WHERE TO REPAIR? WHEN THE ROOT CAUSE IS NOT THE BEST TARGET IN MULTI-AGENT WORKFLOWS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recovering failed multi-agent workflows is essential for reliable Artificial Intelligence (AI) automation: an error passed between agents can prevent an entire task from succeeding. Existing work diagnoses causes and tests repairs, yet diagnosis alone does not determine which permitted change best restores task success. One challenge is to identify feasible recovery choices beyond the error source under permission and budget constraints. Another is to compare interacting repairs under matched conditions, separating the effect of repair locations from search quality. We propose a Cross-Surface Constrained Recovery Model that represents source repairs, communication changes, and downstream checks in one constrained action space. We implement the model with Feasibility-Gated Joint Replay Search through three modules: candidate representation specifies actions, gated joint replay evaluates compatible combinations, and selection and loss audit selects repairs and measures value excluded by location restrictions. On our controlled recovery suite (study identifier: CRPA-R1), full candidate access improves mean recovery-rate Lower Confidence Bound (LCB) by 0.135229 over exhaustive root-only repair across 96 repairable-root traces; on 27 AgentRx retail calibration cases with three seeds, changing only the recovery rule improves executable recovery by 19.75 percentage points over root-following continuation. These findings support designing recovery systems that use diagnosis to guide a broader search for feasible changes that restore task success.

1 INTRODUCTION

Artificial Intelligence (AI) agents can work together to plan a task, use software tools, and check the results (Wu et al., 2024; Hong et al., 2024). When an error passes from one agent to another, the whole task can fail. We study how to recover that failed task: given its execution record, diagnosed cause, available edits, and permission and budget limits, choose an allowed change or combination of changes that can restore success. Figure 1 illustrates two choices: correct the original error or contain its consequences later in the workflow. If no change is allowed, the system abstains.

In an example from our controlled recovery suite (study identifier: CRPA-R1), a planner issues a faulty instruction that a worker propagates to final acceptance. Diagnosis correctly identifies the planner. Recovery can repair its output or check the propagated state before acceptance. Both choices are permitted and equally costly, yet the downstream check recovers three more of twelve replays. The planner remains responsible; recovery asks which change produces success.

Existing research describes multi-agent failure patterns (Cemri et al., 2025), attributes failures to agents or steps (Zhang et al., 2025; West et al., 2025), and tests interventions for debugging (Ma et al., 2026). Algorithmic recourse studies feasible changes to an outcome, while decision-focused learning assesses predictions through resulting decisions (Karimi et al., 2021; Wilder et al., 2019). These approaches supply diagnostic and intervention tools. We ask how much recovery value is lost when diagnosis also restricts where those tools may act.

Two challenges arise. First, identifying the cause does not define feasible recovery choices. Responsibility concerns the failed execution; permissions, available actions, and budgets concern recovery now. A new downstream check may be absent from the original record. We call a mismatch between responsibility-derived locations and those needed for optimal recovery Responsibility–Recourse

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

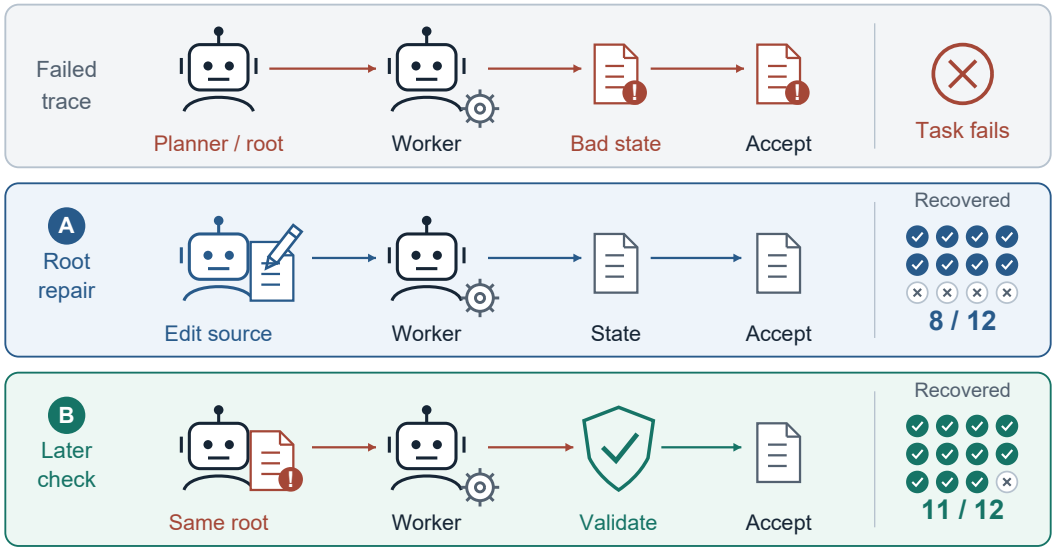

Figure 1: Recovery after a planner–worker failure. A edits the diagnosed source; B leaves it unchanged and adds a check before acceptance. Both are feasible under matched costs and replay conditions. Markers show aggregate outcomes: B recovers three more of twelve replays and is selected in this constructed CRPA-R1 example.

Separation (RRS). Restricting changes to diagnosed locations can lose value even when the correct root remains repairable.

Second, comparisons must separate location restrictions from search quality. Actions may interact, and a weak root-repair rule may never test the best root repair. Compatible combinations must therefore be optimized within each allowed set under shared conditions; otherwise, a recovery gap cannot be assigned to location restrictions.

For the first challenge, we propose the Cross-Surface Constrained Recovery Model: it represents root, communication-path, and downstream actions and constrains their combinations. A surface means a location an action modifies. For the second, we design Feasibility-Gated Joint Replay Search, which applies compatible actions together and compares the best values under full, root-only, and responsibility-based access. Shared replay conditions isolate value excluded by location restrictions; full access supplies an offline reference over observed candidate outcomes.

We make three contributions:

- We propose a Cross-Surface Constrained Recovery Model that defines RRS through excluded recovery value and separates location, information, and selection losses.
- We design Feasibility-Gated Joint Replay Search to screen and jointly evaluate compatible actions, enabling matched, exhaustive comparisons of intervention locations.
- We design a controlled evaluation with fixed diagnosis, constraints, and within-location search: full access gains 0.135229 mean recovery Lower Confidence Bound (LCB) on 96 repairable-root traces; a fixed-attribution AgentRx comparison gains 19.75 percentage points in executable recovery.

2 Related Work

2.1 Failure Attribution and Evaluation

Failure attribution locates responsible agents and steps; evaluation specifies which failure evidence and annotations are available. Cemri et al. (2025) introduced the Multi-Agent System Failure Taxonomy (MAST), categorizing failures into system design, inter-agent misalignment, and task verification. Zhang et al. (2025) introduced Who&When, linking logs to responsible agents and decisive steps. West et al. (2025) proposed Abduct-Act-Predict (A2P), using abduction, a proposed

intervention, and trajectory prediction to locate decisive errors. Ma et al. (2025) combined reversed performance dependencies, Shapley attribution, and causal discovery for agent- and step-level attribution. Wang et al. (2026) proposed a hierarchical causal attribution framework using causal graphs, backtracking, and counterfactual screening to separate causes from symptoms. In et al. (2026) introduced the Multi-Perspective Failure Attribution Benchmark (MP-Bench) to evaluate multiple plausible attributions. Chen et al. (2026) introduced TraceElephant, providing full traces and reproducible environments. These studies vary the causal representation, observed evidence, and treatment of attribution ambiguity.

2.2 Feedback and Intervention-Based Recovery

Intervention-based debugging changes execution and checks the outcome; feedback-based refinement instead uses critiques to guide later attempts. Shinn et al. (2023) proposed Reflexion, storing verbal reflections from feedback to improve subsequent trials. Madaan et al. (2023) proposed Self-Refine, iterating feedback and revision with the same language model. Ma et al. (2026) proposed Do-Then-Verify (DoVer), testing failure hypotheses through message and plan edits and measuring recovery or progress; distinct interventions can independently repair the same failure. Bonagiri et al. (2026) proposed CausalFlow, using counterfactual step interventions and minimal successful edits for repair and training supervision. Zhu et al. (2026) proposed AgentDebugX, combining diagnosis, recovery, and rerunning. These methods assess or revise agent behavior using outcome feedback, textual critique, or executable interventions.

2.3 Actionable Recourse and Decision Quality

Algorithmic recourse searches for feasible changes that produce a desired outcome, while decision-focused learning evaluates predictions through the decisions they support. Ustun et al. (2019) proposed integer-programming tools that identify changes to actionable features capable of reversing a linear classifier's unfavorable decision. Karimi et al. (2021) proposed recourse through minimal causal interventions, accounting for how an action changes other variables through structural dependencies. Sabu et al. (2026) proposed Plausibility-Aware Amortized Recourse Generation (PAR), which learns an amortized recourse generator using class-conditional likelihoods, recourse constraints, and neighborhood conditioning to produce plausible changes. Wilder et al. (2019) proposed a decision-focused learning framework that differentiates through continuous relaxations of discrete optimization problems to train predictors for downstream decision quality. These approaches specify desirable outcomes together with action constraints, causal effects, data plausibility, or optimization objectives, making intervention quality an explicit object of evaluation.

3 Method

The complete recovery framework is shown in Figure 2. Given a failed execution, attribution, candidate actions, and current constraints, it returns an intervention bundle or abstention and an audit of location restrictions. The candidate representation module records what actions change at the root, along propagation paths, or at downstream checks, exposing alternative locations. The gated joint replay module screens and combines compatible actions, then replays each combination to measure recovery under shared conditions. The selection and loss audit module selects the highest-value admitted bundle and repeats selection under root-only and responsibility-based restrictions to measure excluded value. The following subsections define the recovery model, its search algorithm, and the theoretical analysis of recovery loss.

3.1 Cross-Surface Constrained Recovery Model

3.1.1 Representing the Execution and Recovery Locations

This module records the diagnosis alongside the locations that recovery can change. Let $T = (G, X, Y^{(0)})$ be a failed execution: G is its event graph, X its states and actions, and $Y^{(0)}$ its outcome. Locations $\mathcal{S}_T$ include recorded events and edges as well as new checks. Historical responsibility R_T describes causal contributions of recorded locations; r_T is the diagnosed root and $\widehat{H}_T$ the responsibility-selected locations. In Figure 2, a downstream check is an available action even

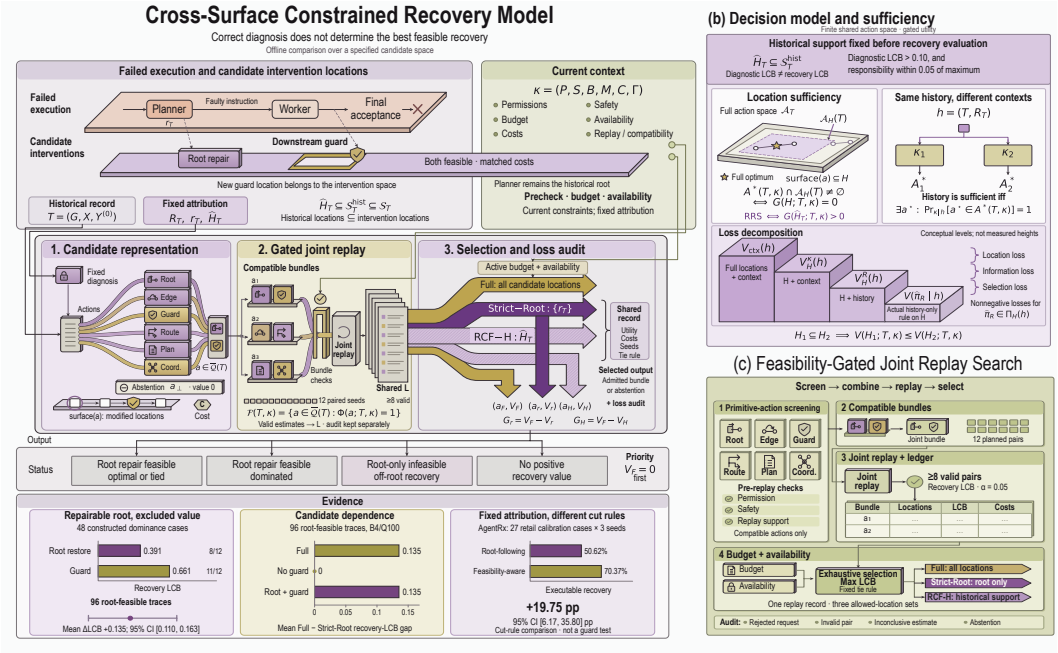

Figure 2: Complete recovery framework. The left panel shows candidate representation, gated joint replay, and selection and loss audit; the right panels summarize the decision model and search procedure. Responsibility-Conditioned Frontier on $\widehat{H}_T$ (RCF-H) restricts changes to historical attribution locations. Evaluation is offline over a specified candidate space.

though it was absent from the failed execution. This representation exposes recovery choices beyond historical attribution.

3.1.2 Forming Bundles and Enforcing Constraints

This module identifies actions that can execute together. A bundle a combines one or more root repairs, edge changes, downstream checks (guards), rerouting, replanning, or orchestration changes; $\mathrm{surface}(a)$ lists its modified locations. A finite universe $\overline{\mathcal{Q}}(T)$ is fixed before outcome evaluation. Context κ specifies permissions, safety, budget, availability, costs, and replay/compatibility conditions. The gate Φ admits only available, permitted, safe, compatible, affordable bundles with valid recovery estimates:

$$F(T,\kappa) = \{a \in \overline{\mathcal{Q}}(T) : \Phi(a;T,\kappa) = 1\}. \tag{1}$$

For example, individually permitted edits may make conflicting writes. Abstention $a_\perp$ changes nothing and has value zero. We use $\mathcal{A}_T = \overline{\mathcal{Q}}(T) \cup \{a_\perp\}$ and extend utility to rejected requests as

$$\widetilde{U}(a;T,\kappa) = \begin{cases} U(a;T,\kappa), & a \in F(T,\kappa), \\ 0, & a \notin F(T,\kappa), \end{cases} \qquad \widetilde{U}(a_\perp;T,\kappa) = 0. \tag{2}$$

Rejected requests retain separate status; selection returns an admitted bundle or abstention. Any hard-gate failure rejects the entire bundle, even when other component actions are individually admissible. The ledger distinguishes such rejection from estimates below the valid-pair threshold and from valid replays with no observed recovery, so missing evidence is not treated as evidence of failure. This prevents unavailable or incompatible proposals from counting as executable recovery.

3.1.3 Scoring and Selecting Recovery Bundles

This module compares locations using the same recovery score. Let $\rho(Y^{(a)})$ indicate task success after applying all bundle actions together. Utility is the recovery-rate LCB:

$$U(a;T,\kappa) = \mathrm{LCB}_\alpha[\rho(Y^{(a)}) \mid T,\kappa], \qquad V_F(T,\kappa) = \max_{a \in \mathcal{A}_T} \widetilde{U}(a;T,\kappa). \tag{3}$$

For k successful recoveries among n valid paired replays, the one-sided exact Clopper–Pearson bound (Clopper & Pearson, 1934) is

$$U(a) = \begin{cases} 0, & k = 0, \\ F^{-1}_{\mathrm{Beta}(k,n-k+1)}(0.05), & k > 0, \end{cases} \tag{4}$$

where F^{-1} is the beta quantile. Each bundle receives 12 planned paired replays; fewer than eight valid pairs makes an estimate inconclusive, distinguishing insufficient evidence from observed failure. Budgets B1–B4 cap (modified surfaces, dispatch-cost proxy) at $(1, 2)$, $(1, 4)$, $(2, 6)$, and $(3, 10)$, respectively. Both caps must hold; recovery, token, and time values do not enter this gate. Budget changes therefore alter admission while preserving recorded estimates.

Descending LCB determines value. Ties use ascending collateral damage, modified-surface count, dispatch cost, rendered-plus-generated tokens, time cost, and then lexicographic bundle identifier. Bundle size and action index are excluded. The Full-Candidate Replay Oracle (Full) returns an admitted maximizer or abstention, giving an offline reference over observed candidates.

3.1.4 Auditing the Loss from Location Restrictions

This module measures value excluded by restricting intervention locations. For allowed locations H, define $\mathcal{A}_H(T) = \{a \in \mathcal{A}_T : \mathrm{surface}(a) \subseteq H\}$ and

$$V(H; T, \kappa) = \max_{a \in \mathcal{A}_H(T)} \widetilde{U}(a; T, \kappa), \qquad G(H; T, \kappa) = V_F - V(H; T, \kappa). \tag{5}$$

Strict-Root uses $H = \{r_T\}$; Responsibility-Conditioned Frontier on $\widehat{H}_T$ (RCF-H) uses $H = \widehat{H}_T$. Their values are V_r, V_H and their losses are $G_r = V_F - V_r$ and $G_H = V_F - V_H$. Both optimize exhaustively within allowed locations. Responsibility–Recourse Separation (RRS) occurs when $G(H; T, \kappa) > 0$: every optimal bundle needs a location outside responsibility-derived H. A root is repairable if a non-abstention root-only bundle is feasible. A positive gap then identifies dominance-induced separation; an infeasible root instead gives feasibility-induced separation. Root-optimal/tied and no-positive-recovery cases remain in reporting denominators. This audit distinguishes value lost despite an executable root repair from value lost because root repair is unavailable; Appendix B.1 gives the expanded definitions.

3.2 Feasibility-Gated Joint Replay Search

The search turns candidate descriptions into executable recovery choices (Algorithm 1). Screening checks permission, safety, replay/scorer availability, and execution support before enumerating compatible bundles; this excludes conflicting joint edits. Joint replay applies all bundle actions together under common seeds and scoring, producing a shared record L that captures interactions. Selection filters that record by budget, availability, and allowed locations, then maximizes LCB with the fixed tie rule; this isolates location loss while retaining rejected and inconclusive statuses.

Algorithm 1 Feasibility-Gated Joint Replay Search

Require: T, bundles $\overline{\mathcal{Q}}(T)$, root r_T, support $\widehat{H}_T$, context κ, shared seeds/scorer/ties.

1: Screen primitive actions and enumerate compatible bundles; initialize $L = \emptyset$.
2: Jointly replay each bundle on 12 paired seeds, excluding invalid pairs.
3: Store $(a, U(a), \text{damage}, \text{costs})$ in L if at least 8 pairs remain; otherwise record an inconclusive estimate.
4: Filter L by budget and availability to obtain L_F; set $H_F = \mathcal{S}_T$, $H_r = \{r_T\}$, $H_H = \widehat{H}_T$.
5: For $j \in \{F, r, H\}$, restrict L_F to surfaces in H_j and select (a_j, V_j) by exhaustive LCB maximization with the fixed tie rule; use $(a_\perp, 0)$ if empty.
6: Return (a_F, a_r, a_H), $G_r = V_F - V_r$, $G_H = V_F - V_H$, and root feasibility; retain rejection, inconclusive, and abstention counts.

For N bundles and m seeds, replay costs at most Nm paired evaluations; enumeration can grow exponentially before pruning. Full uses observed outcomes. Section 4.6 evaluates selectors without them; Appendix A.1 expands the pseudocode.

3.3 Theoretical Analysis of Recovery Loss

This analysis distinguishes loss caused by restricted intervention locations from loss caused by missing current information or imperfect selection. Hold the history $h = (T, R_T)$ fixed and let current context κ vary. Write V_{ctx} for the expected optimum with full locations and observed context, V_H^κ for the corresponding optimum on responsibility-derived locations H, and V_H^R for the best policy on H that sees only history. For any such history-only rule $\widehat{\pi}_R$, nested decision classes give

$$V_{\mathrm{ctx}} - V(\widehat{\pi}_R \mid h) = \underbrace{V_{\mathrm{ctx}} - V_H^\kappa}_{\text{location loss}} + \underbrace{V_H^\kappa - V_H^R}_{\text{information loss}} + \underbrace{V_H^R - V(\widehat{\pi}_R \mid h)}_{\text{selection loss}}. \tag{6}$$

All terms are nonnegative; the first equals the expected location gap $G(H; T, \kappa)$. With a finite shared action space, history alone attains the context-aware optimum exactly when one action is optimal in almost every compatible context. Thus a correct diagnosis can still leave decision-relevant uncertainty about current permissions or budgets. Appendix A.2 gives the formal assumptions, statements, and proofs. This analysis identifies what each evaluation measures: CRPA-R1 controls context and within-location search to isolate location loss, while the fixed-attribution comparison examines recovery-rule dependence.

4 Experiments

The experiments test location-restriction loss, its dependence on downstream validation, and recovery-rule differences under fixed attribution. CRPA-R1 supplies controlled replay evidence; AgentRx supplies an executable continuation test. We also test alternative recovery objectives and selection rules under common candidate access; the appendix retains the full sensitivity record.

4.1 Experimental Setup

CRPA-R1 contains 144 template-generated traces: 24 root-optimal controls, 48 repairable-root dominance cases, 24 root-inadmissible cases, 24 cost-inversion boundaries, and 24 no-recovery controls. The widest budget (B4) and full candidate coverage (Q100) are denoted B4/Q100; the boundaries are root-feasible ties, giving 96 repairable roots. Construction and replay use no external service or model calls.

Full, Strict-Root, and RCF-H share candidates, gates, budget, seeds, scorer, and ties, with exhaustive selection in each support. We report recovery LCBs, paired differences, and trace-level Win/Tie/Loss (W/T/L) counts. Each 95% Confidence Interval (CI) uses 20,000 trace-level bootstrap repetitions (Efron, 1979).

The AgentRx continuation comparison (Barke et al., 2026) uses 27 previously examined retail calibration cases from the Tool-Agent-User Interaction Benchmark (τ-bench) (Yao et al., 2025) and three seeds. Both arms share attribution, trajectories, continuation policy, tools, database state, and scorer. Root-following cuts at the predicted step; the alternative selects a legal cut from visible execution conditions. Recovery intervals and W/T/L use seed-averaged cases.

4.2 Root-Only Repair Can Exclude Better Recovery

We test whether restricting repair to a correct, editable root excludes better recovery. At B4/Q100, we compare Full with exhaustive Strict-Root and RCF-H. In 48 dominance cases, root restore and downstream validation each change one surface, have matched synthetic costs, and pass the same gates, controlling diagnosis, root feasibility, and within-location search.

Table 1: Primary B4/Q100 contrasts. Effects are mean paired recovery-LCB differences with unadjusted 95% bootstrap intervals; W/T/L denotes trace-level wins/ties/losses.

Contrast	n	Gain [95% CI]	W/T/L
Full − Strict-Root, all	144	$+0.207622$ $[0.166200, 0.251019]$	72/72/0
Full − Strict-Root, root-feasible	96	$+0.135229$ $[0.109873, 0.163401]$	48/48/0
Full − Strict-Root, dominance	48	$+0.270457$ $[0.270457, 0.270457]$	48/0/0
Full − RCF-H, all	144	$+0.180570$ $[0.142673, 0.219809]$	67/77/0

Table 1 quantifies a 0.135229 mean recovery-LCB gain over exhaustive root-only repair on the 96 repairable roots. Across the 48 dominance cases, downstream validation recovers three more of 12 replays than root restore, a 25-percentage-point gain. The all-trace contrast additionally includes infeasible roots. Restricting access by responsibility yields a 0.180570 loss against Full, so broadening beyond a single root still need not retain all recovery value. The dominance interval collapses because its 48 traces share a template-imposed effect; it does not imply zero uncertainty on natural failures. Root-optimal and no-recovery controls remain in the all-trace denominator, so the gain does not result from reporting only traces with successful downstream checks. Exhaustive within-location search also prevents a weak root-selection rule from creating the apparent advantage. These constructed cases establish the mechanism, not its prevalence.

4.3 Downstream Validation Explains the Controlled Gap

We test whether downstream validation drives the controlled recovery gap by removing each action family and every bundle containing it, then reselecting both arms at B4/Q100. Root-and-guard-only access tests sufficiency. Both comparisons use all 144 traces and the 96 repairable-root traces.

Table 2: Candidate-family ablation. Entries are mean Full–Strict-Root recovery-LCB gaps after both arms reselect from the remaining bundles.

Candidate access	All traces	Repairable roots
Full B4/Q100	0.207622	0.135229
Remove guard	0	0
Remove edge / reroute / plan	0.207622	0.135229
Remove orchestrator	0.190182	0.135229
Root and guard only	0.188674	0.135229

Table 2 shows that guard removal eliminates every gap; root-and-guard access preserves the full repairable-root effect. Removing orchestrator actions lowers only the all-trace gap by 0.017440; root-and-guard access lowers it by 0.018948. All 72 positive-gap traces originally select guard-containing bundles. Thus downstream validation is necessary for this constructed gap and sufficient for its repairable-root component. Removing edge, reroute, or plan actions does not change it.

4.4 The Recovery Rule Matters Even with Fixed Attribution

We test whether changing the recovery rule changes executable recovery with attribution held fixed. On 27 retail calibration cases with three seeds, we compare Root-following with feasibility-aware continuation cuts under shared execution conditions. Parse failures remain abstentions among 81 paired units.

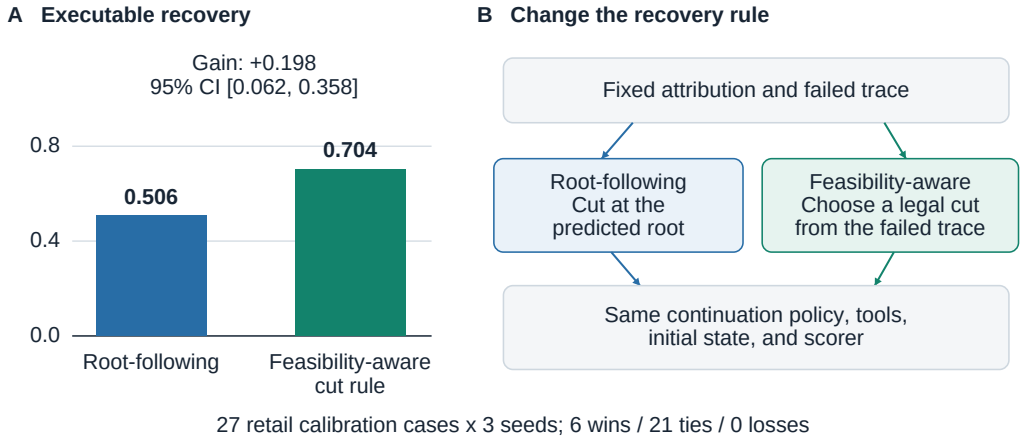

Figure 3: AgentRx with fixed attribution and execution conditions. Panel A reports pooled recovery over 81 paired units; the confidence interval and W/T/L use 27 seed-averaged cases. Panel B shows the changed recovery rule.

The feasibility-aware rule improves pooled executable recovery by 19.75 percentage points (Figure 3). The 95% case-bootstrap interval for this gain is $[6.17, 35.80]$ percentage points; six seed-averaged cases improve, 21 tie, and none decline (two-sided sign-test $p = 0.03125$). The gain isolates cut-rule dependence under shared attribution and continuation conditions. Exact-step attribution accuracy is 0.074074; CRPA-R1 supplies the separate test with correct diagnosis. These previously examined calibration cases leave generalization to fresh cases untested.

4.5 The Recovery Objective Changes the Comparison

We test whether the location gap depends on the recovery objective, re-ranking the B4/Q100 replay record with unchanged gates and ties. Point recovery uses the success fraction; lower-tail Conditional Value At Risk (CVaR) averages the worst 25% or 50% of valid replay deltas; worst-valid-seed uses their minimum. Pareto filtering excludes bundles dominated in recovery, damage, and costs before LCB ranking. These post-result analyses require no new execution.

Table 3: Objective sensitivity: mean Full–Strict-Root gaps within each objective. Magnitudes across objectives are on different scales.

Recovery objective	All 144 traces	96 repairable roots
Recovery LCB	0.207622	0.135229
Point recovery	0.240741	0.125000
Lower-tail CVaR, worst 25%	0.351852	0.333333
Lower-tail CVaR, worst 50%	0.314815	0.250000
Pareto filtering then LCB	0.207622	0.135229
Worst valid seed	0.090278	0

The repairable-root advantage remains positive under point recovery, both CVaR objectives, and Pareto filtering, but disappears under worst-valid-seed evaluation (Table 3). Point recovery and both CVaR rules retain 48 wins, 48 ties, and no losses in this cohort; point recovery and Pareto filtering preserve both arms' original selections on all 144 traces. Thus the gap is not unique to confidence-bound scoring, but an objective that gives decisive weight to the worst replay can erase it. The result supports reporting the recovery objective alongside the intervention policy. This sensitivity concerns the existing paired replay samples; it does not estimate robustness to an unobserved distribution of future executions.

4.6 Full Candidate Access Still Requires Selection

We test whether common candidate access suffices for equal recovery by comparing four selectors on the same feasible B4/Q100 candidates across all 144 traces, with zero-valued abstention. Minimum-cost chooses the cheapest feasible bundle; random selection uses seed 17011; the metadata heuristic ranks information available before execution. Full additionally observes replay outcomes.

Table 4: Selection rules with common candidate access. Full is an offline replay oracle; the other rules do not observe replay outcomes.

Selection rule	Mean recovery LCB
Minimum-cost feasible	0.339921
Seeded random feasible	0.333162
Outcome-blind metadata heuristic	0.421524
Full-Candidate Replay Oracle	0.597602

The metadata heuristic gains 0.081603 over minimum-cost selection and 0.088362 over random selection, while remaining 0.176078 below Full (Table 4). Candidate access alone therefore does not determine recovery quality: a selector must identify valuable feasible actions within that space. Full's extra outcome information and evaluation cost prevent interpreting its advantage as a deployment gain under matched resources; the comparison instead quantifies the remaining selection problem. A selector can fail at this stage without violating any constraint: every chosen candidate may be feasible yet have lower recovery value. This differs from the root-only comparison, where a useful candidate may never be available. The two comparisons therefore identify different limitations even when both yield a gap to Full.

4.7 Discussion and Limitations

CRPA-R1 isolates location-restriction loss under constructed mechanisms and synthetic costs; AgentRx identifies rule dependence within a calibrated retail cohort. Together they motivate reporting attribution, recovery, intervention locations, and abstentions separately. Guard and worst-seed analyses delimit the controlled result. Fresh failures, measured costs, and executable guard tests are needed to establish prevalence and deployment efficiency. Practical selectors require matched-budget comparisons without Full's observed outcomes. Replaying the same failed history under different permissions would test the predicted value of current context; external attribution adapters would extend the comparison to different diagnostic methods.

5 Conclusion

We studied constrained intervention selection after multi-agent failures to connect diagnosis with executable recovery value. Existing attribution and debugging identify causes and repairs, but do not establish the value retained when repair locations are restricted. The first challenge is to represent feasible recovery locations beyond a correctly diagnosed, repairable root. The second is to compare interacting actions under matched conditions and separate location from selection effects. We proposed the Cross-Surface Constrained Recovery Model to represent actions and compatible combinations across the workflow and measure responsibility–recourse separation. We designed Feasibility-Gated Joint Replay Search: candidate representation enumerates changes, joint replay evaluates compatible bundles, and selection returns the best recorded recovery and its location-restriction loss. Full access improves mean recovery LCB over exhaustive root-only repair by 0.135229 on 96 CRPA-R1 traces with repairable roots; the fixed-attribution AgentRx calibration comparison improves executable recovery by 19.75 percentage points. The controlled downstream-interception evidence remains limited to one validation-guard family, leaving other checking mechanisms untested. Guided by the observed benefit of intercepting propagated errors before acceptance, we plan to test alternative interception mechanisms on fresh failure cohorts to assess their generality.

REFERENCES

Shraddha Barke, Arnav Goyal, Alind Khare, Avaljot Singh, Suman Nath, and Chetan Bansal. AgentRx: Diagnosing AI agent failures from execution trajectories. arXiv:2602.02475, 2026. Preprint.

Akash Bonagiri, Devang Borkar, Gerard Janno Anderias, Setareh Rafatirad, and Houman Homayoun. CausalFlow: Causal attribution and counterfactual repair for LLM agent failures. arXiv:2605.25338, 2026. Preprint.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei A Zaharia, Joseph Gonzalez, and Ion Stoica. Why do multi-agent LLM systems fail? In *Advances in Neural Information Processing Systems*, volume 38, pp. 135676–135729. Curran Associates, Inc., 2025. doi: 10.52202/085713-4082. Datasets and Benchmarks Track.

Mengzhuo Chen, Junjie Wang, Fangwen Mu, Yawen Wang, Zhe Liu, Huanxiang Feng, and Qing Wang. Seeing the whole elephant: A benchmark for failure attribution in LLM-based multi-agent systems. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19888–19905. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.912.

C. J. Clopper and E. S. Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26(4):404–413, 1934. doi: 10.1093/biomet/26.4.404.

B. Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979. doi: 10.1214/aos/1176344552.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. JSTOR archival record; no resolvable DOI was found.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations*, volume 2024, pp. 23247–23275, 2024.

Yeonjun In, Mehrab Tanjim, Jayakumar Subramanian, Sungchul Kim, Uttaran Bhattacharya, Wonjoong Kim, Sangwu Park, Somdeb Sarkhel, and Chanyoung Park. Rethinking failure attribution in multi-agent systems: A multi-perspective benchmark and evaluation. arXiv:2603.25001, 2026. Preprint.

Amir-Hossein Karimi, Bernhard Schölkopf, and Isabel Valera. Algorithmic recourse: From counterfactual explanations to interventions. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 353–362. ACM, 2021. doi: 10.1145/3442188.3445899.

Guoqing Ma, Jia Zhu, Hanghui Guo, Weijie Shi, Jiawei Shen, Jingjiang Liu, and Yidan Liang. Automatic failure attribution and critical step prediction method for multi-agent systems based on causal inference. arXiv:2509.08682, 2025. Preprint.

Ming Ma, Jue Zhang, Fangkai Yang, Yu Kang, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. DoVer: Intervention-driven auto debugging for LLM multi-agent systems. In *The Fourteenth International Conference on Learning Representations*, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46534–46594. Curran Associates, Inc., 2023. doi: 10.52202/075280-2019.

Anagha Sabu, Vidhya S, and Narayanan C Krishnan. PAR: Plausibility-aware amortized recourse generation. arXiv:2601.17309, 2026. Preprint.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. doi: 10.52202/075280-0377.

Berk Ustun, Alexander Spangher, and Yang Liu. Actionable recourse in linear classification. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pp. 10–19. ACM, 2019. doi: 10.1145/3287560.3287566.

Yawen Wang, Wenjie Wu, Junjie Wang, and Qing Wang. From flat logs to causal graphs: Hierarchical failure attribution for LLM-based multi-agent systems. arXiv:2602.23701, 2026. Preprint.

Alva West, Yixuan Weng, Minjun Zhu, Zhen Lin, Zhiyuan Ning, and Yue Zhang. Abduct, act, predict: Scaffolding causal inference for automated failure attribution in multi-agent systems. arXiv:2509.10401, 2025. Preprint.

Bryan Wilder, Bistra Dilkina, and Milind Tambe. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):1658–1665, 2019. doi: 10.1609/aaai.v33i01.33011658.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *International Conference on Learning Representations*, volume 2025, pp. 9965–10017, 2025.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and Qingyun Wu. Which agent causes task failures and when? On automated failure attribution of LLM multi-agent systems. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 76583–76599. PMLR, 2025.

Kunlun Zhu, Xuyan Ye, Zhiguang Han, Yuchen Zhao, Bingxuan Li, Weijia Zhang, Muxin Tian, Xiangru Tang, Pan Lu, James Zou, Jiaxuan You, and Heng Ji. AgentDebugX: An open-source toolkit for failure observability, attribution, and recovery in LLM agents. arXiv:2607.18754, 2026. Preprint.

AI USE STATEMENT

Language-model assistance was used for manuscript organization, wording, literature discovery and metadata checks, technical checks, LaTeX editing, and preparation of explanatory diagrams. The reported numerical results were retained from the study records. Language-model-based attribution within the experiment is described in Appendix E. The authors are responsible for the final claims, calculations, citations, and disclosures.

A ALGORITHM, THEORY, AND PROOFS

Throughout, the observed history is $h = (T, R_T)$, the requested-action set $\mathcal{A}_T$ is finite, and all contexts share this set. Write $A^*(T, \kappa) = \arg\max_{a \in \mathcal{A}_T} \widetilde{U}(a; T, \kappa)$ for the full optimal-action set. Utilities are evaluated through the feasibility gate in Eq. 2; recovery values lie in $[0, 1]$, and abstention has value zero. Expectations use $\mu(\kappa \mid h)$. A history-only policy cannot condition on κ. These assumptions ensure that the maxima and expectations used below are well defined.

A.1 SEARCH PSEUDOCODE

The pseudocode expands the three search stages in the main method. It preserves the same feasibility gates, joint replay record, and within-location optimization for all comparison arms.

Algorithm 2 Feasibility-Gated Joint Replay Search

Require: Failed execution T; specified candidate bundles $\overline{\mathcal{Q}}(T)$; root r_T and responsibility support $\widehat{H}_T$; context κ; paired seeds, recovery scorer, and tie-breaking rule.
Ensure: Best admitted bundles (a_F, a_r, a_H), their values (V_F, V_r, V_H), and location-restriction losses (G_r, G_H).
1: Screen candidate actions by permission, safety, replayability, scoring, and execution support.
2: Enumerate $a \in \overline{\mathcal{Q}}(T)$ whose actions are admitted and whose joint changes are compatible; record costs and budget memberships. Initialize $L = \emptyset$.
3: **for** each enumerated bundle a **do**
4: Apply all actions in a jointly and run the 12 paired replays.
5: Exclude invalid replay pairs. If at least 8 pairs remain, store $U(a)$, collateral damage, and costs in L; otherwise mark the estimate inconclusive.
6: **end for**
7: Filter L by the active budget and candidate availability to obtain L_F.
8: Form $L_r = \{a \in L_F : \text{surface}(a) \subseteq \{r_T\}\}$ and $L_H = \{a \in L_F : \text{surface}(a) \subseteq \widehat{H}_T\}$.
9: **for** $j \in \{F, r, H\}$ **do**
10: Select the bundle with highest recovery LCB in L_j, using the fixed tie-breaking rule; return $(a_\perp, 0)$ if L_j is empty.
11: **end for**
12: Return selected bundles, $G_r = V_F - V_r$, $G_H = V_F - V_H$, and root feasibility. Retain rejection, inconclusive-estimate, and abstention records; keep every trace in the reporting denominator.

A.2 FORMAL RECOVERY-LOSS ANALYSIS

The model and search quantify loss from limiting intervention locations at a fixed context. We now relate that loss to two other constraints on recovery: missing information about the current environment and imperfect action selection. This analysis explains which aspect of recovery the empirical gaps measure. Proofs follow the formal statements below.

A.2.1 WHEN ARE HISTORY AND ATTRIBUTION SUFFICIENT?

This analysis asks whether a diagnosis is enough to choose a recovery action when current operating conditions may change. For example, the same failed trace may be replayed under two permission settings that admit different repairs; the historical cause is unchanged, but the preferred action may change. Let $h = (T, R_T)$ contain the failed execution and its historical responsibility, and let $\mu(\kappa \mid h)$

be a distribution over current contexts compatible with that same history. The finite action universe $\mathcal{A}_T$ is shared across contexts. A context-aware policy observes κ. Let $\Pi_R(h)$ contain all distributions over $\mathcal{A}_T$ that depend on h but not on κ; each such history-only policy uses the same action distribution across compatible contexts. Their optimal values are

$$V_{\mathrm{ctx}}(h) = \mathbb{E}_{\kappa|h}\left[\max_{a\in\mathcal{A}_T} \widetilde{U}(a;T,\kappa)\right], \tag{7}$$

$$V_R(h) = \sup_{\pi\in\Pi_R(h)} \mathbb{E}_{\kappa|h}\mathbb{E}_{a\sim\pi(\cdot|h)}[\widetilde{U}(a;T,\kappa)]. \tag{8}$$

Define the value of observing the current context as

$$\Delta_{\mathrm{ctx}}(h) = V_{\mathrm{ctx}}(h) - V_R(h). \tag{9}$$

Theorem 1 (Value of Post-Failure Context). For the finite action space and gated utility above, $\Delta_{\mathrm{ctx}}(h) \geq 0$. History and attribution are decision-sufficient when $\Delta_{\mathrm{ctx}}(h) = 0$: observing current conditions provides no additional optimal recovery value.

Proposition 1 (Zero-Gap Characterization). $\Delta_{\mathrm{ctx}}(h) = 0$ if and only if one action is optimal for almost every compatible context:

$$\exists a^\star \in \mathcal{A}_T : \quad \Pr_{\kappa|h}[a^\star \in A^*(T,\kappa)] = 1. \tag{10}$$

For a finite context family with positive probability on each member, this means that their optimal action sets have a common member. The same history and attribution suffice when current conditions leave a common best action. If current conditions require different actions, additional information has decision value.

Corollary 1 (Context-Ambiguity Regret Bound). Suppose two contexts sharing h have probabilities $p \in (0,1)$ and $1-p$, disjoint optimal sets $A_j^* = A^*(T,\kappa_j)$, and gaps $\gamma_j = V_j - \max_{a\in\mathcal{A}_T\setminus A_j^*} \widetilde{U}(a;T,\kappa_j) > 0$, where $V_j = V_F(T,\kappa_j)$. For $\mathrm{Regret}(\pi;\kappa) = V_F(T,\kappa) - \mathbb{E}_{a\sim\pi(\cdot|h)}[\widetilde{U}(a;T,\kappa)]$, every history-only policy has expected regret

$$\mathbb{E}_\kappa[\mathrm{Regret}(\pi;\kappa)] \geq \min\{p\gamma_1, (1-p)\gamma_2\}. \tag{11}$$

Current permissions or budget can therefore have decision value even with the same complete failure record and the same correct diagnosis.

A.2.2 Separating Location, Information, and Selection Loss

This analysis separates three reasons why a recovery rule can fall short: it cannot act at a useful location, it cannot observe a relevant current condition, or it chooses poorly among its available actions. We compare nested decision classes to assign a distinct value loss to each restriction. Let $H(h)$ be a responsibility-derived support and $\Pi_H(h)$ the history-only policies supported on $\mathcal{A}_{H(h)}(T)$. Define the context-aware and history-only optima within that support:

$$V_H^\kappa(h) = \mathbb{E}_{\kappa|h}[V(H(h);T,\kappa)], \tag{12}$$

$$V_H^R(h) = \sup_{\pi\in\Pi_H(h)} \mathbb{E}_{\kappa|h}\mathbb{E}_{a\sim\pi(\cdot|h)}[\widetilde{U}(a;T,\kappa)]. \tag{13}$$

For an actual rule $\widehat{\pi}_R \in \Pi_H(h)$, write $V(\widehat{\pi}_R \mid h)$ for its expected gated utility under the same distribution.

Theorem 2 (Diagnosis-to-Recourse Loss Decomposition). The recovery loss of any such rule decomposes as

$$\begin{aligned}\mathcal{L}_{\mathrm{total}}(h) &= V_{\mathrm{ctx}}(h) - V(\widehat{\pi}_R \mid h) \\ &= \underbrace{V_{\mathrm{ctx}}(h) - V_H^\kappa(h)}_{\mathcal{L}_{\mathrm{support}}(h)} + \underbrace{V_H^\kappa(h) - V_H^R(h)}_{\mathcal{L}_{\mathrm{context}}(h)} + \underbrace{V_H^R(h) - V(\widehat{\pi}_R \mid h)}_{\mathcal{L}_{\mathrm{selector}}(h)}.\end{aligned} \tag{14}$$

All three terms are nonnegative under these nested support and policy classes.

The first term is the value excluded by the allowed intervention locations, even when current conditions are known. The second is the additional loss from hiding those conditions. The third is the remaining loss of the actual selection rule within its information and action restrictions. They correspond to three improvements: expand intervention locations, observe current conditions, and improve action selection.

For the first term,

$$\mathcal{L}_{\text{support}}(h) = \mathbb{E}_{\kappa|h}[G(H(h); T, \kappa)]. \tag{15}$$

Thus the algorithm's G_r and G_H measure this loss at individual contexts. CRPA-R1 evaluates these quantities with attribution, context, and search within each support controlled. The information and selection terms specify complementary evaluation targets: repeated contexts with the same history, and policies compared under matched information and computation budgets.

A.3 Pointwise Support-Sufficiency Equivalence

For finite $\mathcal{A}_T$, pointwise recourse sufficiency of H_T is equivalent to $V(H_T; T, \kappa) = V_F(T, \kappa)$. Since abstention belongs to every restricted set, $\mathcal{A}_{H_T}(T)$ is nonempty and finite, and its maximum is attained. If $A^*(T, \kappa) \cap \mathcal{A}_{H_T}(T)$ contains an action, that action attains V_F inside the restricted set. Conversely, equality of the restricted and full maxima means that a restricted maximizer belongs to $A^*(T, \kappa)$. Thus $G(H_T; T, \kappa) = 0$ exactly when the restricted support contains a full optimum.

The zero extension also covers supports containing no feasible intervention. When $V_F > 0$, every optimizer has positive value and is feasible. When $V_F = 0$, $a_\perp$ witnesses equality while rejected proposals tied at zero retain distinct recorded statuses.

A.4 Proof of Theorem 1: Value of Post-Failure Context

Fix observed history $h = (T, R_T)$. For every history-only policy $\pi \in \Pi_R(h)$ and every context κ,

$$\mathbb{E}_{a\sim\pi(\cdot|h)}[\widetilde{U}(a; T, \kappa)] \le \max_{a\in\mathcal{A}_T} \widetilde{U}(a; T, \kappa). \tag{16}$$

Taking expectation over $\kappa \sim \mu(\cdot \mid h)$ and then the supremum over π gives

$$V_R(h) \le \mathbb{E}_{\kappa|h}[\max_a \widetilde{U}(a; T, \kappa)] = V_{\text{ctx}}(h). \tag{17}$$

Hence $\Delta_{\text{ctx}}(h) \ge 0$.

A.5 Proof of Proposition 1: Zero-Gap Characterization

If there exists $a^\star$ that is optimal for $\mu(\cdot \mid h)$-almost every context, the point-mass policy $\pi(a^\star \mid h) = 1$ achieves $V_{\text{ctx}}(h)$, so $\Delta_{\text{ctx}}(h) = 0$.

Conversely, suppose $\Delta_{\text{ctx}}(h) = 0$ and let $\pi^\star$ attain $V_R(h)$; an optimizer exists because the action simplex is compact and the objective is linear for finite $\mathcal{A}_T$. Define the pointwise regret

$$r_{\pi^\star}(\kappa) = V_F(T, \kappa) - \mathbb{E}_{a\sim\pi^\star(\cdot|h)}[\widetilde{U}(a; T, \kappa)] \ge 0. \tag{18}$$

Its expectation is exactly $\Delta_{\text{ctx}}(h) = 0$, so $r_{\pi^\star}(\kappa) = 0$ for $\mu(\cdot \mid h)$-almost every κ. For any such context,

$$0 = \sum_{a\in\mathcal{A}_T} \pi^\star(a \mid h) \left[V_F(T, \kappa) - \widetilde{U}(a; T, \kappa)\right].$$

Every summand is nonnegative, so every action receiving positive probability under $\pi^\star$ must lie in $A^*(T, \kappa)$ almost surely. The support of $\pi^\star$ is nonempty; choosing any $a^\star$ in that support proves Eq. 10. For a finite discrete context family with positive mass on each context, "almost surely" reduces to membership in every optimal set.

A.6 Proof of Corollary 1: Context-Ambiguity Regret

Let $q_j = \Pr_{a\sim\pi(\cdot|h)}[a \in A_j^*]$. Optimal actions have zero gap and each nonoptimal action has gap at least γ_j, so

$$\mathrm{Regret}(\pi;\kappa_j) \geq \gamma_j(1-q_j). \tag{19}$$

Because $A_1^* \cap A_2^* = \emptyset$, $q_1 + q_2 \leq 1$. Therefore

$$\begin{aligned}\mathbb{E}_\kappa[\mathrm{Regret}(\pi)] &\geq p\gamma_1(1-q_1) + (1-p)\gamma_2(1-q_2)\\ &= p\gamma_1 + (1-p)\gamma_2 - [p\gamma_1 q_1 + (1-p)\gamma_2 q_2]\\ &\geq p\gamma_1 + (1-p)\gamma_2 - \max\{p\gamma_1, (1-p)\gamma_2\}(q_1+q_2)\\ &\geq \min\{p\gamma_1, (1-p)\gamma_2\},\end{aligned}$$

which proves Eq. 11.

A.7 Proof of Theorem 2: Diagnosis-to-Recourse Loss Decomposition

Add and subtract $V_H^\kappa(h)$ and $V_H^R(h)$:

$$\begin{aligned}V_{\mathrm{ctx}}(h) - V(\widehat{\pi}_R \mid h) = [V_{\mathrm{ctx}}(h) - V_H^\kappa(h)] + \left[V_H^\kappa(h) - V_H^R(h)\right]\\ + \left[V_H^R(h) - V(\widehat{\pi}_R \mid h)\right].\end{aligned}$$

This is Eq. 14. The support term is nonnegative because $\mathcal{A}_{H(h)}(T) \subseteq \mathcal{A}_T$ pointwise in κ. The context term is nonnegative because a context-aware policy restricted to $H(h)$ can imitate every history-only policy in $\Pi_H(h)$ and may additionally condition on κ. The selector term is nonnegative because $\widehat{\pi}_R \in \Pi_H(h)$ while $V_H^R(h)$ is the supremum over that same policy class.

Moreover,

$$\begin{aligned}\mathcal{L}_{\mathrm{support}}(h) &= \mathbb{E}_{\kappa|h}\left[V_F(T,\kappa) - V(H(h);T,\kappa)\right]\\ &= \mathbb{E}_{\kappa|h}\left[G(H(h);T,\kappa)\right].\end{aligned}$$

Hence the trace-level quantities G_r and G_H used by CRPA-R1 are pointwise realizations of the support-deficiency term for singleton-root and responsibility-conditioned supports.

A.8 Support Monotonicity

For $H_1 \subseteq H_2$, $\mathcal{A}_{H_1}(T) \subseteq \mathcal{A}_{H_2}(T)$. Maximizing the same gated value over the larger set cannot decrease value, so $V(H_1;T,\kappa) \leq V(H_2;T,\kappa)$ and $G(H_1;T,\kappa) \geq G(H_2;T,\kappa)$.

B Interventions, Feasibility, and Joint Replay

B.1 Expanded Recovery Model

B.1.1 Representing the Execution and Recovery Locations

This module records the supplied diagnosis alongside the locations that available recovery actions can change. Let the failed execution be $T = (G, X, Y^{(0)})$, where $G = (V, E)$ is an event graph, X records states and actions, and $Y^{(0)}$ is the failed outcome. The set $\mathcal{S}_T$ contains the intervention locations, including recorded events and edges and newly introduced locations such as a downstream check. Let $\mathcal{S}_T^{\mathrm{hist}} \subseteq \mathcal{S}_T$ contain those present in the original execution.

Historical responsibility R_T describes the causal contribution of recorded locations under a specified diagnostic intervention model. We write r_T for the designated root and $\widehat{H}_T \subseteq \mathcal{S}_T^{\mathrm{hist}}$ for locations selected by estimated responsibility. CRPA-R1 supplies the correct root to control diagnosis correctness; Appendix B defines the threshold rule for $\widehat{H}_T$. In the planner–worker example, the planner is the root, while the downstream check is an additional recovery location. Keeping these two sets distinct allows recovery to consider changes that historical attribution alone would omit.

B.1.2 Forming Bundles and Enforcing Constraints

This module determines which actions can be applied together under the current execution conditions. Candidates include root repair, edge modification, downstream validation, rerouting, replanning, and orchestration changes. A validation guard checks a propagated state before final acceptance and contains a detected violation. A bundle a is one action or a combination of actions; $\mathrm{surface}(a) \subseteq \mathcal{S}_T$ lists the locations it changes. The finite bundle universe $\overline{\mathcal{Q}}(T)$ is specified before outcome evaluation and shared across comparisons.

Let $\kappa = (P, S, B, M, C, \Gamma)$ describe current permissions, safety constraints, budget, candidate availability, costs, and replay/compatibility conditions. The admitted bundles form

$$F(T, \kappa) = \{a \in \overline{\mathcal{Q}}(T) : \Phi(a; T, \kappa) = 1\}. \tag{20}$$

The gate Φ requires actions to be available, permitted, and supported by execution; their joint changes must be safe, compatible, and within budget. For example, two individually permitted edits cannot be combined if they make conflicting writes. Admission also requires a valid recovery estimate under the replay protocol. Appendix B specifies the gates and four evaluated budgets.

Abstention $a_\perp$ has zero value and changes no location. Define $\mathcal{A}_T = \overline{\mathcal{Q}}(T) \cup \{a_\perp\}$ and extend utility to rejected requests by

$$\widetilde{U}(a; T, \kappa) = \begin{cases} U(a; T, \kappa), & a \in F(T, \kappa), \\ 0, & a \notin F(T, \kappa), \end{cases} \qquad \widetilde{U}(a_\perp; T, \kappa) = 0. \tag{21}$$

This extension lets the theory evaluate the same request when its feasibility changes across contexts. The search selects only admitted bundles or abstention, and records rejected requests separately. The module therefore prevents unavailable or incompatible proposals from being treated as executable recovery choices.

B.1.3 Scoring and Selecting Recovery Bundles

This module gives admissible bundles a common score so that changes at different locations can be compared. We use the Lower Confidence Bound (LCB) on replay recovery rate,

$$U(a; T, \kappa) = \mathrm{LCB}_\alpha \left[\rho\Big(Y^{(a)}\Big) \mid T, \kappa \right], \tag{22}$$

where $\rho(Y^{(a)}) \in \{0, 1\}$ indicates whether the original task succeeds after jointly applying the bundle. The LCB accounts for uncertainty in a success rate estimated from a limited number of replays. CRPA-R1 uses the one-sided exact Clopper–Pearson bound with $\alpha = 0.05$, 12 planned paired replays, and at least eight valid pairs per estimate. Selection solves

$$a_F \in \arg \max_{a \in F(T,\kappa) \cup \{a_\perp\}} \widetilde{U}(a; T, \kappa), \qquad V_F(T, \kappa) = \widetilde{U}(a_F; T, \kappa). \tag{23}$$

Safety and budget are constraints; recovery LCB is the objective. A fixed rule breaks ties using collateral damage and costs (Appendix B). The resulting Full-Candidate Replay Oracle, abbreviated to Full, selects the bundle with the best recorded value in the admitted candidate space. This module supplies a common offline reference for comparing recovery choices within the specified constraints.

B.1.4 Auditing the Loss from Location Restrictions

This module measures how much recovery value is excluded when interventions are limited to diagnosed locations. For allowed locations $H \subseteq \mathcal{S}_T$, let $\mathcal{A}_H(T) = \{a \in \mathcal{A}_T : \mathrm{surface}(a) \subseteq H\}$. We call H the intervention support: it is simply the set of locations where changes are allowed. Its best value and the loss relative to Full are

$$V(H; T, \kappa) = \max_{a \in \mathcal{A}_H(T)} \widetilde{U}(a; T, \kappa), \qquad G(H; T, \kappa) = V_F(T, \kappa) - V(H; T, \kappa). \tag{24}$$

Strict-Root uses $H = \{r_T\}$, and the Responsibility-Conditioned Frontier on $\widehat{H}_T$ (RCF-H) uses $H = \widehat{H}_T$. Their values are V_r and V_H, with losses

$$G_r = V_F - V_r, \qquad G_H = V_F - V_H. \tag{25}$$

Both comparisons optimize exhaustively within their allowed locations. A positive G_r therefore means that even the best available root repair loses value, rather than that a weaker search rule was used at the root.

Definition 1 (Responsibility–Recourse Separation). Let $A^*(T,\kappa) = \arg\max_{a\in\mathcal{A}_T} \widetilde{U}(a;T,\kappa)$. Responsibility-derived support H_T exhibits RRS when

$$A^*(T,\kappa) \cap \mathcal{A}_{H_T}(T) = \emptyset, \tag{26}$$

or equivalently $G(H_T;T,\kappa) > 0$: every best recovery bundle requires a location outside that support.

A root is repairable when a non-abstention root-only bundle is feasible. With positive Full value, separation can arise because the root is infeasible (feasibility-induced) or because its best feasible repair is less effective (dominance-induced). The latter is our central test. Other cases are root-optimal or tied recovery and no positive-value recovery ($V_F = 0$). Distinguishing these outcomes makes the audit explain whether an alternative location is necessary for feasibility or beneficial despite an executable root repair.

B.2 Responsibility-Based Intervention Support

Let $\widehat{R}_T(s)$ estimate historical responsibility at location s, and let $\mathrm{LCB}_T(s)$ be its diagnostic confidence bound. The responsibility-based support is

$$\widehat{H}_T(\epsilon_R;\delta_R) = \left\{ s \in \mathcal{S}_T^{\mathrm{hist}} : \begin{array}{c} \mathrm{LCB}_T(s) > \delta_R, \\ \widehat{R}_T(s) \geq \max\limits_{s'\in\mathcal{S}_T^{\mathrm{hist}}} \widehat{R}_T(s') - \epsilon_R \end{array} \right\}. \tag{27}$$

The primary comparison uses $\delta_R = 0.10$ and $\epsilon_R = 0.05$. This rule retains sufficiently supported locations whose estimated responsibility is close to the maximum. Its diagnostic confidence bound is distinct from the recovery-outcome LCB used to rank interventions. The support is specified before recovery outcomes are evaluated.

B.3 Feasibility and Budgets

Feasibility-Gated Joint Replay Search evaluates a shared candidate set through three eligibility stages:

1. Action admission. Permission, safety, replayability, scorer availability, intervention granularity, and execution support must pass. Unsupported action types and failed replay-drift prechecks are recorded as rejected.
2. Bundle admission. All members must be admitted, their simultaneous writes must be compatible, and the bundle must satisfy the active budget. B1–B4 respectively bound (modified surfaces, dispatch-cost proxies) by $(1,2)$, $(1,4)$, $(2,6)$, and $(3,10)$. Recovery value, token proxies, and time proxies do not enter this budget gate.
3. Estimate eligibility. Each bundle receives 12 planned paired replays. Runtime-drift pairs are invalid and excluded from estimation; at least eight valid pairs are required. A candidate-availability mask then determines which eligible bundles enter a comparison.

All rejected and inconclusive proposals remain in the accounting. If a support has no eligible bundle, its selected action is abstention and its value is zero. Historical responsibility thresholds, $\delta_R = 0.10$ and $\epsilon_R = 0.05$, define $\widehat{H}_T$ separately from these feasibility conditions.

B.4 Recovery Value and Tie-Breaking

The primary value is the one-sided exact Clopper–Pearson Lower Confidence Bound on binary recovery, with $\alpha = 0.05$. For k successes among n valid paired replays, the value is zero when $k = 0$ and otherwise the α quantile of $\mathrm{Beta}(k, n-k+1)$. Safety and the active budget are hard constraints.

Eligible bundles are ordered by recovery LCB descending, followed by collateral damage, modified-surface count, dispatch-cost proxy, rendered-plus-generated token proxy, and time-cost proxy ascending, then lexicographic bundle identifier. Only the first coordinate defines the reported scalar value; the remaining coordinates resolve equal values. Bundle size and action index are excluded from the tie-breaking suffix. The auxiliary Pareto record uses recovery value $U(a)$, collateral damage $D(a)$,

and the recorded five-coordinate cost vector $C(a)$: dominance requires no worse value, damage, or cost coordinate and a strict improvement in at least one. Pareto membership does not change primary LCB ranking.

B.5 Execution and Comparison

All members of a bundle are applied simultaneously to the same failed trace before a unified replay, preserving their interaction effects. Failure of any hard gate makes the whole bundle infeasible. The comparison shares the trace, candidates, gates, compatibility rules, budgets, paired seeds, scorer, tie-breaking, and missingness conventions. Full, Strict-Root, and RCF-H differ only in permitted intervention surfaces. The replay record is generated once and reused for budget and coverage comparisons.

The Full-Candidate Replay Oracle selects after observing replay outcomes and defines the attainable value within the specified candidate set. Exhaustive selection within each restricted support isolates the value excluded by that support. Evaluating an outcome-blind selector under a deployment budget requires the separate protocol in Appendix G.

C CRPA-R1 Construction

CRPA-R1 tests whether restricting interventions to a correctly diagnosed, repairable root can exclude better recovery. The suite deliberately constructs 144 traces without external service or model calls. The historical root is known by construction. Table A1 separates its two RRS mechanisms from root-optimal and abstention controls. B4/Q100 uses the widest budget and full candidate coverage; the first, second, and fourth cells contain the 96 root-feasible traces.

Table A1: CRPA-R1 construction. Each cell isolates a specified relation between root repair, feasible alternatives, and abstention.

Cell	n	Role
Root feasible and optimal	24	Positive controls where root support reaches the full value.
Root feasible and dominated	48	A feasible downstream guard has greater recovery value.
Root inadmissible	24	Positive-value recovery remains available off root.
Cost-inversion boundary	24	Root dispatch requirements and plan replay drift vary; B4/Q100 yields a tie.
No feasible recovery	24	Negative controls for abstention.

In the 48 dominance cases, root restore edits the corrupted planner surface. A downstream guard checks propagated state before final acceptance and contains a detected violation. Both candidates modify one surface, pass the same feasibility gates, and have matched synthetic costs: one dispatch, 139 rendered-input tokens, 34 generated tokens, and 0.1 seconds. Edge blocking/substitution is permission-rejected in this cell; lower-valued reroute, plan, and orchestrator candidates remain feasible. The complete study contains 1,440 declared actions, 6,456 compatible bundles, 77,472 planned paired replays, and 73,152 valid paired replays. Candidate generation, evaluation rules, and analysis specifications preceded access to outcomes.

The guard recovers three more replays than root restore in each dominance case: 11/12 versus 8/12, a 25-percentage-point improvement. Its LCB exceeds root restore by 0.270457 (0.661319 versus 0.390862). Root-following coincides with the exhaustively optimized Strict-Root Frontier in all 11,520 evaluated budget/coverage/candidate-subset/seed combinations. Thus, the controlled gap reflects excluded intervention locations rather than weaker selection within root support. The 48/48 repeated wins establish this constructed mechanism; they do not estimate its frequency in natural failures.

D SENSITIVITY, SELECTION, AND EARLIER CONTROLLED EVIDENCE

D.1 GRAPHICAL VIEW OF THE PRIMARY CONTRAST

This comparison tests whether off-root recovery can outperform an executable root repair in the 96 repairable-root traces and their 48-case dominance subset. At B4/Q100, both arms share gates, costs, paired seeds, and exhaustive search within allowed locations. Figure A1 shows that allowing off-root actions improves 48 of the 96 repairable-root traces without reducing recovery on the others. In each dominance case, downstream validation recovers three more of 12 replays than root restore, a 25-percentage-point gain under matched costs. Correct attribution and an editable root therefore do not guarantee that the best recovery acts at that root.

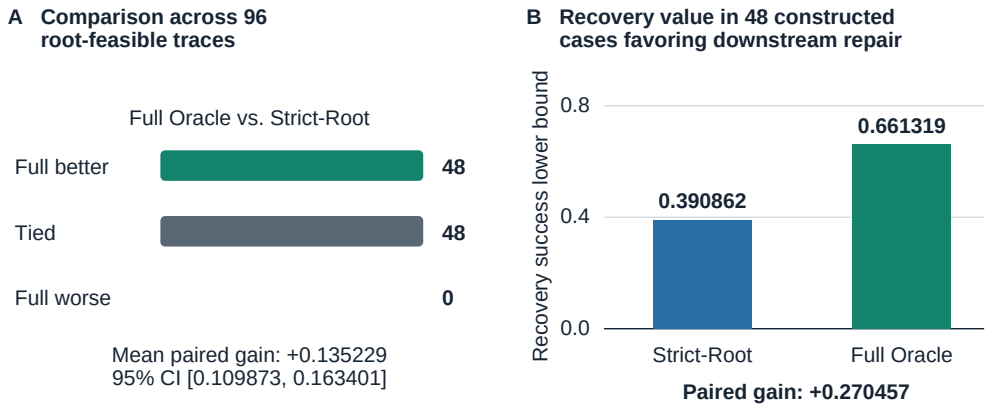

Figure A1: CRPA-R1 with repairable roots. Left: Full wins on 48 of 96 traces and ties on the others. Right: recovery LCBs in the 48 dominance cases, which share one constructed mechanism.

D.2 CRPA-R1 SENSITIVITY

These post-result analyses test which action families sustain the CRPA-R1 gap and whether it depends on the recovery objective or budget, reusing the replay record without new execution. Family exclusions remove all bundles containing that family, including mixed bundles, before both arms reselect. Table A2 reports principal conditions from 19 non-primary variants, with 20,000 trace-level bootstrap repetitions. Objective variants replace the first ranking coordinate and retain the tie-breaking suffix; CVaR-25/50 average the worst 25%/50% of valid binary replay deltas.

Table A2: CRPA-R1 sensitivity: Full minus Strict-Root mean value and 95% bootstrap interval endpoints. Repairable roots number 96 except at B1 ($n = 72$). The final column gives the dominance mean. − denotes family removal; edge/reroute/plan are each removed separately.

Condition	All traces ($n = 144$)			Repairable roots			Dominance ($n = 48$)
	Mean	95% CI		Mean	95% CI		
		Lower	Upper		Lower	Upper	
Full B4/Q100 LCB	0.207622	0.166200	0.251019	0.135229	0.109873	0.163401	0.270457
− guard	0	0	0	0	0	0	0
− edge/reroute/plan	0.207622	0.166200	0.251019	0.135229	0.109873	0.163401	0.270457
− orchestrator	0.190182	0.153811	0.228605	0.135229	0.109873	0.163401	0.270457
Root + guard only	0.188674	0.152805	0.226585	0.135229	0.109873	0.163401	0.270457
Point recovery	0.240741	0.187500	0.298611	0.125000	0.101563	0.151042	0.250000
CVaR, worst 25%	0.351852	0.291667	0.414352	0.333333	0.270833	0.402778	0.666667
CVaR, worst 50%	0.314815	0.258102	0.372685	0.250000	0.203125	0.302083	0.500000
Pareto then LCB	0.207622	0.166200	0.251019	0.135229	0.109873	0.163401	0.270457
Worst valid seed	0.090278	0.048611	0.138889	0	0	0	0
B1 LCB	0.293774	0.252062	0.336446	0.180305	0.150254	0.210355	0.270457
B2 LCB	0.188674	0.152805	0.226585	0.135229	0.109873	0.163401	0.270457
B3 LCB	0.203423	0.162914	0.245850	0.135229	0.109873	0.163401	0.270457

Removing the guard eliminates the support gap; retaining only root and guard preserves its full root-feasible component while reducing the all-trace mean by 0.018948. Orchestrator removal reduces only the all-trace mean, by 0.017440; edge, reroute, and plan removal leave the summaries unchanged. Point recovery, both CVaR objectives, and Pareto filtering retain a positive root-feasible gap, whereas worst-valid-seed makes this cohort tie. The 0.135229 root-feasible LCB gap persists across B2–B4. These findings identify guard availability as the mechanism in this construction and show its dependence on the chosen recovery objective. Objective scales differ, and B1's smaller root-feasible cohort prevents its mean difference from isolating a budget effect.

The selection records provide additional detail about the preceding objective variants. Point recovery preserves both arms' selections in 144/144 traces and yields 48/96 root-feasible wins. CVaR-25 changes one Full selection (143/144 agreement), whereas CVaR-50 and Pareto filtering before LCB each preserve both arms in 144/144 traces. Both CVaR objectives yield root-feasible W/T/L of 48/48/0. Worst-valid-seed takes the minimum valid replay delta and agrees with the LCB Full selection in 109/144 traces.

D.3 Full-Support Selection

This comparison tests how selection rules affect recovery under identical candidate access. All four rules use the same feasible B4/Q100 candidates and all 144 traces, with zero-valued abstention. Minimum-cost selects the least-cost feasible bundle; seeded random draws a feasible bundle with seed 17011; the metadata heuristic ranks information available before execution. Full uses replay outcomes. Table A3 therefore compares selection rules with common candidate access and a common denominator.

Table A3: Full-support selection on B4/Q100, using all 144 traces and zero-valued abstention.

Rule	Mean recovery LCB
Minimum-cost feasible	0.339921
Seeded random feasible (seed 17011)	0.333162
Outcome-blind metadata heuristic	0.421524
Full-Candidate Replay Oracle	0.597602

The metadata heuristic improves mean recovery LCB by 0.081603 over minimum-cost selection and 0.088362 over random selection. Full exceeds these three rules by 0.257681, 0.264440, and 0.176078, respectively. The differences show that common candidate access does not determine the selected recovery value. Full's additional outcome information and evaluation cost make this a descriptive comparison rather than a gain established under matched deployment resources.

D.4 Computational Coverage

This analysis checks how much replay workload smaller candidate prefixes save while retaining every action stratum. At B4, Q25/Q50/Q75/Q100 are nested prefixes within outcome-blind strata defined by action-type tuple, surface-type tuple, bundle size, and cost quartile. Each nonempty stratum retains at least one bundle. Table A4 compares realized bundle coverage and scorer workload using the existing replay record.

Table A4: Realized B4 coverage and approximate reduction in scorer evaluations relative to Q100. The reduction is computed as 100% minus realized coverage.

Prefix	Eligible bundles retained (%)	Scorer reduction (%)
Q25	75.64	24.36
Q50	86.27	13.73
Q75	100	0
Q100	100	0

Q25 reduces scorer evaluations by approximately 10.63 percentage points more than Q50, whereas Q75 saves no evaluations relative to Q100. Retaining every nonempty stratum raises realized coverage above the nominal prefix fraction. These counts measure replay workload: dispatch, token, and time fields are synthetic cost proxies, and selection wall time measures the analysis process. End-to-end deployment efficiency has not been measured.

D.5 Earlier Controlled Studies

This comparison distinguishes feasibility-induced gains from gains over executable root repairs in the earlier studies. The eight-scenario study (identifier: C3) compares Full with root repair across four budgets (32 cells), using scenarios as the inference unit. The earlier held-out study retrospectively evaluates 160 traces from eight source/template blocks across four budgets (640 cells), weighting the eight blocks equally. Table A5 summarizes both experiments under their respective aggregation schemes.

Table A5: Earlier controlled evidence. Gains compare Full with root repair; cell counts distinguish feasibility-induced separation from dominance over feasible root repair. W/T/L uses scenarios for C3 and equally weighted source/template blocks for the 160-trace study.

Quantity	C3	160-trace study
Mean recovery LCB gain	0.371445	0.365923
95% bootstrap interval	[0.138717,0.610937]	[0.130118,0.594945]
W/T/L	5/3/0	5/3/0
Root-feasible optima/ties (cells)	11	220
Root infeasible; off-root recovery (cells)	17	340
No feasible recovery (cells)	4	80
Feasible-but-dominated root (cells)	0	0

Full improves mean recovery LCB over root repair by 0.371445 in C3 and 0.365923 in the 160-trace study. In the latter, responsibility-support gaps occur in 65/160 traces and oracle-root decision divergence in 100/160. Both studies obtain their positive differences where root repair is infeasible; neither contains a cell in which another action outperforms a feasible root repair. They establish feasibility-induced separation and motivate CRPA-R1's separate test of feasible-root dominance.

The historical candidate record contains 84 actions (53 feasible, 31 rejected), 546 bundles (510 eligible, 36 inconclusive), and 6,552 planned replays (6,300 valid, 252 invalid). Of the valid executions, 6,048 belong to frontier-eligible bundles; the rest belong to inconclusive bundles below the valid-pair threshold. Bundle sizes are one ($n = 53$), two ($n = 173$), and three ($n = 320$). These counts refer to distinct action, bundle, and execution accounting levels.

D.6 C3 Sensitivity

This retrospective analysis checks whether C3's feasibility-induced gap depends on responsibility thresholds, allowed action families, or the ranking statistic, without new replay. Table A6 reports the Area Under The Curve (AUC) of recovery value against budget, using trapezoidal integration over B1–B4 and equal scenario weighting; selection agreement uses the 32 scenario-budget cells.

Table A6: Retrospective C3 sensitivity without new replay. The first three rows summarize eight-scenario mean budget-AUC contrasts; agreement counts use 32 cells. The comparison reference for the candidate-access sequence in row 2 is unspecified in the available record, so the sequence is retained only as descriptive data.

Condition	Result
Responsibility $\epsilon_R = 0.05/0.10/0.20$	0.266135/0.087028/0.016791
Root-only / root+edge / +guard / full	0/0/0.264064/0.266135
Full with guard family removed	0
Point versus LCB selection	32/32 identical
Worst-seed versus LCB selection	23/32 identical

Broadening responsibility support from $\epsilon_R = 0.05$ to 0.20 reduces the mean budget-AUC gap by 0.249344. Guard removal eliminates the historical C3 gap. Point recovery preserves all 32 LCB selections, whereas worst-seed ranking changes nine. These comparisons show dependence on guard availability and on the ranking criterion; they add no evidence for feasible-root dominance. We draw no directional comparison from the candidate-access sequence because its reference is unspecified.

E AGENTRX EXECUTION AND RESULTS

The AgentRx comparison tests whether recovery depends on the continuation rule when attribution is held fixed. It evaluates 27 previously examined τ-bench retail calibration cases (Barke et al., 2026; Yao et al., 2025) over three seeds. Two of the original 29 cases were used for interface development and excluded before this run. A local Phi-3.5 Mini Instruct adapter produces A2P-style attribution from the failed trace, without gold labels or scorer fields. Both arms share its predictions: 81 attribution calls yield 81 paired recovery units, retaining parse failures as abstentions. Retail-domain knowledge informs calibration and case selection, so this is a within-domain test of fixed-attribution rule dependence, rather than independent external generalization or an official A2P reproduction.

Root-following cuts the trajectory at the predicted attribution step. The feasibility-aware rule scores legal steps using Table A7, selects the highest score, and breaks ties in favor of the earliest step. If no rule fires, it uses the middle assistant step. Both cuts share the continuation policy and model, tools, initial database state, and scorer. This changes the downstream cut rule; it does not instantiate CRPA-R1's guard comparison.

Table A7: Priority scores for the AgentRx feasibility-aware cut rule. Only legal steps are eligible for selection.

Score	Trigger
9.0	Available-count inconsistency with preceding tool output.
8.8	Rejected tool before modification, return, or exchange.
8.6	Order-level cancellation following a single-item request.
8.0	Consequential tool use without explicit confirmation.
7.0	Tool-output error.
4.0	Tool use before authentication.
3.0	Non-authentication, non-consequential tool fallback.

E.1 PAIRED RECOVERY WITH SHARED ATTRIBUTION

This paired comparison tests whether feasibility-aware continuation increases executable recovery under shared attribution, using 27 cases and three seeds (Table A8). Recovery rates pool the 81 paired units; the confidence interval, W/T/L, and sign tests use cases after averaging the three seeds. Attribution accuracy includes abstentions in the denominator of all 81 calls.

Table A8: AgentRx recovery and shared-attribution diagnostics on 27 calibration cases with three seeds. The recovery gain compares the feasibility-aware rule with Root-following.

Quantity	Value
Root-following pooled executable recovery	0.506173
Feasibility-aware pooled executable recovery	0.703704
Case-averaged recovery gain	0.197531
95% case-bootstrap interval for gain	[0.061728,0.358025]
Case-level W/T/L	6/21/0
Exact-sign p, one-sided / two-sided	0.015625 / 0.03125
Attribution parse coverage	0.975309
Attribution accuracy: exact step	0.074074
Attribution accuracy: within ± 3 steps	0.444444
Attribution accuracy: agent	0.691358

The feasibility-aware rule improves executable recovery by 19.75 percentage points; six seed-averaged cases improve and none decline. The paired design assigns the difference to the cut rule under shared attribution and continuation conditions. Low exact-step attribution accuracy limits this finding to rule dependence rather than dominance over a correctly identified root. Calibration and common rule design further limit the conclusion to these retail cases.

E.2 Earlier AgentRx Cohort

This historical comparison checks whether the same recovery-rule contrast appears in the earlier 29-case cohort. Table A9 preserves this result separately from the 27-case, three-seed comparison. The feasibility-aware rule improves recovery by 3.45 percentage points, with only one case improving. The sign test does not provide evidence of a systematic gain, so this remains descriptive historical evidence.

Table A9: Historical AgentRx comparison on 29 cases. This cohort is not pooled with the later three-seed evaluation.

Quantity	Value
Root-following executable recovery	0.689655
Feasibility-aware executable recovery	0.724138
Recovery gain (percentage points)	3.45
W/T/L	1/28/0
One-sided exact-sign p	0.500

F Statistics and Reproducibility

Primary thresholds, candidate generation, and decision rules were specified before evaluation outcomes; the C3 and CRPA-R1 sensitivity analyses are retrospective. Inference units are paired traces for CRPA-R1, scenarios for C3, eight source/template blocks for the 160-trace study, and cases after seed averaging for AgentRx. Replay seeds estimate recovery within these units. Bootstrap resampling uses the corresponding unit; W/T/L counts positive, zero, and negative paired differences after aggregation. CRPA-R1 uses 20,000 paired bootstrap repetitions.

F.1 Multiplicity Correction

This sensitivity analysis checks whether the two primary contrasts remain supported after correcting for multiple comparisons. The preregistration specified Holm correction (Holm, 1979) for the primary contrast family, but the original analyzer omitted it. We therefore retain the original unadjusted results and report the subsequent sensitivity calculation in Table A10. This calculation uses two-sided exact paired sign tests with a two-member Holm family and conservative 97.5% bootstrap intervals.

Table A10: Subsequent multiplicity sensitivity for the two primary CRPA-R1 contrasts. Both compare Full with Strict-Root; intervals refer to the mean recovery-LCB difference.

Cohort	Holm-adjusted p	97.5% bootstrap interval
All 144 traces	8.47×10^{-22}	$[0.160614, 0.258090]$
96 repairable-root traces	7.11×10^{-15}	$[0.104239, 0.166218]$

Both intervals remain above zero after the correction. These tests compare nested frontiers on a constructed suite and do not estimate the prevalence of the mechanism in natural failures. The dominance cell's original zero-width interval, $[0.270457, 0.270457]$, follows from identical template-imposed effects.

F.2 Reproduction Materials and Reporting

The anonymized supplement contains per-trace outcomes, candidate and action specifications, paired seeds, protocol, generator and analysis specifications, and scripts that reconstruct the 144-trace split, frontier values, paired contrasts, coverage, and primary result summaries. Attribution prompts and model configurations accompany the AgentRx procedure. The AI Use Statement following the references discloses assistance with manuscript preparation. The local model and inference procedure used within the experiment are specified in Appendix E.

G Future External and Outcome-Blind Evaluation

External validation should map an attribution method's agent, step, or ranked set to a common event representation before revealing recovery outcomes. Gold roots, replay scores, and scorer fields remain hidden during inference and mapping; empty, unmapped, and parse-failed outputs remain abstentions. The resulting predicted-root and responsibility-support comparisons share the original candidates, gates, budgets, paired seeds, joint replay, and scoring, with Full as the offline reference. Attribution coverage and accuracy should be reported separately from recovery value and support gaps on fresh domains.

An outcome-blind full-support selector can rank bundles using failed-trace features, action specifications, current execution constraints, and cost coordinates. Replay outcomes, gold responsibility, verifier outputs, and test-family identifiers are excluded; source/template cross-fitting separates development and test families. A preregistered top-k policy then applies the same gates, with abstention when no eligible action remains. Comparison against root-only and responsibility-conditioned policies should match realized replay counts, scorer calls, rendered and generated tokens, dispatches, and wall-time caps. Reports should include executable recovery, regret to Full, coverage and abstention, invalid proposals, and calibration, retaining replay failures. Hyperparameters and stopping rules must be set before opening fresh test outcomes. These protocols address external generalization and practical selection beyond the current controlled and calibration evidence.