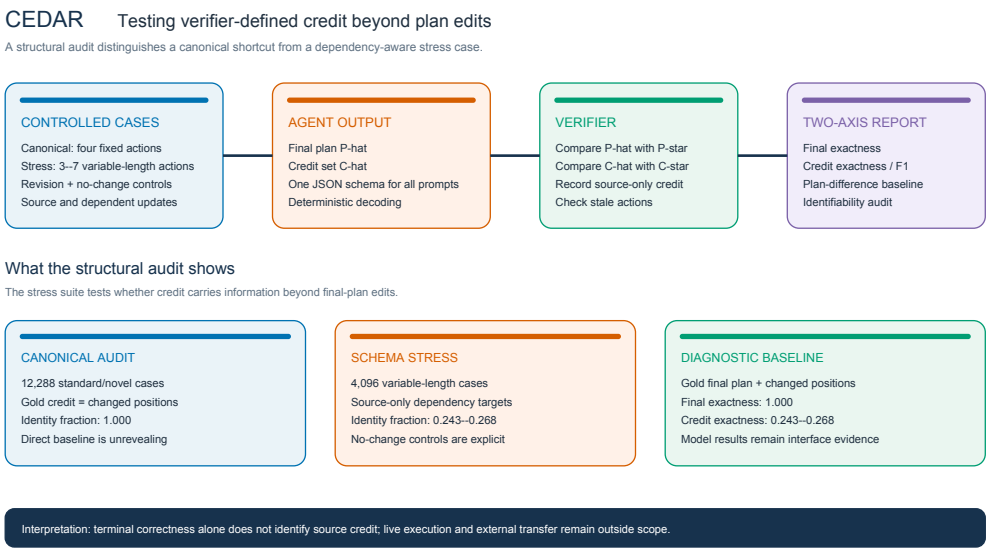
CEDAR: An Identifiability Audit for Verifier-Defined Credit in Plan Revision

Anonymous authors
Paper under double-blind review

Abstract

Multi-step agent benchmarks often score only the final plan, even when a late instruction changes dependencies among earlier actions. We ask whether verifier-defined credit supplies information beyond the edits between an initial and gold final plan. In 12,288 cases from CEDAR's original four-action schema, credit equals the changed-position set in every case, so a direct plan-difference baseline fully recovers the target. We then test a 4,096-case variable-length schema with source-only credit, dependent edits, and no-change controls. Its identity fraction is 0.253662: all 1,039 controls satisfy the identity by construction, while all 3,057 revisions do not; an oracle reaches 1.000 on both final-plan and credit metrics. A bounded check with Qwen2.5-7B-Instruct covers two case seeds and 256 cases per condition. We compare its emitted credit sets with edit sets implied by its predicted plans. The audit separates final-plan scoring from source-credit localization and gives a four-step procedure for testing whether intermediate benchmark labels add information beyond serialized outputs.



Graphical abstract. CEDAR first audits whether credit labels are already determined by initial-to-final plan edits, then evaluates a variable-schema stress suite that separates source actions from dependent edits.

1 Introduction

Interactive agents are commonly evaluated by whether a final answer or plan satisfies a task. This terminal score is useful, but it does not reveal which earlier actions deserve credit when a later instruction changes a dependency. A benchmark may therefore expose a credit field

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.

without testing whether the field contains information that is unavailable from the final plan and the initial plan. This distinction matters for diagnosis, reranking, and training: a model can reproduce a post hoc edit pattern while failing to identify the source action that caused downstream changes.

We study this issue in CEDAR[1], a controlled benchmark for plan revision with verifier-defined credit targets. Our testable question is: when a late instruction changes a multi-step plan, does an explicit credit target provide information beyond the positions that changed between the initial and gold plans? The closest competing explanation is a direct plan-difference rule. Given an initial plan and a correct final plan, this rule marks every changed position. If the benchmark's credit target always equals that set, terminal-plan supervision and credit supervision are not independently identified by the task interface. The audit suggests a test for other benchmarks that score intermediate labels from serialized artifacts: check whether a simpler post hoc rule already fixes the label. We do not test that transfer on an external benchmark here.

We find that this is exactly what happens in the original four-action schema. Across standard and held-out surface families, including wording changes and position permutations, the gold credit set equals the changed-position set in all 12,288 audited cases. We therefore treat the earlier canonical prompt comparisons as measurements of the original interface, not as evidence for an independent credit channel. We then construct a variable-length stress suite in which a source action receives credit while dependent actions also change. The suite includes single- and multi-source revisions, variable plan lengths, and no-change controls. Its pooled identity fraction is 0.253662 across 4,096 cases, with a per-seed range of 0.235352–0.267578, because the controls remain 1.000 while the revision subset is 0.000.

The resulting paper makes three contributions:

1. We provide a deterministic identifiability audit for verifier-defined credit, together with a direct plan-difference baseline, machine-readable provenance, and a reusable four-step audit protocol for intermediate-label benchmarks.
2. We introduce a schema stress test that separates source-only credit from dependent plan edits and includes explicit no-change controls.
3. We report a bounded open-model stress check on the revised interface, using paired predictions to show when emitted credit labels collapse back to the prediction-derived plan-difference set and to make clear that revision itself remains unsolved in this run.

The contribution is an audit of what the benchmark measures. That audit is the main result. The bounded model check is included only to show how the revised interface behaves on concrete outputs after the structural shortcut is removed. The paper is not a causal attribution method, a live-tool evaluation, or evidence of transfer to open-world agent tasks. The authored release includes the repository link together with the executable audit scripts.

2 Relation to Agent Evaluation

Benchmarks such as PlanBench (Valmeekam et al., 2022), SWE-bench (Jimenez et al., 2024), WebArena (Zhou et al., 2023), and OSWorld (Xie et al., 2024) evaluate planning, software, web, or computer-use behavior through task-level outcomes. Tool-agent interaction suites such as tau-bench (Yao et al., 2024), AppWorld (Trivedi et al., 2024), and ToolSandbox (Lu et al., 2024) add stateful tools, user turns, and structured environments. These benchmarks establish why terminal success is a practical evaluation axis, but they do not by themselves establish that a verifier-defined intermediate label is identifiable from the available observations.

Agent Planning Benchmark (Sun et al., 2026) provides planning-specific diagnostics across domains and tests whether its feedback improves downstream execution. PlanBench-XL (Liu et al., 2026) probes long-horizon planning with tool retrieval and runtime dis-

[1]The supplementary source archive submitted with this paper contains the reproduction code.

ruptions. CEDAR instead tests whether its verifier's source-credit label is recoverable from edits between the initial and gold plans. We have not audited either external benchmark's labels, so the present results do not establish that the same shortcut occurs there.

CEDAR focuses on source-credit recoverability from plan edits. Its credit target is not inferred from a human annotation of an execution trace. It is generated by a deterministic verifier from the source positions specified by the revision and is scored separately from the final plan. This design makes it possible to test a necessary prerequisite for credit-based diagnosis: the target must differ from a trivial function of the final plan and the initial plan. The audit is complementary to execution-centric evaluation. It identifies an interface property before a model or a downstream repair procedure is interpreted.

This question is related to dataset artifacts in natural language inference. Hypothesis-only classification exposed annotation cues that predict labels without the premise (Gururangan et al., 2018), while HANS tested models on cases where common syntactic shortcuts fail (McCoy et al., 2019). CEDAR applies the same concern to an intermediate verifier label in plan revision. Its canonical shortcut is an exact equality between two evaluator-defined sets, and the stress suite changes that equality by separating source positions from dependent edits. The audit tests label construction before drawing conclusions from model scores.

Recent work moves closer to this question by studying failure diagnosis or credit tracing over agent trajectories. AgentRx (Barke et al., 2026) diagnoses manually annotated failed trajectories; Long-Horizon Agent Trajectory Attribution (Chen et al., 2026) studies fine-grained attribution on annotated long-horizon trajectories; Verifier-Instrumented Credit Tracing (VICT) (Li et al., 2026) traces verifier-backed credit in agent reinforcement learning; and Causal Agent Replay (Shah, 2026) studies replay-based causal analysis. CEDAR differs along the task, method, data, and evidence axes: it audits serialized plan-revision labels before execution, uses deterministic source/dependent constructions instead of trajectory annotations or replay interventions, and provides a direct structural baseline plus an evaluator-only information-boundary test. We therefore do not compare the present controlled numbers with those execution-centric results; the purpose here is to test whether a credit field is separately identified by its own benchmark interface. The proposed broader use is methodological: benchmark-side label identifiability can be checked before interpreting an intermediate supervision target scored from serialized observations. Testing that protocol on other benchmarks remains open.

3 Benchmark and Identifiability Test

3.1 Case representation

Each case contains an initial ordered plan P0, a revision instruction, a gold final plan P-star, and a gold credit set C-star. A model emits a final plan P-hat and, when requested, a predicted credit set C-hat. The model-visible information set is X-model = (initial_instruction, revision_instruction, distractor_instruction, P0). The evaluator-only information set is X-eval = (P-star, C-star, required_updates, forbidden_stale_actions, verifier metadata). The post hoc gold-plan edit set is D-star = D(P0, P-star) = {i : P0[i] != P-star[i]}; the prediction-derived edit set is D-hat = D(P0, P-hat). We define the identity indicator as 1[C-star = D-star] and report its mean as the identity fraction.

The target C-star is a verifier-defined source label. In the canonical schema it contains every changed position. In the stress schema it contains only the source positions named by the revision; dependent actions may change in P-star without entering C-star. No result in this paper assigns causal responsibility to an action. The word credit refers only to this specified verifier target.

3.2 Canonical audit

The canonical audit enumerates two template families and three surface families over four deterministic case-generation seeds, with 512 cases per configuration. It covers the orig-

inal standard surface, held-out wording, position-permuted actions, and novel templates, together with no-change controls. The resulting audit contains 12,288 cases (2 templates x 3 surfaces x 4 seeds x 512 cases). For each case, the script checks target validity, recomputes D-star, and records whether it equals C-star.

Figure 1 summarizes this information split and the separate final-plan and source-credit scoring axes.

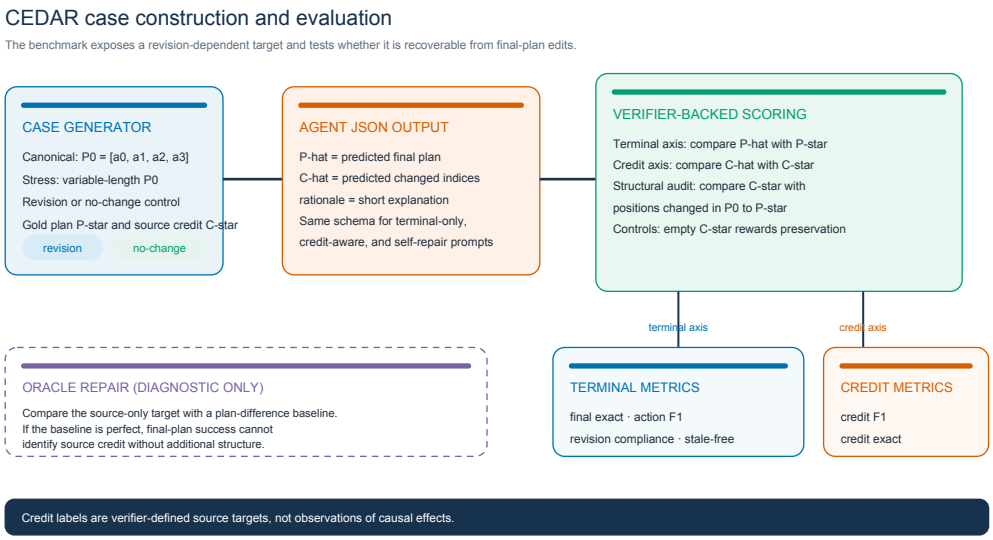


Figure 1: CEDAR separates final-plan exactness from source-credit exactness. The structural audit compares the verifier target with initial-to-gold plan edits before open-model results are interpreted.

3.3 Variable-schema stress suite

The stress generator samples plans with three to seven actions across data, operations, security, and workflow domains. A revision can update one or two source positions and propagate the change to dependent positions. The source positions form C-star; the full set of changed actions forms D-star. Approximately one quarter of the cases are no-change controls with an empty credit set. The generator writes the source and dependent positions into evaluator-only case metadata, so the verifier does not need hidden information at evaluation time. The build_prompt function serializes only the model-visible fields listed above; it excludes the gold plan, credit labels, update lists, stale-action lists, and verifier dictionary. The executable scripts/audit_prompt_surface.py probes all three prompt modes with unique sentinels and checks both sentinel values and evaluator-only field names.

The stress suite contains 1,024 cases for each of four seeds, for 4,096 cases in total. Its controls always satisfy the identity by construction, while revision cases are generated so that source-only credit differs from the full edit set. The direct plan-difference baseline is therefore a diagnostic: it receives the gold final plan, predicts D-star as credit, and tests whether terminal correctness alone identifies the verifier target. Additional structural baselines predict a fixed first position, a random source set, a stale initial plan, or a late repair. These baselines are not agent policies; they expose which metrics are determined by the task specification.

3.4 Metrics and paired transformation

We report final-plan exactness, action F1, credit exactness, credit F1, revision compliance, and stale-action-free rate. All metrics lie in [0, 1]. Final-plan exactness requires the full

predicted plan to equal P-star; credit exactness requires a valid, requested predicted set to equal C-star; credit F1 measures set overlap on those scoreable outputs. Credit F1 is 1 when both sets are empty and 0 when exactly one is empty. An omitted field in plan_only is not an empty prediction, and a malformed or duplicate-index credit field is excluded from the original-credit denominator. Revision compliance checks the required source updates and preservation of unaffected actions. Stale-action-free rate checks that an action invalidated by the revision is not retained.

The gold-plan structural baseline uses D-star and is given P-star; it is not an inference procedure available to the model.

For every model prediction, we create a derived plan-difference output by keeping P-hat unchanged and setting the diagnostic credit field to D-hat. Final-plan metrics must be identical between the original and transformed outputs. We compare credit metrics only for cases with valid, requested original credit, using the same case IDs in both outputs. The derived score on all 256 cases is reported separately and is not itself a paired difference. A gap on matched cases indicates that the emitted credit set is not equivalent to the edits in the emitted plan. When P-hat is wrong, D-hat can also reflect omissions, unrelated changes, or formatting errors, so this is an output-consistency diagnostic, not a pure source-localization estimate. This transformation does not create a stronger model and is not used as a deployed repair procedure.

3.5 Audit protocol

The audit is reusable as a four-step protocol. (1) Enumerate the model-visible and evaluator-only information sets. (2) Recompute the simplest post hoc target functions, including D-star, on every gold case. (3) Generate variable-schema revisions in which source labels and dependent edits differ, together with no-change controls. (4) Evaluate terminal and credit metrics separately, then apply the paired D-hat transformation while checking that terminal metrics are invariant. This protocol distinguishes a label-construction property from model behavior and makes the required evidence explicit before downstream utility is studied.

3.6 Reproducibility and scope

The structural audit and stress generator are deterministic Python programs with unit tests for empty edits, length changes, position permutations, and no-change cases. The prompt-surface audit probes all three prompt modes with 32 generated cases per mode and found no hidden-field leak. The structural report and bounded open-model evaluation were regenerated from frozen manifests and deterministic settings. The six model conditions use Qwen2.5-7B-Instruct with deterministic decoding (do_sample=False) on plan_only, plan_and_credit, and self_repair at case-generation seeds 7 and 19; these seeds define deterministic 256-case stress manifests and are not sampling seeds. Seed 7 contains 77 controls and 179 revision cases, while seed 19 contains 67 controls and 189 revision cases. For each condition, the stored outputs include the case manifest, predictions, semantic metrics, paired plan-difference metrics, and provenance needed to recompute the reported tables. The authored release provides the executable audit scripts together with the records needed to regenerate the reported numbers.

No human ratings, live tool execution, or external transfer set is used. The study is an interface and verifier audit on controlled serialized plans. The scope is stated here because reproducibility and evaluation conditions determine how the results should be interpreted.

4 Results

4.1 The canonical schema identifies credit by construction

Table 1 reports the structural audit. The canonical suite has an identity fraction of 1.000 across all standard and novel configurations. A direct plan-difference output therefore reaches 1.000 credit exactness when supplied with the gold final plan. This is a property of the label construction, not a model result. The stress suite breaks this equality in the

revision subset itself: all 3,057 revision cases have identity 0.000, while the 1,039 no-change controls remain 1.000 by construction. The pooled identity fraction is therefore 0.253662, with per-seed values ranging from 0.235352 to 0.267578.

Figure 2 shows how the controls determine the pooled stress value: all 1,039 controls satisfy the identity, while none of the 3,057 revisions do. The canonical cases all satisfy it as well. This decomposition separates the pooled rate from the result on revision cases.

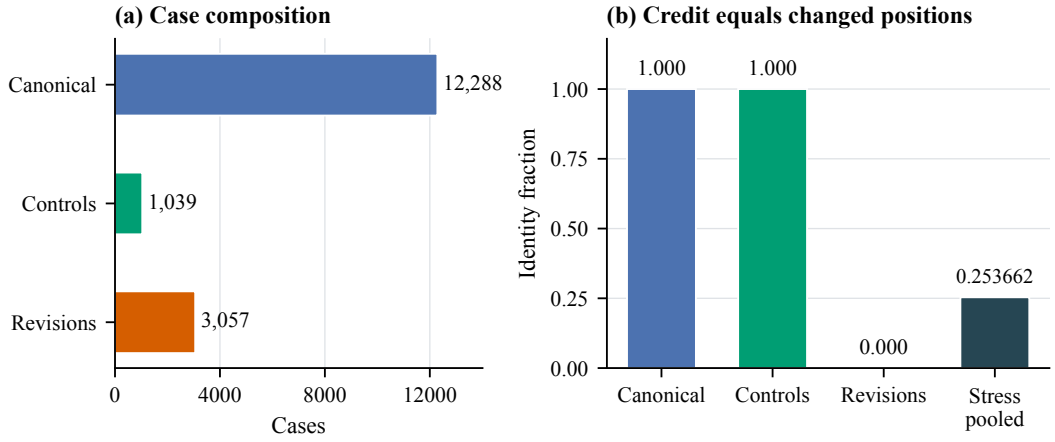


Figure 2: Identity between verifier-defined credit and the set of positions changed between the initial and gold plans. The canonical audit contains 12,288 cases, all with identity. In the 4,096-case stress suite, all 1,039 controls and none of the 3,057 revisions satisfy identity, yielding a pooled fraction of 0.253662. The panels show case composition and identity rates by group; all cases in each group were enumerated, so no uncertainty intervals are shown.

Table 1: Structural identifiability audit.

Suite	Cases	Plan lengths	Identity fraction	Plan difference credit exact ↑	Gold-plan final exact ↑
Canonical standard + novel	12,288	4	1.000	1.000	1.000
Schema stress, pooled	4,096	3–7	0.253662	0.253662	1.000
Schema stress, controls only	1,039	3–7	1.000	1.000	1.000
Schema stress, revision only	3,057	3–7	0.000	0.000	1.000
Schema stress, per-seed range	4,096	3–7	0.235352–0.267578	0.235352–0.267578	1.000

The stress construction also rules out several simpler explanations. With the gold final plan, the changed-position baseline has final exactness 1.000 but credit exactness 0.253662. The control-aware first-position probe reaches credit exactness 0.401123 using the evaluator-only no-change flag; on revisions alone, 604/3,057 cases have source set $\{0\}$. A random source-set probe reaches 0.126709. These values show that the source target is neither the full edit set nor a constant position rule. The late-repair baseline has final exactness 0.253662 and credit exactness 0.000, illustrating that a plan can remain stale while the target itself is well specified.

4.2 A bounded open-model stress check remains secondary evidence

Table 2 tests whether the revised interface exposes plan-credit disagreement in actual outputs; it is not evidence that current models solve CEDAR. Final exactness ranges from 0.145 to 0.301 on the 256-case conditions. Every revision case has final exactness 0.000, although revision-only action F1 ranges from 0.594 to 0.626. The nonzero pooled final scores therefore come entirely from no-change controls. Among the four conditions that request credit, original credit F1 ranges from 0.566 to 0.601; neither plan_only condition has an original-credit score. In seed-7 self_repair, two duplicate-index fields leave 254 scoreable original-credit cases; all other credit-requesting conditions have 256. The derived plan-difference score covers all 256 cases in each condition, including plan_only. The complete denominators and a matched-case comparison are explained in Appendix A. These outputs do not establish that the model recovered source responsibility.

Table 2: Deterministic Qwen2.5-7B-Instruct stress check. All plan and derived-plan scores use 256 cases per condition. Original credit uses 0, 256, or 254 valid cases as described in the text; n/a means that credit was not requested. The derived plan-difference score covers all 256 cases even when original credit is n/a. Revision-only final exactness is 0 in every condition, so nonzero pooled final exactness comes from controls.

Prompt mode	Seed	Final exact ↑	Action F1 ↑	Credit exact ↑	Credit F1 ↑	Plan difference credit exact ↑	Revision compliance ↑
plan_only	7	0.156	0.592	n/a	n/a	0.391	0.301
plan_only	19	0.145	0.583	n/a	n/a	0.281	0.262
plan_and_credit	7	0.301	0.738	0.375	0.586	0.461	0.301
plan_and_credit	19	0.262	0.724	0.266	0.566	0.352	0.262
self_repair	7	0.246	0.674	0.413	0.586	0.418	0.301
self_repair	19	0.223	0.671	0.352	0.601	0.344	0.262

The transformation preserves final exactness and revision compliance because it changes only the credit field. On the same scoreable cases, derived-minus-original credit exactness ranges from -0.008 to +0.086, and credit F1 from +0.141 to +0.231. For seed-7 self_repair, the 256-case derived exactness of 0.418 in Table 2 must not be subtracted from the 254-case original exactness of 0.413; the matched-case values are 0.421 and 0.413. The two credit-requesting modes have nearly equal original credit F1 at seed 7 and reverse order at seed 19; these single deterministic runs do not establish a reliable ranking. No condition achieves exact revision on the revision subset. Because the derived credit set is D-hat, these gaps diagnose consistency between a model's plan and its credit output, not source localization when the final plan is wrong.

4.3 Structural baselines explain the stress metrics

Table 3 compares the non-model strategies on all 4,096 stress cases. The oracle supplies both the gold final plan and the source-only credit target; the changed-position, control-aware, and random-credit probes also receive the gold final plan and differ only in their credit sets. The gold-plan plus changed-positions strategy isolates the exact shortcut removed by the stress suite: it keeps terminal exactness at 1.000 but reaches only 0.253662 credit exactness. The control-aware first-position strategy also receives the evaluator-only no-change flag: it predicts empty credit for 1,039 controls and position zero otherwise. Its 0.401123 credit exactness comprises those controls and 604 of 3,057 revision cases. An unconditional position-zero prediction would instead reach $604/4{,}096 = 0.147461$ credit exactness. These are structural probes, not model-available policies. The stale and late-repair strategies expose the separate cost of terminal failure and stale actions. Late repair copies the initial plan, substitutes only the gold final action at the last position, and marks that position as credit; it therefore leaves earlier dependencies unrepaired.

Table 3: Structural stress baselines. The control-aware first-position strategy uses the evaluator-only no-change flag; it is not a model-available baseline.

Strategy	Final exact ↑	Credit exact ↑	Credit F1 ↑	Revision compliance ↑	Stale-action-free ↑
Oracle final plan + source credit	1.000	1.000	1.000	1.000	1.000
Stale initial plan	0.253662	0.253662	0.253662	0.253662	0.253662
Gold final plan + changed positions	1.000	0.253662	0.751221	1.000	1.000
Control-aware first position	1.000	0.401123	0.526449	1.000	1.000
Random source set	1.000	0.126709	0.226318	1.000	1.000
Late repair	0.253662	0.000	0.000	0.513428	0.253662

The baseline comparison gives the main measurement result. Final-plan correctness and source-credit localization are separable on the stress schema, but the model stress check does not show successful final revision. The safe conclusion is that CEDAR can measure the two fields separately after the schema change; it is not that the six open-model conditions demonstrate reliable plan repair.

5 Interpreting Results from the Four-Action Schema

The earlier four-action model studies show why strong scores on that interface do not answer the present measurement question. Across six Qwen2.5-7B-Instruct prompt conditions, final-plan exactness ranged from 0.922 to 0.977 and credit exactness from 0.719 to 0.805. Across the six Qwen2.5-14B-Instruct conditions, the corresponding ranges were 0.914–1.000 and 0.906–1.000. These runs compare prompt behavior on the fixed four-action schema. Since its gold credit set equals the changed-position set in every audited case, those credit scores cannot establish that the models identified source actions independently of their final plans.

The supervised fine-tuning results give the same caution under held-out cases. Qwen2.5-0.5B-Instruct reached 1.000 final-plan and credit exactness on a 256-case split from the same finite four-domain family, and retained 1.000 final-plan exactness with 0.953 credit exactness on a 512-case wording holdout. In the separate seed-4040 position-permuted evaluation, final-plan exactness was 0.959 and credit exactness was 0.352 overall, falling to 0.153 on revisions. A separate seed-3030 audit of the domain-and-case-type position rule scored 0.361 credit exactness overall and 0.144 on revisions. These audits use different case seeds and are not a matched model-versus-baseline comparison. Their similar point estimates are descriptive; they do not establish a paired difference. Both remain within the four-action schema and do not test transfer to the variable-length source-only targets studied here.

A separate candidate-selection study also shows that terminal-plan and credit metrics can move in different directions. In its finite synthetic candidate pool, a selector trained for final-plan exactness reached mean final exactness 0.976 across four seeds but mean credit exactness 0.044. With eight candidates in their original order, the comparison policy reached 0.000 final exactness and 1.000 credit exactness. All cases in that test used a fixed four-action schema with the same singleton credit target, so these values diagnose an objective mismatch in that construction, not general reranking performance.

A sensitivity audit varied the candidate pool from two to eight and changed candidate order. In the original order, the learned selector's mean final exactness across four seeds was 0.000, 1.000, 1.000, and 0.976 for pools of two, four, six, and eight candidates; its credit exactness was 0.000, 1.000, 0.397, and 0.044. At eight candidates, reversing the order changed the comparison policy's final and credit exactness to 1.000 and 0.076; 16 random orders averaged 0.782 and 0.619. Its perfect credit score in the original order therefore depends on candidate order and tie-breaking under a constant target. The learned selector's eight-candidate means remained 0.976 final exactness and 0.044 credit exactness across the tested orderings. This sensitivity is specific to the finite text-sampler candidates; it does not establish performance

on model-generated trajectories or live tools. Taken together, these experiments describe the original interface but do not establish independent source-credit localization or downstream utility on the variable-length stress suite.

The four-action experiments remain useful for comparing prompt behavior and checking generator regressions. The variable-schema audit is the appropriate test for whether credit adds information beyond final-plan edits. Any future training comparison must use the stress schema, include the direct transformation, and measure downstream repair utility separately from credit localization.

6 Limitations and Responsible Interpretation

The cases are controlled serialized plans with three to seven actions. They do not reproduce the open-ended state, tool side effects, partial observability, or execution failures of interactive environments. The verifier target is source-only by construction and is not a human judgment or a causal effect estimate. The model evidence covers one 7B instruction-tuned model, two generation seeds, three output interfaces, and 256 cases per condition. It is sufficient to test the paired transformation on this suite, but it does not support broad model or domain generalization.

The current results do not measure live execution, tool safety, user satisfaction, external transfer, or whether source-credit predictions improve a deployed repair policy. They also do not compare learning procedures on the stress suite. Those questions require separately authorized environments, labels, and experiments. The present paper keeps them out of the reported evidence.

7 Conclusion

CEDAR exposes a measurement failure that is easy to miss in plan-revision benchmarks. In the canonical four-action schema, verifier-defined credit is identical to the evaluator-computed set of changed plan positions, so a credit score does not by itself establish an independent signal. A variable-length schema with source-only targets and explicit controls removes that identity in most cases while preserving a clean oracle. The bounded Qwen stress check then shows that the revised interface is informative for measurement but does not make revision reliable: models preserve controls and fail the revision cases in this run.

The practical recommendation is to report terminal-plan success and source-credit localization as separate axes, audit their identifiability before interpreting model comparisons, and require a downstream repair or selection result before attributing utility to credit. This makes the benchmark's conclusions narrower, reproducible, and easier to falsify. More broadly, benchmark-side identifiability should be checked before any intermediate label is used for diagnosis, model selection, reranking, or training, instead of being treated as a CEDAR-only cautionary note.

AI use statement

Generative AI tools were used to aid and polish the paper's writing and to support literature retrieval and discovery, including finding related work. These uses are described in the relevant parts of the paper.

Reproducibility statement

Sections 2 and 3 describe the case construction, visible and evaluator-only fields, scoring rules, and structural comparisons; Appendix A gives the model-output denominators and matched comparisons. The supplementary MIT-licensed source archive provides the deterministic audit code and saved outputs needed to recompute the six Qwen2.5-7B-Instruct stress conditions. It also contains commands for rebuilding the structural checks and prompt-surface test. Model weights are not part of the archive, so generating new model outputs requires separately obtaining Qwen2.5-7B-Instruct. The earlier four-action model studies discussed in Section 5 are not rerun by this lightweight package.

References

Shraddha Barke, Arnav Goyal, Alind Khare, Avaljot Singh, Suman Nath, and Chetan Bansal. Agentrx: Diagnosing ai agent failures from execution trajectories, 2026. URL https://arxiv.org/abs/2602.02475.

Jing Chen, Yang Sun, Li Zhang, Lin Xu, and Jie Shi. Long-horizon agent trajectory attribution: A unified benchmark and fine-grained annotation framework, 2026. URL https://arxiv.org/abs/2608.06909.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pp. 107–112, 2018. doi: 10.18653/v1/N18-2017. URL https://aclanthology.org/N18-2017/.

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?, 2024. URL https://arxiv.org/abs/2310.06770.

Pengcheng Li, Zhengyang Zhang, Dongxu Zhang, Sui Huang, and Shaohua Ma. Vict: Verifier-instrumented credit tracing for long-horizon llm agent reinforcement learning, 2026. URL https://arxiv.org/abs/2608.28128.

Jiayu Liu, Qihan Lin, Cheng Qian, Rui Wang, Emre Can Acikgoz, Xiaocheng Yang, Jiateng Liu, Zhenhailong Wang, Xiusi Chen, Heng Ji, and Dilek Hakkani-Tur. Planbench-xl: Evaluating long-horizon planning of llm tool-use agents in large-scale tool ecosystems, 2026. URL https://arxiv.org/abs/2606.22388.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Felix Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, Zirui Wang, and Ruoming Pang. Toolsandbox: A stateful, conversational, interactive evaluation benchmark for llm tool use capabilities, 2024. URL https://arxiv.org/abs/2408.04682.

R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 3428–3448, 2019. doi: 10.18653/v1/P19-1334. URL https://aclanthology.org/P19-1334/.

Jaineet Shah. Causal agent replay: Counterfactual attribution for llm-agent failures, 2026. URL https://arxiv.org/abs/2606.08275.

Haoyu Sun, Wenxuan Wang, Mingyang Song, Jujie He, Weinan Zhang, Yang Liu, Yang Yang, and Yu Cheng. Agent planning benchmark: A diagnostic framework for planning capabilities in llm agents, 2026. URL https://arxiv.org/abs/2606.04874.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. Appworld: A controllable world of apps and people for benchmarking interactive coding agents, 2024. URL https://arxiv.org/abs/2407.18901.

Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. Planbench: An extensible benchmark for evaluating large language models on planning and reasoning about change, 2022. URL https://arxiv.org/abs/2206.10498.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments, 2024. URL https://arxiv.org/abs/2404.07972.

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/2406.12045.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents, 2023. URL https://arxiv.org/abs/2307.13854.

A Credit-field applicability and matched comparisons

The credit column in Table 2 scores what the model actually emitted, not what can be inferred from its plan. The plan_only prompt requests a plan but no credit set. Its original credit exactness and F1 are therefore not applicable in both seeds (0 of 256 scoreable cases). This does not prevent a diagnostic from computing D-hat from each emitted plan: its credit exactness is 0.391 at seed 7 and 0.281 at seed 19, each over 256 cases. Those numbers describe derived edits and cannot be interpreted as the quality of an unrequested credit output.

In the two plan_and_credit conditions and seed-19 self_repair, all 256 outputs have valid requested credit. Seed-7 self_repair has two duplicate-index credit fields (cases stress-7-00181 and stress-7-00185), leaving 254 scoreable outputs. Its original exactness is 105/254 = 0.413386 and mean credit F1 is 0.585958. The derived score of 0.417969 uses 256 cases. On the intersection of the 254 valid case IDs, derived exactness is 0.421260 and derived F1 is 0.772628; original exactness is 0.413386 and original F1 is 0.585958. Thus the matched differences are +0.007874 in exactness and +0.186670 in F1. The raw saved predictions were replayed under the corrected field-validity rules; these are not new model generations.

The other three credit-requesting comparisons use 256 matched cases each. Derived-minus-original exactness and F1 are respectively +0.085938 and +0.230636 for plan_and_credit seed 7, +0.085938 and +0.201339 for its seed 19, and -0.007812 and +0.140709 for self_repair seed 19. The negative exactness difference in the last condition matters: replacing emitted credit with plan edits is not uniformly beneficial even on this synthetic suite. The F1 differences measure set overlap with the same source-only target under a different predicted set; they do not show that source responsibility was recovered. The validity rule is essential to the interpretation: the earlier evaluator treated unrequested or malformed fields as empty sets, which made the plan-only credit entries and their apparent paired gaps spurious.