a B30/native11 tie at $w = 0$. The earlier mixed endpoint selects B30 uniquely for $0 \leq w < 1$, with R10/native11/B30 tied at $w = 1$. Under each complete reading, $S_{R10}(w) - S_{B30}(w) = (3w - 2)/6$, so $\mathcal{A}(w)$ is $\{B30\}$ for $0 \leq w < 2/3$, $\{R10, B30\}$ at $w = 2/3$ and $\{R10\}$ for $2/3 < w \leq 1$.

Adding nominal weight does not move this complete-evaluation boundary. With a fault fraction f, define $T^e_\pi(f, w) = (1 - f)c^e_{\pi,n}/6 + fS^e_\pi(w)$. Within each complete reading, R10, native11 and B30 have equal nominal counts: five under broad and six under truthy-only. The equality is a property of the screened population rather than of the policies: Appendix B.1 admitted only tasks with three nominal passes, which suppressed nominal variance among the leading policies before any fault was injected, so the invariance below is conditional on that screen. B10 has the same count under coherent dependency and one fewer under strict precedence. Thus for any $f > 0$, the maximizing regions above remain, and $T_{R10} - T_{B30} = f(3w - 2)/6$. At $f = 0$, all four policies tie under coherent dependency and R10/native11/B30 under strict precedence: there is no fault contrast. Here f and w parameterize mixtures of the two controlled fault regimes.

The task dependence is sharp. Across all four complete readings, omitting budget task 054 gives $S_{R10} - S_{B30} = -(1 - w)/5$: B30 uniquely leads all four policies for $w < 1$, and R10 can only tie at one. That endpoint also includes native11 under truthy/coherent. Omitting schema task 080 gives $(2w - 1)/5$, moving the B30/R10 boundary to $1/2$. Omitting any of 051, 060, 068 or 085 gives $(3w - 2)/5$, retaining the $2/3$ boundary. Direct enumeration of all four policies in each of these 24 reading/omission cases confirms the stated leaders; no pairwise result becomes a global maximum without that check. These are different five-task populations, not intervals or independent repetitions.

F.8 Verdict-change sensitivity

Under one reading and execution, a verdict change flips the verdict $v(c) \in \{0, 1\}$ of one fault cell c with a deliverable. Let $T_\pi = c_{\pi,r} + c_{\pi,s}$ (F.7), $D = T_{R10} - T_{native11}$, $M = \arg\max_\pi T_\pi$, g the gap between the two largest totals and W^* the set of $w \in (0, 1)$ where $\arg\max_\pi S_\pi(w)$ changes; for Q the sign of D, M or W^*, $k(Q)$ is the fewest verdict changes after which Q differs.

Proposition 1. (i) $k(D) = |D|$ if $D \neq 0$, else 1: a change moves D by one on an R10 or native11 cell and by zero elsewhere, so the sign survives $|D| - 1$ changes; withdrawing $|D|$ verdicts from the pair's leader reaches zero. (ii) $k(M) = \max(g, 1)$: a change moves any $T_L - T_\pi$ by at most one, so a unique leader L survives $g - 1$ changes and withdrawing g of its verdicts ties it; at $g = 0$ one change to a member of M alters M. (iii) W^* is empty exactly when one line is weakly maximal at both ends; a boundary then costs at least one plus a challenger's smaller endpoint gap, the gaps of (ii) on $c_{\pi,r}$ and $c_{\pi,s}$ alone; where exactly two lines cross, $k(W^*) = 1$, since any change to either line moves the crossing. The minimizing changes of (i) and (ii) withdraw satisfied verdicts, which always have deliverables, so the restriction is inactive; the released exhaustive search, imposing it, agrees in all fifteen cells ($k(W^*)$ is 1, or 2 where two identical lines lead).

Proposition 2. Let $N(e, e')$ count fault cells whose verdicts differ between executions e and e', N_{AB} those of the compared pair. Then $|D_e - D_{e'}| \le N_{AB} \le N$, and $N \ge k(M_e)$ if M differs, likewise W^*: flipping the differing cells one by one turns e into e', each flip moving D by at most one and no quantity before k flips (Proposition 1). So $m = |D_e - D_{e'}| \ge k(D_e)$ says re-execution has re-decided at least $k(D_e)$ verdicts of the compared pair, not that D changes sign: under the original oracle $m = 4 \ge k = 2$ on two executions, yet D stays negative, the re-decisions pushing it further from zero.

k is small partly because the grid is coarse, twelve fault cells per policy, so one verdict is a twelfth; the rule therefore compares k with movement measured on the same grid, never across grids. Nothing here assumes independence, estimates a flip probability or forms an interval; three executions give an observed m, and $m < k$ would not certify stability. k is the fragility index on a deterministic aggregation rather than a test statistic, and the margin of victory with verdicts as votes (Walsh et al., 2014; Xia, 2012); pass^k estimates reliability from repeated trials and facet decompositions fit a variance model, whereas k and m are two counts on one grid compared by inequality (Yao et al., 2024; Hardy et al., 2026).

F.9 Variance decomposition

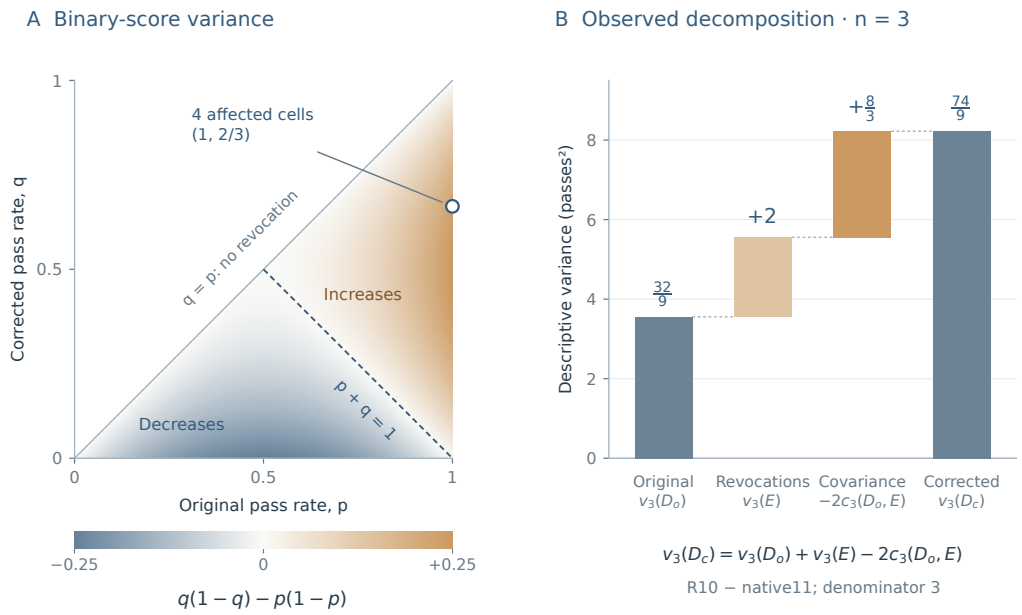


Figure 8: Variance under credit revocation. (a) For binary pass rates $p \ge q$, corrected minus original variance is $(p - q)(p + q - 1)$; its sign depends on $p + q$, and it is zero when $q = p$. The observed affected cells lie at (1,2/3). (b) For the three frozen R10-native11 margins, original variance 32/9, differential-revocation variance 2 and the covariance contribution 8/3 sum to corrected variance 74/9, using divisor three. The decomposition is exact for these fixed observations; its components retain their observed covariance.

For a fixed pair of policies and execution r, let $D_{o,r}$ and $D_{c,r}$ be the original and checked pass-count margins and $R_{h,r}$ the number of original passes revoked from policy h. With $E_r = R_{1,r} - R_{2,r}$ the identity $D_{c,r} = D_{o,r} - E_r$ holds exactly, and for any set of executions the empirical variance satisfies $v(D_c) = v(D_o) + v(E) - 2c(D_o, E)$. The correction amplifies the movement of the comparison when $2c(D_o, E) < v(E)$, preserves it at equality (a constant E is the simplest case), and attenuates it when $2c(D_o, E) > v(E)$. In the six-task executions, $\bar{D}_o = -10/3$, $\bar{D}_c = -7/3$, $\bar{E} = -1$, $v_3(D_o) = 32/9$, $v_3(E) = 2$, $c_3(D_o, E) = -4/3$ and $v_3(D_c) = 74/9$; with divisor two the three variances are $16/3$, 3 and $37/3$ with covariance -2, and the direction is unchanged. Of the 72 six-task cells, four increase their three-point variance from 0 to $2/9$, 68 are unchanged and none decreases; the two 054 recoverable cells of native11 and B30 both move from $(1, 1, 1)$ to $(0, 1, 1)$ while their

difference stays $(0, 0, 0)$, and the six-task R10-minus-B10 fault totals remain $(9, 9, 6)$ before and after correction. A strict counterexample to any monotone claim is the sequence $(1, 0, 0)$ corrected to $(0, 0, 0)$, whose variance falls from $2/9$ to 0; this constructed sequence demonstrates the variance decrease algebraically.

F.10 Source-check transfer to Kimi

We ran the six-task plan three times on Kimi K3, applying the task-054/068 source checks unchanged. Each execution assigns 72 cells across six tasks, four policies and three conditions. The plan was frozen at 01:14:24 Beijing time on 26 September 2026, and the three-execution extension at 01:17:41. The first task-054 cell ran from approximately 01:18 to 01:22. Analysis code was frozen at 01:19:59, during that cell and before its result was read (SHA-256 prefixes 265649de…, 58494521…, b4ab0f00…, respectively).

Each execution completed 70 cells and retains two provider-quota interruptions as unknown values in $[0, 1]$. The interrupted cells are 060 R10 silent and 080 native11 silent on execution 1, and 080 native11 recoverable and silent on executions 2 and 3. Each records ten HTTP 429 responses. Executions 2 and 3 ran alongside execution 1 from 09:46 on the same account; their durations reflect the shared-account execution conditions.

Table 27: The source checks confirm no revoked pass in three Kimi executions, with unresolved cases retained. Each task contributes twelve assignments per execution. A parsing-inconclusive case lacks a unique largest-overrun clause under the fixed checker and counts as neither a pass nor a confirmed violation. Across the two tasks, 39 checks pass, 22 are inconclusive and 11 have no output.

Execution	Task 054: pass / inconclusive / no output / violation	Task 068: pass / no output / violation	Confirmed revocations
First	2 / 8 / 2 / 0	11 / 1 / 0	0
Second	3 / 7 / 2 / 0	10 / 2 / 0	0
Third	3 / 7 / 2 / 0	10 / 2 / 0	0
Total	8 / 22 / 6 / 0	31 / 5 / 0	0

Table 28: Kimi fault-contrast intervals include zero under the fixed source checks. Units are fault passes out of twelve per policy. Bounds propagate quota-interrupted cells and parsing-inconclusive checks in the original denominators. All 216 planned cells are retained, with 210 complete and six quota-unknown.

Execution	Original R10 - native11	Checked R10 - native11	Definite sign reversal
First	[-3, -1]	[-3, +1]	no
Second	[-1, +1]	[-3, +2]	no
Third	[-2, 0]	[-3, +2]	no

The prespecified prediction that at least one originally credited source violation would move a policy contrast was not confirmed on this model; the corrected lower bounds fall because unresolved cells widen the interval, not because any revocation was observed.

G Agreement at artifact and obligation levels

G.1 Constructed artifacts and reader exposure

The frozen development merge contains twelve hand-built artifacts, six each for interface repair (083) and schema-drift audit (092). Author expectations were frozen after construction and before

sharing the expectation-masked bundle with the blinded reader. The reader froze its source-derived rules before inspecting candidates and froze judgments before author unblinding. The reader had previously evaluated the 61-workspace package and retained that context and model family. The original author performed the merge.

Author and reader agree sample by sample on four satisfied and eight violated whole artifacts. Their obligation sets are not interchangeable: the authors' private adjudication records retain 192 old-machine, 156 author-final and 60 reader-final cells, with overlapping rather than one-to-one clause groups. Clause-level agreement therefore requires aligning these overlapping groups first. Historical compliance remains unknown for all twelve artifacts, separately from final validity. These constructed artifacts form a separate author–AI comparison; the 72-assignment recovery denominator remains fixed.

G.2 Scope of the legacy-interface requirement

Task 083 requires preserving the new Product/Money model, supporting legacy dictionaries and retaining currency validation. In sample 002, the dictionary branch adds one to the supplied price; order pricing multiplies that value by quantity, and reports aggregate those order totals. Saved reader checks record 979 rather than 977 for a same-currency mixed-shape order and report totals GBP 576/USD 54 rather than GBP 575/USD 51. These are recorded counterexamples on supported inputs, not new executions in this analysis.

The author marks legacy support C03 and order/report behavior C04 violated. The reader marks F1 satisfied and legacy-support F2 violated; both whole-artifact labels are violated. F1's stated wording includes normal pricing/report aggregation for source-supported inputs. If it covers all supported shapes, the counterexamples also contradict its satisfied label. If F1 is restricted to the new Product path, F2 contains those failures instead. Both readings locate the failure, while assigning it to clauses with different input domains. A report of agreement must align the input domain of each clause and carry the observed counterexample through the call chain, rather than treating a shared final failure as clause-wise confirmation.

G.3 Cross-file consistency of rejection counts

Task 092 requires rejection records with source-row identities, a summary including rejected_row_count, and audit notes. Its source contains five invalid rows, E004-E008. E008 has two defects, so the provided per-reason representation of a complete rejection table would have six records for five identities. Sample 005 omits E004, the row with disallowed trial_pause: the actual table has five records but only four distinct identities. Its unchanged notes still describe six records and five distinct rows, and explicitly say the summary counts rows, not records. The JSON summary says five.

The author's D16 count check is satisfied because five matches the bad-row total in the source. The reader's F4 is violated because the table, summary convention and notes do not agree. Both already reject the artifact for missing E004, so this additional inconsistency changes no whole-artifact label. D16 verifies the source total, while F4 exposes the delivered table's inconsistent identities and counting convention. Their frozen values remain unchanged. Joining the actual rejected identities to the source reconciles the table, declared unit and notes.

The frozen merge accepts two supplied, qualified positive artifacts with different severity conventions, CSV representation and change-log interpretations. April 30's missing required header directly contradicts the log; April 29's bad values do not, by themselves, establish a persistent schema migration. The tested positive artifacts define the demonstrated representation alternatives; an explicit six-rejection-record summary remains an untested case.

G.4 Interpretation

The two cases distinguish a clause-domain disagreement from an additional cross-file contradiction. The clause-level evidence resolves differences hidden by whole-artifact agreement: define each clause's input domain, trace failures through dependent outputs, and reconcile identities and declared units across files. Each check follows an obligation exposed by the corresponding artifact.

The artifact checks identify concrete original-oracle acceptance errors while preserving the accepted choice-region calculation. The complete selection map follows the specified final-artifact evaluation. Testing new real outputs against obligations fixed in advance provides a direct next evaluation of the measurement result.

H Recovery from saved partial work

H.1 Paired continuation and fresh sessions

The initial midtask experiment uses task 085, already seen in screening and startup evaluation. Its two independently generated prefixes have completed tool work and changed implementation files, but have not written the required report. The next eligible request is interrupted before provider forwarding. Each checkpoint is restored at the same absolute workspace path for a resumed and a fresh session, with byte-identical files, remaining time, request opportunities and residual obstruction. Nominal and patient-wait controls start from four separately generated prefixes; they are not eight arms of one shared checkpoint.

Each full strategy receives a 600-second original round and 48 local POST opportunities, with native request and stream retries disabled and a 300-second idle setting. The interrupted local request counts against opportunities even though it never reaches the provider. Archival copying is outside the measured strategy; waiting and launching the suffix are charged. Prefix 1 leaves 558.151 seconds and 41 opportunities; prefix 2 leaves 580.460 seconds and 43. This matches offered allowances, not actual token consumption or monetary cost.

Table 29: Paired midtask suffixes and independently generated controls have different accounting units. Paired policies each include their shared prefix and charged recovery action; independent controls report their actual complete-policy use. The hypothetical paired totals must not be summed into a physical bill. All eight terminal artifacts are retained.

Source prefix	Policy	Original score	Whole-policy time (s)	Whole-policy local POSTs	Whole-policy input tokens
Paired checkpoint 1	Resume	0.95	143.057	15	157620
Paired checkpoint 1	Fresh	1.00	146.140	17	133623
Independent nominal prefix 1	Nominal	1.00	85.715	12	127468
Independent wait prefix 1	Wait	1.00	120.660	12	127273
Independent wait prefix 2	Wait	1.00	101.357	14	132742
Independent nominal prefix 2	Nominal	1.00	80.366	11	100792
Paired checkpoint 2	Fresh	1.00	173.496	20	189444
Paired checkpoint 2	Resume	1.00	200.275	19	227894

The paired policies each include their shared prefix in these hypothetical complete-policy columns; the independently generated controls report their actual complete-policy use. These columns must not be summed as a physical bill. Fresh passes twice and resume once. The only deduction is checkpoint 1's resumed report under in/out rather than the required out directory. Implementation, stability and test-integrity checks pass. Every paired suffix still modifies the sorting implementation, so the intervention is not merely a request to write a report after completed work. The original workspaces and scores are retained without moving the report to repair its grade.

Resume's report includes earlier failures while fresh concentrates on problems encountered after the checkpoint. The task asks for flaky behavior, cause and verification, but not a complete historical postmortem; we do not add a penalty to fresh for omitting earlier events. Resume uses fewer local POSTs but more input tokens in both pairs. Its first paired execution is slightly faster, its second slower. The pairs expose a trade-off between request count, input tokens and elapsed time.

The initial comparison has a treatment difference. Resume's newest input is a 344-character continuation notice; fresh's 1568-character input includes the full original task, its absolute report path and the same notice. The original task remains in resume's older history, but the latest-turn cue is unequal. Appendix H.2 matches the latest task text on these same checkpoints to separate task restatement from history retention.

Physical accounting for this initial experiment totals 108 local POSTs, including two planned cuts, and 106 provider POSTs, all HTTP 200 with completed responses. Observed input/output totals are 1,126,567/44,248 tokens, with 1,036,736 cached tokens already included in input. Monetary cost is unknown. One relay usage record remains null; a derived record recovers its usage from the matching original completed payload. Shared prefix use is counted once physically and once in each hypothetical full strategy. Earlier invalid endpoint attempts and their unknown usage remain separately disclosed in Appendix B.5; they are instrument failures, not model-solving failures.

H.2 Matching the latest task text

We reuse each saved checkpoint for three suffixes: R_notice resumes with the short notice; R_recap resumes with the full original public task followed by that notice; F starts a fresh session with the same latest task text as R_recap. No new hint identifies the earlier wrong directory. Each arm restores the original files and absolute workspace path, residual obstruction, and remaining allowance: 558.1513542500325 seconds and 41 opportunities at checkpoint 1, 580.4595804589335 seconds and 43 at checkpoint 2. Both resumed arms retain exactly the ordered old tool history, 12 and eight items respectively; fresh has none. The forwarded nonhistory request fields match. Native message identifiers differ, but the latest task text matches exactly within each R_recap/F pair. This is not equality of complete request payloads, input tokens or realized cost.

All six suffixes were attempted. Five complete normally; checkpoint 1's R_notice ends with an instrument-affected request failure. R_recap and F score 0.95 and 1.00 at checkpoint 1; R_notice leaves a 0.95 artifact. All three checkpoint 2 artifacts score 1.00. Every artifact passes the original implementation checks, and every suffix performs a remaining sorting repair. The two deductions concern only the report location, in/out/root_cause.md instead of out/root_cause.md. Run status is recorded separately from artifact score (Appendix B.6). Checkpoint 2 was executed in a later block after the instrument interruption, without regenerating prefixes or repeating checkpoint 1. These are two familiar sites on one task, not six independent tasks or a heldout replication; the earlier independent-prefix nominal/wait controls are not contemporaneous arms of this intervention.

The normally completed R_recap error changes the interpretation: restoring the full task to the latest turn did not repair this saved case, so unequal latest-task text cannot explain this R_recap/F observation. The trace shows R_recap checking, writing and finally rereading ../out/root_cause.md from in/flakyqueue, a relative path already present in the archived history. It also initially repeats the stable-priority-order assumption, then corrects the implementation after a failing test. The trace identifies the repeated path and the subsequent correction sequence. Fresh also makes mistaken intermediate assumptions, notably at checkpoint 2, before correcting them. Both session types revise intermediate assumptions during the observed repairs.

Restating the full task leaves the path error intact in the completed R_recap run at checkpoint 1; a short notice suffices at checkpoint 2. The equal R_notice and R_recap artifact scores at checkpoint 1 retain an execution asymmetry: the notice-only run ends with an instrument failure, leaving its eventual repair outcome unknown. Speed and token counts describe these checkpoint-specific executions.

Physical suffix execution uses 57 upstream send attempts, split 29/28 across the original/later blocks, with 56 complete responses and one unknown usage record. Known input/output subtotals are 618,354/22,680 tokens; complete totals and monetary cost are unknown. Historical prefixes are not new expenditure in this batch. The token accounting separates cached, uncached and reasoning tokens (Appendix B.6). The failed notice run counts 291.377 native monotonic seconds,

but recorded Unix timestamps span at least 21,354.881 seconds and lack a normal suffix-end event. Its recorded wall-clock span exceeds the 600-second recovery deadline. Appendix B.6 distinguishes the upstream/recording boundary from a separate, recoverable post-completion relay error.

H.3 Scheduling revisions

The scheduling-revision study derives a public contract from task 103. After a real first-round plan for twelve decisions across three regions, fresh and retained suffixes each run once from identical restored files. The retained suffix receives its native session with eighteen earlier tool calls; the fresh suffix receives no history.

Both exit normally and produce byte-identical revised plans and diffs, changing seven decisions and retaining five. The frozen evaluator nevertheless marks initial, revision and complete outcomes false for both. It expects bare roster keys instead of regional owner suffixes and a list instead of the delivered state container. An author audit under four global interpretations finds that accepting both representations satisfies every fixed obligation (Appendix B.8). This is an output-informed self-review with known identities.

Fresh takes 371.556 seconds and retained 200.966, with twelve POSTs each. Retained uses more input and fewer output tokens (Appendix B.7). The identical revisions achieve equal delivered results under the accepted representation, with distinct measured resource use under fixed execution order, cache exposure and recorded tool failures. The supplementary archive gives the stage-level accounting.

H.4 Finite decisions and a program baseline

A development task chooses among four binary configurations using a published operating loss and a per-bit switching loss from the approved incumbent 00. We restore two real initial recommendations for eight suffixes each: four conditions crossed with fresh and retained sessions. Each phase has a 540-second deadline and a 48-POST cap, with no injected fault.

Both initial recommendations are 00 and correct. All sixteen suffixes are legal, immutable, timely and exactly optimal, with zero regret. The prespecified conjunction required retained to fail where fresh succeeded in both low-cost coupled pairs; this prediction is false.

Retained takes less time and uses fewer POSTs, input tokens and output tokens in all eight pairs. Fresh totals are 266.705 seconds, 34 POSTs and 223,972/5,738 input/output tokens; retained totals are 132.118 seconds, 24 POSTs and 196,474/2,822 tokens. The resource advantage accompanies equal solution quality. The public enumerator also produces correct artifacts in all nine specified executions without a model call: 0.047839 seconds for the initial solve and 0.027074 to 0.042129 seconds per suffix (Appendix B.9). Retention shortens the observed model workflow, while an ordinary program solves this finite computation directly.

H.5 Visible-history and procedure-card controls

A synthetic SQL report task uses four independently generated histories on two data seeds. Each history receives a data-update revision and a semantic revision that changes revenue from gross to net of refunds. Each revision has four arms: F starts a fresh native session, R restores the original session, H quotes the visible historical log as JSON in the latest message, and P supplies a fixed public procedure card there.

All 36 model phases and twelve ordinary-program phases finish normally and correctly. The prediction that retained sessions would continue reporting gross revenue is therefore refuted. The frozen resource criterion requires correctness in all eight conditions and, in at least six suffixes, no more than one extra POST and no more than 1.10 times R's input and output tokens. P meets the per-suffix resource limits in seven of eight conditions; H meets them in none.

Suffix totals are 28 POSTs and 168.724 seconds for F, 16 and 85.744 for R, 16 and 127.837 for H, and 16 and 92.723 for P. R dominates H under the recorded resource comparison in all eight pairs. The ordinary program takes 0.036 to 0.073 seconds per stage with no model calls. The procedure card

achieves the required task outcome through explicit guidance, while the native-history arm retains its full context. Per-arm totals and stage accounting are in the supplementary archive.

I Extended related work

I.1 Evaluation repair and harness reliability

EvalPlus adds valid test inputs, differentially tests saved programs and reports model-ranking changes under the base and extended suites. It uses input contracts, repairs faulty reference implementations and evaluates reduced suites on a model excluded from their construction. We build on this fixed-artifact evaluation design by comparing the induced selections on other executions. In the recovery study, the continuation-versus-retry reversal on the initial execution does not recur under the same plan, and the observed execution variation exceeds variation across the shared requirement interpretations (Appendix F.5). (Liu et al., 2023)

BenchGuard cross-verifies instructions, reference solutions, evaluators and environments, with agent programs and execution logs available as additional diagnostic evidence. Its findings are supported by human expert review and benchmark-author confirmation. Identifying unmet obligations and cross-artifact inconsistencies is already part of this approach. Our budget and CSV counterexamples use direct file evidence; the other labels retain their disclosed AI-reader provenance. (Tu et al., 2026)

Benchmarks Are Not Monolithic describes benchmark items with model-generated attributes and uses them to construct conditional cross-benchmark subsets, with a human-label comparison. That population intervention differs from changing acceptance labels for the same outputs. Both item-level auditing and conditional comparisons are established methods. Our decision audit focuses on whether a changed scoring rule improves the choice evaluated on another execution. (Siedler & Sassoon, 2026)

Harness-Bench compares complete model-harness configurations under shared task conditions; ReliabilityBench studies repeated-run consistency, task perturbations and infrastructure-fault intensity. We reuse artifact-based tasks and preserve native execution behavior, while comparing recovery choices within one shell. Our timed windows change request exposure through each policy's own timeline. Equal scheduled downtime and equal opportunity counts do not imply equal encountered faults, an important distinction from attributing the whole configuration effect to conversational state. (Yao et al., 2026; Gupta, 2026)

I.2 Interrupted execution and handoff

ContinuityBench studies recovery in a live environment. It resumes the same live environment after controlled interruptions and varies the handoff from no context to a deterministic summary or summary plus full action history, while keeping the source benchmark's native evaluator fixed. Across tau-bench, AppWorld and TerminalBench, interruption lowers success and the richest handoff is not monotonically best. These results establish interrupted-state recovery and handoff fidelity as measurable reliability axes. We study an additional measurement choice: acceptance obligations for the resulting artifacts. Within each execution, rescoring holds those artifacts fixed; across executions, the decision audit tests the recovery choice induced by each scoring rule. (Gulati, 2026)

Three further handoff studies vary what a successor receives. Handoff Debt interrupts a coding agent at deterministic points after its first edit, first validation result or first post-failure edit, freezes the repository and evaluates successor takeover under repository-only, raw-trace, summary-note and structured-note views: 75 SWE-bench Verified source tasks, 181 handoff points, four views and three successor models, scored by official resolution together with agent events and prompt tokens. Its notes are generated after the interruption by a separate note-writing model, and their generation cost is not included in the successor's event or token metrics; the evaluator is fixed and no service fault is injected. We do not vary the handoff format. (KC & Budathoki, 2026)

The Handoff Tax continues a trajectory produced by a different model under raw, compacted and trajectory-drop interfaces, sweeping switch direction and timing over 500 SWE-bench Verified tasks and about 58,000 runs; the interruption is a model switch rather than a fault, and the evaluator is again fixed. (Ganz et al., 2026)

Do Not Restart supplies a successor-side residual-completion contract that freezes the remaining obligations before the successor proposes, closes the proposal into an evidence-linked graph and checks execution against it, reporting accuracy comparable to a strong full-task agent at 22.0 to 34.6 percent of its cost across five stateful surfaces; no fault is injected and no native-recovery control is present. Our continuation policy keeps the same harness and model and resumes the conversation after native recovery ends. Its cost is charged to the interrupted round. We compare it with the other assigned recovery policies under the original and strengthened acceptance rules. (Deng et al., 2026)

I.3 Rollback, compaction and durable writes

Rollback methods recover an earlier state after a detected failure. AgentRewind records aligned checkpoints of the agent context and a controlled environment; when the agent judges that it is unlikely to progress, it selects an earlier checkpoint, restores both states and continues with a summary of the abandoned attempt as rewind memory, evaluated on MettleBench with paired continue-versus-rewind runs launched from identical copies of failed endpoints. EvoUndo verifies and repairs the recoverability of an agent's runtime self-modifications across counterfactual harness states, with exact effect-scoped snapshots as a comparator; the object recovered is a harness mutation, not task completion under a service interruption. Learning to Undo adds a reversibility signal and selective rollback to tabular value-based reinforcement learning; it has no agent task, harness or evaluator. Our policies never roll back: they retry, wait or continue within a fixed round allowance under an injected outage whose window is shared by every opportunity. (Zhuang et al., 2026; Sah et al., 2026; Sorstkins et al., 2025)

Compaction and durable-write methods preserve different forms of progress. TRACE evaluates individual compaction events with a boundary-local paired verifier that continues from the same environment state before and after compression, targeting the loss of execution position under recurrent context compression on AppWorld; the handoff is self-compaction, with no service fault and no native-recovery control. Resilient Write interposes a six-layer durable write surface over the Model Context Protocol, including a task-continuity handoff envelope, and evaluates it on a six-scenario tool-use benchmark; it provides a write surface rather than a study of completion after an interruption. Recoverability as a System Primitive makes reuse of saved progress an explicit decision, a supported continuation source plus a permitted recovery route or a withheld continuation, evaluated in four constructed studies, including four eligibility challenges and twenty paired file-restoration challenges; its evaluated loop neither constructs nor consumes a handoff, so cross-session recovery is not established. Our fault begins before any deliverable exists and our endpoint is the delivered artifact, so none of these compaction or write surfaces is exercised. (Min et al., 2026; Agyemang et al., 2026; Zhang & Liu, 2026)

I.4 Policy selection and failure feedback

Harness choice and ranking reliability provide a basis for comparing recovery policies. Stop Comparing LLM Agents Without Disclosing the Harness treats retries, validation, anomaly recovery and stopping as controller choices and reports harness-induced variance and model-rank reversals under a controlled model-by-harness grid. How Reliable Are Agent Leaderboards decomposes model, scaffold, item, benchmark and interaction variance, showing that the object being ranked and the contexts sampled determine whether an ordering is reliable. Our audit connects evaluator obligations to this selection problem: a scoring correction can change the maximizing set on one execution without improving the choice on another. The recovery comparisons remain descriptive because they use six task clusters and one run per execution cell. We therefore retain complete maximizing sets and their dependence on tasks and requirement interpretations. (Zhang et al., 2026b; Hardy et al., 2026)

Agent-Diff separates process from outcome with state-diff contracts and catalogs recovery behavior over enterprise API tasks. Its reported strategy associations are explicitly observational rather than causal. Our assigned policies and shared fault schedules support direct configuration comparisons on the studied tasks, with each execution retaining its complete policy results. Rethinking the Evaluation of Harness Evolution further shows why search and validation on the same task set do not demonstrate reusable improvement: matched-budget baselines and held-out tasks are required. We separate rescoring an existing artifact from assessing the resulting choice on another execution. The

later-execution and held-out-panel comparisons measure the value of each choice on different outputs within the stated tasks and candidate sets. (Pysklo et al., 2026; Wang et al., 2026b)

WAREX measures web-agent task outcomes under injected failures. Its authors' methodological clarification distinguishes end-to-end main experiments from recovery starting at saved error states and checking a subsequent reload action; it also describes sequential compound faults and recovery transfer. Our final deliverable endpoint follows the end-to-end question, with increased retries and longer waiting retained as competing actions. A reconnect or reload is useful explanatory evidence, not by itself a completed task. (Kara et al., 2026)

PROBE starts from a common population of initially unresolved runs and compares recovery feedback under the same repair interface and execution budget, using the same feedback model for its structured pipeline and summary baselines. Its failure anchors and guidance already specify the target, operation, verification signal and boundary condition. Feedback-generation latency is measured separately from repair execution, which separates feedback cost from repair cost. Our continuation experiment instead charges recovery within the interrupted round's remaining allowance. The public-card comparison measures advance-written guidance on the SQL task as a complete intervention. (Zhao et al., 2026)

Diagnosis-guided recovery also requires separate content and cost comparisons. PROBE compares structured feedback with outcome-only and trace-summary alternatives on common failed runs. Isolating the value of an explicit explanation would require holding failure evidence and action guidance fixed: removing a named field may leave the same diagnosis in the action text. Evaluating the online policy would additionally charge each route its actual preparation before repair. These interventions are outside our experiments. (Zhao et al., 2026)

I.5 Conversation history and resource allocation

LLMs Get Lost in Multi-Turn Conversation compares consolidated fresh inputs with continuing histories, adds Recap and Snowball interventions, and recommends consolidation before retrying. Its v1 Recap uses actual simulated user utterances; the later author response also describes corrective feedback and an additional attempt. Any correct answer attempt can end its simulation, unlike our final delivered workspace. Its translation discussion invokes both sentence-level decomposability and BLEU's limited document sensitivity. Our saved-workspace contrasts concern modified files, charged re-observation and remaining allowances; these differences require measurement, not an imported history-harm explanation. (Laban et al., 2025a;b)

Agent Workflow Memory (AWM) induces reusable subroutines from past trajectories and supplies them as prompt memory, combining goals, observations, reasoning and actions. It compares offline and online induction, concrete and abstract experiences, rule- and model-based induction, and text and code representations. Its shorter trajectories already establish process reuse as a plausible source of savings, alongside reported success improvements and failures from inappropriate workflow reliance or dynamic intermediate states. Our fixed public card is advance-written guidance, not an AWM induction method; its sufficiency on this SQL task does not identify what native continuation remembers. (Wang et al., 2024)

MCPP studies online allocation under joint time and budget constraints, and BAGEN estimates remaining resource requirements from trajectory prefixes. We evaluate fixed recovery actions rather than a learned feasibility estimator or online selector. Equal remaining wall time and a request cap do not impose equal token expenditure. Any adaptive policy must be evaluated through subsequent task execution and the information actually available when it chooses. (Wang et al., 2026a; Lin et al., 2026)

J Request timing and task-level contrasts

J.1 Opportunity exhaustion

Two R10 failures under permanent silence, tasks 054 and 080, use twelve connections, six before and six after continuation. The last successful created-event send receipts occur at 143.595 and 142.547 seconds after the fault anchor. Both therefore enter permanent silence before the 144-second endpoint, and no new POST follows it. Connection closure is observed at 153.598/152.548

seconds; the recovery adapter returns at 153.647/152.587 seconds, leaving 1644.285/446.215 seconds of the original allowance. Each attempt exits with a stream failure rather than a round timeout, and the continuation limit ends recovery. These are observed send, closure and return times, not inferred POST-initiation or exact operating-system exit times. The failure is exhaustion of allowed opportunities with time remaining.

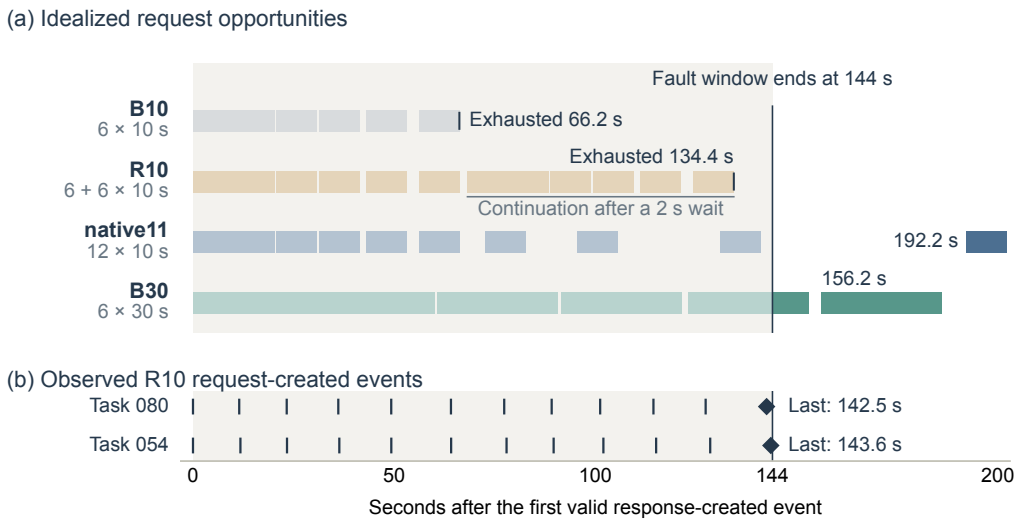


Figure 9: Idealized request opportunities and observed request-created events during a 144-second silence. (a) Bars use zero pre-event and launch delays and unit jitter. B10 exhausts at 66.2 s, R10 at 134.4 s; native11 and B30 reach new opportunities at 192.2 s and 156.2 s. B30 also retains an open connection at 144 s under recoverable silence; under permanent silence, a blocked connection stays silent. (b) Ticks mark observed R10 events; diamonds mark the last, at 142.5 s (080) and 143.6 s (054). B10/R10/native11/B30 denote few retries/continuation/more retries/wait longer.

The frozen protocol already admitted that opportunity count is insufficient. Under zero first-event delay and central retry waits, restarting two six-opportunity cycles can exhaust R10 at 134.4 seconds, while unreset native11 backoff can reach another opportunity at 192.2 seconds. The schedule behind these numbers is arithmetic on the frozen settings. With idle the stream-idle setting (10 s for B10, R10 and native11, 30 s for B30), the created event of the k-th native resend falls at $t_k = k \cdot \mathrm{idle} + 0.2\,(2^k - 1)$ seconds after the anchor for $k = 0, \ldots, n$, n the resend allowance of 5 or 11: each resend waits its idle timeout and then a backoff of $0.2 \cdot 2^{k-1}$ seconds (Codex CLI 0.154.0 doubles a 200 ms base per resend and multiplies it by a jitter factor, taken as one, with no cap and no server-specified wait), and each connection's opportunity closes at $t_k + \mathrm{idle}$, with zero pre-event latency and zero launch overhead. B10's created events fall at 0, 10.2, 20.6, 31.4, 43.0 and 56.2 s, so its last opportunity closes at 66.2 s. R10 then waits the continuation delay of 2 s and restarts the same cycle at 68.2 s, its twelfth created event at 124.4 s and its last opportunity closing at 134.4 s. Native11 continues the one cycle, 72.6, 95.4, 131.0 and 192.2 s for its sixth to ninth resends, so its tenth opportunity is the first after the window, with 304.6 and 519.4 s to follow. B30's created events fall at 0, 30.2, 60.6, 91.4, 123.0 and 156.2 s, its fourth resend's connection open until 153.0 s. This conditional calculation refutes a guaranteed-crossing argument. It is not the measured timing of the two failures, and it does not explain the other two failed point predictions.

J.2 Task-level continuation-versus-retry contrast

The two disputes behind the shared readings are narrow. In task 085, six implementations replace falsey injected clock and random objects with ambient defaults, so whether falsey interface-conforming objects are valid injections decides those labels; the clock fallback already exists in the initial constructor, so a surviving violation need not be agent-introduced. In launch operations, a dependency can denote prior approval or a condition achieved at the completion milestone.

Under truthy-only/coherent evaluation the two lexical false negatives are also accepted and cancel; three native11 violations then move the R10-minus-native11 average from $-2/12$ to $+1/12$ and B30 loses one budget pass, so the first-execution direction stands without the disputed falsey or approval readings, and both AI judgments leave the prose of the two disputed requirements unresolved. Task omission identifies the contributions supporting the contrast. In task order 051, 054, 060, 068, 080 and 085, the broad/coherent R10-minus-native11 numerators are 0, +1, 0, +1, -1 and +2 over the common denominator twelve. Removing any one task leaves that contrast positive, now over ten conditions. Under truthy/coherent, the 085 contribution is zero; omitting either 054 or 068 leaves a tie. All four shared readings remain nonnegative across the six omissions, but not strictly positive throughout. These are different five-task populations of the original execution, not confidence intervals or independent replications, and Appendix F.5 shows that the reversal itself does not survive re-execution of the same plan. The unfavorable 080 result remains in every full-sample calculation.

B10 is the one policy whose fault outcome no label set disputes. All twelve of its six-task fault cells carry the no-output label under each of the four shared readings on each of the three executions, all sixteen of its held-out fault cells carry it in every reader world of both GLM-5.3 executions and of the first Kimi K3 execution, and the sixteen of the second Kimi K3 execution and the sixteen of the third GLM-5.3 execution, which no reader labeled, end in a visible request error with a final workspace holding no file outside the inputs: 116 cells in which short native recovery left no required deliverable behind. One of them, B10 under permanent silence on task 090 in the first GLM-5.3 execution, met the provider's quota wall before its fault anchor was established (Appendix K.1), so the count on anchored fault cells is 115 of 115; Table 4, Table 5 and the B10 row of Table 30 give the corresponding scores and counts.

K Eight-task executions and artifact evaluation

K.1 Original scores and grading environment

The first execution ran serially from one frozen plan; its terminal states are 74 normal exits, 16 visible request errors, 3 round deadlines (the R10 and native11 cells of 086 under recoverable silence and its B30 cell under permanent silence) and 3 other failed turns, and every R10 continuation reused the original conversation within the original round. Task 090's four permanent-silence cells ran last: R10 failed after its continuation on an idle timeout, and the native11, B30 and B10 cells recorded a provider quota error, of which native11 had already established a legal anchor while B30 and B10 had not. The eight tasks are frontend state bug (041), API schema migration (042), metric-definition audit (052), funnel drop-off analysis (055), inventory forecast (056), SQL migration preflight and rollback (086), CLI parser bug tests (087) and time-series anomaly attribution (090). Figure 10 shows the first execution's first-round original-oracle score of every held-out cell with its native terminal state. First-round score ranges by task are 041 [0.2039, 1], 042 [0.15, 0.74], 052 [0.1059, 1], 055 [0.0874, 0.9836], 056 [0.0947, 0.69], 086 [0.1329, 0.9674], 087 [0.2867, 0.6607] and 090 [0.0508, 0.5424]. The environment correction regrades all 96 copies with the same oracle after freezing an interpreter that can reach pytest: ten 087 scores rise by 0.25, the two 087 B10 fault deliverables still fail the tests, and the other 84 scores are unchanged. Only tasks 041, 042, 086 and 087 return a grade band; the bands are fail, pass, good and excellent in ascending score order, and Table 30 counts pass or better. On the first execution, counting pass or better gives 38 of 96 passes, all on those four tasks, and the full-score endpoint of Section 2 degenerates to five cells scoring exactly 1.0 on two tasks. The second execution ran serially from the same frozen plan on 19 September 2026; its terminal states are 78 normal exits and 18 visible request errors, all 64 fault cells established a valid anchor, R10 continued its conversation 16 times, and the same environment correction changed nine 087 scores; the third execution is in Appendix K.4. The batch issued 1,204 upstream POST attempts, 974 with complete usage and 230 unknown; the known token lower bound is 13,291,664, and total tokens and monetary cost remain unknown.

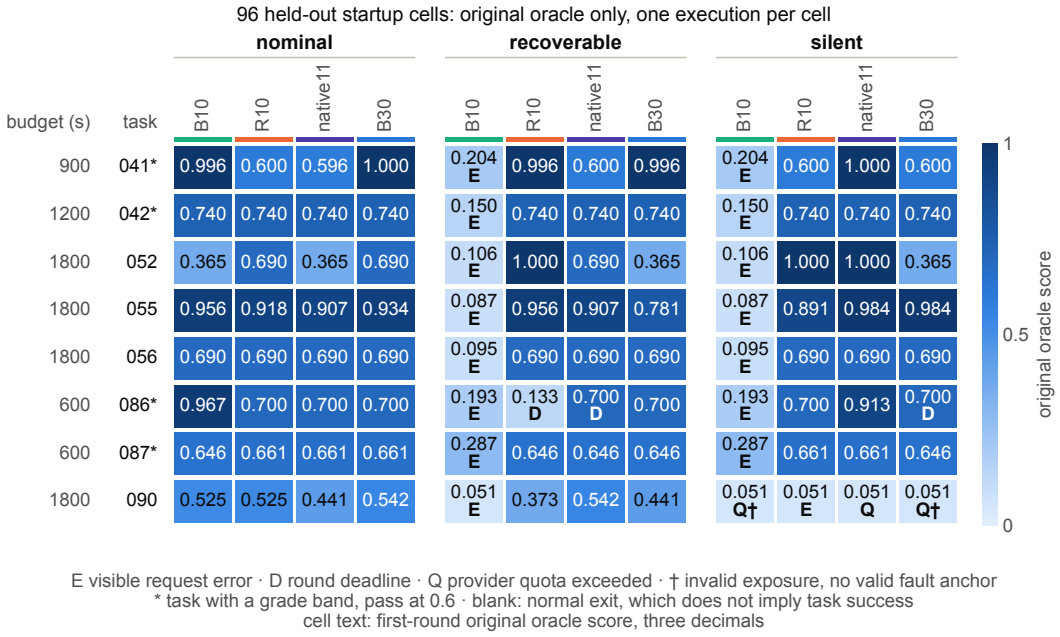


Figure 10: Original-grader scores for the first eight-task execution. The matrix contains all 96 task-policy-regime cells. Color and three-decimal text show first-round scores; E, D and Q denote a visible request error, round deadline and provider quota error. A dagger marks an invalid fault anchor. Starred tasks alone emit a pass band at 0.6: task 041's 0.596 and 0.600 therefore receive different band labels. Blank terminal annotations indicate normal exit, not successful task completion. Table 30 reports the separate complete-task readings.

K.2 Reader comparisons and task sensitivity

Table 30: Held-out counts and maximal sets vary across evaluation readings and executions. First-execution entries give nominal/recoverable/silent counts out of eight and equal-weight fault mean M; the corrected-score row gives M only. Only four tasks emit pass bands. Unknown, absent-output and invalid-exposure cells receive zero in the stated base reading. Later rows reapply frozen rules using the named reader implementations. The disclosed satisfied completion of the third execution's unknown B30 cell changes its count but neither maximal set.

Endpoint	B10	R10	native11	B30	Equal-weight maximal set
Original oracle, pass or better	4/0/0; 0	4/3/4; 7/16	3/4/4; 1/2	4/4/4; 1/2	native11, B30
Original oracle, corrected score over all eight tasks	0.147	0.710	0.753	0.665	native11
First reader, as labeled	7/0/0; 0	7/7/7; 7/8	7/7/7; 7/8	7/6/6; 3/4	R10, native11
Second reader, as labeled	3/0/0; 0	3/5/5; 5/8	5/3/5; 1/2	5/3/3; 3/8	R10
Second reader, unknown as satisfied	3/0/0; 0	4/5/6; 11/16	5/4/5; 9/16	6/3/3; 3/8	R10
Third world (31 disputed cells from the third reader, 65 shared)	7/0/0; 0	5/6/7; 13/16	7/6/6; 3/4	7/6/6; 3/4	R10

Table 30 (continued)

Endpoint	B10	R10	native11	B30	Equal-weight maximal set
Second execution, first reader's rules, Claude Opus 5	7/0/0; 0	8/8/6; 7/8	8/7/6; 13/16	6/7/8; 15/16	B30
Second execution, second reader's rules, Claude Opus 5	5/0/0; 0	7/7/4; 11/16	7/6/4; 5/8	6/7/7; 7/8	B30
Second execution, second reader's rules, Codex re-read	5/0/0; 0	6/6/4; 5/8	5/5/4; 9/16	5/5/5; 5/8	R10, B30
Third execution, first reader's rules, Claude Opus 5	7/0/0; 0	8/8/6; 7/8	7/8/5; 13/16	6/5/7; 3/4	R10
Third execution, second reader's rules, Claude Opus 5	6/0/0; 0	7/6/5; 11/16	6/8/4; 3/4	5/4/5; 9/16	native11
Third execution, second reader's rules, Codex re-read	4/0/0; 0	6/6/4; 5/8	5/6/3; 9/16	3/2/5; 7/16	R10

Table 31: All six pairwise contrasts under the held-out evaluation worlds. Each entry gives the equal-weight fault-mean difference, followed by recoverable and silent differences in parentheses, each over eight tasks. Nominal counts are reported separately in Table 30.

Policy pair	Original pass band	Reader 1	Reader 2	Reader 2, unresolved satisfied	Third world
B10 minus R10	-7/16 (-3/8, -1/2)	-7/8 (-7/8, -7/8)	-5/8 (-5/8, -5/8)	-11/16 (-5/8, -3/4)	-13/16 (-3/4, -7/8)
B10 minus native11	-1/2 (-1/2, -1/2)	-7/8 (-7/8, -7/8)	-1/2 (-3/8, -5/8)	-9/16 (-1/2, -5/8)	-3/4 (-3/4, -3/4)
B10 minus B30	-1/2 (-1/2, -1/2)	-3/4 (-3/4, -3/4)	-3/8 (-3/8, -3/8)	-3/8 (-3/8, -3/8)	-3/4 (-3/4, -3/4)
R10 minus native11	-1/16 (-1/8, 0)	0 (0, 0)	+1/8 (+1/4, 0)	+1/8 (+1/8, +1/8)	+1/16 (0, +1/8)
R10 minus B30	-1/16 (-1/8, 0)	+1/8 (+1/8, +1/8)	+1/4 (+1/4, +1/4)	+5/16 (+1/4, +3/8)	+1/16 (0, +1/8)
native11 minus B30	0 (0, 0)	+1/8 (+1/8, +1/8)	+1/8 (0, +1/4)	+3/16 (+1/8, +1/4)	0 (0, 0)

The grade band exists on tasks 041, 042, 086 and 087 only, so the pass-band row of Table 30 carries no information on the other four tasks. Table 32 restricts the three deliverable-bearing fault policies to the eight fault cells of those four tasks. In the third world B30 alone leads there, in the band's direction, while R10's eight-task lead comes from the band-free four tasks, where it credits 6/8 against native11 5/8 and B30 4/8. Nominal cells enter no fault count: in the third world R10 passes 5 of 8 against 7 for native11 and B30, so over all 24 cells the totals are 18 against 19 and 19.

Table 32: Restricting the comparison to the four banded tasks changes what the counts represent. Counts use eight fault cells per policy: two regimes on tasks 041, 042, 086 and 087. Reader 2 retains one additional unresolved R10 cell.

Endpoint	R10	native11	B30
Original oracle, pass or better	7/8	8/8	8/8
First reader, as labeled	8/8	8/8	8/8
Second reader, as labeled	4/8	4/8	2/8

Table 32 (continued)

Endpoint	R10	native11	B30
Third world	7/8	7/8	8/8

On the corrected-score row of Table 30, the recoverable-regime task means are R10 0.7231 against native11 0.7207 and the permanently silent means native11 0.786 against R10 0.698, so the two score lines cross at $w = 3529/3623$; this is a readout of that endpoint, not a finding, and the main text does not use it.

The 31 label disagreements decompose as 27 cells the first reader marks satisfied and the second violated, one satisfied against unknown, and three violated against unknown; by task they are ten on 042, ten on 086, three each on 052 and 087, two each on 055 and 090, one on 041 and none on 056. The three 052 cells are unknown for the second reader because its rules leave an ARR classification undefined. Table 33 lists, task by task, the fault cells that the second reader stops crediting and the resulting movement of the two contrasts on which the maximal sets turn; summed over tasks, they reproduce the first-to-second-reader change of every count in Table 30.

Table 33: Disagreements on particular tasks account for changes in the policy contrasts. Count changes use sixteen fault cells per policy; contrast changes use fault-mean units. The evaluated world credits a cell only when both readers mark it satisfied. N11 denotes native11. The last column counts third-reader assignments to each earlier reading, not votes defining truth.

Task	Disputed cells	ΔR10	ΔN11	ΔB30	Δ(R10 - native11)	Δ(R10 - B30)	Third reader's assignment
042	10	-2	-2	-2	0	0	first, 10
086	10	-2	-2	-2	0	0	first 8, second 2
052	3	0	0	0	0	0	first, 3
087	3	0	0	-2	0	+2/16	first, 3
055	2	0	-1	0	+1/16	0	second, 2
090	2	0	-1	0	+1/16	0	first, 2
041	1	0	0	0	0	0	second, 1
056	0	0	0	0	0	0	none

The twenty cells of 042 and 086, which the second reader rejects on obligations about nested sensitive-field leakage, lost conflict metadata and audit counts (042) and migration reruns and post-checks of defaults (086), lower R10, native11 and B30 alike and leave their differences unchanged; the R10-minus-native11 gap moves from zero to $2/16$ through the recoverable native11 cells of 055 and 090; the two 087 cells lower only B30, and releasing both would still leave R10 above B30 by exactly $1/8$ at every w, so they widen an existing lead rather than create one. The disagreements are of four kinds: confirmed defects (a sensitive field written into unknownFields on 042, large-integer sorting on 087, incident numbering on 090), source-interpretation disputes (the churn definition on 055, empty-filter semantics on 087, the ARR classification on 052), an environment difference (the SQL engine on 086) and a post-check reading (whether an 086 post-check must read the declared DEFAULT out of the schema or assert the defaulted values). The third reader's 31 decisions fall 26 to the first reader's reading and 5 to the second's: 041 one to the second; 042 all ten to the first; 052 three to the first; 055 two to the second; 086 eight to the first and two to the second; 087 three to the first; 090 two to the first. Appendix K.3 gives the provenance and each cell's reason. Relative to the first reader's world the third world therefore withdraws credit from three fault cells, native11 on 055 recoverable, R10 on 086 recoverable and native11 on 086 silent, and from two nominal cells, R10 on 041 and on 055; it agrees with the first reader on 91 cells and with the second on 70, and its label totals are 63 satisfied, 14 violated and 19 no output. The labels are not put to a vote.

Omitting one task at a time changes the equal-weight maximal sets as follows. Under the pass band, omitting 086 adds R10 to the tie and every other omission leaves native11 and B30 tied. Under the first reader, omitting 052 gives native11 alone, omitting 090 gives R10 alone, and the other six omissions keep R10 and native11 tied. Under the second reader, every omission keeps R10 alone in both worlds; in the third world, omitting 052 leaves B30 alone, omitting 055 ties R10, native11 and B30, and every other omission leaves R10 alone. The fault-weight intervals are simple in every world: native11 and B30 coincide for all w under the pass band, with R10 joining them only at $w = 0$; R10 and native11 coincide for all w under the first reader; R10 is the unique maximum for all $w > 0$ under the second reader as labeled and for all w with unknown counted satisfied; and in the third world R10 is the unique maximum for all $w < 1$ and ties native11 and B30 at $w = 1$. Counting the fewest label changes on fault cells with a deliverable, as in Appendix F.8, the equal-weight maximal set moves under one change in the pass-band row, the first reader's world and the third world, and under two in both second-reader worlds; Appendix K.4 sets these margins against the second execution, and the omission results above show task dependence, not a stable ordering. Figure 10 and Table 30 are the only readouts of the first execution's original-oracle layer in the paper; the per-cell labels of all three worlds and the 31 disputed cells are in the archived join records, and the second execution's readouts are in Appendix K.4. The 090 dependence has a mechanical cause: the first execution ran serially task by task with 090 last, and its B30 and B10 cells under permanent silence met the provider's quota wall before their fault clock started (Appendix K.1), so a provider-side event at the tail of the batch fell on one task's cells alone; a plan that interleaves tasks across policies removes that coincidence, and the second execution, which met no wall, has 64 valid anchors.

K.3 Reader procedures and disputed labels

The two readers read a neutral package of the 96 workspaces without policy, regime or score, and the second reader emits unknown where its rules are silent. They agree on 65 cells and disagree on 31, 27 of them cells the first reader accepts and the second rejects, concentrated on the two code tasks 042 and 086. The third reader's labels agree with the first reader on 26 of the 31 and with the second on 5. The five concern reproducible program defects or task definitions, while the 26 include obligation-set differences and one post-check reading (Appendix K.2 and below). Agreement with one reader is not a source ruling against the other: ten task-042 cells omit the second reader's nested-sensitive-field check. The three label sets are reported by evidence type rather than put to a vote. The disagreements move the two contrasts on which the maximal sets turn by $2/16$ each (Table 33), the re-reading movement on this batch.

The third reader is a clean session that had opened no reader package: it read only the public task text and fixtures of the seven tasks involved, wrote 75 obligations with their source lines into a rule file, derived the expected data values by its own scripts, froze rules, scripts and probes together with the source hashes before opening any candidate, then ran the 31 disputed workspaces in a network-denied sandbox (public tests and frozen probes; SQL through two frozen engines, a library build and a command-line build of SQLite at different versions) and left per-obligation evidence for each workspace before stopping on a session limit. A second clean session verified the 47 frozen hashes, kept the rule file unchanged and mapped the evidence to whole and per-obligation labels by the rules' criteria, for 29 workspaces from the frozen evidence directly and for two 086 workspaces, whose probe output had exceeded the driver's retained length, by rerunning the same frozen probe in isolation. Both sessions were told only that these 31 candidates had received different labels from the two readers, not who labeled what or in which direction; neither saw any score, mapping or reader label, and neither had touched the eight tasks before. The reader is an AI session, not human ground truth.

Its label mapper, written after the candidates were opened and outside the frozen files, used the frozen probe signals incompletely: the 086 post-check obligation was scored on a status keyword, leaving the recorded default and p4 signals unused; the 087 test obligation dropped its clause that the new tests exercise the command line; and the 087 quoted-comma obligation added a source-spelling condition the rule does not state. An independent reproduction found that under the rules' own reading policy none of these changes a label, so Table 30 and the possible- and always-maximal sets stand; under the stricter schema reading of the post-check obligation, which would change five third-world labels, R10 remains the third world's unique equal-weight leader and both sets are again unchanged.

The prospective pass is the frozen rules, probes and per-obligation evidence of the first session; the label mapping, the independent reproduction and the stricter-reading sensitivity are post-hoc checks. Its exact queries and raw probe outputs remain in the authors' private adjudication records; the released rule and label projections support the aggregate calculations, not source-probe replay. The reader models of Appendix K.3 are those recorded in each reader session's log; the Codex CLI reader ran at that tool's configured reasoning-effort setting, and no reader had a sampling parameter set for it.

Its 31 labels are 23 satisfied and 8 violated, 26 in the first reader's reading and 5 in the second's. The five: on 041 the nominal R10 deliverable's quantity update with quantity zero empties the whole cart instead of removing the line; on 055 the nominal R10 and recoverable native11 deliverables name the wrong largest drop-off transition for the beta cohort, the source defining it by the drop-off rate from the previous stage; on 086 the recoverable R10 migration fails at a table rename under one of the two frozen SQLite engines while passing under the other, and the silent native11 post-check opens with command-line-only directives that the library driver cannot execute. Eighteen of the 26 are obligation-set differences: the third reader's rules contain no obligation on nested sensitive fields written to the unknown-field list (042, ten cells), on large-integer sorting precision (087, three cells) or on incident numbering (090, two cells), and on 052 they mark the three cells violated, where the second reader had returned unknown, because the frozen reading of the severity rule places an annual-revenue metric in the high class. Enforcing the second reader's nested-field obligation on every credited 042 cell lowers R10, native11 and B30 by two fault cells each, to 11, 10 and 10, and leaves the maximal set {R10} and its margin of one unchanged, so the third reader's 042 decisions are an omission from its obligation set rather than a ruling against that obligation. On 086, the benchmark's own grader executes migration.sql and rollback.sql through Python's sqlite3 module (oracle_grade.py, lines 54 to 58 and 97 to 100) and line 27 of the prompt forbids features outside SQLite, so a post-check that opens with command-line dot directives fails under the benchmark's own engine; the third reader's execution obligations add no engine the grader lacks. The other eight, all on 086, differ on the reading of an obligation both packages carry rather than on its presence: the third reader's post-check obligation names a status-default check, its frozen probe recorded both a status signal and a DEFAULT, DFLT or TABLE_INFO signal, and its label mapper consumed only the first. All eight post-checks assert that every invoice holds the default value, and three of them also read the declared default out of the schema, so the eight labels stand on the value reading; the recorded text signal does not separate the two readings, since it matches a check merely named for defaults and misses schema-free checks identical in effect. The third world's labels are 63 satisfied, 14 violated and 19 no output; it agrees with the first reader on 91 cells and with the second on 70. These are three source readings; the paper reports them by evidence type and does not vote.

The two GLM-5.3 re-executions and the first Kimi K3 execution were read under each frozen rule set on Claude Opus 5 in Claude Code. The second reader's rule file was applied once more, unchanged, by a GPT-6-configured Codex CLI team to those two re-executions and both Kimi K3 executions. Each such world is one rule set held by one implementation, and the second Kimi K3 execution has that one reader world only. The later team's runtime did not expose its exact model identifier, effort or sampling parameters, which it records as unknown, and it is one root execution body with three native partitions of a single reading team, so it counts as one reader rather than four.

It read a mandatory group channel before the task's prohibition list, which exposed existing aggregates, the policies' execution status and some disagreement summaries, and its first receipt carried a group message with aggregate results. While locating its fourth package, the second Kimi K3 execution, its filter over the production task document was incomplete and additionally exposed an execution summary and the numeric scores of two replacement rows, and a directory search showed regime names in the paths; these exposures followed the freezing of the first three packages, so by its own disclosure that package cannot be called fully blind. It opened no other reader's labels, no outcome file, no manuscript and no private mapping, and an earlier separately authorized session that had read the mapping was withdrawn and produced no label used here.

Partitions without pytest installed executed the test bodies directly and claim no pytest run. Two prospective amendments were recorded during the sequence, the first applying from the third GLM-5.3 execution and the second from the first Kimi K3 execution, and no frozen package was re-judged under either. The rule file it was given holds 50 obligations over the eight tasks, fewer than the count named in its task text, and it used that file unchanged.

Task 041 exposes why an unchanged obligation file is not an unchanged probe. After the second GLM-5.3 execution was read, the later implementation added a test that a saved old cart object retains its redo outcome after another snapshot is traversed. For the third execution's recoverable native11 artifact, the prior six checks all pass, while the added check observes the saved redo subtotal changing from 900 to 1200. This program behavior is reproducible in a network-denied run, but the public task specifies immutable old cart values, undo/redo and branch clearing without explicitly deciding whether that old object's redo history must remain fixed after traversing another snapshot. Restoring this one disputed fault label changes the later implementation's third-execution counts from R10/native11/B30 10/9/7 to 10/10/7 and its maximal set from {R10} to {R10,native11}; it does not erase the two other shared-package set differences. A further probe amendment before the first Kimi reading also was not applied retroactively. The 3/3 set difference is an observed comparison of reading procedures, not a controlled test of reader-model identity.

The maximal policy sets under every reader world are listed in Table 34 for reference; the main text uses deterministic checks only.

Table 34: Complete maximal sets under the original score and frozen reader procedures. Original-score selection uses the original grader's numeric scores after the documented environment correction, over eight tasks. Reader identities and rule versions are specified in Appendix K.3–K.6. A dash means no reading. A possible B30 tie remains conditional on an unresolved label. Kimi execution 1's original-score selection is unresolved; Kimi execution 2 was not fully blind.

Run	Original score	A: first reading	B: first reading	B: independent re-reading
GLM-5.3 1	native11	R10, native11	R10	-
GLM-5.3 2	B30	B30	B30	R10, B30
GLM-5.3 3	B30	R10	native11	R10
Kimi K3 1	unresolved	B30	B30	native11
Kimi K3 2	B30	-	-	native11
GPT-6 Sol 1	native11	B30	native11	-
GPT-6 Sol 2	native11	native11	native11 + B30?	-
Qwen3.7 Plus 1	native11	-	-	-
Qwen3.7 Plus 2	B30	-	-	-

K.4 Repeated executions and reader implementations

Under the environment-repaired continuous score, the four-policy means of executions 2 and 3 are B10 0.147/0.147, R10 0.708/0.697, native11 0.685/0.676 and B30 0.769/0.735, so B30 leads the first-execution winner native11 by 0.085 and 0.059.

The GLM-5.3 second execution was predicted before it was read in a file whose SHA256 (d0d1f83d7dbab6c42fe1b71ea5aa09d9fee85499961e161f1a4eb74c671a6828) was posted to a group thread with a server timestamp before any replicate cell existed; the judge was a session other than the author, its inputs the two executions' outcome files, the two readers' label files and that document, and its verdicts are the six rows of Table 35. The second execution ran serially from the same frozen plan on 19 September 2026 with every fault anchor valid (Appendix K.1). Two clean sessions re-read its 96 deliverables from a neutral package under the first and second readers' frozen rules, unchanged, with their exposure disclosed. The two label sets agree on 85 cells and differ on 11, all cells the first reader accepts and the second rejects under three obligations only the second's rules carry: nested sensitive fields copied into the unknown-field list (042, three cells), a migration that resets a preserved status or drops its triggers when run twice (086, six cells) and prose that contradicts the artifact's own numbers (055, two cells); the same rule sets disagree on 31 cells of the first execution (Appendix K.3), and the 11 went to no third reader. Under the first reader's rules the second execution counts R10, native11 and B30 at 14, 13 and 15 of 16 fault cells, under the second reader's at 11, 10 and 14,

so B30 is alone at the top of both Claude Opus 5 worlds with margins of 1 and 3; the same rule sets on the first execution gave 14, 14 and 12 ({R10, native11}) and 10, 8 and 6 ({R10}). Re-execution re-decides 8 fault cells under the first reader's rules and 17 under the second's; the two rule sets disagree on 17 fault cells of the first execution and 7 of the second. B30's second-execution lead comes from its own deliverables: it gains three fault cells under the first reader's rules and eight under the second's, while R10 loses two permanent-silence cells to visible request errors that left no deliverable and regains the two task-090 cells that provider quota exhaustion had emptied in the first execution. On the original evaluator's endpoints, 28 of the 64 fault cells and 19 of the 32 nominal cells re-score and 17 deliverables are byte-identical; the band endpoint, defined on tasks 041, 042, 086 and 087, keeps its totals of 7, 8 and 8 of 16 in both executions, two R10 fault cells re-graded in opposite directions, and the full-score endpoint changes 13 cells and its maximal set from {R10, native11} to {R10, B30}.

Table 35: Predictions made before reading the second held-out execution include failures as well as successes. Predictions and zones follow the sealed record identified in the text. Probabilities are the author's bets, not empirical confidence estimates. P6 is conditional on re-reading under both frozen rule sets; side bets remain separately identified.

Registered prediction	Author probability	Observed second execution	Registered outcome zone
P2. Long waiting is not the corrected-score leader	75%	leader, 0.769 over 0.708 and 0.685	rejection
P1. Short waiting: no full-score fault cell, fault mean at most 0.30	90%	0 of 16; 0.147	support
P3. Continuation minus retries moves by at least 0.043	65% (sign reversal 40%)	moves 0.066; sign reversed	support
P4. 55 to 85 percent of corrected scores byte-identical; more on band tasks	70% (65%)	49 of 96; 28 of 48 against 21 of 48	middle (side bet holds)
P5. Largest group shift under permanent silence; specifically retries	65% (45%)	long waiting under silence, +0.183; retries third	support (side bet fails)
P6. Continuation in both, one or neither reader world; pairwise move of at least 2 of 16 in one world	40 / 40 / 20%; 70%	neither; both worlds move by 1	neither (20% zone); rejection

A third GLM-5.3 execution ran from the same frozen plan on 19 and 20 September 2026 and was assembled by row segment: rows 1 to 19 from a batch that met the provider's balance wall at cell 20, rows 20 to 41 from a continuation started after the account was recharged, during which a proxy on the executing machine went down in the middle of cell 42 and refused its remaining requests and every request of cell 43, and rows 42 to 96 from a second continuation. The three truncated records are kept unchanged and unused under the rule written for the Kimi K3 continuation, fixed for cells 42 and 43 before either was scored; cell 20 had been scored with its batch, and its truncated and complete records both score 0.74. All 96 cells are scored and all 64 fault cells have a valid anchor; the terminal states are 73 normal exits, 17 visible request errors and 6 round deadlines. Against the first execution 53 of 96 corrected scores are byte-identical, 27 of the 64 fault cells and 16 of the 32 nominal cells re-score and 16 deliverables are byte-identical; against the second, 57, 23, 16 and 17; 42 scores and 16 deliverables are identical in all three. Taking the proxy-cut record of cell 42, R10 under recoverable faults on task 087 scoring 0.287 in place of the complete re-run's 0.882, would move the corrected-score difference of native11 over R10 from -0.021 to +0.016 and R10's band count from 6 to 5 of 16; taking cell 43's, native11 on the same task at 0.287 in place of 0.896, would move them to -0.059 and native11 to 4 of 16; under every alternative B30 leads alone on both endpoints. The band endpoint loses six cells between the second and third executions and gains none: four 041 cells, three fault cells and the B30 nominal cell, end at 0.596 or 0.578, just under the 0.6 pass line, and two 087 cells end at the round deadline. The largest group shifts from the first

execution are B30 under permanent silence, +0.178, and native11 and R10 under permanent silence, -0.165 and -0.122; from the second they are native11 under permanent silence, -0.069, and B30 under recoverable faults, -0.064, while B30 under permanent silence moves by 0.004. The third execution was not pre-registered; on the same six items it gives short waiting 0 of 16 at 0.147, long waiting ahead by 0.038, a pairwise move of 0.064 from the first execution with the sign reversed, 53 of 96 byte-identical scores with 30 of 48 on the band tasks against 23 of 48, and long waiting under permanent silence as the largest group shift, +0.178.

Two clean sessions re-read the third execution's 96 deliverables under the same two frozen rule sets (label files SHA256 8475830c... and dfd7e3ba...). Under the first reader's rules the third execution counts R10, native11 and B30 at 14, 13 and 12 of 16 fault cells and under the second reader's at 11, 12 and 9, R10 alone at the top in the first world and native11 alone in the second (Table 30); the readers agree on 84 cells and differ on 12, 11 of them cells the first reader accepts and the second rejects: nested sensitive fields copied into the unknown-field list (042, three cells), an error entry without its index (042, one), a second migration run resetting a preserved status (086, four), a rollback that runs under one SQLite build only (086, one), a redo that restores the wrong state after two undos (041, one) and prose contradicting the artifact's own table (055, one); the twelfth is a 052 cell the first reader rejects and the second leaves unknown. Re-execution changes verdicts on 7 to 9 fault cells under the first reader's rules and 17 to 20 under the second's; the two rule sets disagree on 8 fault cells of the third execution. B30 loses three fault cells under the first reader's rules and five under the second's between the second and third executions, all to defects in its own deliverables, an unchanged cli.py on 087, a lost-user list contradicting its own funnel table on 055 and a double-counted revenue total on 090 among them, and two of those cells score higher under the original oracle than their second-execution counterparts. On the two conditional questions of the pre-registration, stated for the third execution without a zone, R10 is in the first reader's maximal set and outside the second's, and its R10-minus-native11 count moves by 1 and 3 of 16 from the first execution's worlds.

The second reader's rule file was then applied once more, unchanged, by the later GPT-6-configured Codex CLI team to the second and third GLM-5.3 executions and both Kimi K3 executions (Appendix K.3; label files SHA256 9033d205..., a217e207..., 4520d378... and 67d0200c..., rule file 7526b924...). That world counts R10, native11 and B30 at 10, 9 and 10 of 16 fault cells on the second execution, maximal set {R10, B30}, and 10, 9 and 7 on the third, maximal set {R10} (Table 30). Against the Claude Opus 5 world holding the same rules it agrees on 86 and 84 of those 96 cells and on 79 of the first Kimi K3 execution's 96. The two implementations mark the same cells as carrying no deliverable, 18, 18 and 23 of 96; every one of the 10, 12 and 17 disagreements is a cell Claude Opus 5 labels satisfied and the later Codex CLI team does not, so on each package the Claude violated set is contained in the later team's. Task 042 supplies 7, 6 and 9 of them, the rest falling on 041, 055, 086, 087 and 090. The shift falls unevenly across the policies: R10, native11 and B30 lose 1, 1 and 4 fault cells on the second execution, 1, 3 and 2 on the third and 3, 2 and 7 on the first Kimi K3 execution, which is why the maximal set changes in each of the three pairs. They concentrate on nested sensitive fields copied into the unknown-field list, an error entry without its index and an audit that counts preserved unknown fields as zero, with a post-check that never tests the column default and a redo that restores the wrong state after two undos. Across the pairs on which each axis is defined, the R10-minus-native11 count moves by at most 2 of 16 under re-execution, at most 2 under a rule swap and at most 2 under a reader-implementation swap, each reversing its sign once; the equal-weight leader changes in three of the four execution pairs, one of the three rule pairs and three of three reader-implementation pairs; and re-execution changes verdicts on 7 to 17 fault cells against 1 to 7 for a rule swap and 6 to 12 for a reader-implementation swap. Read under the second reader's rules by the later Codex CLI team, the first Kimi K3 execution counts B10, R10, native11 and B30 at 0, 9, 11 and 7 of 16, maximal set {native11} with B30 last, and the second at 0, 10, 11 and 10, maximal set {native11}, 10 fault cells re-decided between them. That is the only execution pair in this grid whose leader does not change; it has one reader, and that reader's disclosure withdraws any claim that the second package was read blind (Appendix K.3). Omitting task 041 leaves different maximal sets on GLM execution 2 (Claude {B30}, later Codex {R10,B30}) and Kimi execution 1 (Claude {B30}, later Codex {R10,native11}), but both implementations then select {native11} on GLM execution 3.

The eight held-out tasks were also run twice with Kimi K3 in place of GLM-5.3 under the same frozen 96-cell plan, oracle and environment correction. The first execution's 96 completed cells were assembled from three runs of that plan, rows 1 to 37 from the frozen batch and rows 38 to 76 and 77 to 96 from two continuations after provider quota walls, with the wall-stopped cells kept only as

provider evidence. One cell, B30 under recoverable faults on task 086, produced no migration file, so the original oracle raised an error and it carries no score; both readers label it no output. Two clean sessions re-read the 96 deliverables from a neutral package under the first and second readers' frozen rules, unchanged, with their exposure disclosed. The two label sets agree on 94 cells and differ on two, both on 086 and both B30 deliverables the first reader accepts and the second rejects for one reason its rules carry and the first's do not: the migration rebuilds the invoices table on every run and resets a preserved status to its default, the defect behind all seven of the second reader's 086 rejections, five of which also lack package files and are rejected by the first reader for that. On this batch the first reader counts R10, native11 and B30 at 12, 13 and 15 of 16 fault cells and the second at 12, 13 and 14, so B30 is alone at the top of both worlds and R10 is in neither maximal set; the original evaluator's band endpoint gives R10 7, native11 5 and B30 7 of 16, tying R10 with B30, and its corrected score over tasks gives R10 0.773 and native11 0.703 with B30 between 0.767 and 0.829 depending on the ungraded cell, so that endpoint's maximal set is undetermined. B30's reader-world lead comes from the four tasks with a grade band, where it leads 7 to 5 and 5 under the first reader and 6 to 5 and 5 under the second; on the other four it ties native11 at 8 with R10 at 7. Every maximal set on this batch moves under one or two verdict changes, and the labels are AI readings.

The second Kimi K3 execution ran serially from the same plan on 19 and 20 September 2026 and was likewise assembled by row segment: rows 1 to 70 from the batch that met a provider quota wall at cell 70, whose record, short waiting under permanent silence on task 055 with a valid anchor and no deliverable, is kept as that row; rows 71 to 96 from a continuation started after the account was recharged; and rows 79 and 80 from a replacement run, because a proxy on the executing machine was down for 47 minutes during the continuation and every upstream request of those two cells in that window was refused before reaching the provider (129 of 133 attempts on row 79, 85 of 93 on row 80). The rule for such external truncation was written before any cell of the continuation was scored: the truncated records are kept unchanged, the same rows are executed once more in a separate directory, and the complete execution replaces the truncated one in the 96-row table rather than the better of the two being chosen; the replacement changed row 79 from a round deadline scoring 0.0947 to a normal exit scoring 0.69 and row 80 from an invalid exposure, its first request refused before the fault clock started, to a valid one at the same score of 0.69. All 96 cells are scored, all 64 fault cells have a valid anchor, and the terminal states are 70 normal exits, 17 visible request errors and 9 round deadlines. Against the first execution, 59 of 96 corrected scores are byte-identical, 27 of the 64 fault cells and 10 of the 32 nominal cells re-score, 83 terminal states agree and 19 deliverables are byte-identical. Every definable original-evaluator endpoint changes leader: the full-score maximal set moves from {R10} (3, 0 and 0 of 16 for R10, native11 and B30) to {native11} (2, 4 and 3), the band endpoint from {R10, B30} (7, 5 and 7) to {native11, B30} (6, 7 and 7), and the corrected score over tasks, undetermined on the first execution, is 0.713, 0.840 and 0.852 on the second with B10 at 0.147, so B30 leads it and R10 sits in no second-execution maximal set; each defined binary maximal set moves under one verdict flip; the first corrected-score maximal set remains unresolved. The largest group shifts are B30 and native11 under recoverable silence, +0.208 and +0.179, and R10 under recoverable silence, -0.128. The second execution has one reader world: the later GPT-6-configured Codex CLI team applied the second reader's frozen obligation file to its deliverables; this fourth package was not fully blind (Appendix K.3). The two-rule-set Claude worlds above belong to the first execution only. These outcomes show why a corrected comparison on one execution cannot by itself select a policy; they also mean that the first execution's two held-out statements are read as GLM-5.3 single-execution results, not as facts about the policies.

K.5 Sensitive-key probes

The public task-042 prompt and PII policy require unmapped legacy values to be retained in `metadata.unknownFields` except for sensitive identity or payment fields; `ssn` and `credit_card` are named explicitly. The publisher grader tests these keys only at the root and awards 0.74 to a conversion that otherwise reaches its pass band. We selected this task after seeing reader disagreements. The authors resolved only the source-rule scope: an explicitly named sensitive key remains prohibited when nested in an unknown legacy field. The violation labels below come from exact-key canary checks on fixed synthetic inputs, rather than visual inspection or a vote among readers. Interpretive synonyms such as `social_security_number` are excluded.

Four fixed shapes test a nested dictionary, an object list, a nested list and an object-valued payment field, each with a separate benign campaign value to show that the preservation branch ran. In four

completed executions, task 042 contributes 48 assigned cells: 39 deliverables all score exactly 0.74 and nine no-output cells score 0.15. Among the 39, the exact-key checks expose a sensitive canary under `unknownFields` in 29: 6/10 in GLM-5.3 execution 2, 4/10 in GLM-5.3 execution 3, 9/9 in Kimi K3 execution 1 and 10/10 in Kimi K3 execution 2. All 39 retain the benign value, though four flatten its structure. A flattened `credit_card.number` can also expose a sensitive value; it is disclosed separately from the 29-cell exact-key count. The other ten pass-band cells serve as negative controls for this detector. Each fixed input and candidate was replayed in a disposable network-denied copy; the 39/39 probe outputs matched the frozen observations. The released per-cell projection reproduces these counts; source-probe replay additionally requires the omitted benchmark inputs and candidate code.

On the publisher's four-task band, whose fault denominator is eight per policy, revoke only the task-042 cells with demonstrated exact-key leakage and leave every other verdict unchanged. GLM-5.3 execution 2 moves from R10/native11/B30 counts 7/8/8 to 5/6/8, changing {native11,B30} to {B30}. GLM-5.3 execution 3 moves 6/5/7 to 5/5/6 and keeps B30 alone. Kimi K3 execution 1 moves 7/5/[7,8] to 5/4/[5,6]; a separate grader error leaves B30 unresolved. Kimi K3 execution 2 moves 6/7/7 to 4/5/5 and retains its native11/B30 tie. The comparison isolates the selection effect of the exact-key check on task 042; its 29/39 detector count covers outputs sharing the source, plan and implementations.

GPT probe transfer. We then applied the same four inputs and exact-key detector to task 042 from two complete GPT-6 Sol executions of the eight-task plan, each with 96 scored cells and 64 valid fault anchors. The model, provider interface and execution date differ from the earlier executions, so this is a new execution environment, not an isolated model effect. Both executions use the same provider organization. Their 24 assigned task-042 cells include 18 outputs in a passing grade band, nine per execution: 17 score 0.74 and one scores 0.9592 with the excellent grade. The other six retain the starter code and receive no candidate replay.

All 18 pass-band outputs were replayed in disposable network-denied copies. The detector finds zero hits in the first execution and one in the second: native11 under nominal conditions copies the nested-list canary to `metadata.unknownFields.promotion[0][0].ssn`. All 18 retain the benign canary; the other 17 are negative on the same detector. The four-task fault-band counts remain B10/R10/native11/B30 = 0/3/8/7 out of eight in both executions after the narrow correction. Thus native11 stays the unique leader, and R10-minus-native11 stays at $-5/8$. A surviving grader blind spot can therefore coexist with no change in the fault-policy decision on a new output population.

The four input probes were retained on 20 September, before the GPT executions began on 25 September. The exact-key counting implementation was written during those executions, and the transfer analysis was initiated after their original-score summaries were known. Its plan fixed the inputs, detector and all 24 cells before their candidate code was opened for replay. This chronology defines a post-hoc transfer of the existing probes. The supplement exposes all 24 cells, probe hashes and the four-task fault-band projection for recounting; it omits the candidate code and benchmark inputs, as for the earlier executions.

K.6 GPT ARTIFACT EVALUATIONS

Under the environment-repaired continuous score, the two GPT-6 Sol executions give B10 0.147/0.147, R10 0.487/0.490, native11 0.860/0.876 and B30 0.799/0.801, so native11 leads R10 by 0.373 and 0.385 and B30 by 0.061 and 0.075.

A written analysis contract dated 25 September at 21:26 Beijing time, while the GPT executions were in progress and before their completed results or aggregates were available, fixed the earlier B rules as the primary endpoint and A as rule sensitivity. The contract fixes the reading analysis during execution. B contains 50 task obligations frozen on 16 September; A contains 68 frozen on 18 September with its nine existing amendments. The two complete 96-cell executions use GPT-6 Sol, but the new readers are Codex procedures whose exact server model versions are unverified. These comparisons vary the generator, provider interface and execution date together.

A separate packager mixed all 192 workspaces under random neutral IDs, retaining every assignment, and froze the private mapping separately. Two new Codex sessions, each partitioned by task family, read the public sources and workspaces under their respective old rules without the mapping, current

per-cell original scores or the other reader's labels. Each froze 192 task labels and all obligation labels before the packager joined identities: 1,632 obligation labels for A and 1,200 for B. They report incidental exposure to historical group aggregates, so the reading is not free of earlier-result exposure. A also used an old probe script named by its rule materials. Neither procedure accessed the new task-042 source-transfer results in K.5 before label freezing. Their task partitions form one procedure each, not additional independent readers.

Table 36: Complete GPT reader worlds retain unresolved-label bounds. Counts are out of sixteen per policy. Each unresolved label takes both binary completions; no-output labels remain zero. B was designated primary before the completed batches' results were known.

Reading	Run	Fault passes: B10 / R10 / native11 / B30	Possible maximal sets	R10 - native11
B, primary	1	0 / 7 / 15 / 13--14	{native11}	-8
B, primary	2	0 / 8 / 14 / 13--14	{native11} or {native11, B30}	-6
A, sensitivity	1	0 / 7 / 15 / 16	{B30}	-8
A, sensitivity	2	0 / 8 / 16 / 15	{native11}	-8

A labels each execution 69 satisfied, three violated and 24 no output. B labels execution 1 as 62 satisfied, nine violated, one unresolved and 24 no output, and execution 2 as 64, seven, one and 24. Both unresolved labels concern the recoverable task-041 B30 output: the legacy coupon's representation is unspecified, so a particular legacy-persistence probe is interpretation-dependent. The current-version round trip passes. We retain both completions rather than promote an unspecified input representation to a mandatory requirement.

The procedures agree on 89/96 and 91/96 task labels; every disagreement is an A satisfaction that B marks violated or unresolved. They differ on the legacy coupon, task-042 sensitive-field and audit obligations, and task-086's default-schema post-check. These full readings test more obligations than the four exact-key probes in K.5; their labels are not interchangeable with that narrower detector. Across executions, A changes six task labels, three on fault cells, and B changes eight, four on fault cells. A changes the unique leader; B's second-execution leader is unique or tied depending on the unresolved label. R10 remains outside every maximal set. These counts compare the two implemented reading procedures, including their implementation differences.

The supplement releases an anonymous 192-cell label and original-score projection, the rule hashes and exposure summary. It permits recounting all four worlds, both unresolved-label completions and the original-score means. It reproduces arithmetic from the released labels; rerunning the judgments additionally requires the benchmark inputs and candidate workspaces.

K.7 Qwen execution results

Qwen3.7 Plus completed two further 96-cell executions of the same eight-task plan, each with all 64 fault anchors valid and every cell scored. The environment-corrected mean fault scores for B10/R10/native11/B30 are 0.14652/0.70341/0.82851/0.81364 on execution 1 and 0.14652/0.75600/0.80091/0.82485 on execution 2. The unique winner changes from native11 to B30; B30 exceeds the first-run winner by 0.02394 on execution 2. The uncorrected scores select the same winner on each execution. These are original-grader endpoints, not source-corrected task truth.

Endpoints remain distinct. Full-score fault counts are 0/2/2/2 and 0/1/3/2 out of sixteen, selecting a three-way tie then native11. On the four tasks with a grade band, passing counts are 0/6/7/7 then 0/8/7/8 out of eight, changing the maximal set from {native11, B30} to {R10, B30}. The continuous endpoint and the full-score endpoint therefore name different second-execution winners. Neither arbitrary endpoint choice nor a tied first-execution set establishes a unique future policy.

Both batches preserve the planned failure denominator. There are no missing assignments or round timeouts; 19 and 18 cells end in visible request errors. The request records include 8 and 10 statuses logged as 502, with local TLS or other connection failures affecting six and eight cells. One 60.8-second pause on the second execution occurs between cells; no incomplete cell was substituted on that account. Provider, model, dates and interface vary together relative to the earlier environments. This pair has no complete AI source reading. Its anonymous per-cell score projection and standalone verifier are included in the supplement; the verifier recomputes reported scores rather than rerunning the publisher grader.

K.8 Combined sensitive-key and migration checks

The third GLM-5.3 held-out execution supplies one fixed set of artifacts and original publisher scores. Only tasks 041, 042, 086 and 087 emit grade bands; their two fault regimes provide eight assignments per policy. The original fault-band counts in policy order B10/R10/native11/B30 are 0/6/5/7. Apply the existing explicit-key task-042 probe and revoke only original pass-band credits where it finds the prohibited output: R10 and B30 each lose one, giving 0/5/5/6. Apply the task-086 state-preservation probe under the interpretation below: R10 loses one more and B30 loses two, giving 0/4/5/4. No original failure is promoted, and tasks 041 and 087 retain their original labels. Thus the maximal policy set changes from {B30} to {native11} on one unchanged scoring endpoint.

Task 086 requires an idempotent, transaction-safe migration that adds a non-null status column with default open, preserves invoices and payments, and does not duplicate or corrupt data on repetition. Our executable interpretation includes preserving a valid post-migration invoice status. On disposable copies, the existing probe initializes the provided schema, executes the submitted migration twice, sets inv1.status to paid, executes the migration a third time, and reads the status. The first two executions also expose failures on an unchanged state; the intervening update distinguishes that static check from preservation of the current state. This extra update is an author-declared operationalization, not a quoted task instruction.

All twelve task-086 assignments, including missing outputs, are retained. The five state-resetting artifacts among ten original pass-band artifacts include three fault assignments: one R10 and two B30. The R10 and B30 silent artifacts receive original scores 0.9889/excellent and 0.9834/excellent, respectively, while both native11 fault artifacts preserve paid. Running the frozen products through Python SQLite and the sqlite3 command line gives matching outcomes across the twelve assignments. The combined correction was selected after inspecting the artifacts and is reported as a diagnostic of this execution. A retained pass has no recorded violation under the implemented checks and their specified inputs.

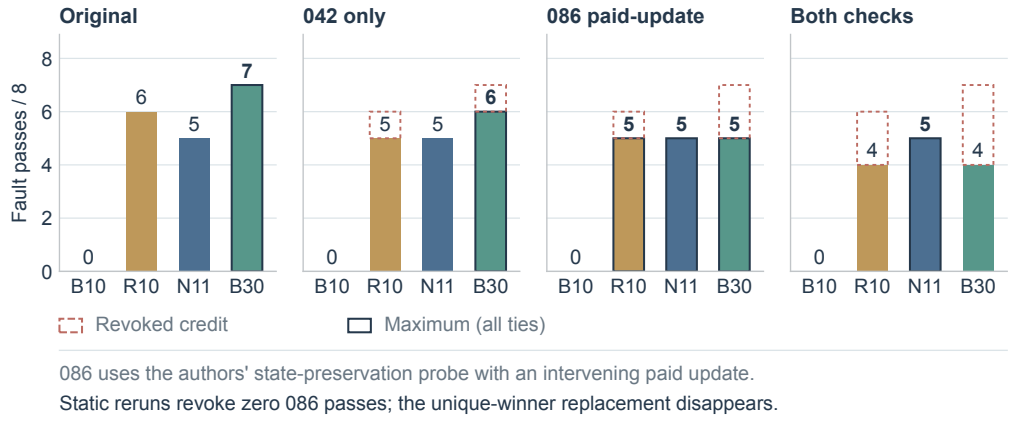


Figure 11: Fault passes under two source checks on GLM-5.3 execution 3. Each policy has eight assignments on four banded tasks. Outlines mark all maxima; dashed extensions show revoked credit. Task 042 checks nested sensitive keys. Task 086 migrates twice, commits an invoice as paid, and migrates once more, implementing the authors' state-preservation reading. Static reruns revoke zero 086 passes and do not produce this unique-winner replacement. B10, R10, N11 (native11) and B30 denote few retries, continuation, more retries and wait longer. These are retrospective checks of fixed outputs, distinct from the eight-task endpoint of sixteen fault cells.

K.9 Checker development and selection rules

Development chronology. The checkers, the application script, the selection protocol and the loss calculator were written by an author process restricted to the public task directories of the eight held-out tasks and the two readers' frozen obligation files, with every read file listed with its SHA-256 in the seal. Version 1 was sealed at 10:55:37 on 26 September 2026 (Beijing time) before that author process had opened any execution output, and version 2 at 11:01:16, byte-identical in its checkers, repaired only the adapter for the historical directory layout and the original-score wrapper. Of the three later executions of Section 2, each model's first row (the nominal probe cell 041-nominal-B10, outside the fault pool) started at 10:53:35 before version 1, and rows 2 to 96 started at 11:01:43 and later, after version 2.

An adversarial stress test on 157 hand-built correct, violating and edge-case workspaces then found that missing single deliverables were recorded as `no_output` and never revoked, that the 086 checks omitted foreign-key enforcement, transactions, repeated-run orphans and rollback contents, that the 042 checker lacked the nested sensitive-key probe, that 087's repeated `--where` conjunction was untested, and that 086 verdicts depended on the SQLite engine version.

Versions 3, 4 and 5 (sealed 11:22:28, 11:31:49 and 11:36:40) repaired these in isolation, the last also repairing a `BEGIN;` statement-detection defect introduced in version 3; every earlier seal is retained with its hashes.

The process that requested the repairs had seen the first application's results, whereas the isolated author of each version had not. The recovery-policy comparisons in Sections 5 and 6 use version 5, which was sealed while the three later executions were already generating; the version-1 and version-2 seals are the references that precede rows 2 to 96 of those executions.

Version comparisons. Applied by their own sealed scripts to the same cells, version 2 (checkers byte-identical to version 1) revokes 5 original passes over the nine historical executions (three on task 055, two on task 086) and changes no fault-pool maximal set; versions 3, 4 and 5 revoke 107, 106 and 106 and change three (version 3's extra cell is the engine-dependent 086 case).

On the three later executions version 2 revokes 0, 0 and 2 passes and version 5 revokes 1, 18 and 9; no maximal set changes under either.

On tasks 042 and 086, version 5 and the retrospective checks of Appendix K.10 revoke the same 58 task-042 cells and agree on 213 of 215 graded cells overall; the two disagreements are a GLM

2 cell whose state resets only after a third migration, which the shorter version-5 sequence misses, and a GLM 3 cell with an invalid invoice reference that the retrospective probe does not test. The four task-042 canaries were reconstructed from the public policy and implement the same exact-key criterion, which explains the identical decisions without making the inputs independent.

Coverage. Version 5 covers 25 of the 50 frozen obligations: 041 (public tests unchanged and passing), 042 (protected files unchanged, public tests, and the four nested-key shapes rebuilt from the public policy and the published description, not copied byte for byte from the earlier probe, so this is a mechanical check of those four inputs rather than a reproduction of the earlier 29-of-39 count), 052, 055, 056 and 090 (deliverable presence, CSV structure, and values recomputed from the public fixtures, with tolerance for orderings the sources leave open), 086 (seed preservation, orphaned payment handling, idempotent re-execution, transaction use and rollback, evaluated on two SQLite engines with a pass if either complete run passes), and 087 (unchanged public tests, the repeated `--where` conjunction, and a bounded presence check for new nonempty regression-test files).

The remaining 25 obligations concern prose, interface or semantic requirements and are listed with their assessment needs in the coverage file. One 056 obligation exposes a difference between the public fixture arithmetic and the original grader's ground truth for `more_than_one_pack_low_skus`; the checker follows the fixtures and revokes nothing on 056 in the twelve earlier executions.

Coverage across subsequent outputs. Holding version 5 and its 25 covered obligations fixed, later outputs add violations of known obligations that were absent among the historical revoked passes (Table 37). Across twenty executions, 232 of 1,920 assigned cells lose original passing credit. The cumulative union grows from seven to eleven to fifteen obligation IDs. The four IDs first observed in the three later executions all occur in the same task-052 cell, so they are not four independent defect causes.

Table 37: Known obligation IDs violated among revoked original passes, with the same version-5 checker throughout. Cells include nominal and fault assignments; cumulative columns add groups in the displayed order. An ID is counted once per group regardless of the number of violating cells.

Output group	Executions	Cells	Revoked passes	IDs in group	Newly observed IDs	Cumulative IDs
Historical	9	864	106	7	7	7
Later executions	3	288	28	9	4	11
New cloud environments	8	768	98	14	4	15

The nine historical executions predate version 5; the three later executions were generating when it was sealed; the eight new-cloud executions follow the seal. Group unions are invariant to order within each group, whereas intermediate plateaus in an execution-by-execution curve depend on that order. These counts track already stated obligations whose violations appear in subsequent outputs. The continued additions show that observed coverage expands beyond the historical group. The version comparison separately varies both development information and implemented coverage, from twenty obligations in versions 1–2 to twenty-five in versions 3–5.

Application and endpoint. Each execution's 96 cells are mapped to a uniform layout, including the completed continuation segments of the three interrupted historical executions, and each cell's final workspace is checked in a disposable copy without network access. A cell is a checked pass when its original grade is a pass band or a score of at least 0.60 and no covered obligation is violated; `no_output` on an obligation never revokes a pass. The fault pool counts the two fault cells of every task, sixteen per policy; the nominal cells are reported but excluded from selection.

Selection rules and loss. For a single-execution choice, take the equal-weight maximal set of the fault pool on execution i; on another execution j of the same model the loss is the best policy's count on j minus the mean count on j of the chosen set. For a two-execution choice, defined only where a

model has at least three executions, take the maximal set of the mean checked counts of executions i_1, i_2 and evaluate on every other execution. Rules, ties and evaluation were fixed before any table was read.

A supplementary common-target comparison, specified after these tables were read, scores both the original-score and the checked-score selection of execution i on the checked fault pool of execution j: the mean losses are $11/6$ versus $67/36$ for GLM (six ordered pairs; one improved, three tied, two worse), $5/4$ versus 2 for Kimi, $1/4$ versus $1/4$ for GPT and for Qwen, and $4/3$ versus $5/2$ for the GLM two-execution mean. The difference between the two original tables decomposes exactly as $L_{CC} - L_{OO} = (L_{OC} - L_{OO}) + (L_{CC} - L_{OC})$; for GLM the $7/36$ gap is $1/6$ from the change of target plus $1/36$ from the change of selection.

The historical mean-selection rule was saved at 12:51 before the new GLM scores were read, after generation had started and with partial exposure to GPT and Qwen results: original and checked means both select B30 for Kimi and Qwen and native11 for GPT, and for GLM the original mean selects B30 while the checked mean selects native11.

Scored on the three later executions' checked fault pools, the original-mean choice loses 0 and the checked-mean choice 8 of 16 on GLM (B30 15, native11 7), and both lose 0 on GPT and Qwen, whose choices coincide; with task 086's revocations omitted from both the historical means and the test scores, the two choices coincide in all three families and lose 0.

Under the static task-086 reading the historical single-execution comparison gives mean gains of +1/6 for GLM (two improved, four tied) and 0 for Kimi, GPT and Qwen, and the GLM two-execution mean -5/6.

Table 38: Losses under the frozen original-target and checked-target selection protocols. Values are fault-pass losses out of sixteen over all eligible within-model transfers. Ties receive equal weight. Two-execution selection is available only for GLM-5.3. This table changes selection and test scoring together between its first two columns; the common-target decision effect is reported separately in Table 43.

Model	Original selection and target, one execution	Checked selection and target, one execution	Checked selection and target, two-execution mean
GLM-5.3	0, 0, 2, 2, 2, 4	0, 0, 4/3, 7/3, 7/2, 4	0, 7/2, 4
Kimi K3	0, 2	1, 3	–
GPT-6 Sol	0, 1/2	0, 1/2	–
Qwen3.7 Plus	0, 1/2	0, 1/2	–

Table 39: The sealed checker revokes credit without changing every maximal set. Vectors are B10/R10/native11/B30, each out of sixteen fault cells. Revocation counts include all 96 cells, including nominal cells, and therefore have a different denominator. Every assigned cell is retained; an obligation marked no output does not revoke credit. Rows generated after the seals remain labeled post-seal; checker-development exposure is disclosed separately. Task 086 includes a committed paid update between migrations; static historical reruns revoke zero 086 passes.

Run	Original fault passes	Checked fault passes	Maximal set: original → checked	Revocations across all cells, by task
GLM 1	0/13/14/12	0/12/12/8	native11 → R10, native11	11 (042:8, 086:2, 055:1)
GLM 2	0/13/12/14	0/9/10/13	B30 → B30	12 (042:6, 086:5, 055:1)
GLM 3	0/12/10/14	0/10/10/10	B30 → R10, native11, B30	11 (042:4, 086:6, 055:1)

Table 39 (continued)

Run	Original fault passes	Checked fault passes	Maximal set: original → checked	Revocations across all cells, by task
Kimi 1	0/14/11/15	0/10/9/12	B30 → B30	16 (042:9, 086:7)
Kimi 2	0/13/15/15	0/11/12/11	native11, B30 → native11	16 (042:10, 086:6)
GPT 1	0/7/15/15	0/7/15/15	native11, B30 → native11, B30	0
GPT 2	0/8/15/14	0/8/15/14	native11 → native11	1 (042:1, nominal)
Qwen 1	0/11/13/13	0/8/9/9	native11, B30 → native11, B30	19 (042:10, 086:9)
Qwen 2	0/14/14/15	0/10/10/11	B30 → B30	20 (042:10, 086:10)
GPT 3 (post-seal)	0/7/15/14	0/7/15/14	native11 → native11	1 (042:1, nominal)
Qwen 3 (post-seal)	0/7/12/14	0/5/8/10	B30 → B30	18 (042:9, 086:9)
GLM 4 (post-seal)	0/13/10/15	0/12/7/15	B30 → B30	9 (042:5, 086:2, 055:1, 052:1; five nominal)

K.10 Retrospective checks across nine executions

Table 40: Each retrospective check has a distinct effect on the four-task maximal set. Counts are B10/R10/native11/B30 out of eight fault cells on the four banded tasks; brackets give the complete maximal set. R, N and B abbreviate R10, native11 and B30. Kimi execution 1 retains a missing-grade interval; daggers mark inconclusive checks that retain credit. The 086 sequence runs the migration twice, commits a paid update, and runs the migration once more. Static reruns revoke zero 086 passes.

Run	Original	Task 042 only	Task 086, paid update only	Both checks
GLM 1	0, 7, 8, 8 (N, B)	0, 6, 6, 6 (R, N, B)	0, 7, 8, 7 (N)	0, 6, 6, 5 (R, N)†
GLM 2	0, 7, 8, 8 (N, B)	0, 5, 6, 8 (B)	0, 5, 7, 7 (N, B)	0, 3, 5, 7 (B)
GLM 3	0, 6, 5, 7 (B)	0, 5, 5, 6 (B)	0, 5, 5, 5 (R, N, B)	0, 4, 5, 4 (N)
Kimi 1	0, 7, 5, 7–8 (B or R, B)	0, 5, 4, 5–6 (B or R, B)	0, 5, 4, 6–7 (B)	0, 3, 3, 4–5 (B)
Kimi 2	0, 6, 7, 7 (N, B)	0, 4, 5, 5 (N, B)	0, 6, 6, 5 (R, N)	0, 4, 4, 3 (R, N)
GPT 1	0, 3, 8, 7 (N)	0, 3, 8, 7 (N)	0, 3, 8, 7 (N)	0, 3, 8, 7 (N)
GPT 2	0, 3, 8, 7 (N)	0, 3, 8, 7 (N)	0, 3, 8, 7 (N)	0, 3, 8, 7 (N)
Qwen 1	0, 6, 7, 7 (N, B)	0, 4, 5, 5 (N, B)	0, 5, 5, 5 (R, N, B)	0, 3, 3, 3 (R, N, B)
Qwen 2	0, 8, 7, 8 (R, B)	0, 6, 5, 6 (R, B)	0, 6, 5, 6 (R, B)	0, 4, 3, 4 (R, B)

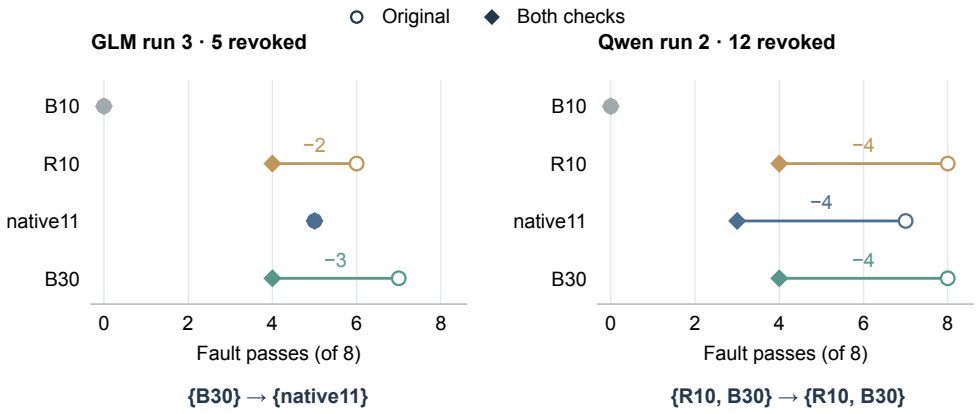


Figure 12: Revoked fault passes and the resulting maximal sets on two historical executions. Circles and diamonds show original and checked counts out of eight; segments join scores of the same outputs. Five revocations on GLM-5.3 execution 3 replace {B30} with {native11}; twelve on Qwen3.7 Plus execution 2 preserve {R10, B30}. Braces denote sets, not summed scores. The 086 check migrates twice, commits paid, then migrates once more; static reruns revoke zero 086 passes. B10, R10, native11 and B30 denote few retries, continuation, more retries and wait longer. The nine-execution historical matrix retains the other outcomes.

Of the 108 task-042 assignments across the nine executions, 87 originally pass and 58 of those are confirmed violations; of the 108 task-086 assignments, 77 originally pass and 45 reset the committed state under both engines, with one further pass left unresolved by an engine difference and one original grade missing. The combined correction measures grader-gap effects across these nine recorded executions. The per-cell projection, per-policy contributions and the recomputation script are released.

Fault passes / 8		Original				042 only				086 only paid update				Both paid update			
Agent model	Run	B10	R10	N11	B30	B10	R10	N11	B30	B10	R10	N11	B30	B10	R10	N11	B30
GLM-5.3	1	0	7	**8**	**8**	0	**6**	**6**	**6**	0	7†	**8**	7	0	**6†**	**6**	5
	2	0	7	**8**	**8**	0	5	6	**8**	0	5	**7**	**7**	0	3	5	**7**
	3	0	6	5	**7**	0	5	5	**6**	0	**5**	**5**	**5**	0	4	**5**	4
Kimi K3	1	0	**7**	5	**7-8**	0	**5**	4	**5-6**	0	5	4	**6-7†**	0	3	3	**4-5†**
	2	0	6	**7**	**7**	0	4	**5**	**5**	0	**6**	**6**	5	0	**4**	**4**	3
GPT-6 Sol	1	0	3	**8**	7	0	3	**8**	7	0	3	**8**	7	0	3	**8**	7
	2	0	3	**8**	7	0	3	**8**	7	0	3	**8**	7	0	3	**8**	7
Qwen3.7 Plus	1	0	6	**7**	**7**	0	4	**5**	**5**	0	**5**	**5**	**5**	0	**3**	**3**	**3**
	2	0	**8**	7	**8**	0	**6**	5	**6**	0	**6**	5	**6**	0	**4**	3	**4**

Figure 13: Source-check counts across nine historical executions. Each cell reports pass-band fault passes out of eight on tasks 041, 042, 086 and 087. Solid outlines mark policies maximal under every completion of missing grades; dashed outlines mark additional possible ties. Kimi execution 1 retains one missing grade. Daggers mark inconclusive checks that revoke no credit. The 086 check migrates twice, commits paid, then migrates once more; static reruns revoke zero passes. B10, R10, N11 (native11) and B30 denote few retries, continuation, more retries and wait longer. This retrospective matrix excludes subsequent executions and is separate from the sealed eight-task checker evaluation.

K.11 Migration interpretation and revocation placement

The task-086 probe of Section 3 migrates twice, commits invoice `inv1` as `paid`, migrates again and reads the status. A static reading executes schema, migration, snapshot of every user table, migration, snapshot, and passes when the second migration raises no error, changes no row and duplicates no orphaned payment. Run on disposable network-denied copies under Python SQLite 3.53.3 and the system SQLite 3.51.0, with a revocation only where both engines confirm a failure, this reading confirms no static-idempotence failure on any of the 132 task-086 artifacts (129 pass on both engines, 2 fail only on the Python engine, 1 has no migration file); the same artifacts show 89 double-engine state resets and 40 preserved states under the `paid` reading. On the four-task pass band the changed maximal sets fall from five of eight fully graded historical executions to two (GLM 1 and 2), and the unique-winner replacement of GLM 3 disappears; omitting task 086's revocations while keeping the task gives the same two. On the sealed eight-task fault pool, three of nine become one (GLM 1). Both later executions included in this static-probe comparison retain their maximal sets under every reading.

Table 41: Maximal-set changes depend on check scope and revocation placement. Task 086 stays in every denominator. Paid checking commits a payment between migrations; static reruns revoke zero 086 passes. The placement reference redistributes each execution's observed revocations uniformly among originally passing fault cells within task, using 10,000 draws and seed 20260926. Incomplete Kimi-execution-1 pass-band grades remain interval-valued and are excluded from its changed/unchanged count.

Endpoint	Observed: paid update	Observed: static 086	Observed: omit 086 revocations	Random placement: mean changes	Reference upper-tail fraction
Four-task pass band, 8 graded executions	5	2	2	4.00	0.25
Sealed eight-task fault pool, 9 executions	3	1	1	2.19	0.35

The placement reference conditions on the number and task of each execution's revocations; it says how often a perturbation of that size moves the maximal set, not that the corrections are random. The per-execution results, SQL observations and the probe script are released.

K.12 MARGIN SIZE AND WINNER RETENTION

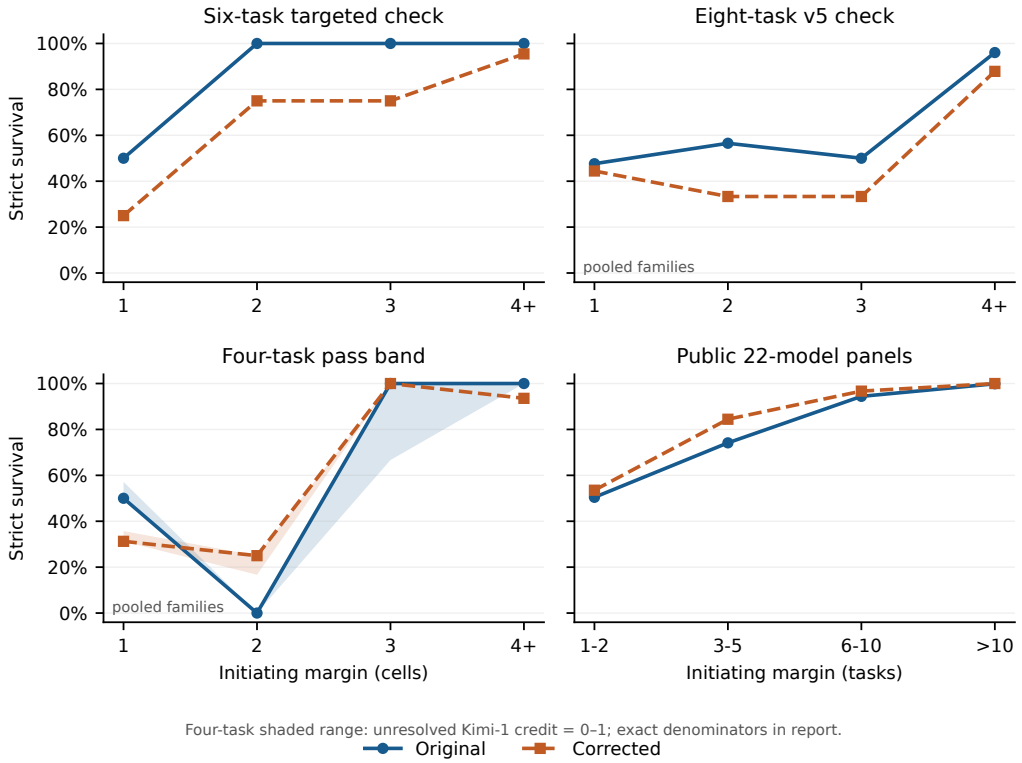


Figure 14: Strict lead retention by initial margin. Each point reports the fraction of strict leads that remain strict on another execution or generation panel, grouped by the initial margin. Later ties count as failures to retain the lead. Recovery panels use fault cells; public panels use passed tasks. Eight-task and four-task rates pool model families. Four-task shading spans the unresolved Kimi-execution-1 credit; exact numerators and denominators appear in Table 42. Intermediate bins need not be monotone, and repeated candidates and executions make these descriptive comparisons dependent.

For every ordered pair of executions within a model (or of panels on the public release) and every pair of policies (or models), we record the strict winner and its margin on the first and whether the same winner still leads strictly on the second. The largest-margin bins have high retention under both scorings, but intermediate bins are not consistently monotone (Figure 14, Table 42). For example, the checked eight-task rates are 8/18, 1/3 and 3/9 before reaching 94/107 in the largest bin.

We also retrospectively flag a corrected lead when it is no larger than the largest absolute revocation difference on the other execution, using the identity in Section 4. On the public panels, this diagnostic identifies 55 of the 95 observed losses, misses 40 and raises 27 false alarms. On the sealed eight-task fault pool, it identifies 9 losses, misses 10 and raises 2 false alarms. The diagnostic explains realized losses using test-execution revocations. The margin bins describe retention by initial lead; both analyses characterize observed transfers, while the selection rules remain fixed.

Table 42: Strict lead retention by initial margin on every eligible transfer. Each entry is retained leads / eligible leads. Recovery counts pool model families; public counts pool the twenty ordered panel pairs. A later tie is non-retention. Cells and tasks are different units as labeled, and observations reuse candidates and executions.

Dataset	Scoring	1 cell / 1–2 tasks	2 cells / 3–5 tasks	3 cells / 6–10 tasks	≥ 4 cells / >10 tasks
Eight-task fault pool, four models	original	10/21	13/23	2/4	97/101
Eight-task fault pool, four models	checked	8/18	1/3	3/9	94/107
Six-task plan, GLM-5.3	original	2/4	4/4	4/4	22/22
Six-task plan, GLM-5.3	checked	1/4	3/4	3/4	21/22
Public panels, 22 models	base tests	119/236	175/236	574/608	3456/3460
Public panels, 22 models	strengthened tests	137/256	331/392	584/604	3308/3308

K.13 Common-target selection losses

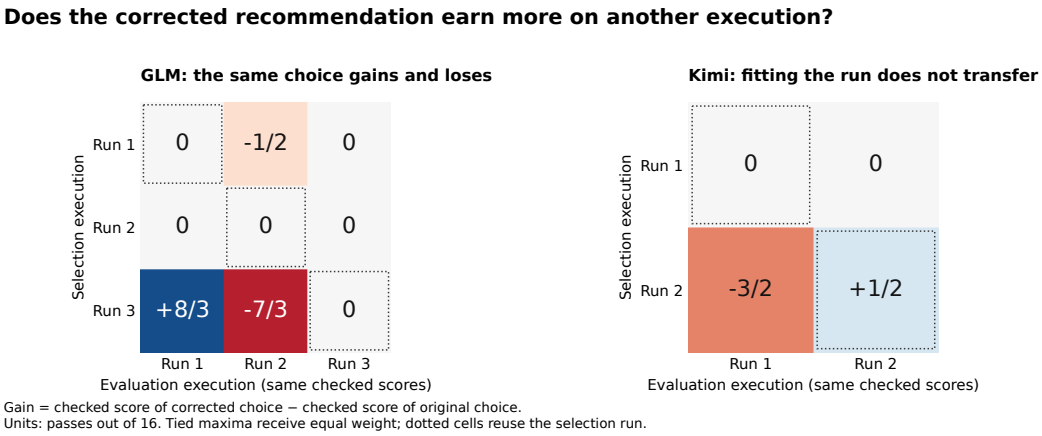


Figure 15: Common-target selection benefit across recovery executions. Rows identify the selection execution and columns the test execution. Values are corrected-choice minus original-choice passes under the full checker, out of sixteen; tied maxima receive equal weight. Dotted diagonal cells reuse the selection execution and are nonnegative by construction. The GLM execution-3 choice gains 8/3 passes on execution 1 and loses 7/3 on execution 2. The Kimi execution-2 choice loses 3/2 on execution 1. Under the narrower task-086 reading these highlighted choices coincide; GLM transfers 1 to 2 and 1 to 3 each gain 1/2, yielding a mean of +1/6.

Comparison definition. A common corrected target on another execution measures whether following the changed winner improves the outcome. Let $S_O(i)$ and $S_C(i)$ be the complete maximizing policy sets under the original and the checked fault-pool scores of training execution i; on a different execution j of the same model, both choices incur loss $\max_p C_j(p)$ minus the uniform mean of C_j over the selected set, where C_j is the checked score. The difference $L_{OC}(i,j) - L_{CC}(i,j)$ changes only the choice while holding the test execution, task weights and scoring criterion fixed; positive values favor checked-score selection. These losses use C, the score under the covered checks. The within-endpoint rules of Appendix K.9 were fixed before any table was read; this common-target contrast was specified after those tables had been seen and is reported for every ordered pair without selection.

Table 43: Historical mean benefits on the common full-checker target are negative for GLM and Kimi and zero for GPT and Qwen. Both choices are scored on the same checked execution, in fault passes out of sixteen, over all ordered within-model transfers. Ties receive uniform weight. Full checking includes the paid-update 086 scope; static historical checking revokes zero 086 passes and yields the separate sensitivity analysis.

Model	Transfers	Loss: original choice	Loss: corrected choice	Lower / equal / higher loss
GLM-5.3	6	11/6	67/36	1 / 3 / 2
Kimi K3	2	5/4	2	0 / 1 / 1
GPT-6 Sol	2	1/4	1/4	0 / 2 / 0
Qwen3.7 Plus	2	1/4	1/4	0 / 2 / 0

Checked-score selection yields negative historical mean benefits for GLM and Kimi and zero for GPT and Qwen (Table 43). GLM's small adverse mean combines opposite effects of the same changed choice. On its third execution, original scores select B30 and checking expands the choice to a three-policy tie. This reduces loss by 2.7 passes on the first test execution and increases it by 2.3 on the second. In Kimi, resolving a tie into native11 alone increases loss by 3/2 on one transfer. GPT and Qwen have identical selectors under both scorings, so their selection effect is exactly zero despite Qwen's revoked passes.

Averaging two GLM training executions before choosing gives a mean loss of 4/3 under original-score selection and 5/2 under checked-score selection, both evaluated on the checked remaining execution. Keeping the test criterion fixed matters: the apparent 7/36 increase between original-on-original and checked-on-checked loss decomposes into 1/6 from the changed test criterion and 1/36 from the changed choice. Under the static task-086 reading, checked selection instead gains 1/6 on average for GLM and ties for the other families. The adverse transfers above therefore depend on the paid-update scope (Appendix K.9).

Historical-mean choices were saved before the new GLM result was read, after partial exposure to the new GPT and Qwen results (Appendix K.9). The checked-score choice loses 8 of 16 possible fault passes on that GLM execution: B30 scores 15 and the selected native11 scores 7. Original and checked choices coincide for GPT and Qwen.

The checker revokes one nominal task-042 credit on the third GPT-6 Sol execution, leaving its fault pool and native11 leader unchanged. It revokes eighteen credits on the third Qwen3.7 Plus execution, nine each on tasks 042 and 086 and ten in the fault pool. B30 remains the sole maximum at 10 of 16; continuation minus more retries narrows from -5 to -3. On the fourth GLM-5.3 execution, nine credits are revoked, five nominal. B30 remains the sole maximum at 15 of 16, and continuation minus more retries widens from $+3$ to $+5$.

These three later executions use four concurrent row slices of the same plan. They finish in 1.1, 1.6 and 2.5 hours, compared with 4 to 10 hours previously. Concurrency changes service load within the 144-second windows, so these results combine re-execution with a changed execution environment. Appendix K.9 reports the complete frozen-choice evaluation.

Under the narrower reading of task 086, which removes only that task's revocations and keeps its original labels and denominator, the GLM single-execution mean turns to +1/6 (two ordered pairs improve by 1/2 and four are unchanged) and Kimi, GPT and Qwen stay at zero; the +8/3, -7/3 and -3/2 examples above disappear because those selections no longer change. The two-execution means keep a negative case: with executions 1 and 3 as the training pair, original scores choose B30 and the narrow corrected scores choose R10, each a sole winner, and on execution 2 their losses are 0 and 3. Enumerating all 24 fixed priority orders over the four policies, applied identically to both scorings and every execution, moves the full-checker GLM mean between -1/6 and +1/6 and the Kimi mean between -3/2 and 0, with GPT and Qwen at zero.

K.14 ADDITIONAL CLOUD EXECUTIONS

The further cloud study assigns two 96-cell executions to each of four model environments: GLM-5.1, DeepSeek-V4-Pro, Qwen3.6-Plus and MiniMax-M3 through one provider. The analysis rule was sealed at 13:44 Beijing time on 26 September 2026, before the controller read these outputs. It applies version 5 unchanged and selects over all four policies using their sixteen-cell fault counts, with uniform weight over tied maxima.

Each complete pair supplies two directed transfers: choose on a and test on b, then choose on b and test on a. Both selectors use the same checked test target. The sensitivity restores task-086 original labels while retaining the task and its denominator. An environment contributes no transfer until both executions and their checks are complete.

Table 44: Completion and results of the new executions. Fault-pool passes are B10/B30/R10/native11 out of sixteen under the original labels and under the sealed checker; the benefit is that of the corrected choice on the other execution under its checked target.

Environment	Run	Cells	Original fault pool	Checked fault pool	Revoked	Benefit of corrected choice, full / omit 086
MiniMax-M3	a	96	0/9/4/11	0/6/3/9	10	a→b: 0 / 0
MiniMax-M3	b	96	0/7/5/7	0/3/4/6	12	b→a: +1.5 / +1.0
Qwen3.6-Plus	a	96	0/9/11/9	0/6/9/7	10	a→b: 0 / 0
Qwen3.6-Plus	b	96	0/10/12/10	0/8/9/8	15	b→a: 0 / 0
GLM-5.1	a	96	0/10/9/9	0/8/8/8	10	a→b: +0.33 / +1.0
GLM-5.1	b	96	0/9/11/10	0/6/7/6	16	b→a: 0 / 0
DeepSeek-V4-Pro	a	96	0/11/10/11	0/7/8/9	13	a→b: -0.5 / -0.5
DeepSeek-V4-Pro	b	96	0/11/12/11	0/10/9/9	12	b→a: -1.0 / 0

On MiniMax-M3 the original scores choose native11 on the first execution and tie B30 with native11 on the second; the checker resolves the tie toward native11, which is the better policy on the other execution, so the corrected choice loses nothing in one direction and gains 1.5 passes in the other. The few-retries policy passes no fault cell on either execution. Both directions reuse the same execution pair.

On Qwen3.6-Plus, R10 is the unique fault-pool leader under original scores and every checked variant on both executions. The recommendation is unchanged and both transfers have zero benefit. The few-retries policy again passes no fault cell. On DeepSeek-V4-Pro the original scores tie B30 with native11 on the first execution and choose R10 on the second; the checker resolves the first tie toward native11 and moves the second choice to B30, and both corrected choices score lower on the other execution, losses of 0.5 and 1.0 passes. On GLM-5.1 the original scores choose B30 on the first execution and R10 on the second; the checker turns the first choice into a three-way tie that gains 1/3 of a pass on the other execution and leaves the second unchanged, a mean of +0.17 under the full checker and +0.50 with the task-086 revocations omitted. The four new environments thus show gains, a zero and a loss.

At the original reporting deadline of 17:15 Beijing time on 26 September 2026, four of the eight assigned executions were complete, covering MiniMax-M3 and Qwen3.6-Plus; the completion window was then extended, without changing the eight assignments or the sealed scoring and selection rules, after those results had been inspected. DeepSeek-V4-Pro settled at 17:26:16 and 17:52:28 and GLM-5.1 at 18:27:38 and 18:30:25; all eight executions and their checks were complete

before the manuscript was finalized, so every environment contributes both directions in Table 44, and all 768 assigned cells remain in the completion record.

Why the narrow DeepSeek mean remains negative. In policy order B10/B30/R10/native11, execution A's original counts are (0, 11, 10, 11), and its checked counts with only task-086 labels restored are (0, 8, 8, 9). The original choice weights B30 and native11 equally; the checked choice selects native11. Task 042 removes two passes from each and preserves their tie. The additional revoked B30 output on permanent-silence task 087 lacks the required regression test, breaking the tie toward native11.

Execution B's original counts are (0, 11, 12, 11), and its narrow checked counts are (0, 10, 10, 9). Testing A's original mixture on B gives 9.5 passes; testing its corrected choice gives nine, a benefit of -1/2. Under original grading, native11's one-pass advantage on task 042 offsets B30's one-pass advantage on recoverable task 090. Checking removes that task-042 advantage while preserving the task-090 difference: B30 passes and native11 fails. Thus task 087 changes the selection, task 042 removes erroneous target credit, and task 090 supplies the remaining performance difference. In the reverse direction, B changes R10 to a B30/R10 tie, both scoring eight on A, so the benefit is zero. The two-direction mean is -1/4. This explanation decomposes the existing frozen labels; it introduces no new execution or scoring rule.

L PUBLIC TRIALS AND GENERATION PANELS

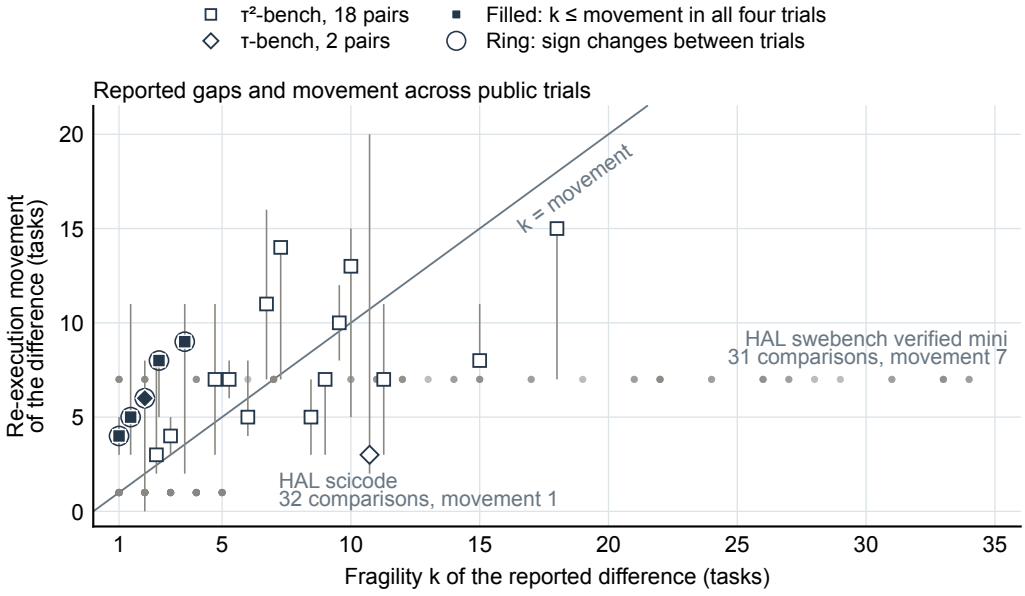


Figure 16: Reported pairwise gaps and their observed movement across public trials. Squares and diamonds show eighteen τ^2-bench and two τ-bench comparisons under trial-index pairing. Vertical lines span movement over all eligible pairings, not confidence intervals. Filled markers have k no larger than movement in all four trials; rings identify sign reversals. Gray points represent 63 one-sided HAL comparisons, each drawn at both runs' k and the repeated entry's own movement, a lower bound. Thirteen of twenty reported gaps are within their index-paired movement. Table 45 gives full counts and pairing definitions.

We use public per-task, per-trial records to measure how pairwise differences change across executions of the same configuration. Figure 16 combines the comparisons in Table 45 with the one-sided comparisons for the two matching HAL configurations in Table 46. For each pair, the numerator is the difference in passes on a trial. Its fragility k is the distance from zero, set to one for a zero numerator. Movement is the range of that numerator across index-paired trials; a trial is inside the observed movement when k is no greater than this range.

Pairing is part of the measurement. For τ^2-bench, three of the four comparisons inside movement on all index-paired trials remain inside on all 96 trial cells across 24 pairings; the fourth does so on 85 of 96. The τ-bench airline comparison is inside on 6,124 of 6,720 trial cells across 1,680 injective pairings of four of Claude's eight trials with GPT-4o's four. The data, calculation scripts and checksums are in the supplementary archive under external-replicates/.

Other releases provide less direct evidence of movement. The captured HAL reliability dashboard associated with (Rabanser et al., 2026) records three or five runs per GAIA task but releases per-task pass counts rather than individual trial assignments. These counts imply that, per model, 29 to 85 of 165 tasks change verdict across runs, while adjacent ranks differ by 1.1 to 21 tasks on average. SWE-bench Verified and Lite release one verdict per task for each of 182 submissions (Jimenez et al., 2024); that release supports comparisons across submissions, with one verdict per task and submission.

Table 45: Public trial-level gaps and their observed movement under alternative pairings. Gaps and movement are in tasks. τ-bench index-pairs the first four of Claude's eight trials with GPT-4o's four; all injective pairings enter the final range. τ^2-bench provides four trials per model and domain. Inside means k is no larger than the paired difference's range. The final range is a pairing sensitivity range, not a confidence interval. Recorded temperature/reasoning settings remain in the accompanying text.

Benchmark	Domain	Model pair	Gap	Paired pass differences	Fragility k	Index-paired movement	Trials within movement	All pairs range
τ-bench	airline	Claude 3.5 Sonnet vs GPT-4o	2	+4, -2, +2, +3	4, 2, 2, 3	6	4/4	0-8
τ-bench	retail	Claude 3.5 Sonnet vs GPT-4o	10.12	+13, +10, +12, +12	13, 10, 12, 12	3	0/4	2-20
τ^2-bench	airline	GPT-4.1 vs Claude 3.7 Sonnet	3	+2, +2, +5, +3	2, 2, 5, 3	3	3/4	2-8
τ^2-bench	airline	GPT-4.1 mini vs Claude 3.7 Sonnet	0.25	-2, +1, 0, +2	2, 1, 1, 2	4	4/4	3-5
τ^2-bench	airline	o4-mini vs Claude 3.7 Sonnet	4.5	+1, +6, +8, +3	1, 6, 8, 3	7	3/4	3-11
τ^2-bench	airline	GPT-4.1 vs GPT-4.1 mini	2.75	+4, +1, +5, +1	4, 1, 5, 1	4	3/4	3-5
τ^2-bench	airline	o4-mini vs GPT-4.1	1.5	-1, +4, +3, 0	1, 4, 3, 1	5	4/4	3-11
τ^2-bench	airline	o4-mini vs GPT-4.1 mini	4.25	+3, +5, +8, +1	3, 5, 8, 1	7	3/4	6-8
τ^2-bench	retail	Claude 3.7 Sonnet vs GPT-4.1	5.25	+7, +8, +3, +3	7, 8, 3, 3	5	2/4	4-8
τ^2-bench	retail	Claude 3.7 Sonnet vs GPT-4.1 mini	14.5	+13, +12, +20, +13	13, 12, 20, 13	8	0/4	8-11

Table 45 (continued)

Benchmark	Domain	Model pair	Gap	Paired pass differences	Fragility k	Index-paired movement	Trials within movement	All pairs range
τ^2-bench	retail	Claude 3.7 Sonnet vs o4-mini	8.25	+12, +6, +5, +10	12, 6, 5, 10	7	2/4	3-7
τ^2-bench	retail	GPT-4.1 vs GPT-4.1 mini	9.25	+6, +4, +17, +10	6, 4, 17, 10	13	3/4	5-15
τ^2-bench	retail	GPT-4.1 vs o4-mini	3	+5, -2, +2, +7	5, 2, 2, 7	9	4/4	2-11
τ^2-bench	retail	o4-mini vs GPT-4.1 mini	6.25	+1, +6, +15, +3	1, 6, 15, 3	14	3/4	7-14
τ^2-bench	telecom	Claude 3.7 Sonnet vs GPT-4.1	17.25	+8, +23, +18, +20	8, 23, 18, 20	15	1/4	7-15
τ^2-bench	telecom	Claude 3.7 Sonnet vs GPT-4.1 mini	6.25	+1, +12, +4, +8	1, 12, 4, 8	11	3/4	7-16
τ^2-bench	telecom	Claude 3.7 Sonnet vs o4-mini	8.25	+1, +11, +11, +10	1, 11, 11, 10	10	2/4	8-12
τ^2-bench	telecom	GPT-4.1 mini vs GPT-4.1	11	+7, +11, +14, +12	7, 11, 14, 12	7	1/4	3-11
τ^2-bench	telecom	o4-mini vs GPT-4.1	9	+7, +12, +7, +10	7, 12, 7, 10	5	0/4	3-7
τ^2-bench	telecom	GPT-4.1 mini vs o4-mini	2	0, -1, +7, +2	1, 1, 7, 2	8	4/4	5-8

Table 46: Repeated leaderboard names do not always identify the same configuration. The seven entries were each submitted twice under one name. Only the two rows marked same match the scaffold and agent arguments used for the same-configuration analysis; the remaining rows retain their specific model, scaffold, budget or argument differences.

Leaderboard	Entry	Passes on the two submissions	Tasks with changed verdicts	Configuration comparison
swebench verified mini	SWE-Agent / Claude Opus 4.1 (August 2025)	34, 27 of 50	11	same
scicode	Scicode Tool Calling Agent / Claude Opus 4.1 High (August 2025)	5, 4 of 65	5	same
gaia	HAL Generalist Agent / Claude Sonnet 4.5 (September 2025)	123, 123 of 165	30	scaffold hal_generalist_agent vs hal_generalist_agent_v2

Table 46 (continued)

Leaderboard	Entry	Passes on the two submissions	Tasks with changed verdicts	Configuration comparison
swebench verified mini	HAL Generalist Agent / Claude Haiku 4.5 (October 2025)	12, 12 of 50	14	model id claude-haiku-4-5-20251001 vs anthropic/claude-haiku-4-5; also budget
gaia	HAL Generalist Agent / DeepSeek V3 (March 2025)	60, 37 of 165	45	scaffold hal_generalist_agent vs hal_generalist_agent_v2; model id together_ai/deepseek-ai/DeepSeek-V3 vs openrouter/deepseek/deepseek-chat-v3-0324; also budget
corebench hard	CORE-Agent / DeepSeek R1 (January 2025)	4, 2 of 45	4	model id together_ai/deepseek-ai/DeepSeek-R1 vs openrouter/deepseek/deepseek-r1; also max_steps, openrouter_provider_only
corebench hard	HAL Generalist Agent / DeepSeek R1 (January 2025)	3, 1 of 45	4	model id together_ai/deepseek-ai/DeepSeek-R1 vs openrouter/deepseek/deepseek-r1; also budget, max_steps, openrouter_provider_only

L.1 Strengthened tests on the EvalPlus release

The EvalPlus v0.1.0 release provides 200 sampled HumanEval programs per task for each of 22 code models, generated at temperature 0.2. We use sample indices 0–4 for every task and model, giving $22 \times 164 \times 5 = 18{,}040$ programs. We verify each archive's published size and member grid, then rescore its programs in network-denied containers with the official v0.3.1 evaluator and the maintainers' `find_zero` repair (commit 26d6d00). Evaluation uses the base tests and HumanEval+ v0.1.10 strengthened tests; a strengthened pass requires both to pass.

All 110 model-panel blocks complete without an infrastructure failure. Base outcomes are 4,195 passes, 13,839 failures and 6 timeouts; strengthened outcomes are 3,695 passes, 14,037 failures and 308 timeouts. Generation indices define five aligned panels of 164 tasks, rather than five benchmark runs by the maintainers. A preregistered sensitivity analysis applies 100 seeded within-task permutations of these indices; no panel's top set changes in any repetition. The protocol, endpoints and selection-loss rules were frozen before results were read.

Under both scorings, phind-code-llama-34b-v2 leads every panel by at least 26 tasks. Leave-one-panel-out top-model selection therefore loses zero tasks in all 140 evaluation cells across the one-panel, two-panel and cross-scoring rules. No pair of panels has different top sets, and three of 22 models change Top-5 membership under each scoring. Pairwise comparisons are less stable: 20 of 231 model pairs reverse their base-margin sign across panels and 19 reverse their strengthened-margin sign.

Strengthening changes each pair's margin by a panel-dependent amount. This change is constant across panels for no pair and changes sign for 45 pairs. It flips a strict winner on 12 of 1,155 pair-panels and changes tie status on 27. Restricting attention to changed choices with a unique strengthened winner gives 28 selection events and 112 transfers to the other four panels. The corrected winner remains strictly ahead in 71 transfers (63%), including 34 of 48 strict-reversal transfers (71%). Unchanged choices retain their winner in 4,289 of 4,448 transfers (96%). Under base tests, the changed group retains its base winner in 48 of 92 eligible transfers (52%).

Every changed choice has a strengthened margin of at most four tasks: the transfer counts are 40 at one task, 48 at two, 20 at three and four at four. Post-hoc stratification of unchanged choices gives retention of 83/168 (49%) at margins of one or two tasks and 314/368 (85%) at three to five. The pooled retention difference reflects comparisons with different original margins; margin-stratified results characterize this composition.

Table 47: Strengthened tests change some pairwise choices while leaving the full-leaderboard choice unchanged. Retention includes selections with a unique strengthened winner. The 28 changed pair-panels yield 112 transfers; 1,112 unchanged pair-panels yield 4,448. Fifteen pair-panels without a unique strengthened winner are excluded from retention. These denominators differ from the all-choice common-target analysis. Top-model loss uses leave-one-panel-out selection.

Quantity	Base tests	Strengthened tests
Top model identical on all five panels	yes	yes
Pairs whose margin reverses sign across panels (of 231)	20	19
Corrected winner retained on another panel, pairs flipped by the correction	–	71 of 112 (63%)
Winner retained on another panel, pairs not flipped by the correction	4276 of 4448 (96%)	4289 of 4448 (96%)
Leave-one-panel-out selection loss for the top model	0	0

The rescored programs, per-program verdicts, pair-by-panel margins and the frozen analysis code are released.

Table 48: Complete common-target losses and gains on the public generation panels. For every fixed pair, select on panel i and score both choices on panel $j \neq i$ against the same strengthened target. L_OC and L_CC denote the two loss columns; units are passed tasks out of 164. Tied maxima receive uniform weight. The unchanged row stays in the overall mean; transfers share models, tasks and panels.

Selection category	Choice events	Transfers	Loss: base choice	Loss: plus choice	Mean gain	Gain / zero / loss
All	1,155	4,620	0.130	0.102	+0.028	88 / 4,484 / 48
Changed by the correction	39	156	1.497	0.667	+0.830	88 / 20 / 48
Strict reversals	12	48	2.375	0.604	+1.771	34 / 1 / 13
Tie resolved into a winner	16	64	1.305	0.516	+0.789	37 / 12 / 15
Winner turned into a tie	11	44	0.818	0.955	-0.136	17 / 7 / 20
Unchanged	1,116	4,464	0.083	0.083	0	0 / 4,464 / 0

Per fixed pair over its 20 transfers, 14 pairs gain, 212 are zero and 5 lose; per test panel the mean gain is +0.048, +0.017, -0.001, +0.060 and +0.017 tasks. Resampling the 231 pairs with replacement 10,000 times gives quantile ranges of 0.008 to 0.051 for the overall mean and 0.29 to 1.49 for the changed-transfer mean; these describe that reweighting procedure, not the uncertainty of the enumerated means, since pairs share models, tasks and panels. All transfers reuse the same five panels.

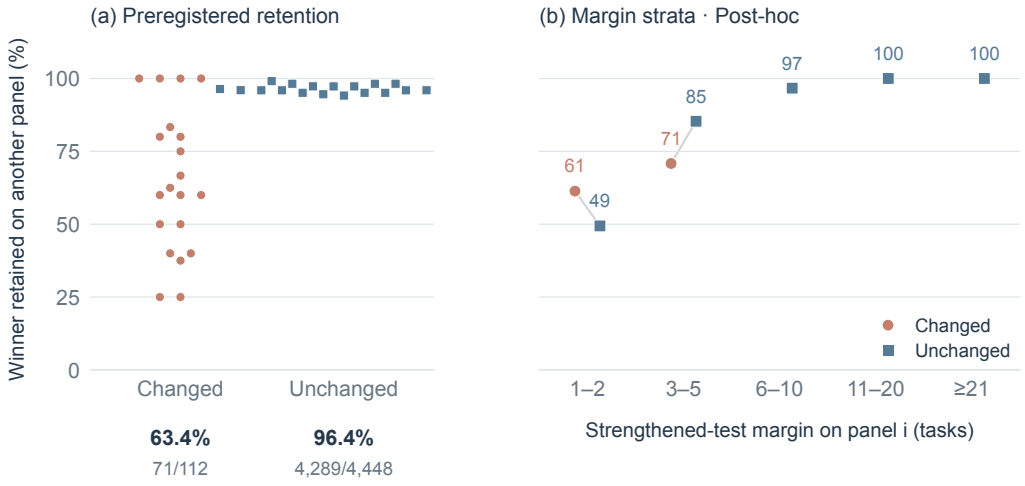


Figure 17: Winner retention and margin composition on public panels. (a) Among selections with a unique strengthened-score winner, retention is 71/112 for choices changed by the tests and 4,289/4,448 for unchanged choices; later ties count as non-retention. Small marks report the twenty ordered panel-pair rates. (b) Post-hoc strata use the strengthened-score margin on the selection panel; no changed choices occur above five tasks. The pooled contrast is confounded by margin composition. Models and panels recur across observations, so the plot reports descriptive rates without independence-based intervals.

L.2 Frozen choices on held-out panels

The plan and complete choice sets were frozen at 13:23 Beijing time on 26 September 2026, before the first new scoring block began (plan SHA-256 cca16e2e…, choices c13c1c61…). For each model pair, original-score and corrected-score choices are the equally weighted maximal sets under the mean scores of panels 0–4. The development results and research question were known at this point; the new panel results had not been read.

We then rescore sample indices 5–9 for all 22 models and 164 tasks using the same evaluator and containers as Appendix L.1. All 110 blocks complete. On each new panel, both frozen choices are evaluated against the same strengthened target. Each of the 231 pairs contributes five transfers; tied choices retain uniform weight.

Table 49: Frozen choices improve mean performance on unread generation panels, with adverse transfers retained. Choices were fixed from the mean scores of panels 0–4 before scoring panels 5–9. Each of the 231 fixed pairs contributes five transfers, all evaluated on the same strengthened target in tasks out of 164. Uniform ties are retained. The 1,155 transfers describe the enumerated comparisons, reusing tasks and models.

Selection category	Model pairs	Test transfers	Loss: base-score choice	Loss: plus choice	Mean gain	Gain / zero / loss
All pairs	231	1,155	0.098	0.052	+0.046	28 / 1,115 / 12
Changed choice	9	45	1.789	0.611	+1.178	28 / 5 / 12
Strict reversal	7	35	1.886	0.543	+1.343	22 / 4 / 9
Tie resolved	1	5	0.900	0.600	+0.300	3 / 0 / 2
Tie created	1	5	2.000	1.100	+0.900	3 / 1 / 1
Unchanged choice	222	1,110	0.030	0.030	0	0 / 1,110 / 0

Table 50: Mean benefit varies across the five held-out generation panels. Each row weights the same 231 fixed pairs equally and evaluates the frozen choices from panels 0–4. Units are passed tasks out of 164. Panel indices label samples, not chronological execution order.

Test-panel index	Loss: base-score choice	Loss: plus choice	Mean gain	Gain / zero / loss
5	0.154	0.058	+0.095	7 / 223 / 1
6	0.093	0.026	+0.067	7 / 223 / 1
7	0.078	0.089	-0.011	3 / 223 / 5
8	0.067	0.032	+0.035	5 / 224 / 2
9	0.100	0.056	+0.043	6 / 222 / 3

The frozen 22-model choice is identical under both scorings and loses no tasks on any new panel. Pairwise mean loss falls from 0.098 to 0.052 tasks. The nine pairs whose choices change contribute 45 transfers, including twelve losses; four panels have positive mean benefit and one has negative benefit (Tables 49-50).

A separate adoption rule follows a pair's corrected recommendation only when its leave-one-panel-out benefit on panels 0–4 is positive; otherwise it retains the original choice. This rule was frozen at 14:08, after new-panel scoring had started but before the analyst read the new results. Its held-out mean loss is 0.063 tasks, compared with 0.052 for always adopting and 0.098 for never adopting. It differs from always adopting on four pairs and from never adopting on five. Thus the rule loses 0.010 tasks relative to always adopting and gains 0.035 relative to never adopting. The 1,155 transfers reuse tasks, models and generation conditions and are descriptive comparisons.

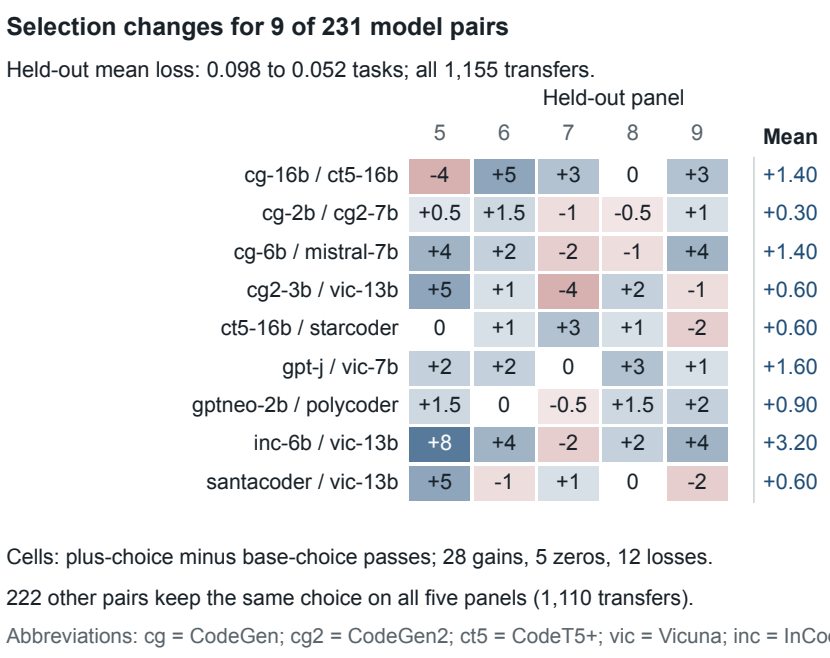


Figure 18: Held-out benefit for all nine model pairs whose base-score and plus-score selectors differ. Selectors are fixed from panels 0–4. Each cell gives the difference in passed tasks on the same strengthened target for one held-out panel; the final column averages that row. Positive values favor plus-score selection. The remaining 222 pairs keep the same choice, accounting for 1,110 of 1,155 transfers. These comparisons share models, tasks and panels.

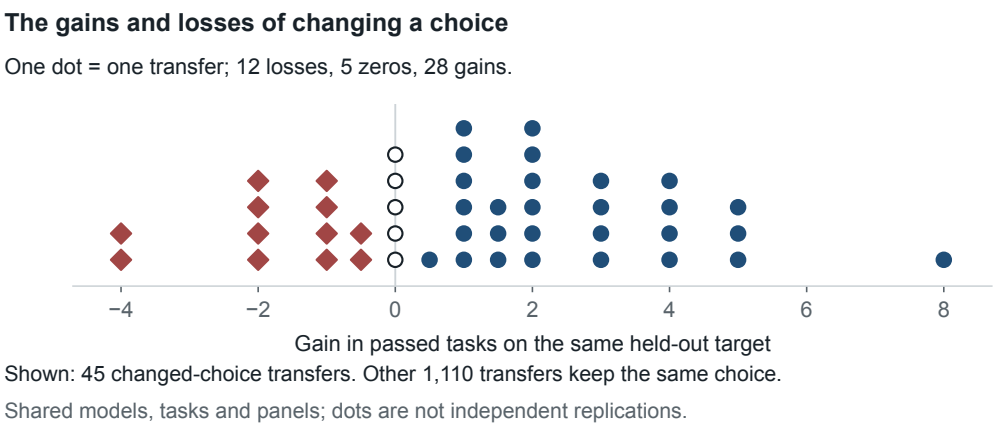


Figure 19: Exact distribution of held-out benefit among the 45 transfers with a changed choice. Each dot is one transfer; dots sharing a value are stacked vertically. Benefit is the plus-score choice's passed-task count minus the base-score choice's count on the same strengthened target. The other 1,110 transfers keep the same choice. Dots share models, tasks and panels across the enumerated transfers.

L.3 Banking benchmark scoring correction

The τ^2-bench 1.0.1 changelog states that `banking_knowledge` scores are not comparable across versions and that earlier trajectories can be rescored with the corrected tasks. Score-changing repairs include one task's amount, ten tasks' gold trajectories, and the treatment of additional read operations and numeric parameters across the domain. The audit records all 97 task identifiers in the release tag.

We use the official Gemini 3 Flash and Claude Sonnet 4.5 trajectories: four trials of 97 tasks, or 388 simulations per model, with no missing simulation. Each simulation is rescored offline using the 1.0.1 source and its recorded tool configuration. The recovered totals exactly match the published 106/388 (27.32%) and 98/388 (25.26%). Original scores are the rewards stored in the trajectories; corrected scores are the 1.0.1 rewards. Sixteen task identifiers change verdict, all promotions.

The audit gives tasks equal weight and selects each trial's complete maximizing model set, with uniform ties. Both original-score and corrected-score choices from trial i are evaluated on trial $j \neq i$ under the corrected target. The event was found after the official aggregate scores had been seen. The rule, scripts, input files and task set were frozen at 15:39 Beijing time on 26 September 2026 before any successful per-task rescoring. The first frozen implementation failed while reading the older Gemini file format, before computing a score. The second added only a structural migration for missing tool-message identifiers; both freezes are retained.

Table 51: A documented scoring correction changes choices with both positive and negative transfer gains. All twelve ordered transfers use the four published τ^2-bench banking trials and the 1.0.1 corrected target, with 97 tasks per trial. Tied choices receive uniform weight. The correction event was found after aggregate scores were seen; the audit rule and inputs were frozen before rescoring, with the disclosed structural-format repair retained. These reused trials are dependent.

Selection trial → test trial	Corrected passes: Gemini / Claude	Original choice	Corrected choice	Loss: original choice	Loss: corrected choice	Gain
0 → 1	25 / 25	Claude	Claude	0	0	0
0 → 2	28 / 24	Claude	Claude	4	4	0
0 → 3	27 / 22	Claude	Claude	5	5	0
1 → 0	26 / 27	Claude	tie	0	0.5	-0.5

Table 51 (continued)

Selection trial → test trial	Corrected passes: Gemini / Claude	Original choice	Corrected choice	Loss: original choice	Loss: corrected choice	Gain
1 → 2	28 / 24	Claude	tie	4	2	+2
1 → 3	27 / 22	Claude	tie	5	2.5	+2.5
2 → 0	26 / 27	Gemini	Gemini	1	1	0
2 → 1	25 / 25	Gemini	Gemini	0	0	0
2 → 3	27 / 22	Gemini	Gemini	0	0	0
3 → 0	26 / 27	Claude	Gemini	0	1	-1
3 → 1	25 / 25	Claude	Gemini	0	0	0
3 → 2	28 / 24	Claude	Gemini	4	0	+4

Over all twelve pairs the mean loss falls from 1.92 to 1.33 tasks (benefit +0.58; three gains, seven zeros, two losses); over the six pairs whose choice changes, from 2.17 to 1.00 (+1.17; three, one, two); the six unchanged pairs have zero benefit. The four trials come from one published archive and the twelve pairs reuse them, so these are descriptive comparisons.

M EXECUTION DATES, USAGE AND COST

Table 52 reports execution dates, summed cell time, recorded upstream attempts and known token usage for the listed executions. Input includes cached tokens; total known tokens are input plus output. Attempts that end without a usage frame have unknown consumption and are neither estimated nor priced. A dash denotes an unrecorded quantity rather than zero.

For the first eight rows, retained records give exact input totals of 4,961,189, 4,475,544, 4,820,055, 12,497,638, 13,147,839, 13,275,406, 11,778,924 and 11,878,303 tokens. Corresponding output totals are 278,707, 258,053, 265,794, 794,026, 799,510, 905,070, 1,026,249 and 1,014,903. The table displays these in thousands.

The price column applies the providers' recorded September 2026 list prices to known usage frames. GLM-5.3 prices are 8, 2 and 28 CNY per million uncached-input, cached-input and output tokens (open.bigmodel.cn); Kimi K3 prices are 20, 2 and 100 CNY (platform.kimi.com). GLM frames report no cache-write tokens. Kimi reports 593,134 on its first execution and 1,192,677 on its second; these are billed separately at 20 or 40 CNY per million depending on a cache lifetime absent from the frames. They are excluded from the price column. The amounts are therefore lower bounds on spending.

Summed cell time differs from elapsed execution time. The six-task plan runs three cells concurrently, so each execution spans about one hour; the original held-out plan runs one cell at a time. The third GLM held-out execution combines three batches after a balance interruption at cell 20 and a proxy outage at cells 42 and 43. Each Kimi execution also combines three batches: the first follows two quota interruptions; the second follows one quota interruption and a 47-minute local proxy outage. During that outage, requests were refused before reaching the provider and two cells were rerun (Appendix K.4).

Both providers report a model identifier without the serving-stack version. Re-execution movement includes any serving-stack changes across these dates (Section 9).

Table 52: Execution dates and recorded usage. Cell time sums wall-clock hours; token counts are in thousands and input includes cached input. Known usage is priced at recorded list prices: USD for GPT-6 Sol and CNY otherwise. Qwen3.7 Plus counts are provider-reported totals; its prices were not retrieved. Dashes denote unavailable quantities.

Execution	Dates in 2026	Cell time (h)	Up-stream attempts	Usage calls	Input tokens (k)	Cached input (k)	Output tokens (k)	Cost
Six-task plan, GLM-5.3, execution 1	15 Sep	3.0	805	619	4,961	4,536	279	CNY 20.3
Six-task plan, GLM-5.3, execution 2	19 Sep	3.0	764	580	4,476	4,086	258	CNY 18.5
Six-task plan, GLM-5.3, execution 3	19 Sep	3.0	785	594	4,820	4,424	266	CNY 19.5
Held-out plan, GLM-5.3, execution 1	16 Sep	6.7	1,204	974	12,498	11,326	794	CNY 54.3
Held-out plan, GLM-5.3, execution 2	19 Sep	6.0	1,263	1,011	13,148	11,931	800	CNY 56.0
Held-out plan, GLM-5.3, execution 3	19-20 Sep	6.7	1,212	967	13,275	12,077	905	CNY 59.1
Held-out plan, Kimi K3, execution 1	18-19 Sep	10.4	1,134	888	11,779	10,357	1,026	CNY 151.8
Held-out plan, Kimi K3, execution 2	19-20 Sep	10.6	1,144	887	11,878	10,577	1,015	CNY 148.7
Held-out plan, GPT-6 Sol, execution 1	25-26 Sep	4.1	–	–	5,044	4,276	319	USD 5.96
Held-out plan, GPT-6 Sol, execution 2	25-26 Sep	4.0	–	–	4,830	4,071	313	USD 5.84
Held-out plan, Qwen3.7 Plus, execution 1	25-26 Sep	5.1	–	–	8,680	7,040	730	–
Held-out plan, Qwen3.7 Plus, execution 2	25-26 Sep	5.4	–	–	10,340	8,490	780	–
Held-out plan, GPT-6 Sol, execution 3 (four slices)	26 Sep	1.1	913	–	5,071	4,290	333	USD 6.14
Held-out plan, Qwen3.7 Plus, execution 3 (four slices)	26 Sep	1.6	–	–	–	–	–	–
Held-out plan, GLM-5.3, execution 4 (four slices)	26 Sep	2.5	–	–	–	–	–	–

M.1 Natural service errors

The request-log snapshot at 19:12:26 on 26 September 2026 (Beijing time) includes the eight completed new-cloud executions analyzed in Section 5.1. Their 768 cells contain 10,222 provider-

side requests and 53 natural error responses. A natural error is a request without a planned fault marker whose status is 429, 500, 502 or 503, or whose status is absent because the connection fails; no connection-level cases occur here. Client cancellations (HTTP 499) are separate: 88 occur at the deadline and 1,914 before it. We count the provider-side log once, without adding the outer proxy's copy. An unmarked request can belong to a cell with a planned 144-second injection; request classification does not change the cell's experimental condition.

For a more restricted description, Table 53 uses only nominal cells from executions whose recorded operational checks all passed: GLM-5.1 A, Qwen3.6-Plus B and both MiniMax-M3 executions. This gives 128 cells. The other four executions completed all assigned cells but had process-exit or port-closure checks marked false. That operational restriction defines this supplementary exposure subset; all eight executions remain in the scoring analysis.

Table 53: Natural-error exposure in 128 nominal cells from four executions with passing operational checks. Entries are exposed cells / exposed cells with solver completion; completion is distinct from a passing task grade. A zero exposure supplies no estimate of recovery performance. Each GLM and Qwen policy has eight assigned cells; each MiniMax policy has sixteen.

Environment	B10	B30	R10	native11
GLM-5.1	0 / 0	0 / 0	0 / 0	1 / 1
Qwen3.6-Plus	1 / 1	1 / 1	0 / 0	0 / 0
MiniMax-M3	3 / 3	1 / 1	1 / 1	1 / 0

GLM and Qwen have no exposure for some policies. MiniMax has exposure under all four, but only one to three cells per policy, on unmatched tasks. The logs quantify natural-error exposure and completion in these cells. A comparison with the controlled-interruption ranking requires matched natural-error exposures across policies. The released projection retains family, execution, condition and policy denominators and the input hashes for this fixed snapshot.