LONGRCA BENCH: ROOT-CAUSE LOCALIZATION IN LONG-HORIZON FAILED AGENT TRAJECTORIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Failure attribution is a critical component of the failure-driven agent improvement loop: identifying which workflow role introduced a decisive error and when can turn failed executions into actionable feedback for repair, retry, and workflow refinement. However, existing benchmarks are dominated by short execution histories. In Who&When, 71% of trajectories contain at most 10 recorded steps, a regime in which the visible erroneous step is often close to the final failure and failure attribution can collapse toward local failure detection. Long-horizon failures create a different diagnostic problem: an early error may be propagated, reused, validated, or left uncorrected across dozens or hundreds of subsequent steps before the execution terminates. We introduce LONGRCA BENCH, comprising 1,140 complete naturally failed trajectories from five sources, all human-annotated with responsible roles and earliest decisive root-cause steps. Trajectories have a median length of 145 recorded steps and a maximum of 728. Annotated root-cause steps occur a median of 48 steps before execution ends. Under the same DeepSeek-V4-Flash backbone and scoring protocol, five existing methods achieve 30.4–36.4% exact root-step accuracy on Who&When but only 2.8–13.2% on LONGRCA BENCH, exposing a substantial gap between detecting nearby errors and tracing distant error origins. Motivated by the benchmark's challenges, we propose Root-Cause Trajectory Attribution (RCTA), a training-free method that combines hierarchical trajectory summarization, original-record retrieval, and upstream instruction–execution comparison to distinguish inherited errors from newly introduced ones. RCTA achieves 41.8% exact root-step accuracy on Who&When and 24.1% on LONGRCA BENCH, compared with 36.4% and 13.2% for the strongest evaluated baselines, respectively.

1 INTRODUCTION

When an agent run fails, locating the source of failure can help developers identify weaknesses in agent behavior and workflow design. It can also help users decide what to correct before retrying the task. In long agent executions, an early error may persist through subsequent planning, tool use, and verification. Diagnosing such failures requires connecting evidence across the execution history (Barke et al., 2026). Root-cause attribution must distinguish the step that introduced the decisive error from later actions that propagated it or failed to correct it (Zhang et al., 2025).

Who&When evaluates responsible-agent and decisive-step attribution (Zhang et al., 2025), while RootSE annotates earliest decisive errors in repository-level coding trajectories (Wang et al., 2026a). However, existing resources offer limited coverage of tracing error origins across long execution histories, as summarized in Section 5. Evaluating this capability requires complete failed trajectories with human-annotated attribution targets.

We introduce **LongRCA Bench**, comprising 1,140 complete, human-annotated failed trajectories from five task sources. Each trajectory is labeled with a responsible workflow role and a root-cause step, which are predicted and scored independently. We define the root as the earliest recorded step introducing a decisive error that makes the subsequent execution irrecoverable. Trajectories have a median length of 145 recorded steps and a maximum of 728; annotated root-cause steps occur a median of 48 steps before execution ends. These histories support evaluation of error-origin localization across extensive subsequent execution.

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.

Our evaluation reveals substantial room for improvement in the tested attribution methods. With DeepSeek-V4-Flash (DS-V4-Flash), the five evaluated baselines achieve 30.4%–36.4% exact root-step accuracy on Who&When (Zhang et al., 2025), compared with 2.8%–13.2% on LongRCA Bench, as shown in Figure 1. The datasets also differ in domains, workflows, and recording granularity, so this gap does not isolate a trajectory-length effect.

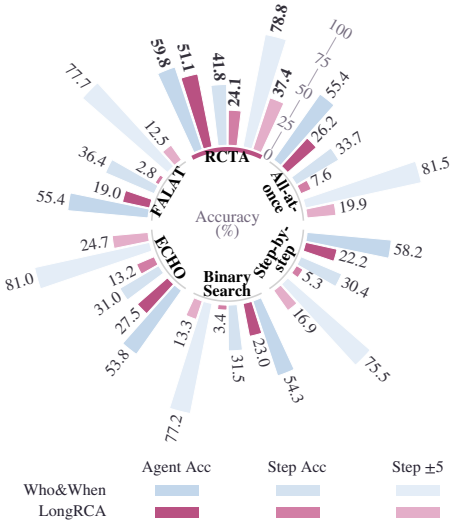


Figure 1: All six evaluated methods score lower on LongRCA across Agent Acc, Step Acc, and Step ±5. Accuracy (%) uses DS-V4-Flash on the complete Who&When ($n = 184$) and LongRCA ($n = 1{,}140$) sets, with each trajectory equally weighted within its dataset.

One attribution challenge arises when an agent faithfully executes an instruction that already contains an error. We propose **Root-Cause Trajectory Attribution (RCTA)**, a training-free method that retrieves original trajectory records and compares them with available upstream handoff instructions to distinguish inherited errors from newly introduced ones. On the full benchmark, RCTA achieves 24.1% exact root-step accuracy, compared with 13.2% for the strongest evaluated baseline under the same DS-V4-Flash backbone and scoring protocol, as detailed in Section 4.2.

We also provide **LongRCA-Mini**, a fixed 200-trajectory subset for lower-cost preliminary comparisons. Among the six methods evaluated on Mini, RCTA achieves the highest observed exact root-step accuracy on each of five backbones. In RCTA's component study with DS-V4-Flash, retrieving the record at the annotated root-cause step does not guarantee correct localization, as discussed in Section 4.3, highlighting a remaining gap between evidence retrieval and accurate attribution.

Our contributions are threefold:

- **A benchmark for tracing error origins in long agent executions.** LongRCA Bench provides complete failed histories with human-annotated responsible roles and earliest decisive root-cause steps. LongRCA-Mini supports lower-cost preliminary comparisons.
- **A method combining evidence recovery and upstream instruction comparison.** RCTA retrieves original candidate records and compares them with available handoff instructions to distinguish inherited errors from errors newly introduced during execution.
- **A systematic evaluation of attribution performance and its limitations.** We evaluate six methods on the full benchmark, compare five backbones on Mini, and conduct component diagnostics. These analyses reveal a remaining gap between recovering annotated root-step records and correctly localizing error origins.

2 LongRCA Bench

2.1 Task Definition and Illustrative Case

Given a complete failed execution history $H = (h_0, \ldots, h_{T-1})$ with stable 0-based indices, methods predict a responsible workflow role and a root-cause step, scored independently against human references. The responsible role need not be the recorded sender of the root step. The reference root r^* is the earliest recorded step introducing a decisive error that makes the subsequent execution irrecoverable. An earlier minor, recoverable error that is not necessarily fatal does not displace a later decisive error. Repaired errors and later steps that merely propagate, expose, or fail to correct an existing decisive error are excluded. An explicit logged delegation is a *handoff*; an erroneous handoff can be the root even when another participant executes it. The benchmark targets single-root attribution and excludes trajectories suspected of containing multiple independent decisive root causes across subtasks. Appendix G details the annotation boundaries.

Illustrative case. Figure 2 illustrates a 223-step travel-planning trajectory in which nightly lodging prices are recorded as two-night totals. The reference marks step 73, where the agent first records a nightly price as a two-night total. Later verification and a correction to separate budget arithmetic leave this interpretation unchanged. Appendix K.2 presents the trajectory and reference annotation.

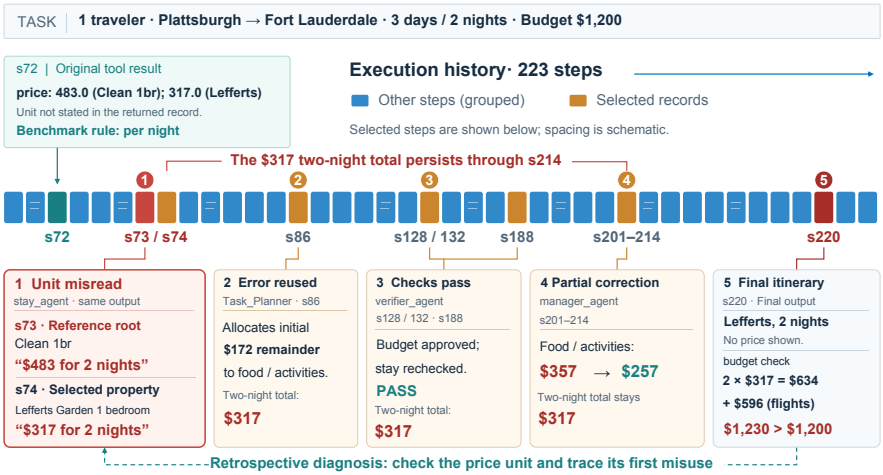


Figure 2: A lodging-price misinterpretation persists through planning, verification, and a local budget correction. The human reference identifies `stay_agent` and step 73. Green arrows indicate retrospective diagnosis.

2.2 Data Sources and Collection

We generated execution records from five public task benchmarks using the multi-agent workflows in Table 1 (Deng et al., 2025; Merrill et al., 2026; Xie et al., 2024; He et al., 2026; El Hattami et al., 2025). Failures arise from ordinary execution without injected errors. Infrastructure and evaluator errors, smoke tests, and debug runs are excluded; eligible budget-limited failures retain their histories through termination. Appendix E documents task coverage, generators, execution limits, annotation completion, and source-specific protocols.

Table 1: Task sources and workflows. Counts refer to released trajectories after annotation, quality control, and deduplication.

Source	Task domain	Workflow	Released N
SWE-bench Pro	Software repair	Diagnosis–execution–verification	128
Terminal-Bench 2	Terminal tasks	Diagnosis–execution–verification	42
TravelPlanner	Travel planning	Magentic-One group chat	685
VitaBench	Service-oriented tool use	Action expert–computer terminal	108
WebArena Verified	Web interaction	Planner–web-surfer–critic	177
Total	**5 domains**	**5 workflows**	**1,140**

Before admission to the formal annotation pool, trajectories were checked against the single-root scope. All eligible failures entered the pool without further selection by length or difficulty. Source, trajectory length, and task difficulty informed annotation assignments. Distinct attempts at the same task remain separate trajectories; duplicate records are merged.

Normalization preserves original content, roles, and temporal order without synthesizing or artificially subdividing events; event granularity remains source-dependent. Methods receive only `question_ID` and `history`, including task instructions and any recorded evaluator feedback. Reference labels and rationales are withheld (Appendix D).

2.3 Human Annotation and Quality Control

Twenty-two master's and doctoral students in computer science, experienced in analyzing agent trajectories and identifying failure root causes, annotated the trajectories. The same pool participated in calibration and formal annotation. Calibration on 100 trajectories refined history navigation,

root-cause search, and evidence recording. Four trajectories were excluded because they potentially contained independent root causes across subtasks; 96 were retained.

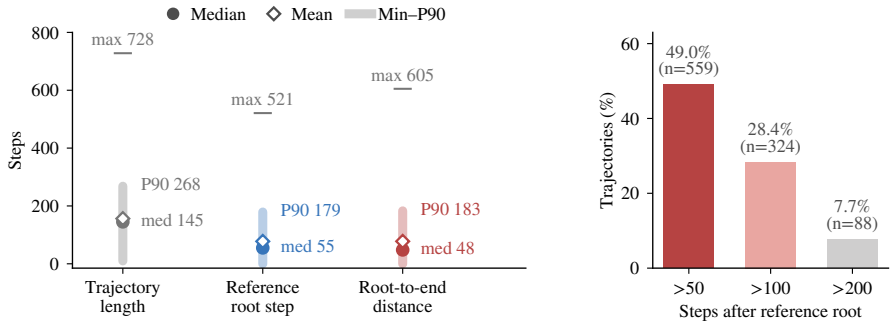
Annotators independently reviewed task instructions, complete histories, and source evaluator outcomes, providing a responsible role, root step, and evidence-grounded rationale. DeepSeek V4 Flash Preview supplied factual navigation summaries without root-cause judgments; annotations required original trajectory evidence. Before adjudication, pairwise agreement across 310 annotation pairs from 294 multiply annotated formal-phase trajectories was 66.1% for responsible roles, 40.3% for exact root steps, and 50.3% within five steps.

Two experts adjudicated disagreements using the original tasks, histories, evaluator outcomes, and rationales under the attribution criteria. Final quality control re-examined all retained trajectories, including calibration cases, and excluded unresolved attribution ambiguity, invalid labels, or insufficient evidence. The released dataset combines 96 calibration trajectories with 1,044 formal-phase trajectories, yielding 1,140 unique human-annotated failed executions.

All released records pass structural and file-consistency checks and retain verified mappings to their original execution histories. Appendix K.1 provides the annotation protocol, disagreement analysis, and sensitivity to observed root-step labels.

2.4 Benchmark Characteristics and LongRCA-Mini

TravelPlanner accounts for 60.1% of the released dataset. Trajectories have a mean length of 156.3 recorded steps, a median of 145, and a maximum of 728. Root-to-end distance, $T-1-r^*$, has a median of 48 steps; 28.4% of trajectories contain more than 100 steps after the reference root, as shown in Figure 3. This distance measures subsequent recorded activity, rather than error propagation or delay until failure becomes observable. Appendix K.3 provides additional source-level statistics.



(a) Trajectory statistics (b) Root-to-end distance

Figure 3: Long-horizon characteristics ($n = 1,140$). (a) Medians (circles), means (diamonds), minimum–P90 ranges, and maxima of trajectory length, reference root step, and root-to-end distance. (b) Fractions with more than 50, 100, or 200 recorded steps after the reference root.

LongRCA-Mini. LongRCA-Mini supports preliminary comparisons with 200 fixed trajectories, sampled uniformly without replacement within each source, with 40 trajectories per source. Equal allocation matches the equal-source Macro-5 objective. Using cached Full predictions from six methods under DS-V4-Flash, 200 is the smallest tested size meeting the predefined score-error criteria. Individual trajectories are selected without predictions or reference labels, and observed results do not trigger resampling.

Across six methods and three accuracy metrics, the fixed subset has mean and maximum absolute deviations from Full Macro-5 of 1.665 and 3.771 pp, respectively. Full Macro-5 weights the five sources equally. Computed from the same cached predictions, these deviations quantify subset-induced differences under the evaluated configuration. Appendix A gives the calibration criteria, sampling protocol, and detailed fidelity results.

3 Root-Cause Trajectory Attribution

As shown in Figure 4, RCTA separates evidence retrieval from error-origin attribution. Stage I gathers original records from the long execution history. Stage II evaluates where the decisive error was

introduced, using preceding handoff instructions when available. This comparison addresses cases in which a downstream action inherits an error from an earlier instruction. An auxiliary validation layer checks the output fields and cited evidence.

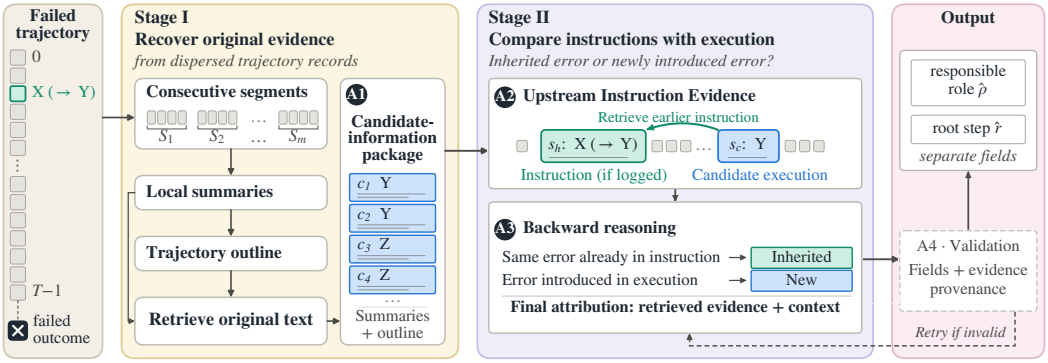


Figure 4: RCTA separates evidence retrieval (Stage I, A1) from error-origin attribution (Stage II, A2–A3). Logged handoffs augment the evidence when available (A2). The instruction–execution pair illustrates a local comparison; final attribution considers the retrieved evidence package and trajectory context (A3). Auxiliary validation checks output fields and textual provenance (A4).

3.1 Stage I: Failure-relevant candidate recall

Stage I uses summaries to navigate the long trajectory and retrieves original records as evidence for the final attribution decision. A *segment* contains consecutive recorded steps with original IDs. Local summaries propose failure-relevant steps from each segment, using available preceding overlap records as context. A trajectory outline aggregates subgoals and conflicts across summaries. RCTA pools and prioritizes proposed step IDs, then retrieves their original text to form the base-candidate set before handoff augmentation. The retrieved set includes records of possible error origins and downstream effects, leaving the final root-cause decision to Stage II. Appendix H specifies segmentation, retrieval rules, and prompts; Algorithm 1 gives the inference sequence.

3.2 Stage II: Upstream instruction comparison

Using recorded order and recipient information, RCTA retrieves preceding logged handoff instructions when available (A2). These instructions are candidate sources of the error to be assessed during attribution. Candidate records and retrieved instructions retain their original text and step IDs for comparison. Retrieved handoffs can add possible root steps outside the base-candidate set.

Backward instruction–execution reasoning (A3) examines the retrieved evidence against task requirements, using summaries and the outline as context. For a candidate with a retrieved handoff, it assesses whether the instruction already contains the same decisive error or whether the error was newly introduced during execution. The final decision considers the retrieved evidence package. Following Section 2.1, it targets the earliest supported introduction of a decisive error that makes the subsequent execution irrecoverable. An instruction introducing that error can therefore be selected instead of its downstream candidate. Without a retrieved handoff, attribution proceeds from the available candidate records and trajectory context.

The final attribution call returns separate responsible-role and root-step fields. Validation (A4) checks field validity and the textual provenance of cited handoff evidence, preserving valid fields independently. It does not verify causal correctness. Appendix H details retries and output handling.

4 Evaluation

4.1 Experimental Setup

We evaluate exact error-origin localization on LongRCA Bench (RQ1), examine what RCTA's component diagnostics reveal about evidence retrieval and attribution (RQ2), and compare performance across model backbones (RQ3).

The full-benchmark comparison evaluates six training-free methods on all 1,140 failed trajectories using DeepSeek-V4-Flash (DS-V4-Flash). Alongside RCTA, All-at-once reads a compressed full trajectory in one attribution prompt, Step-by-step scans it sequentially, and Binary search recursively bisects it (Zhang et al., 2025). ECHO uses hierarchical context and consensus voting (Banerjee et al., 2025); FALAT traces dependencies among decisions, tool outputs, and messages to localize error-introducing steps (Rafi et al., 2026). All methods use the same LongRCA Bench attribution targets and scoring rules.

Cross-backbone evaluation compares the same six methods on the fixed 200-trajectory LongRCA-Mini across five backbones. Component diagnostics use DS-V4-Flash on the same Mini subset with dedicated ablation inference settings, distinct from the Full and cross-backbone runs (Appendix H). Component conditions share reference labels, segmentation, and summary caches. Mini fidelity reuses cached Full predictions.

Agent Acc compares the explicit responsible-role output with its reference. Step Acc requires an exact match to the 0-based reference root step; Step ± 5 permits absolute step error at most five. Role and step are scored independently, with invalid fields counted as errors for the corresponding targets. Missing or unrecognized roles are invalid; missing, nonnumeric, or out-of-range steps fail both step metrics. All expected trajectories remain in accuracy denominators. Full weights trajectories equally, preserving source proportions. Mini uses $S_{\text{Macro-5}} = \frac{1}{5}\sum_{b=1}^{5} S_b$, where S_b is the current metric's source score. Both sides of Full–Mini fidelity use Macro-5. With equal source quotas, pooled accuracy equals Macro-5; this equality need not hold for MAE when valid-output counts differ. Appendix L.1 provides normalization, MAE, and protocol details.

4.2 Root-Cause Attribution on LongRCA Bench (RQ1)

Table 2: Full-benchmark accuracies (%) using DS-V4-Flash ($n = 1{,}140$). Overall weights trajectories equally; per-source columns report Step Acc. Bold marks the highest observed value in each column.

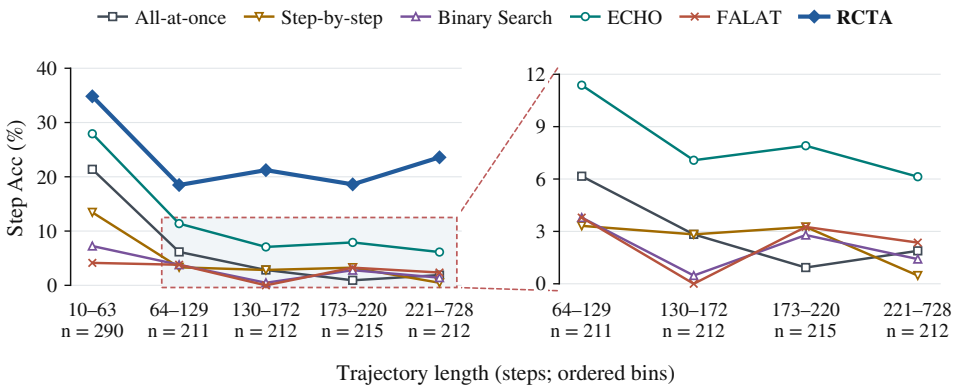
Method	Overall			Per-source Step Acc				
	Agent Acc	Step Acc	Step ± 5	SWE	TB2	TP	Vita	WA
All-at-once	26.2	7.6	19.9	1.6	7.1	4.5	19.4	16.9
Step-by-step	22.2	5.3	16.9	0.8	4.8	3.5	12.0	11.3
Binary search	23.0	3.4	13.3	0.8	4.8	2.0	12.0	5.1
ECHO	27.5	13.2	24.7	7.8	21.4	8.9	22.2	26.0
FALAT	19.0	2.8	12.5	2.3	2.4	2.9	3.7	2.3
RCTA	**51.1**	**24.1**	**37.4**	**38.3**	**26.2**	**18.8**	**25.0**	**33.3**

As shown in Table 2, the five baselines achieve 2.8%–13.2% exact root-step accuracy with DS-V4-Flash on LongRCA Bench. RCTA reaches 24.1% overall and achieves the highest observed Step Acc in each of the five sources. With a five-step tolerance, RCTA reaches 37.4%, compared with 24.7% for the strongest baseline. Between-source comparisons remain descriptive because the represented tasks and workflows differ.

Role attribution is a separate target. Predicting each source's most frequent reference role yields 40.3% Agent Acc, compared with 51.1% for RCTA. This descriptive prior uses frequencies from the evaluated set; role accuracy does not establish root-step accuracy. Appendix L.2 reports the role-prior comparison and complete metrics, including Root MAE. Appendix C provides additional Who&When cross-dataset context under the same backbone and details the comparison protocol and interpretation.

Localization across log characteristics.

Figure 5 groups all 1,140 trajectories by recorded length, keeping equal lengths together. RCTA has the highest observed Step Acc in every group, and all six methods score lower in each longer group than in the shortest group. Variation among longer groups is non-monotonic: RCTA rises from 18.6% to 23.6% between the final two groups. The longest group mainly contains 221–400-step trajectories (170 of 212). Source and workflow composition differ across groups, so these



(a) Overall accuracy (0–40%) (b) Zoom: bins 2–5 (0–12%)

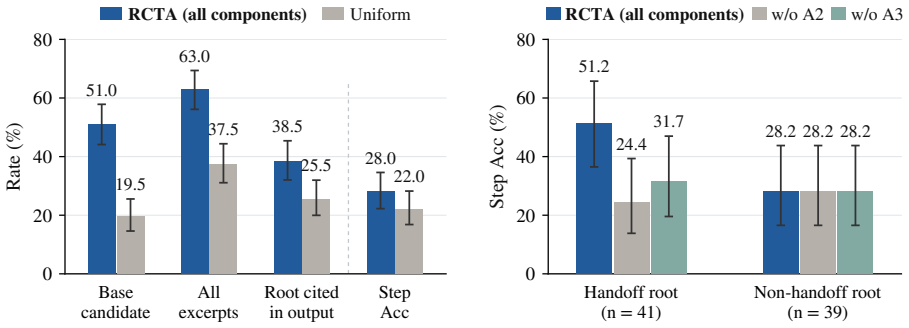
Figure 5: Step Acc on all 1,140 trajectories with DS-V4-Flash. Methods share length-only groups without splitting equal lengths. Panel (a) shows all groups; panel (b) magnifies the 0–12% region of groups 2–5. Unequal-width groups are equally spaced; connecting lines are visual guides.

comparisons do not isolate a causal effect of length. Appendix F gives the source visualization, numeric results, partition algorithm, and four- and six-group sensitivity checks.

Within-source analyses show lower Step Acc and Step ± 5 in the group with the most log characters than in the group with the fewest log characters for all six methods on TravelPlanner and VitaBench. Across root-to-end distance groups, accuracy varies by source and shows no consistent decline with increasing distance (Appendix F.1).

4.3 Retrieval and Attribution Diagnostics in RCTA (RQ2)

We examine reference-root record coverage, handoff-based attribution, and output validity under the dedicated component settings. All comparisons use the same 200 Mini trajectories; the complete RCTA configuration reaches 28.0% Step Acc in this study.



(a) Evidence coverage and localization (b) Step Acc by annotated root type

Figure 6: Component diagnostics. (a) Reference-root hit rates in base candidates, handoff-augmented input excerpts, and evidence IDs cited in the final output, alongside Step Acc ($n = 200$). (b) Step Acc on handoff-containing trajectories with handoff roots ($n = 41$) or non-handoff roots ($n = 39$). RCTA (all components) and w/o A3 use identical candidate and handoff excerpts. Error bars show 95% Wilson intervals; subgroup contrasts are exploratory.

Evidence retrieval and localization. The control without A1 uses equally many uniformly positioned excerpts and masks location and provenance cues. The A1 comparison evaluates excerpt selection together with location and provenance cues. After handoff augmentation, reference-root record coverage is 37.5% for this control and 63.0% for complete RCTA, while final Step Acc is 22.0% and 28.0%, respectively, as shown in Figure 6(a). Input coverage measures whether the retrieved excerpts contain the annotated root record. The root citation rate measures whether the final output cites that step, while Step Acc measures whether it selects that step as the root. In 21 of the

77 cases where RCTA's final output cites the annotated root step, it selects a different step as the root.

Upstream instructions and inherited errors. The handoff diagnostic covers 80 trajectories from SWE-bench Pro and Terminal-Bench 2 that contain explicit logged handoffs. On the 41 handoff-root cases, complete RCTA and variants without handoff evidence (A2) or the backward-attribution package (A3) achieve 51.2%, 24.4%, and 31.7% Step Acc, respectively, as shown in Figure 6(b). On the 39 non-handoff-root cases, all three obtain 28.2%. Both groups contain handoffs. The variant without A3 retains the original candidate and handoff records. It removes backward-attribution instructions and associated cues and checks. The observed differences are concentrated in handoff-root cases. These exploratory results concern trajectories collected under the shared diagnosis–execution–verification workflow in the two source benchmarks.

Table 3: Paired component ablations with DS-V4-Flash on LongRCA-Mini ($n = 200$). Accuracies are Macro-5 percentages; Cov. reports valid-output coverage under each condition's output checks. p_{Holm} adjusts the four exact McNemar tests on Step Acc (Appendices H and L.1).

Setting	Agent Acc↑	Step Acc↑	Step ±5↑	Cov.↑	p_{Holm}
RCTA (all components)	56.0	28.0	41.0	97.0	–
w/o candidate-information package (A1)	53.5	22.0	39.0	91.5	.17
w/o handoff evidence (A2)	53.5	24.5	38.0	97.0	.75
w/o backward attribution (A3)	51.0	25.0	41.0	97.5	.75
w/o validation (A4)	54.0	26.5	39.5	89.0	.75

Overall ablations and output validity. As shown in Table 3, all four removals lower Step Acc point estimates, but none passes Holm correction across the four comparisons ($p_{\mathrm{Holm}} \geq .17$). Non-significance does not establish equivalence. Removing A4 omits validation-driven retry and output sanitization. With A4, valid-output coverage rises from 89% to 97%, and three additional trajectories receive correct root-step predictions. Appendices H and L.1 specify the controls and scoring rules; Appendix L.3 reports paired intervals and complete results.

4.4 Cross-Backbone Evaluation on LongRCA-Mini (RQ3)

Table 4: Agent and Step Acc (%, Macro-5) on the fixed 200-trajectory Mini (40 per source). V4 denotes DeepSeek-V4. Bold marks the highest point estimate per backbone and metric; complete metrics and paired analyses are in Appendix B.

Method	V4-Flash		V4-Pro		GPT-5.6 Sol		Kimi K3		Qwen3.5-35B	
	Agent	Step	Agent	Step	Agent	Step	Agent	Step	Agent	Step
All-at-once	42.5	7.5	50.0	11.5	53.0	17.0	54.5	11.0	**51.0**	3.5
Step-by-step	38.5	6.5	47.5	10.5	49.0	13.0	44.0	9.5	46.5	4.5
Binary search	40.5	7.5	45.5	10.0	51.5	15.5	42.0	12.0	40.0	5.0
ECHO	45.0	13.0	53.5	18.0	55.5	18.5	55.0	20.5	48.5	10.0
FALAT	35.5	2.5	48.0	14.5	52.0	14.0	52.0	18.0	41.0	11.0
RCTA	**60.5**	**27.0**	**61.0**	**28.0**	**64.5**	**30.0**	**61.5**	**31.5**	48.5	**13.5**

As shown in Table 4, RCTA achieves the highest observed Step Acc on each of the five evaluated backbones on LongRCA-Mini. Its Step Acc ranges from 13.5% to 31.5% across these backbones. RCTA also has the highest Agent Acc on four backbones; on Qwen3.5-35B-A3B, All-at-once has higher Agent Acc and ECHO has higher Step ±5.

These results are point estimates and do not establish significant superiority on every backbone. Appendix B reports the complete metrics, paired tests, and their endpoint-family sensitivity, together with an analysis of complementary exact localizations between RCTA and the five-baseline union.

5 Related Work

Failure-attribution benchmarks provide complementary forms of supervision and execution coverage. Who&When establishes responsible-agent and decisive-step attribution (Zhang et al., 2025), while Who&When Pro expands this setting through controlled error injection at larger scale (Liu

Table 5: Comparison of resources for agent failure analysis.

Benchmark	Error Source	Diagnostic Targets	Trajectories Total (Reviewed)	Avg. length (Steps)
Who&When (Zhang et al., 2025)	Natural	Agent, Step[a]	184 (184)	22.2
Who&When Pro (Liu et al., 2026)	Synthetic	Agent, Step[a], Type	12,326 (100)	7.5
TraceElephant (Chen et al., 2026)	Natural	Component, Step[b]	220 (220)	–
RootSE (Wang et al., 2026a)	Natural	Step[a]	102 (102)	51.7
TrajErrBench (Qi et al., 2026)	Natural	Step[a], Type	486 (486)	–
AgentRx (Barke et al., 2026)	Natural	Step[d], Type	170 (170)	–
Failure as a Process (Zhao et al., 2026)	Natural	Steps[c], Type	1184 (1184)	42.0
HORIZON (Wang et al., 2026b)	Natural	Type	1995 (40)	–
LongRCA Bench (Ours)	**Natural**	**Agent, Step[a]**	**1140 (1140)**	**156.3**

[a]Earliest decisive error; [b]Earliest point at which failure becomes inevitable; [c]Decisive error, empirical lock-in, and first observable sign of failure; [d]Earliest unrecovered failure in the causal chain. Parentheses indicate human-reviewed counts. HORIZON and Failure as a Process report failed-trajectory subsets. Annotation protocols and step granularity differ across benchmarks, so mean lengths are not directly comparable measures of difficulty; – denotes an unreported overall mean. See Appendix J for details.

et al., 2026). Other resources broaden the diagnostic setting: TraceElephant emphasizes full execution observability (Chen et al., 2026); RootSE, introduced with TrajAudit, annotates earliest decisive errors in repository-level coding trajectories (Wang et al., 2026a); and AgentRx provides critical-step and failure-category annotations across diverse task settings (Barke et al., 2026). TrajErrBench annotates decisive error steps and error categories in tool-use and coding trajectories (Qi et al., 2026).

Related studies examine how failures develop and how task horizon affects agent behavior. Failure as a Process distinguishes decisive errors, empirical failure lock-in, and first observable signs of failure (Zhao et al., 2026). HORIZON varies intrinsic task horizon to study performance degradation and failure patterns across domains; this horizon is distinct from recorded execution length (Wang et al., 2026b). LongRCA Bench covers long failed trajectories from five task sources and independently evaluates responsible-role and root-step attribution. The responsible role may differ from the sender of the root-step record.

Attribution methods combine evidence organization with reasoning about error origins. ECHO uses hierarchical context and analyst consensus (Banerjee et al., 2025), while SAFARI supports active trace investigation through reading, search, and persistent short-term memory (Zhu et al., 2026b). FALAT and CHIEF use typed dependencies and hierarchical causal structure, respectively, to distinguish error sources from downstream propagation (Rafi et al., 2026; Wang et al., 2026c). AgentRx uses constraint-checking logs to support diagnosis (Barke et al., 2026). TrajDebug tracks error propagation and repair to locate decisive errors. Its diagnostics also show that retaining the correct candidate does not guarantee correct attribution (Qi et al., 2026). RCTA retrieves candidate records and available explicit upstream handoff instructions, then compares instructions with execution to distinguish inherited errors from newly introduced ones.

6 Conclusion

LongRCA Bench provides 1,140 complete, human-annotated failure trajectories for retrospective error-origin localization in long agent executions. RCTA retrieves original records and compares them with available upstream handoff instructions to distinguish inherited errors from newly introduced ones. On the full benchmark with DS-V4-Flash, it achieves 24.1% exact root-step accuracy, compared with 13.2% for the strongest evaluated baseline. Component diagnostics show that RCTA can retrieve the annotated root record and still select a different step. Future work should distinguish missing evidence from failures to attribute errors using available records, while assessing uncertainty in reference root annotations.

AI USE STATEMENT

We used GPT-5.6 Sol and Codex for literature discovery, drafting, editing, and LaTeX formatting, and DeepSeek V4 Flash Preview for annotation-navigation summaries. We reviewed the AI-assisted content and references and take responsibility for the final work.

ETHICS STATEMENT

Annotators were informed about the study and participated as part of the research group's work. We screened the released data for sensitive information and reviewed the source licenses.

REPRODUCIBILITY STATEMENT

Sections 2–4 describe benchmark construction, RCTA, and evaluation. Appendices A and K.1 document the sampling manifest and annotation protocol. Prompt and inference details are provided in Appendix H, and scoring rules in Appendix L.1.

REFERENCES

Adi Banerjee, Anirudh Nair, and Tarik Borogovac. Where did it all go wrong? a hierarchical look into multi-agent error attribution. *arXiv preprint arXiv:2510.04886*, 2025. URL https://arxiv.org/abs/2510.04886.

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

Mert Cemri, Melissa Z. Pan, Shuyi Yang, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Why do multi-agent LLM systems fail? In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2503.13657.

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. URL https://aclanthology.org/2026.acl-long.912/.

Xiang Deng, Jeff Da, Edwin Pan, Yannis Yiming He, Charles Ide, Kanak Garg, Niklas Lauffer, Andrew Park, Nitin Pasari, Chetan Rane, Karmini Sampath, Maya Krishnan, Srivatsa Kundurthy, Sean Hendryx, Zifan Wang, Chen Bo Calvin Zhang, Noah Jacobson, Bing Liu, and Brad Kenstler. SWE-Bench Pro: Can AI agents solve long-horizon software engineering tasks? *arXiv preprint arXiv:2509.16941*, 2025. URL https://arxiv.org/abs/2509.16941.

Darshan Deshpande, Varun Gangal, Hersh Mehta, Jitin Krishnan, Anand Kannappan, and Rebecca Qian. TRAIL: Trace reasoning and agentic issue localization. *arXiv preprint arXiv:2505.08638*, 2025. URL https://arxiv.org/abs/2505.08638.

Amine El Hattami, Megh Thakkar, Nicolas Chapados, and Christopher Pal. WebArena Verified: Reliable evaluation for web agents. In *Workshop on Scaling Environments for Agents*, 2025. URL https://openreview.net/forum?id=94tlGxmqkN.

Yu Ge, Linna Xie, Zhong Li, Yu Pei, and Tian Zhang. Who is introducing the failure? automatically attributing failures of multi-agent systems via spectrum analysis. *arXiv preprint arXiv:2509.13782*, 2025. URL https://arxiv.org/abs/2509.13782.

Wei He, Yueqing Sun, Hongyan Hao, Xueyuan Hao, Zhikang Xia, Dunwei Tu, Qi Gu, Chengcheng Han, Dengchang Zhao, Hui Su, Kefeng Zhang, Man Gao, Xi Su, Xiaodong Cai, Xunliang Cai, Yu Yang, and Yunke Zhao. VitaBench: Benchmarking LLM agents with versatile interactive tasks in real-world applications. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.26490.

Yeonjun In, Md 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 preprint arXiv:2603.25001*, 2026. URL https://arxiv.org/abs/2603.25001.

Jan-Christoph Klie, Richard Eckart de Castilho, and Iryna Gurevych. Analyzing dataset annotation quality management in the wild. *Computational Linguistics*, 50(3):817–866, 2024. doi: 10.1162/coli_a_00516. URL https://aclanthology.org/2024.cl-3.1/.

Jiale Liu, Huajun Xi, Shaokun Zhang, Yifan Zeng, Tianwei Yue, Chi Wang, Jian Kang, Qingyun Wu, and Huazheng Wang. Who&when pro: Can LLMs really attribute failures in AI agents? *arXiv preprint arXiv:2607.09996*, 2026. URL https://arxiv.org/abs/2607.09996.

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 preprint arXiv:2509.08682*, 2025. URL https://arxiv.org/abs/2509.08682.

Mike A. Merrill, Alexander G. Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, et al. Terminal-Bench: Benchmarking agents on hard, realistic tasks in command line interfaces. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2601.11868.

Robert G. Newcombe. Interval estimation for the difference between independent proportions: comparison of eleven methods. *Statistics in Medicine*, 17(8):873–890, 1998. doi: 10.1002/(SICI)1097-0258(19980430)17:8<873::AID-SIM779>3.0.CO;2-I. URL https://pubmed.ncbi.nlm.nih.gov/9595617/.

Yunjia Qi, Zehua Yin, Xintong Shi, Hao Peng, Songyuanyi Lu, Yixian Liu, Richeng Xuan, Yuhong Liu, Zhichao Hu, Xiaozhi Wang, Lei Hou, Bin Xu, and Juanzi Li. TRAJDEBUG: Tracing error lifecycle to identify critical failures in long-horizon agent trajectories. *arXiv preprint arXiv:2608.06346*, 2026. URL https://arxiv.org/abs/2608.06346.

Md Nakhla Rafi, Md Ahasanuzzaman, Dong Jae Kim, Zhijie Wang, and Tse-Hsun Chen. Falat: Tracing failures in llm agent trajectories via dependency-guided search. *arXiv preprint arXiv:2606.00765*, 2026. URL https://arxiv.org/abs/2606.00765.

Minxing Wang, Xiaofei Xie, and Yintong Huo. Trajaudit: Automated failure diagnosis for agentic coding systems. *arXiv preprint arXiv:2605.26563*, 2026a.

Xinyu Jessica Wang, Haoyue Bai, Yiyou Sun, Haorui Wang, Shuibai Zhang, Wenjie Hu, Mya Schroder, Bilge Mutlu, Dawn Song, and Robert D. Nowak. The long-horizon task mirage? diagnosing where and why agentic systems break. *arXiv preprint arXiv:2604.11978*, 2026b. URL https://arxiv.org/abs/2604.11978.

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 preprint arXiv:2602.23701*, 2026c. URL https://arxiv.org/abs/2602.23701.

Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. TravelPlanner: A benchmark for real-world planning with language agents. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54590–54613. PMLR, 2024. URL https://proceedings.mlr.press/v235/xie24j.html.

Yifan Yu, Moyan Li, Shaoyuan Xu, Jinmiao Fu, Xinhai Hou, Fan Lai, and Bryan Wang. Correct: Condensed error recognition via knowledge transfer in multi-agent systems. *arXiv preprint arXiv:2509.24088*, 2025. URL https://arxiv.org/abs/2509.24088.

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. URL https://proceedings.mlr.press/v267/zhang25cq.html.

Xiangxin Zhao, Han Li, Shuaiting Li, Tianyi Zhao, Earl T. Barr, Federica Sarro, and He Ye. Failure as a process: An anatomy of CLI coding agent trajectories. *arXiv preprint arXiv:2607.09510*, 2026. URL https://arxiv.org/abs/2607.09510.

Chenyang Zhu, Spencer Hong, Jingyu Wu, Kushal Chawla, Yuhui Tang, Youbing Yin, Nathan Wolfe, Erin Babinsky, and Daben Liu. RAFFLES: Reasoning-based attribution of faults for LLM systems. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7659–7688. Association for Computational Linguistics, 2026a. doi: 10.18653/v1/2026.eacl-long.359. URL https://aclanthology.org/2026.eacl-long.359/.

Chenyang Zhu, Jiayu Yao, Kushal Chawla, Youbing Yin, Nathan Wolfe, Pengshan Cai, Jingyu Wu, Spencer Hong, Sangwoo Cho, Shi-Xiong Zhang, Daben Liu, Sambit Sahu, and Erin Babinsky. SAFARI: Scaling long horizon agentic fault attribution via active investigation. *arXiv preprint arXiv:2606.24626*, 2026b. URL https://arxiv.org/abs/2606.24626.

A LongRCA-Mini sampling and reproducibility

Sampling protocol. The sample-size calibration uses master seed 20260820 and 10,000 draws for each candidate size under equal-quota simple-random sampling. The released 200-QID manifest is generated with the independent master seed 20260821. It records a SHA-256 hash of the source dataset and a hash for every selected record. Final manifest selection is without replacement and uses only source membership, QIDs, and the fixed seed; no prediction outcomes or reference labels influence which QIDs are selected. The manifest is not rerolled to improve an observed result and contains 40 QIDs per source. Full-benchmark predictions from six evaluated methods under DS-V4-Flash are used to calibrate sample size, not to select individual trajectories. All methods share the sampled QIDs within each calibration draw and the same released manifest for the fixed-subset check. Agent Acc follows the independent explicit-output scoring protocol in Section 4.

The candidate sizes are 75, 100, 125, 150, 175, 200, 225, 250, 300, 400, and 600. Equal allocation caps each source at its population size and redistributes unused slots evenly, with deterministic rounding to preserve the requested total. All sampled scores retain equal source weights in Macro-5 even when source-size caps yield unequal quotas.

Score errors and calibration criteria. Calibration and fixed-manifest fidelity use the same full-benchmark predictions. For method m and metric k, absolute score error in percentage points (pp) is

$$D_{m,k} = 100 \left| S^{\text{Mini}}_{m,k} - S^{\text{Full}}_{m,k} \right|.$$

Both scores are fractions aggregated with Macro-5. Full reaggregates all 1,140 predictions per method using equal source weights, which differ from the original-proportion weighting in Section 4.2. Across the 18 method–metric pairs formed by the six methods and Agent Acc, Step Acc, and Step ± 5, a candidate size passes when

$$\max_{m,k} \text{Median}(D_{m,k}) \le 2 \text{ pp}, \qquad \max_{m,k} Q_{0.95}(D_{m,k}) \le 6 \text{ pp},$$

where each median and quantile is computed over repeated draws for one method–metric pair before taking the maximum across pairs. These predefined criteria depend only on absolute score errors.

As shown in Figure 7 and Table 6, $n = 200$ is the smallest tested size satisfying both error criteria; $n = 175$ exceeds the median-error threshold. The empirical p95 uses the nearest-rank definition; the maximum of per-pair p95 errors is not a simultaneous 18-pair coverage bound.

Equal allocation follows the equal-source Macro-5 objective; calibration determines the smallest passing size among the tested values for that protocol, not an optimum over all possible sampling designs or integer sizes. A supplementary repeat with seed 20260907 and 10,000 draws per candidate size also yields $n = 175$ failing and $n = 200$ as the smallest passing tested size.

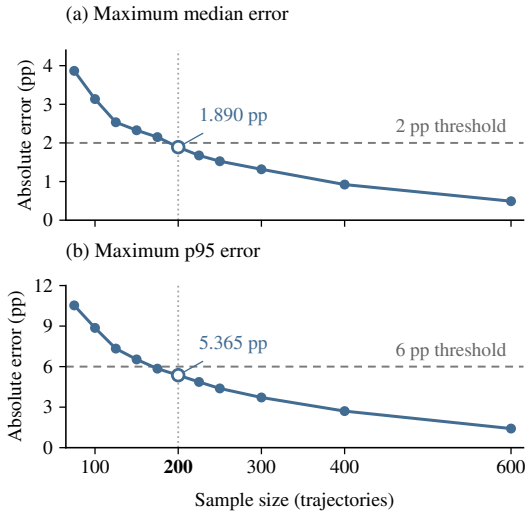


Figure 7: Mini calibration with cached predictions and 10,000 source-stratified draws per size: maximum per-pair median and 95th-percentile absolute Macro-5 errors across 18 method–metric pairs.

Table 6: Six-method sample-size calibration with equal-quota simple-random sampling. Quotas follow the source order SWE-bench Pro, Terminal-Bench 2, TravelPlanner, VitaBench, and WebArena Verified. For each of 18 method–metric pairs, absolute score errors are summarized over 10,000 draws. "Max. median" and "Max. p95" report the maximum per-pair median and 95th-percentile errors (pp). "Pass" requires only max. median ≤ 2 pp and max. p95 ≤ 6 pp.

Size	Quotas	Max. median	Max. p95	Pass
75	15/15/15/15/15	3.865	10.532	No
100	20/20/20/20/20	3.135	8.865	No
125	25/25/25/25/25	2.535	7.335	No
150	30/30/30/30/30	2.327	6.532	No
175	35/35/35/35/35	2.151	5.849	No
200	**40/40/40/40/40**	**1.890**	**5.365**	**Yes**
225	46/42/46/46/45	1.676	4.865	Yes
250	52/42/52/52/52	1.526	4.384	Yes
300	65/42/65/64/64	1.317	3.716	Yes
400	90/42/90/89/89	0.922	2.708	Yes
600	128/42/161/108/161	0.491	1.415	Yes

Fixed-manifest fidelity. The fixed-manifest check compares predictions on the released 200 QIDs with Full under Macro-5. Selection follows the prediction-independent rule in Section 2.4.

Table 7: Fidelity of the fixed LongRCA-Mini manifest. Each cell reports **Full** $\rightarrow$ **Mini** ($|\Delta|$): Macro-5 accuracy (%) and absolute difference (pp). Across the 18 method–metric comparisons, the mean absolute deviation is 1.665 pp and the maximum deviation is 3.771 pp. All results use DS-V4-Flash.

Method	Agent Acc	Step Acc	Step ± 5
All-at-once	48.2 $\rightarrow$ 48.5 (0.3)	9.9 $\rightarrow$ 8.5 (1.4)	23.7 $\rightarrow$ 21.5 (2.2)
Step-by-step	41.8 $\rightarrow$ 41.5 (0.3)	6.5 $\rightarrow$ 6.5 (0.0)	19.8 $\rightarrow$ 17.5 (2.3)
Binary search	40.8 $\rightarrow$ 39.5 (1.3)	4.9 $\rightarrow$ 7.5 (2.6)	18.1 $\rightarrow$ 20.0 (1.9)
ECHO	49.4 $\rightarrow$ 46.5 (2.9)	17.3 $\rightarrow$ 13.5 (3.8)	31.2 $\rightarrow$ 29.0 (2.2)
FALAT	33.1 $\rightarrow$ 34.5 (1.4)	2.7 $\rightarrow$ 3.0 (0.3)	15.5 $\rightarrow$ 18.0 (2.5)
RCTA	**60.9** $\rightarrow$ **61.0** (0.1)	**28.3** $\rightarrow$ **27.0** (1.3)	**44.1** $\rightarrow$ **41.0** (3.1)

Table 7 reports a mean absolute deviation of 1.665 pp and a maximum of 3.771 pp across the 18 method–metric pairs. These deviations are computed before rounding the displayed scores and apply

to this fixed manifest; calibration quantiles instead summarize repeated samples. The comparison measures subset-selection discrepancy under shared predictions and reference labels, not run-to-run inference variance. It supports preliminary comparisons, ablations, and configuration screening under the evaluated protocol, without establishing fidelity for future methods. Close comparisons, per-source claims, headline results, leaderboard submissions, changed data or scoring protocols, and materially different method behavior require confirmation on all 1,140 trajectories.

Table 8: Structural comparison of LongRCA Bench and the fixed LongRCA-Mini manifest. The KS statistic is the maximum across within-source comparisons of trajectory length, root-to-end distance, and normalized root position. History characters are used as a model-independent input-cost proxy.

Structural quantity	**Full**	**Mini**	**Comparison**
Trajectories	1,140	200	17.54% retained
Recorded steps	178,137	32,317	18.14% retained
History characters	207,731,218	41,784,984	20.11% retained
Macro-5 mean trajectory length	167.7	161.6	-6.1 steps
Macro-5 mean root-to-end distance	104.9	104.4	-0.5 steps
Macro-5 mean normalized root position	0.420	0.399	-0.021
Maximum within-source KS distance	–	–	0.177

As shown in Table 8, the fixed Mini retains 20.11% of history characters and 18.14% of recorded steps. The Macro-5 mean trajectory length differs from Full by 6.1 steps, and the mean root-to-end distance differs by 0.5 steps. Score fidelity is summarized in Table 7.

Table 9: Within-source structural audit of the fixed manifest. Entries are Full/Mini means; input length is measured in thousands of history characters, and normalized root position is $r^*/(T-1)$. KS is the largest empirical CDF distance across the four reported characteristics. SWE, TB2, TP, Vita, and WA denote SWE-bench Pro, Terminal-Bench 2, TravelPlanner, VitaBench, and WebArena Verified, respectively.

Source	**Steps**	**Input kchars**	**Root-to-end**	**Root position**	**KS**
SWE	287.0/263.0	450.2/432.9	215.4/218.5	0.250/0.181	0.177
TB2	272.1/265.6	297.2/294.9	186.9/179.5	0.263/0.268	0.040
TP	165.9/167.8	163.3/154.1	65.9/67.8	0.597/0.569	0.134
Vita	54.6/50.8	23.2/26.9	29.7/26.0	0.445/0.473	0.134
WA	58.8/60.8	131.2/135.8	26.8/30.1	0.543/0.502	0.147

As shown in Table 9, the maximum within-source KS distance ranges from 0.040 for Terminal Bench 2 to 0.177 for SWE-bench Pro. The largest change in mean trajectory length occurs in SWE-bench Pro, from 287.0 to 263.0 steps.

As shown in Table 10, RCTA Step Acc changes from 38.3% to 32.5% on SWE-bench Pro and from 25.0% to 32.5% on VitaBench. These source-level shifts describe variation within the fixed manifest; each Mini source contains only 40 trajectories.

For direct comparison with the full benchmark's natural source mixture, the Mini source scores can instead be post-stratified as $S_{\text{original}} = \sum_{b=1}^{5}(N_b/1140)S_b$.

In Table 11, Full and Mini use the original source proportions as their aggregation weights. This secondary estimand is separate from the equal-source Macro-5 aggregate used in the fidelity analysis above; the two types of aggregate should not be compared as if they used the same source weights.

Reproduction artifacts. The release includes scripts/sample_equal_benchmark_dataset.py, scripts/analyze_longrca_six_method_calibration.py, scripts/analyze_longrca_mini_fidelity.py, and scripts/plot_longrca_six_method_calibration.py, together with the fixed QID manifest, repeated-draw indices and scores, and per-size and per-method–metric results. Input hashes and the Python, NumPy, and Matplotlib versions are recorded in the analysis manifests.

The calibration code verifies identical 1,140-QID coverage across the six methods, exact quotas and no within-draw duplicate QIDs, and identical draw indices and scores under same-seed replay. The

Table 10: Per-source Full/Mini accuracy (%) for the fixed manifest. Each cell reports the full source score followed by the Mini source score. Source abbreviations follow Table 9. These values are an audit of source-level variation, not evidence that 40 trajectories support definitive per-source comparisons.

Method	Metric	SWE	TB2	TP	Vita	WA
All-at-once	Agent Acc	47.7/42.5	61.9/62.5	7.2/7.5	82.4/85.0	41.8/45.0
	Step Acc	1.6/2.5	7.1/5.0	4.5/5.0	19.4/20.0	16.9/10.0
	Step ±5	4.7/7.5	21.4/20.0	15.2/10.0	41.7/42.5	35.6/27.5
Step-by-step	Agent Acc	38.3/30.0	40.5/40.0	2.8/5.0	83.3/85.0	44.1/47.5
	Step Acc	0.8/2.5	4.8/5.0	3.5/0.0	12.0/12.5	11.3/12.5
	Step ±5	10.2/7.5	16.7/15.0	13.0/10.0	30.6/27.5	28.8/27.5
Binary search	Agent Acc	39.1/27.5	45.2/45.0	7.4/7.5	82.4/85.0	29.9/32.5
	Step Acc	0.8/2.5	4.8/2.5	2.0/7.5	12.0/17.5	5.1/7.5
	Step ±5	3.9/5.0	23.8/22.5	9.5/15.0	32.4/35.0	20.9/22.5
ECHO	Agent Acc	45.3/32.5	59.5/60.0	7.2/5.0	82.4/85.0	52.5/50.0
	Step Acc	7.8/7.5	21.4/22.5	8.9/7.5	22.2/20.0	26.0/10.0
	Step ±5	18.0/17.5	38.1/37.5	18.8/17.5	42.6/45.0	38.4/27.5
FALAT	Agent Acc	32.0/37.5	28.6/30.0	6.6/10.0	79.6/80.0	18.6/15.0
	Step Acc	2.3/5.0	2.4/2.5	2.9/2.5	3.7/2.5	2.3/2.5
	Step ±5	10.9/10.0	16.7/17.5	10.1/10.0	26.9/32.5	13.0/20.0
RCTA	Agent Acc	63.3/52.5	57.1/55.0	41.2/47.5	82.4/85.0	60.5/65.0
	Step Acc	38.3/32.5	26.2/25.0	18.8/22.5	25.0/32.5	33.3/22.5
	Step ±5	46.9/40.0	45.2/42.5	29.9/27.5	47.2/52.5	51.4/42.5

Table 11: Original-mixture fidelity audit. Cells report Full $\rightarrow$ post-stratified Mini ($|\Delta|$), in accuracy percentages and pp. Both sides use the original source weights, not Macro-5.

Method	Agent Acc	Step Acc	Step ±5
All-at-once	$26.2 \rightarrow 26.6$ (0.4)	$7.6 \rightarrow 6.9$ (0.7)	$19.9 \rightarrow 15.9$ (4.0)
Step-by-step	$22.2 \rightarrow 23.3$ (1.1)	$5.3 \rightarrow 3.6$ (1.7)	$16.9 \rightarrow 14.3$ (2.7)
Binary search	$23.0 \rightarrow 22.4$ (0.6)	$3.4 \rightarrow 7.7$ (4.3)	$13.3 \rightarrow 17.2$ (3.9)
ECHO	$27.5 \rightarrow 24.7$ (2.9)	$13.2 \rightarrow 9.6$ (3.5)	$24.7 \rightarrow 22.4$ (2.3)
FALAT	$19.0 \rightarrow 21.2$ (2.2)	$2.8 \rightarrow 2.8$ (0.0)	$12.5 \rightarrow 14.0$ (1.5)
RCTA	$51.1 \rightarrow 54.6$ (3.5)	$24.1 \rightarrow 24.7$ (0.5)	$37.4 \rightarrow 34.2$ (3.2)

sampling function accepts only source population sizes, sample size, repetition count, and seed; prediction scores are joined after drawing. The fixed-manifest fidelity check reuses the released 200 QIDs without modifying their selection.

B Complete cross-backbone results on LongRCA-Mini

In the complete results in Table 12, each source contributes 40 trajectories, so aggregate accuracies equal Macro-5. Joint Exact and Joint ±5 require both correct role attribution and the corresponding step criterion; Step MAE summarizes valid-output absolute root-step error. The entries in Table 12 are descriptive point estimates rather than repeated-run averages; paired uncertainty analyses and significance tests are reported separately below. This cross-backbone experiment is separate from the cached-prediction fidelity analysis in Appendix A.

Evaluation protocol and prediction coverage. DS-V4-Flash, DS-V4-Pro, GPT-5.6 Sol, and Kimi K3 each have 200 unique predictions per method on the shared 40-per-source sample, providing 4,800 method–trajectory records for the paired analysis below. Kimi K3 is supported by complete per-trajectory predictions, with the same fixed accuracy denominators as the other back-

Table 12: Complete six-metric record for the fixed 200-trajectory cross-backbone experiment. Accuracies are Macro-5 percentages; Step MAE is the reported valid-output mean absolute root-step error in steps. Bold marks the best value within each backbone and metric (including ties). Shading identifies RCTA. Evaluation settings and prediction coverage are detailed in the accompanying text.

Method	Agent Acc ↑	Step Acc ↑	Step ±5 ↑	Joint Exact ↑	Joint ±5 ↑	MAE ↓
DS-V4-Flash						
All-at-once	42.5	7.5	20.5	7.5	13.0	44.915
Step-by-step	38.5	6.5	18.0	6.5	10.0	41.340
Binary search	40.5	7.5	20.5	7.5	12.0	43.465
ECHO	45.0	13.0	28.0	13.0	18.0	40.930
FALAT	35.5	2.5	17.5	2.5	10.5	53.685
RCTA	**60.5**	**27.0**	**41.0**	**27.0**	**34.5**	**28.360**
DS-V4-Pro						
All-at-once	50.0	11.5	29.5	11.5	19.5	36.200
Step-by-step	47.5	10.5	23.0	10.5	16.0	41.865
Binary search	45.5	10.0	28.5	10.0	18.0	34.555
ECHO	53.5	18.0	35.5	18.0	25.5	36.540
FALAT	48.0	14.5	23.5	14.5	20.5	52.370
RCTA	**61.0**	**28.0**	**43.0**	**28.0**	**37.0**	**31.365**
GPT-5.6-sol						
All-at-once	53.0	17.0	34.0	17.0	26.0	42.690
Step-by-step	49.0	13.0	26.0	13.0	21.0	40.635
Binary search	51.5	15.5	31.5	15.5	25.5	32.710
ECHO	55.5	18.5	32.0	18.5	24.5	44.305
FALAT	52.0	14.0	29.0	14.0	21.5	39.820
RCTA	**64.5**	**30.0**	**42.0**	**29.5**	**38.0**	**31.685**
Kimi K3						
All-at-once	54.5	11.0	29.5	10.5	20.5	34.225
Step-by-step	44.0	9.5	23.5	9.5	16.0	48.855
Binary search	42.0	12.0	28.0	12.0	20.5	40.145
ECHO	55.0	20.5	37.0	20.5	28.0	**30.760**
FALAT	52.0	18.0	30.0	18.0	24.0	41.265
RCTA	**61.5**	**31.5**	**41.5**	**31.0**	**37.5**	31.995
Qwen3.5-35B-A3B						
All-at-once	**51.0**	3.5	18.0	2.5	11.0	**42.550**
Step-by-step	46.5	4.5	18.0	3.0	12.0	44.865
Binary search	40.0	5.0	18.0	4.0	12.5	43.560
ECHO	48.5	10.0	**26.5**	9.0	19.5	44.650
FALAT	41.0	11.0	20.0	11.0	15.5	51.345
RCTA	48.5	**13.5**	25.0	**13.5**	**21.0**	45.330

bones. The table retains the MAE exception on Kimi K3: ECHO obtains 30.760 steps, compared with 31.995 for RCTA. The paired RCTA–ECHO MAE difference is 1.235 steps (95% CI: $[-6.975, 9.235]$), so the point-estimate reversal is not a resolved MAE difference. Exact-localization gains need not imply superiority on every reported measure.

Qwen3.5-35B-A3B has 1,200 valid method–trajectory records (200 per method, 40 per source), with no missing, duplicate, or parse-failed predictions. Its scores use explicit role and root-step outputs; RCTA reads `root_cause.agent/step`. On this backbone, RCTA also leads Joint Exact (13.5%) and Joint ±5 (21.0%). Although ECHO has higher Step ±5 (26.5% versus 25.0%),

RCTA has higher Joint ± 5 (21.0% versus 19.5%), indicating more cases where role and tolerant step predictions are simultaneously correct. This aggregate observation alone does not establish a mechanism of improved role–step consistency.

Paired Step Acc comparisons. All paired comparisons retain the same 200 trajectories, with invalid predictions counted as errors. Table 13 reports pointwise 95% source-stratified paired-bootstrap intervals from 20,000 resamples and two-sided exact McNemar tests. Holm-20 adjusts the 20 Step Acc comparisons of RCTA with five baselines on four backbones; Holm-40 adjusts the combined 40-comparison family of Agent Acc and Step Acc. Both are reported to show sensitivity to the endpoint family, rather than treating pointwise intervals as simultaneous guarantees. These analyses use one retained prediction per method and backbone; resampling trajectories does not capture inference-time variability. Qwen3.5-35B-A3B is not included in this paired analysis.

Table 13: Paired RCTA–ECHO Step Acc differences on LongRCA-Mini ($n = 200$). Δ and pointwise 95% intervals are in percentage points; W/L counts RCTA-only/ECHO-only correct predictions. Holm-20 adjusts the Step Acc family; Holm-40 jointly adjusts Agent Acc and Step Acc.

Backbone	Δ	95% CI	W/L	Holm-20 p	Holm-40 p
DS-V4-Flash	$+14.0$	$[7.5, 20.5]$	39/11	0.00072	0.00217
DS-V4-Pro	$+10.0$	$[3.5, 17.0]$	36/16	0.00779	0.06007
GPT-5.6 Sol	$+11.5$	$[4.5, 19.0]$	40/17	0.00640	0.04162
Kimi K3	$+11.0$	$[5.0, 17.0]$	31/9	0.00322	0.01155

For DS-V4-Pro, the adjusted Step Acc comparison changes from $p = .00779$ under Holm-20 to $p = .060$ under Holm-40; the other three RCTA–ECHO comparisons remain below .05 under both families. A source-stratified task-cluster bootstrap that keeps repeated attempts together also gives positive pointwise RCTA–ECHO Step Acc intervals on all four backbones (184 task groups among 200 trajectories). These checks concern the evaluated samples and settings, not unseen benchmarks or backbones.

Sensitivity to source composition. For Table 14, we average the four model outcomes within each trajectory and then average over the retained benchmarks. Excluding each benchmark in turn leaves a mean gain of 8.3–13.0 percentage points on the remaining Mini trajectories. This is a sensitivity of the observed aggregate to source composition. Bootstrap resampling keeps a trajectory's model outcomes together and preserves the retained source proportions; the five intervals are pointwise, not simultaneous coverage guarantees.

Table 14: Leave-one-benchmark-out sensitivity of the four-backbone mean RCTA–ECHO Step Acc gain. Each row retains 160 trajectories from four benchmarks; differences and intervals are in percentage points.

Excluded benchmark	Mean gain	95% CI
SWE-bench Pro	$+8.3$	$[3.6, 13.1]$
Terminal-Bench 2	$+13.0$	$[8.6, 17.5]$
TravelPlanner	$+11.4$	$[6.4, 16.4]$
VitaBench	$+13.0$	$[7.8, 18.1]$
WebArena Verified	$+12.5$	$[7.3, 17.8]$

Complementary correct localizations. Figure 8 compares RCTA with the union of five baseline correct sets separately for the four backbones shown on LongRCA-Mini. Both directions of complementarity are retained: RCTA correctly localizes 23–35 trajectories per backbone that all five baselines miss, while the baselines correctly localize 27–40 that RCTA misses. RCTA's total correct count combines the "Both" and "RCTA only" segments; the latter measures unique coverage relative to the baselines, not total accuracy. The union is a gold-informed set summary, not the performance of a deployed ensemble or a cost-matched competitor. The four rows reuse the same 200 Mini trajectories and must not be added to count unique instances.

Across all four backbones, 26 trajectories have at least one correct RCTA prediction and no correct prediction from any of the 20 baseline method–backbone pairs. Conversely, 43 have a correct base-

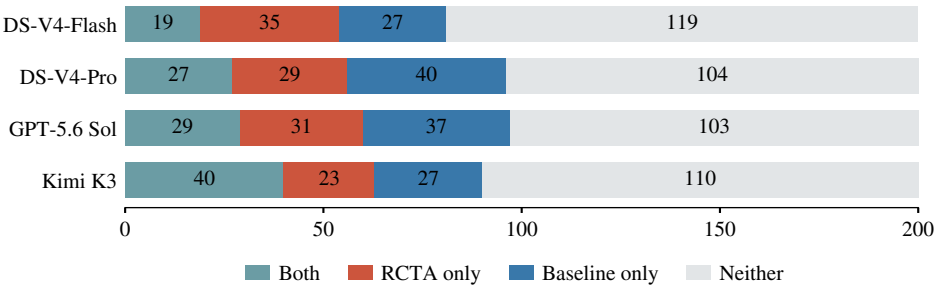


Figure 8: Exact-root correctness on LongRCA-Mini for four backbones ($n = 200$ each). Both: RCTA and at least one of five baselines are correct; RCTA only: RCTA is correct and all five baselines are wrong; Baseline only: RCTA is wrong and at least one baseline is correct; Neither: all six methods are wrong. Segment labels are trajectory counts.

line prediction and no correct RCTA prediction; 75 have successes from both method sets, and 56 have none. These counts summarize differently sized sets of observed predictions, not an ensemble comparison. Among the 26 RCTA-only trajectories, 18 have non-handoff reference roots. Inspecting all recorded role names further identifies 15 trajectories without any explicit handoff marker: eight from TravelPlanner, four from VitaBench, and three from WebArena Verified. The absence of the explicit marker does not establish the absence of implicit coordination or identify which component produced the successful attribution.

Illustrative roots with and without explicit handoffs. Two selected 162-step trajectories illustrate the observed coverage. In `swe_bench_pro__071`, the reference root is a handoff at step 27: the instruction requests a new `repoquery` parser but omits connecting it to `parseInstalledPackages` for the six-field output. In `travelplanner__458`, the reference root is step 84, where the accommodation agent proposes using estimated industry-average prices after its tool results fail to provide suitable listings for the required cities. This trajectory contains no explicit handoff marker. RCTA exactly matches the reference step on both cases with all four backbones, whereas none of the 20 baseline method–backbone results does. The cases illustrate distinct root structures; they do not estimate the prevalence of task-specific failure types or constitute a controlled comparison of components.

C Supplementary comparison with Who&When

We evaluate all six methods on the complete Who&When dataset (Zhang et al., 2025) using DS-V4-Flash. The dataset contains 184 trajectories: 126 from algorithm-generated systems and 58 from hand-crafted systems. We compare these results with the full LongRCA evaluation, weighting each trajectory equally within its dataset.

The five baselines compared in Section 1 obtain 30.4–36.4% Step Acc on Who&When and 2.8–13.2% on LongRCA. RCTA obtains 41.8% and 24.1%, respectively. It has the highest observed accuracy on all three metrics on LongRCA. On Who&When, it leads in Agent Acc and Step Acc, while All-at-once leads in Step ± 5.

All six methods have lower accuracy on LongRCA for each of the three metrics in this evaluation. The median trajectory length is 10 steps in Who&When and 145 in LongRCA, so a five-step localization error has a different relative scale in the two datasets. We use Step Acc as the primary localization comparison and report Step ± 5 as a complementary measure. The datasets also differ in domains and workflows, so the cross-dataset accuracy gaps do not isolate the effect of trajectory length.

D Released data schema

The benchmark distinguishes participant identity from functional role classes. Fields such as `name` and `role` preserve source-record metadata, whereas `mistake_agent` stores the reference partic-

ipant label used for responsible-role evaluation. Labels such as `ActionAgent` and `stay_agent` denote specific workflow participants; planner, executor, and verifier are broader method-internal classes and are not benchmark prediction targets. The responsible participant need not be the outer sender of the reference root-step record.

Normalization maps each retained execution to an ordered sequence of records covering multi-agent dialogue, planning, tool calls, observations, browser or terminal interactions, and evaluator feedback. Records preserve original content and roles, stable 0-based indices, and available tool and verifier metadata. Temporal order is preserved without synthesizing intermediate actions or subdividing source events into artificial steps. Annotators, reference labels, and evaluated methods share these indices within each trajectory; event granularity can still differ across sources.

Source evaluator outcomes guide failure selection and human annotation. Evaluated methods receive only `question_ID` and `history`, which includes the task instruction. Reference annotations are withheld during prediction, and only the responsible-role and root-step fields are scored. This field boundary does not remove evaluator or verifier feedback already recorded within the history.

Table 15: Released LongRCA Bench data schema. Evaluated methods receive `question_ID` and `history`, which includes the task instruction. The three reference annotation fields are withheld during prediction.

Field	Meaning
`question_ID`	Source-prefixed trajectory identifier
`history`	Ordered trajectory records containing `step`, `name`, `role`, and `content`; includes the task instruction
`mistake_agent`	Reference responsible role, scored independently of the root-cause step
`mistake_step`	Reference 0-based root-cause step, scored independently of the responsible role
`mistake_reason`	Trajectory-grounded annotation rationale, withheld during prediction and not scored

E SOURCE COLLECTION AND EXECUTION PROVENANCE

Table 16 reports task coverage and collection protocols; Table 17 records workflow implementations and generator counts in the released dataset. For TravelPlanner, tasks are shuffled within each split and difficulty level (seed 20260413), then assigned to one of the three generators. VitaBench single-domain tasks are shuffled within each domain (seed 20260407) and assigned across generators; its cross-domain tasks use MiniMax-M2.5. These assignments allocate tasks across models rather than running every task with every model.

Candidate failures require source-specific evaluator outcomes and usable records. SWE and Terminal collection accepts reward-zero trials without execution exceptions, with complete agent logs and participation by all three roles. TravelPlanner requires a failed evaluation and available execution, final-output, and evaluation records. VitaBench retains reward-zero trials, including step- or tool-error-budget exhaustion. WebArena retains failed evaluations from normally ended or timed-out runs, while excluding runner and evaluator errors. Infrastructure and evaluator errors, smoke tests, and debug runs are excluded; eligible budget-limited failures retain their histories through termination.

Before admission to the formal annotation pool, candidates are checked against the single-root scope in Section 2.1. Trajectories suspected of containing multiple independent decisive root causes across subtasks are excluded. All eligible failures enter the pool without further selection by length or difficulty; source, trajectory length, and task difficulty inform annotation assignments. Distinct attempts at the same task remain separate trajectories, and duplicate records are merged. Released counts refer to trajectories retained after annotation, quality control, and deduplication.

Source representation is uneven: TravelPlanner contributes 685 trajectories (60.1%), while the other four sources together contribute 455 (39.9%). MiniMax-M2.5 generated 681 trajectories, Kimi-K2.5 generated 276, and Qwen3.5-Plus generated 183. Only TravelPlanner and VitaBench contain

Table 16: Source task coverage and execution protocols. Task counts describe collection coverage, not released failures. Budgets use each runner's native units and are not bounds on normalized history length.

Source	Task coverage	Execution and additional attempts
SWE-bench Pro	Public-task collection batches	OpenHands custom team through Harbor. Tasks without a qualifying failure are rerun until qualification or the default five-round collection limit.
Terminal-Bench 2	Full 89-task suite (2.0)	The same OpenHands–Harbor workflow and conditional collection protocol; five rounds is a collection limit, not a fixed number of trials per task.
TravelPlanner	Train 45; validation 180; test 1,000	One assigned generator per task. Planning ends at `TASK_COMPLETE` or 12 team turns, followed by itinerary generation. Resumed batches skip completed outputs.
VitaBench	Single-domain 300 (100 each for delivery, in-store, and OTA); cross-domain 100	One trial per scheduled task; up to 100 simulation steps or 10 cumulative tool errors, with earlier agent termination. Selected unfinished runs receive additional attempts.
WebArena Verified	All shopping (187) and reddit (106) tasks; MiniMax-M2.5	One formal batch record per task. Up to six coordination cycles or 600 s, with at most ten browser actions per web-surfer step; earlier termination on critic completion/failure or two cycles without progress.

Table 17: Execution provenance and generator counts for LongRCA Bench. Generator columns correspond to MiniMax-M2.5 (M2.5), Kimi-K2.5 (K2.5), and Qwen3.5-Plus (Q3.5+). A dash indicates that the source contains no trajectory from that generator.

Source	N	Workflow	M2.5	K2.5	Q3.5+
SWE-bench Pro	128	OpenHands `DiagnostAgent–ActionAgent–JudgeAgent` team via Harbor	128	–	–
Terminal-Bench 2	42	OpenHands `DiagnostAgent–ActionAgent–JudgeAgent` team via Harbor	42	–	–
TravelPlanner	685	AutoGen `MagenticOneGroupChat` with specialist roles	265	260	160
VitaBench	108	`Action_Expert–Computer_terminal` workflow	69	16	23
WebArena Verified	177	AutoGen `Planner–WebSurfer–Critic` controller	177	–	–
Total	**1,140**		**681**	**276**	**183**

trajectories from all three generators. These counts describe trajectory generation rather than the inference backbones used for attribution.

F Localization by source and log characteristics

As shown in Tables 18 and 19, RCTA has the highest observed Step Acc in each source and each length group. These values underlie Figure 9 and the length-group analysis in Section 4.2. Both use the same full-benchmark predictions reported there. Each accuracy denominator includes all trajectories assigned to the corresponding source or length group.

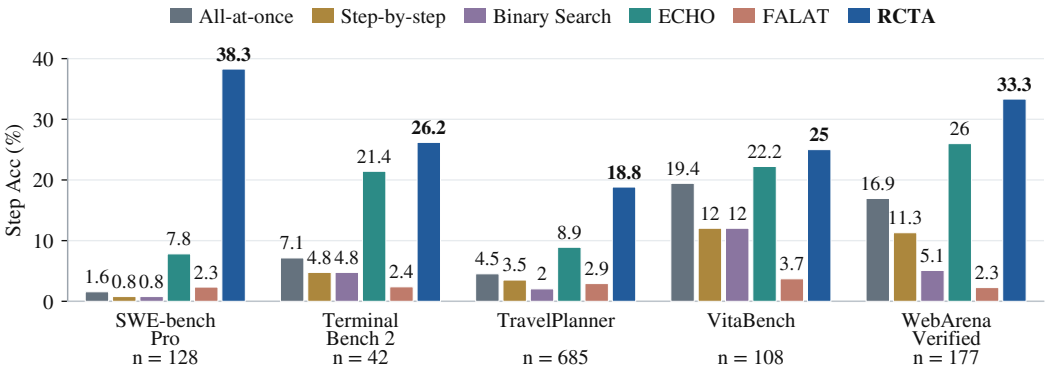


Figure 9: Step Acc (%) across five sources using DS-V4-Flash. All six methods use the same complete source subsets (n below source labels; total 1,140); invalid step outputs count as errors.

Table 18: Full-benchmark Step Acc (%) by source, underlying Figure 9. Source sample sizes are 128, 42, 685, 108, and 177, respectively. Every method uses the complete corresponding source subset; bold marks the highest observed accuracy in each source.

Method	SWE-bench Pro	Terminal Bench 2	TravelPlanner	VitaBench	WebArena Verified
All-at-once	1.6	7.1	4.5	19.4	16.9
Step-by-step	0.8	4.8	3.5	12.0	11.3
Binary search	0.8	4.8	2.0	12.0	5.1
ECHO	7.8	21.4	8.9	22.2	26.0
FALAT	2.3	2.4	2.9	3.7	2.3
RCTA	**38.3**	**26.2**	**18.8**	**25.0**	**33.3**

Table 19: Step Acc (%) across the five shared trajectory-length groups in Section 4.2. The groups contain 290, 211, 212, 215, and 212 trajectories, respectively. Bold marks the highest observed accuracy in each group.

Method	10–63	64–129	130–172	173–220	221–728
All-at-once	21.4	6.2	2.8	0.9	1.9
Step-by-step	13.4	3.3	2.8	3.3	0.5
Binary search	7.2	3.8	0.5	2.8	1.4
ECHO	27.9	11.4	7.1	7.9	6.1
FALAT	4.1	3.8	0.0	3.3	2.4
RCTA	**34.8**	**18.5**	**21.2**	**18.6**	**23.6**

Within the longest group, 170 of 212 trajectories (80.2%) have 221–400 steps. The RCTA–ECHO Step Acc difference in this group is 17.5 percentage points; it does not isolate performance on the extreme longest trajectories.

Shared length partition. The partition is constructed from one length per unique trajectory, with no prediction outcomes as inputs. Dynamic programming selects four cut points between distinct observed lengths to minimize the sum of squared deviations of group sizes from $1{,}140/5$, while keeping equal lengths together. The selected boundaries are 63, 129, 172, and 220 steps. A concentration of 145 trajectories at exactly 63 steps limits how closely the groups can approach equal sizes. Five groups provide a descriptive resolution for the main figure.

Sensitivity to the number of groups. Repeating this length-only partition with four and six groups gives group sizes of 290/281/283/286 and 145/198/199/200/198/200, respectively. RCTA has the highest observed Step Acc in every group under all three choices, whereas local curve shapes vary.

This sensitivity check supports the method comparison across these partitions while motivating caution when interpreting individual fluctuations. Source domains and workflows differ across the resulting groups.

F.1 WITHIN-SOURCE LOCALIZATION PATTERNS

Using the full-benchmark DS-V4-Flash predictions for all six methods, we examine localization within each source along two dimensions: root-to-end distance, $D = T-1-r^*$, and log text volume, measured by the total number of Unicode characters in history content, including whitespace. The former counts recorded activity after the reference root; the latter measures the amount of log text.

For each source and dimension, we choose the largest feasible number of groups up to three, with at least 30 trajectories per group and equal values kept together. Cut points minimize the sum of squared deviations from equal group sizes using only the grouping variable. All six methods share each group's trajectories, and invalid predictions count as errors. Terminal-Bench 2 has 42 trajectories and remains a single group. Figure 10 shows Step Acc. The accompanying analysis data include Wilson 95% intervals for group accuracies and Newcombe 95% intervals for each method's difference between the highest and lowest groups (Newcombe, 1998). These intervals are pointwise and conditional on the fixed groups.

Root-to-end distance shows heterogeneous patterns. On VitaBench, all six methods attain their highest Step Acc and Step ± 5 in the middle group; on TravelPlanner, differences between the highest and lowest groups have mixed signs across methods and metrics. Log text volume shows a more consistent contrast in these two sources: all six methods score lower in the highest-character group than in the lowest-character group on both metrics. Step Acc changes range from -8.8 to -0.9 pp on TravelPlanner and from -33.3 to -2.8 pp on VitaBench, with some intermediate-group reversals. Patterns on SWE-bench Pro and WebArena Verified vary by method. These source-specific associations do not establish a common monotonic trend or isolate a causal effect of subsequent activity or log text volume.

G ROOT-CAUSE DECISION RULES

The rules below specify common boundary cases for root-cause annotation under the task definition in Section 2.1.

- **Error already present in a handoff instruction.** If a handoff instruction contains the earliest recorded introduction of a decisive error that makes the subsequent execution irrecoverable, the handoff step is selected even when a later participant faithfully executes it.
- **Error introduced after a handoff.** A later execution step is selected when it introduces a new decisive error that is absent from the handoff instruction and makes the subsequent execution irrecoverable.
- **Successfully repaired error.** An earlier error is excluded if it is corrected before the execution ends.
- **Error missed during later verification.** A later verifier may fail to detect an earlier error. This omission may be retained as unscored explanatory context, but it does not replace the step that introduced the error.
- **Contradictory task and evaluation requirements.** An input step may be selected only if the task instruction and the official evaluator or test requirement are explicitly incompatible. Both texts must be cited from the recorded trajectory.
- **Earliest supported introduction.** Among candidates supported by the original log, the root is the earliest recorded step that introduced a decisive error making the subsequent execution irrecoverable. An earlier minor, recoverable error that is not necessarily fatal does not displace a later decisive error.

These rules govern root-cause-step annotation. The responsible role is annotated separately, and role and step predictions are scored independently.

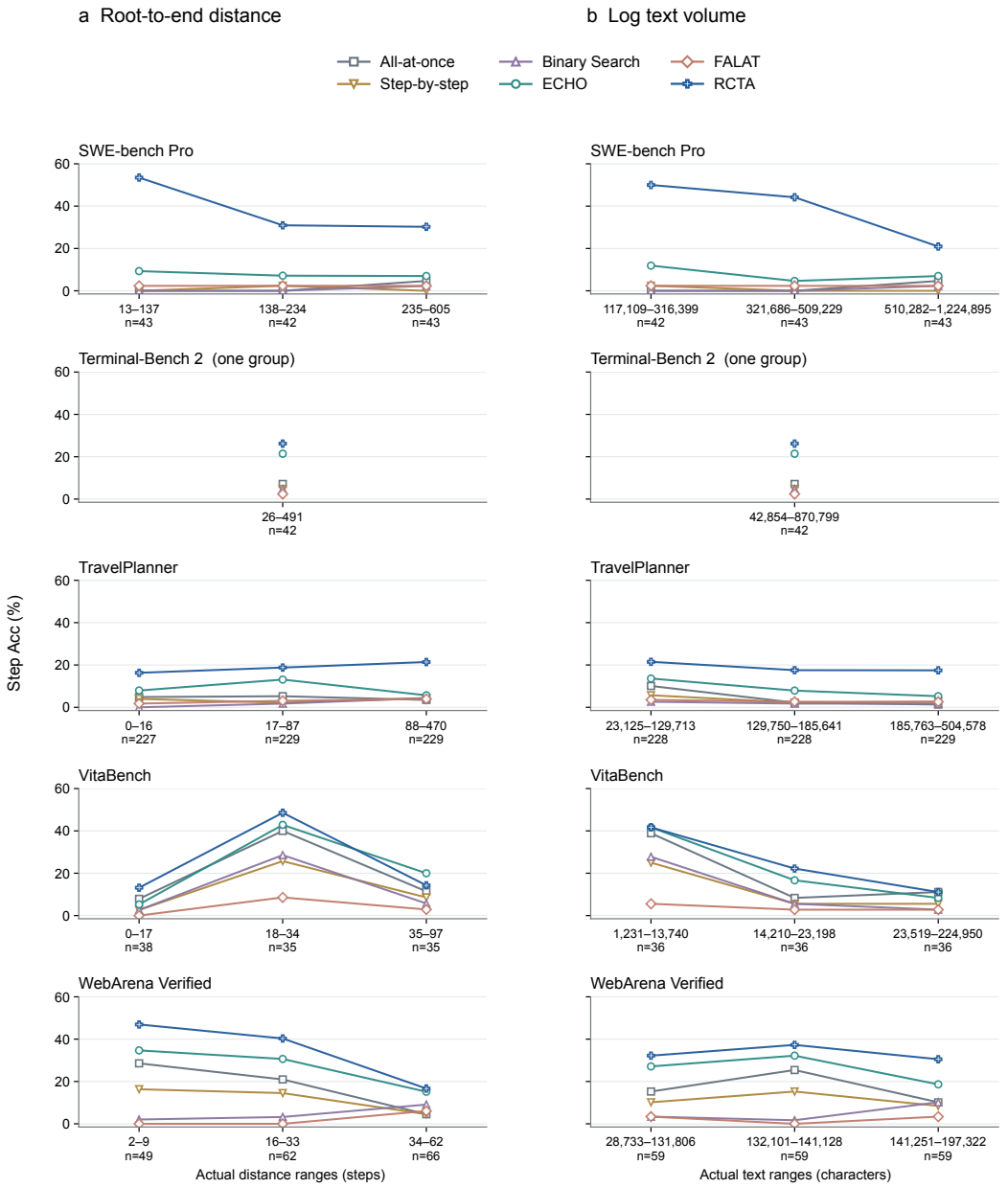


Figure 10: Within-source Step Acc with DS-V4-Flash by (a) root-to-end distance and (b) log text volume. All six methods share the trajectories in each group. Labels give actual ranges and sample sizes; unequal-width groups are equally spaced, and connecting lines are visual guides. Terminal-Bench 2 remains a single group under the minimum-size rule. Table 20 reports both localization metrics.

H Prompt excerpts, baseline adaptations, and implementation details

H.1 Overview

This appendix presents selected prompt excerpts and concise summaries of RCTA and the baseline adaptations. Boxes marked *excerpt* preserve selected instruction text; prose, field tables, and boxes marked *summary* condense the input and output requirements. These are not full executable templates. The full templates are included in the anonymous reproduction archive

Table 20: Within-source localization with DS-V4-Flash. Each method cell reports Step Acc / Step ± 5 (%). All methods share the trajectories in each row. SWE, TB2, TP, Vita, and WA denote SWE-bench Pro, Terminal-Bench 2, TravelPlanner, VitaBench, and WebArena Verified, respectively.

Source	Group	Actual range	n	All-at-once	Step-by-step	Binary search	ECHO	FALAT	RCTA
Root-to-end distance (steps)									
SWE	1	13–137	43	0.0 / 7.0	0.0 / 7.0	0.0 / 4.7	9.3 / 18.6	2.3 / 2.3	53.5 / 62.8
	2	138–234	42	0.0 / 2.4	2.4 / 11.9	0.0 / 2.4	7.1 / 19.0	2.4 / 9.5	31.0 / 38.1
	3	235–605	43	4.7 / 4.7	0.0 / 11.6	2.3 / 4.7	7.0 / 16.3	2.3 / 20.9	30.2 / 39.5
TB2	1	26–491	42	7.1 / 21.4	4.8 / 16.7	4.8 / 23.8	21.4 / 38.1	2.4 / 16.7	26.2 / 45.2
TP	1	0–16	227	4.8 / 10.1	4.0 / 11.9	0.0 / 2.2	7.9 / 13.2	1.8 / 9.3	16.3 / 31.7
	2	17–87	229	5.2 / 15.3	2.2 / 9.2	1.7 / 10.0	13.1 / 23.1	3.1 / 10.5	18.8 / 27.9
	3	88–470	229	3.5 / 20.1	4.4 / 17.9	4.4 / 16.2	5.7 / 20.1	3.9 / 10.5	21.4 / 30.1
Vita	1	0–17	38	7.9 / 28.9	2.6 / 21.1	2.6 / 15.8	5.3 / 26.3	0.0 / 7.9	13.2 / 44.7
	2	18–34	35	40.0 / 57.1	25.7 / 42.9	28.6 / 54.3	42.9 / 60.0	8.6 / 48.6	48.6 / 60.0
	3	35–97	35	11.4 / 40.0	8.6 / 28.6	5.7 / 28.6	20.0 / 42.9	2.9 / 25.7	14.3 / 37.1
WA	1	2–9	49	28.6 / 57.1	16.3 / 42.9	2.0 / 8.2	34.7 / 46.9	0.0 / 12.2	46.9 / 69.4
	2	16–33	62	21.0 / 38.7	14.5 / 29.0	3.2 / 19.4	30.6 / 40.3	0.0 / 3.2	40.3 / 50.0
	3	34–62	66	4.5 / 16.7	4.5 / 18.2	9.1 / 31.8	15.2 / 30.3	6.1 / 22.7	16.7 / 39.4
Log text volume (characters)									
SWE	1	117,109–316,399	42	0.0 / 2.4	2.4 / 9.5	0.0 / 0.0	11.9 / 26.2	2.4 / 9.5	50.0 / 61.9
	2	321,686–509,229	43	0.0 / 7.0	0.0 / 11.6	0.0 / 7.0	4.7 / 9.3	2.3 / 4.7	44.2 / 53.5
	3	510,282–1,224,895	43	4.7 / 4.7	0.0 / 9.3	2.3 / 4.7	7.0 / 18.6	2.3 / 18.6	20.9 / 25.6
TB2	1	42,854–870,799	42	7.1 / 21.4	4.8 / 16.7	4.8 / 23.8	21.4 / 38.1	2.4 / 16.7	26.2 / 45.2
TP	1	23,125–129,713	228	10.1 / 22.8	5.7 / 18.0	2.6 / 14.5	13.6 / 27.6	3.5 / 17.1	21.5 / 38.2
	2	129,750–185,641	228	2.2 / 14.9	2.2 / 11.8	1.8 / 7.0	7.9 / 15.8	2.6 / 8.3	17.5 / 25.9
	3	185,763–504,578	229	1.3 / 7.9	2.6 / 9.2	1.7 / 7.0	5.2 / 13.1	2.6 / 4.8	17.5 / 25.8
Vita	1	1,231–13,740	36	38.9 / 72.2	25.0 / 55.6	27.8 / 55.6	41.7 / 66.7	5.6 / 52.8	41.7 / 80.6
	2	14,210–23,198	36	8.3 / 25.0	5.6 / 19.4	5.6 / 25.0	16.7 / 33.3	2.8 / 19.4	22.2 / 30.6
	3	23,519–224,950	36	11.1 / 27.8	5.6 / 16.7	2.8 / 16.7	8.3 / 27.8	2.8 / 8.3	11.1 / 30.6
WA	1	28,733–131,806	59	15.3 / 42.4	10.2 / 35.6	3.4 / 22.0	27.1 / 39.0	3.4 / 15.3	32.2 / 50.8
	2	132,101–141,128	59	25.4 / 40.7	15.3 / 28.8	1.7 / 20.3	32.2 / 39.0	0.0 / 8.5	37.3 / 50.8
	3	141,251–197,322	59	10.2 / 23.7	8.5 / 22.0	10.2 / 20.3	18.6 / 37.3	3.4 / 15.3	30.5 / 52.5

`longrca-anonymous-review-v1.zip`, at `docs/prompts/appendix_prompt_templates.pdf` (readable PDF) and `docs/prompts/appendix_prompt_templates.tex` (copyable LaTeX source).

RCTA pairs a system prompt with a user message at each of three stages: local summarization, trajectory-outline aggregation, and final attribution. These implement, but are not identical to, the two scientific stages and four components in Section 3. The user input is derived from the released identifier and recorded trajectory. Task instructions are read from the trajectory, and reference roles, root steps, and rationales are excluded. Step identifiers and agent names retain their recorded values.

All 1,140 full-benchmark RCTA predictions use the same three system-prompt templates. They contain task rules and output-schema guidance, but no LongRCA Bench instances or input–output demonstrations. A trajectory with m segments requires $m + 2$ initial LLM calls before any validation-triggered retry: one per segment, one outline call, and one final-attribution call.

Table 21: Prompt requirements and programmatic membership checks for the three RCTA stages. The local-summary template is applied once per segment; the final role and root fields are validated independently.

Stage	User-message inputs	Output summary	Validation
Local	Segment, preceding overlap, valid local IDs, and recorded verification signals	Segment goal, key events, candidate failures, findings, and uncertainties	All IDs occur in the segment
Outline	Local summaries, global step IDs, and verification-signal summary	Contiguous phases, supporting step IDs, and cross-segment conflicts	All IDs occur in the trajectory
Attribution	Outline, summaries, candidate records, upstream handoffs, valid roles and IDs	Responsible role, root step, supporting evidence, and delegation comparison	Role and IDs occur in the trajectory; cited text is grounded

H.2 LOCAL SUMMARY

Scope (excerpt).

```
You are an expert in analyzing failed multi-agent trajectories. At this stage,
examine **only one segment of the trajectory** and extract its key events,
suspected errors, and verification signals. **Do not** determine the root cause
of the entire trajectory; that is the responsibility of a later stage.
```

Evidence identifiers (excerpt).

```
2. Every integer in a `step_ids` array **must** appear in `step_id_list`. If you
need to refer to an overlap step, describe it in the text of `uncertainties`
instead.
```

User message and output (summary). contains the current records; <OVERLAP_CONTEXT> supplies preceding context only. The user message also supplies step_id_list, recorded role names, and verification signals. The model returns a JSON summary with segment_goal, key_events, candidate_failures, verifier_findings, and uncertainties. Suspected errors are retained as candidates rather than treated as final causal conclusions.

Source: prompts/local_summary.md; user-message construction in analyze/local_summarizer.py.

H.3 TRAJECTORY OUTLINE

Aggregation scope (excerpt).

```
You are an expert in analyzing failed multi-agent trajectories. You have received
the LocalSummary list for all segments. Now **aggregate them into 3-8
semantically meaningful phases** and identify conflicts or inconsistencies across
phases. **Do not** make the final root-cause determination; that is the next
stage's responsibility.
```

Continuity and retrieval (summary). Each phase has a step range, goal, involved agents, contiguous sub-phases, failure signals, and supporting step IDs. Sub-phases cover the entire phase without overlaps or gaps. Supporting IDs identify records to retrieve for final attribution. Cross-segment contradictions are kept in conflicts, rather than silently resolved during summarization.

User message (summary). The inputs are <LOCAL_SUMMARIES>, <VALID_STEP_IDS>, and <VERIFIER_SIGNAL_SUMMARY>, together with the identifier-only task brief. No new root-cause decision is made at this stage.

Source: prompts/phase_aggregate.md; analyze/global_reducer.py:aggregate_phases.

H.4 FINAL ATTRIBUTION

H.4.1 ERROR ORIGIN AND COUNTERFACTUAL COMPARISON

Counterfactual criterion (excerpt).

```
Criterion: if the history before this step remains unchanged and only this step's
erroneous behavior is replaced with a reasonable correct action, and the
subsequent task would likely avoid failure or move substantially toward success,
this step may be considered the root-cause step.
```

Error origin (excerpt).

```
In multi-agent trajectories, several agents may appear responsible. **Do not stop
at the first step where a defect becomes visible; trace back to the earliest step
that introduced it.**
```

Selection rule (summary). Compare plausible earlier origins with later visible errors. Prefer the earliest decisive error; an early error that is minor, recoverable, and not necessarily fatal does not displace a later step that makes the execution irrecoverable. Distinguish an error's introduction from its subsequent propagation and manifestation. The counterfactual is an evidence-based diagnostic judgment, not a measured repair outcome. The same selection rule is used in the task definition in Section 2.1.

User-message inputs (summary). `<TRAJECTORY_OUTLINE>` and `<LOCAL_SUMMARIES>` provide context. `<CANDIDATE_RECORDS>` contains retrieved original text, including available upstream handoffs. Separate linkage hints identify possible instruction–execution relationships. The message also supplies `<VALID_ROLE_NAMES>` and `<VALID_STEP_IDS>`. Handoff records are part of the excerpt set, not a separate runtime field.

Source: `prompts/root_cause.md`, Sections 2 and 6;
`analyze/global_reducer.py:diagnose_root_cause`.

H.4.2 UPSTREAM AND DOWNSTREAM RESPONSIBILITY

The final comparison tests whether the downstream agent inherited a defect specified upstream or introduced a new error during execution.

Instruction–execution comparison (excerpt).

```
**"Would implementing this plan literally produce this failure?"**
```

Decision rules (summary).

1. Read the preceding handoff and compare its literal instructions with the recorded task requirements and downstream action. Plan defects can be incorrect values, faulty control flow or omissions, or an explicitly incompatible approach, tool, or data source.
2. Trace the root upstream when faithful execution implements the same defect already present in that handoff. Quote the instruction containing the defect and record the comparison in `delegation_trace`.
3. Keep the root at the execution step when the executor deviates from, misreads, or adds an error outside the plan. Shared identifiers or overlapping scope alone do not establish inherited error.
4. Distinguish a missed safeguard from the step that introduced the defect. Do not attribute an executor's independent error to a planner merely because that planner delegated the work.

H.4.3 INDEPENDENT OUTPUTS AND EVIDENCE CONSTRAINTS

Role and step (excerpt).

```
Output the responsible role and root step separately; in cases of transferred
responsibility, the role need not mechanically match the sender of that step's
message.
```

Evidence rules (summary). The responsible role must occur among the recorded role names, and the root step and supporting evidence IDs must occur in the trajectory. The explanation must cite specific task instructions, observed failures, code, or handoff text. Upstream attribution requires a quotation grounded in the referenced handoff; text matching checks provenance, not causal correctness. If evidence is insufficient, the model can abstain and report low confidence.

Selected output fields	Meaning (schema summary)
`root_cause_event.agent`, `.step`	Separately predicted responsible role and earliest decisive root step
`reason`, `evidence_step_ids`	Causal explanation and recorded supporting steps
`delegation_trace`	Whether upstream instructions were checked, the referenced handoff, its quotation, and whether the quoted instruction contains the defect
`failure_manifestation_event`, `failure_chain`	Observable error and explanatory propagation chain; distinct from the scored root prediction
`abstain`, `confidence`	Uncertainty and evidence insufficiency

Source: `prompts/root_cause.md`, Sections 2, 6, 8, and 9. The expanded archive retains the full taxonomy, JSON schema, exceptional cases, compatibility fields, and validation/retry text.

H.5 BASELINE TASK AND METHOD-SPECIFIC DIFFERENCES

The following summaries describe the local LongRCA adaptations. The shared attribution instruction is shown once for All-at-once, Step-by-step, Binary Search, and ECHO. FALAT uses the stage-specific criteria in the next subsection.

Shared attribution objective (excerpt).

```
Task: root-cause failure attribution. Identify the earliest decisive root-cause
error step leading to the final failure, rather than the final symptom or last
reported error.
```

Shared input and output (summary). The user message supplies task text, recorded agent names, the valid step range, and compressed trajectory text. The final attribution fields are `predicted_agent`, `predicted_step`, `reasoning`, `confidence`, and `evidence_steps`. Step indices are global and 0-based. The ordinal convention is used by default for All-at-once, Step-by-step, and Binary Search; ECHO uses explicit Step/turn indices.

H.5.1 ALL-AT-ONCE

A single attribution call reads the compressed full trajectory. Each history entry is retained, with record content compressed as needed. There is no additional method-specific reasoning instruction beyond the shared objective.

H.5.2 STEP-BY-STEP

The model scans each consecutive chunk for a candidate error, returning a `has_candidate` flag and an agent/step proposal. After all chunks, a global judge compares the collected candidates with the trajectory outline.

Global judgment (excerpt).

```
Consider all candidates and select the earliest decisive root-cause error step
(0-indexed), rather than the final symptom.
```

H.5.3 BINARY SEARCH

Each comparison receives two adjacent trajectory halves and returns `choice:left/right` with a reason. The retained range is recursively narrowed, then passed to the shared attribution call with an explicit range restriction.

Half comparison (excerpt).

```
Decide which half is more likely to contain the earliest decisive root-cause
error leading to the final failure.
```

H.5.4 ECHO

Three analysts receive the shared attribution message with different prefixes: *general* emphasizes global judgment, *detail* emphasizes tool errors and other high-signal steps, and *skeptical* distinguishes early causes from later symptoms. Confidence is accumulated for each normalized agent–step pair; ties favor the earlier step. Consensus is computed in code, without a separate model prompt.

Sources: `Benchmark/early_detection/baselines/full_log.py`: `_attr_task`, `CHUNK_TASK`, `JUDGE_TASK`, `COMPARE_TASK`, and `ECHO_FOCI`.

H.6 FALAT: STAGE STRUCTURE AND DECISIVE-ERROR CRITERIA

FALAT organizes the trajectory into hierarchical representations and typed dependencies before checking candidate errors. Its templates are summarized below; only the key verification and local re-search instructions are quoted.

Stage	Distinctive instruction (summary)
External prior (optional)	Identify task requirements, agent roles, and likely error signals without planning a solution.
Hierarchical representation	Produce high-level task/strategy notes, mid-level key moments, and low-level evidence with original step references.
Typed dependencies	Compare candidate pairs using `follow_up`, `redundancy`, `no_influence`, `error_shift`, `correction`, or `dead_end`; prune candidates and trace value flow.
Candidate selection	Rank 2–3 plausible decisive errors from the refined candidate set using the step-to-agent mapping.
Verification	Check error independence, counterfactual sufficiency, and temporal precedence against retrieved evidence.
Local re-search	Challenge the preliminary diagnosis in the $[t-2, t+2]$ neighborhood, then confirm or revise it.

Candidate verification (excerpt).

```
1. Error independence. Does this step introduce the error, or merely inherit it
from an upstream fault? Use the typed dependency structure to trace whether the
candidate is an error source or a downstream carrier.
2. Counterfactual sufficiency. If this step's output, decision, or assumption
were corrected, would the final answer recover the expected output while keeping
prior steps fixed and allowing subsequent steps to adapt?
3. Temporal precedence. Among candidates satisfying the above criteria, prefer
the earliest step.
```

Local challenge (excerpt).

```
Be critical. Do not simply confirm the current diagnosis; challenge it against
nearby alternatives.
```

Local decision and output (summary). For nearby steps, the model gives arguments for and against causal responsibility, assigns suspicion scores, and distinguishes root cause, propagation, symptom, and contributing error. It applies the counterfactual test to the top three candidates, then returns `CONFIRM` or `REVISE` with the decisive agent and step. Candidates outside the bounded local window are ineligible for that final local decision.

Source: `Benchmark/early_detection/baselines/falat.py`, the six `_build_*_prompt` methods. Full stage templates, output schemas, and parsing-retry messages are included in `longrca-anonymous-review-v1.zip` under `docs/prompts/appendix_prompt_templates.pdf` and the corresponding `.tex` file. See `docs/prompts/README.md` in that archive for the template and code index.

H.7 RCTA IMPLEMENTATION AND INFERENCE SETTINGS

Validation and output recovery. The local and outline validators check the identifier memberships summarized in Table 21. If the phase-construction response cannot be parsed, the implementation assigns one phase to each local summary. The final-attribution validator checks the role and root fields independently; an invalid field does not overwrite a valid prediction in the other field. The marker `system_evaluation` may appear only as an unscored terminal node in the explanatory failure chain; it cannot be the predicted root step. After the single validation-triggered retry, valid role and root fields are preserved independently, unsupported fields are removed, and the output is flagged for review. The system abstains only when an essential field remains invalid (Algorithm 1).

Segmentation and overlap. A *segment* is a consecutive block of original steps; a *phase* groups adjacent segment summaries serving the same subgoal without repartitioning recorded steps. A *handoff instruction* is a logged `X (-> Y)` message. For segmentation, a *verifier signal* is a rule-extracted `PASS` or `FAIL` marker from explicit language in a verifier or terminal record. The signal serves only as an optional boundary cue, not as a semantic judgment, benchmark label, or the source evaluator's final outcome.

Segmentation uses rule-based implementation settings, not learned parameters. When adding the next step would exceed 32,000 characters, the implementation starts a new segment when possible; an individual overlong step may exceed this limit. Each segment contains at most 80 non-overlap steps. A boundary after a finish step, or before a role-changing handoff instruction or completion step, requires at least eight steps and 10,000 characters in the current segment. A boundary before a change in the most recent explicit verifier signal requires 10,000 characters but no eight-step minimum. After the 24,000-character target, an ordinary source-role change can form a soft boundary.

Except for the first, each segment receives up to five preceding steps from the previous segment as overlap context.

Handoff quotation validation. For an upstream attribution, the checker accepts a normalized substring match or sufficient word overlap between the cited text and the referenced handoff instruction. The evidence-provenance boundary is described in Section H.4.

Candidate ordering and role matching. Candidate hints are deduplicated by integer step ID, merging distinct role, source, and note fields. The fixed retrieval priority is possible manifestation (0), possible root (1), terminal context (2), and other candidates (3); a recovered-noise hint adds 3. If several tags occur, the first matching category in this order sets the base priority. Ties use ascending step ID. The first 80 candidates are retained before handoff augmentation; the expanded set is deduplicated without a second 80-step cap. Actual excerpts contain only IDs present in the trajectory. These rules prioritize evidence retrieval, not the final root-cause decision. They are implemented by `_add_candidate`, `_candidate_priority`, and `_collect_candidate_step_hints` in `analyze/global_reducer.py` in the released implementation.

Role classes are determined by recorded names in `preprocess/step_enricher.py`. Planner names are `DiagnostAgent`, `Task_Planner`, `Orchestrator`, and `manager_agent`;

executor names are `ActionAgent`, `Action_Expert`, `WebSurfer`, `FileSurfer`, `Assistant`, `Coder`, `transport_agent`, `stay_agent`, and `food_attr_agent`; verifier names are `JudgeAgent`, `Verification_Expert`, and `verifier_agent`. Classification trims whitespace and extracts the sender from a handoff name. Names ending in `Expert` or `_Expert` map to verifier if they contain `verif` or `validat` (case-insensitive), and otherwise to executor. `Computer_terminal` and `ComputerTerminal` map to terminal, and `human` maps to human; unmatched names remain unknown. An `Orchestrator` status suffix retains the planner class.

For executor or verifier candidates, handoff augmentation requires the logged recipient to equal the candidate's raw agent name, with a case-sensitive comparison; matching the broad role class is insufficient. It selects the nearest strictly earlier matching handoff and uses no cross-recipient fallback. Other candidate roles use the nearest earlier handoff regardless of recipient. The relevant functions are `_delegate_target`, `_nearest_preceding_delegate`, and `_augment_with_preceding_delegates` in the same reducer module.

The implementation also contains role-name-triggered recall rules for a configured family of planner–executor logs. These rules can add visible executor errors, planner decisions, recovered-error context, and terminal context before candidate prioritization. None of the 1,140 LongRCA Bench trajectories matched the configured role-name signature, so these rules did not affect the LongRCA results.

Ablation inference settings. Component comparisons use DS-V4-Flash with output budgets of 4,000 tokens for local summaries (reasoning off), 12,000 for aggregation, and 32,000 for attribution. One length-triggered retry doubles the budget, subject to a 65,536-token cap. Attribution requests temperature 0; aggregation and attribution use reasoning at high effort. All component conditions share these settings. Full-benchmark evaluation uses output budgets of 4,000, 6,000, and 8,000 tokens for local summarization, aggregation, and attribution, respectively.

Component-control implementation. The A3 control omits explicit backward-reasoning instructions while keeping the original candidate and handoff texts and step IDs. It removes linkage and role hints, replaces annotations for added handoff excerpts with neutral context labels, and omits the instruction-evidence check. The A4 control uses the initial attribution output of RCTA (all components) after any length-triggered retry and omits validation-driven retry and output sanitization. Its output is checked only to measure validity.

Ablation statistical protocol. Component comparisons use source-macro aggregation over the five sources and four pre-specified exact McNemar tests on Step Acc against RCTA (all components), with Holm correction (family α=.05). Source-stratified paired bootstrap CIs (10,000 resamples; seed 20260820) are pointwise and quantify trajectory resampling conditional on the observed outputs. At the observed discordance rates ($\approx$0.15), a post-hoc sensitivity calculation estimates 80% power for effects of roughly 9–10 percentage points in the four-test design. Accuracy uses a fixed denominator of 200; abstained, missing, or unparseable predictions count as errors.

Inference-time repeatability. Across three pipeline realizations on a 30-trajectory panel, RCTA (all components) obtains 26.7/30.0/43.3% Step Acc, compared with 26.7/16.7/30.0% without A1 and 23.3/20.0/33.3% without A2. The corresponding differences, computed as RCTA (all components) minus each ablation, are 0.0/13.3/13.3 and 3.3/10.0/10.0 percentage points. Pairwise agreement between root-step predictions from RCTA (all components) is 50.0–56.7%.

Evidence retrieval and output citation. The base-candidate set and the handoff-augmented excerpt set are evaluated separately. Relative to RCTA (all components), the equal-size uniform control changes base-candidate recall by -31.5 pp (pointwise 95% CI $[-39, -24]$), all-excerpt recall by -25.5 pp ($[-33, -18]$), root citation rate by -13.0 pp ($[-19, -7]$), and Step Acc by -6.0 pp ($[-11.0, -1.0]$). The three evidence contrasts are exploratory and are not members of the four-comparison Step Acc test family.

Handoff-root diagnostic. On the 41 handoff-root cases within the 80-trajectory cohort (Appendix L.3), removing A2 or A3 lowers Step Acc by 26.8 or 19.5 pp, respectively; the unadjusted

exact McNemar p-values are .0034 and .077. On the 39 non-handoff-root cases, both Step Acc differences are zero. Among non-abstaining, parseable in-range handoff-root predictions, late-prediction rates are 40.0%/60.0%/51.2% and mean signed errors (prediction minus reference) are +24.0/+32.4/+32.3 steps for RCTA (all components), w/o A2, and w/o A3, respectively. These subgroup diagnostics are exploratory.

I RCTA INFERENCE PSEUDOCODE

The pseudocode exposes the correspondence between the operational inference sequence and A1–A4 while retaining the implementation fallbacks described in Section H. Indices c and h denote steps s_c (a base candidate) and s_h (its retrieved handoff instruction). Handoff retrieval (A2) only augments the evidence package. The instruction–execution comparison and independent role/root prediction occur in COMPAREANDATTRIBUTE (A3).

Algorithm 1 RCTA inference for a failed trajectory.

Require: Failed trajectory $\tau = (s_0, \ldots, s_{T-1})$ with a normalized, step-indexed history.
Ensure: Responsible role $\hat{\rho}$, root step $\hat{r}$, evidence E, or abstention.

Stage I: Failure-relevant candidate recall (A1)
$\mathcal{S} \leftarrow \text{ENRICHANDSEGMENT}(\tau)$
$L \leftarrow [\text{LOCALSUMMARY}(S_j) \mid S_j \in \mathcal{S}]$
$P \leftarrow \text{ORGANIZEPHASES}(L)$
if P is not parseable **then**
  $P \leftarrow \text{SINGLETONPHASES}(L)$
end if
$B \leftarrow \text{RETRIEVESOURCESTEPS}(L, P)$
if $\text{PLANNEREXECUTORSIGNATURE}(\tau)$ matches **then**
  $B \leftarrow B \cup \text{RULEBASEDCANDIDATES}(\tau)$
end if
$B \leftarrow \text{TOPPRIORITIZED}(B, 80)$
$C \leftarrow \text{CANDIDATEPACKAGE}(B, L, P, \tau)$
Stage II: Upstream instruction comparison (A2–A3)
for all $c \in B$ **do**
  if s_c is emitted by an executor or verifier **then**
    $h \leftarrow \text{ROLEMATCHEDHANDOFF}(c, \tau)$
  else
    $h \leftarrow \text{NEARESTEARLIERHANDOFF}(c, \tau)$
  end if
  $C \leftarrow \text{ATTACHIFPRESENT}(C, h)$
end for
$(\hat{\rho}, \hat{r}, E) \leftarrow \text{COMPAREANDATTRIBUTE}(P, C)$
Output and evidence validation (A4)
$v \leftarrow \text{VALIDATEFIELDS}(\hat{\rho}, \hat{r}, E, \tau)$
if v reports an invalid field **then**
  $(\hat{\rho}, \hat{r}, E) \leftarrow \text{RETRYWITHFEEDBACK}(P, C, v)$
  $v \leftarrow \text{VALIDATEFIELDS}(\hat{\rho}, \hat{r}, E, \tau)$
end if
return $\text{SANITIZEORABSTAIN}(\hat{\rho}, \hat{r}, E, v)$

J EXTENDED RELATED WORK AND COMPARISON NOTES

Broader failure-characterization resources describe errors at levels that complement agent and step attribution. MAST organizes recurring failure modes, whereas TRAIL attaches error types and impact to execution spans (Cemri et al., 2025; Deshpande et al., 2025). MP-Bench accommodates multiple plausible failure-inducing steps (In et al., 2026). Among the resources compared in Section 5, Who&When Pro scales data collection through controlled error injection, while TraceElephant emphasizes complete execution observability (Liu et al., 2026; Chen et al., 2026).

RAFFLES uses iterative judge–evaluator feedback, whereas AgentRx grounds diagnosis in checked constraint violations (Zhu et al., 2026a; Barke et al., 2026). CDC combines Shapley-based agent attribution with temporal causal discovery for critical-step prediction (Ma et al., 2025). Diagnostic support can also extend beyond the given trajectory. FAMAS re-executes failed tasks and ranks actions using spectrum statistics across runs, while CORRECT retrieves error schemata distilled from previously annotated failures (Ge et al., 2025; Yu et al., 2025).

In the resource comparison in Section 5, *Error Source* distinguishes failures arising during ordinary task execution (Natural) from failures constructed through deliberate error injection (Synthetic). Controlled task conditions, such as HORIZON's horizon settings, are distinct from deliberate error injection. *Diagnostic Targets* identifies the annotated object: Agent denotes the responsible agent or workflow role, Component an execution component, and Type a failure-mode or error-cause category. Step and Steps retain the localization targets summarized in the table's superscript notes; their annotation rules remain benchmark-specific.

Total (Reviewed) reports the number of released or analyzed trajectories, followed by the human-reviewed count in parentheses. Equal counts indicate review of all counted trajectories, but do not imply identical review or annotation protocols across resources. *Avg. length (steps)* reports each resource's overall mean using its own step-counting convention; – denotes an unreported mean. For HORIZON and Failure as a Process, trajectory counts refer to failed runs; the mean length reported for Failure as a Process also refers to failed runs.

K Benchmark Construction and Supplementary Details

K.1 Human Annotation and Quality Control

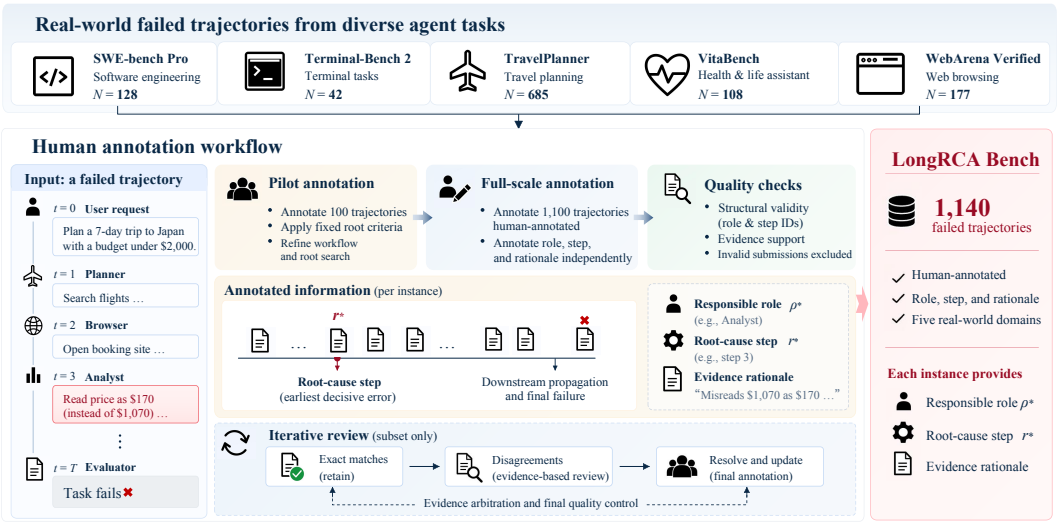


Figure 11: Construction and annotation workflow for LongRCA Bench. Calibration retains 96 of 100 trajectories; together with 1,044 formal-phase trajectories, the release contains 1,140 unique failed executions.

The same 22 master's and doctoral students in computer science participated in calibration and formal annotation. Calibration covered 100 trajectories; four were excluded because they potentially contained independent root causes across subtasks. After annotation, quality control, and deduplication, the release combines 96 calibration trajectories with 1,044 formal-phase trajectories, yielding 1,140 unique failed executions.

Calibration refined how they navigated long histories, traced candidate errors, and documented evidence; the root-cause definition and decision rules remained unchanged. In both stages, annotators independently reviewed task instructions, complete step-indexed histories, and source evaluator outcomes, then submitted a responsible role, root step, and evidence-grounded rationale.

DeepSeek V4 Flash Preview generated factual activity summaries to support navigation. The prompt requested descriptions of recorded activities without root-cause judgments. Summaries remained separate from annotation fields; reference labels and rationales required support from task requirements and original trajectory records.

Pre-adjudication agreement is assessed on 294 multiply annotated formal-phase trajectories, yielding 310 unordered pairs of annotations from different annotators. These counts describe the agreement-analysis subset, not the size of the released formal-phase dataset. Pairwise agreement is computed before label resolution, with role names normalized as in Appendix L.1. Exact role, step, and joint matches number 205, 125, and 121 (66.1%, 40.3%, and 39.0%). Calibration is reported separately in Table 22; the two sets are not pooled. Typical annotation time was 30–40 minutes.

Table 22: Pre-resolution disagreement among completed, structurally valid annotations. Each unordered pair of different annotators has equal weight. Role agreement uses the evaluation normalization. Step-gap percentages sum to 100 within each row; quartiles use linear interpolation. Calibration and full-scale annotations are summarized separately.

Role relation	Pairs	Role	Step ± 5	Median [Q1, Q3]	0	1–5	6–20	> 20
Formal-phase agreement subset: 294 trajectories								
All pairs	310	66.1	50.3	5 [0, 46]	40.3	10.0	11.6	38.1
Same role	205	100.0	67.3	0 [0, 15]	59.0	8.3	8.3	24.4
Different role	105	0.0	17.1	42 [13, 78]	3.8	13.3	18.1	64.8
Calibration: 100 trajectories								
All pairs	296	53.7	37.8	17.5 [1, 54.25]	24.7	13.2	15.5	46.6
Same role	159	100.0	59.7	1 [0, 29]	45.9	13.8	11.9	28.3
Different role	137	0.0	12.4	37 [15, 82]	0.0	12.4	19.7	67.9

The formal-phase agreement subset has a median step gap of 5 and an interquartile range of 0–46. Only 31 of the 185 non-exact pairs (16.8%) differ by 1–5 steps; nearby localization therefore explains a minority of exact-step disagreements. The median gap is 0 for same-role pairs and 42 for different-role pairs. These descriptive results concern the existing multiply annotated samples.

A valid submission must select an integer step within the trajectory range and a role present in the recorded role set. Its rationale must identify trajectory evidence supporting the proposed causal interpretation. Submissions with invalid references, evident annotation defects, or insufficient evidence are excluded; selected steps are not automatically changed to repair submissions.

For trajectories with multiple valid annotations, labels are compared field by field, and exact matches are retained as consensus references. Two experts resolve disagreements using the original task, complete history, source evaluator outcome, and written rationales under the attribution criteria. Final quality control re-examines every retained trajectory, including calibration cases, and excludes unresolved attribution ambiguity, invalid labels, or inadequate evidential support. Each released reference retains separate role and root-step fields with a supporting rationale.

Sensitivity to observed root-step labels. We assess root-step sensitivity on the 294 multiply annotated formal-phase trajectories with predictions from all six methods. We first score against the final reference roots, then repeat scoring over 10,000 draws (seed 20260918). Each draw uniformly selects one complete observed human annotation per trajectory and uses its root step for all six methods. Trajectories and predictions remain fixed, and scores weight trajectories equally.

On the 294-trajectory subset, RCTA strictly leads all baselines in 100.00% of draws for Step Acc and 99.68% for Step ± 5; its gap to the strongest baseline has a central 95% range of [3.1, 8.8] and [1.4, 8.5] pp, respectively. As shown in Table 23, full method rankings vary more than the leading step method. These results describe root-step sensitivity within the available annotation overlap.

Automatic release checks. All 1,140 released records have an existing integer root index, a role in the recorded role set, a nonempty rationale, unique contiguous step indices, matching individual and merged records, and a verified mapping to source trajectory content. These checks establish structural integrity and traceability, not causal correctness; neither agreement nor label sensitivity estimates final-label accuracy or independent expert reproducibility (Klie et al., 2024).

Table 23: Root-step sensitivity to observed human labels on 294 formal-phase trajectories with DS-V4-Flash. Accuracy cells show final-label score → resampled mean [2.5th, 97.5th percentiles] (%). Gap rows use RCTA minus the strongest baseline in each draw (pp). Rank-1 rates require a strict lead; ties do not count as wins. Intervals describe label sensitivity on fixed samples, not population confidence intervals.

Method / comparison	Step Acc	Step ± 5
All-at-once	7.8 → 8.3 [7.1, 9.5]	21.1 → 22.1 [19.7, 24.5]
Step-by-step	6.1 → 6.5 [5.4, 7.5]	15.6 → 17.3 [15.3, 19.4]
Binary search	3.7 → 3.9 [3.1, 4.8]	15.0 → 15.9 [13.9, 18.0]
ECHO	17.0 → 17.6 [15.6, 19.4]	27.9 → 31.1 [28.6, 33.7]
FALAT	1.0 → 2.2 [1.4, 3.1]	13.6 → 14.4 [12.2, 16.7]
RCTA	21.8 → 23.5 [21.1, 25.9]	35.4 → 36.2 [33.7, 38.8]
RCTA gap (pp)	4.8 → 5.9 [3.1, 8.8]	7.5 → 5.1 [1.4, 8.5]
RCTA strict rank 1	100.00%	99.68%
Full ordering retained	98.39%	72.15%

Responsible role and the recorded event. The role and root step are recorded separately, with a rationale grounded in the task and trajectory. A log event can relay another agent's action or completion report. For example, in `swe_bench_pro__047`, step 133 is logged under `Computer_terminal` but relays `ActionAgent`'s delegated completion report. The retained human reference assigns responsibility to `ActionAgent`, with step 133 as the root. Its rationale identifies the missing required configuration file despite the success report, and distinguishes later missed checks from that failure. This example shows why the event's outer sender does not mechanically determine the reference role.

K.2 Complete illustrative case

The illustrative case in Section 2.1 shows a pricing error in a 223-step TravelPlanner trajectory.[1] The user requests a three-day trip for one traveler from Plattsburgh to Fort Lauderdale within a $1,200 budget. The human reference assigns responsibility to `stay_agent` and identifies step 73 as the root-cause step.

Steps 78 and 79 confirm that candidate notes were stored on the blackboard. At step 83, the agent compares the candidates and recommends Lefferts, calculating a $172 remainder using its $317 lodging total. The planner allocates this remainder to food and activities at step 86. The workflow verifier approves the budget and later rechecks the accommodation after transport changes, retaining the same two-night interpretation. During finalization, the manager corrects a separate arithmetic error in the food/activity allocation from $357 to $257, but leaves the lodging total at $317. The correction therefore repairs the allocation arithmetic while preserving the erroneous price interpretation.

The final itinerary at step 220 lists Lefferts for both nights without stating a lodging price. TravelPlanner defines `price` as a nightly rate, but the original accommodation-search response does not specify the unit. Applying this pricing rule to the recorded itinerary gives flight costs of $\$341 + \$255 = \$596$ and lodging costs of $\$317 \times 2 = \$634$. These items alone total $1,230, exceeding the budget before food or local transport. This budget check is part of retrospective diagnosis; the recorded source evaluator reports commonsense failures without executing the budget test. The case shows how an early misinterpretation can persist through verification and local correction, requiring the final plan to be read together with intermediate records and the applicable pricing rule.

K.3 Additional benchmark statistics

Median trajectory length ranges from 51 recorded steps for VitaBench to 257.5 for SWE-bench Pro; source medians for root-to-end distance range from 24 to 186 steps. Section 2.4 reports the overall

[1] Released as instance `travelplanner__021` in LongRCA Bench, corresponding to TravelPlanner test query 104, generated by Kimi-K2.5 with Magentic-One group-chat coordination (run 0001).

distribution summaries and threshold counts. Appendix A reports the total recorded step count and source-level means alongside the Mini comparison.

L SUPPLEMENTARY EVALUATION PROTOCOLS AND DIAGNOSTICS

L.1 SCORING AND EXPERIMENT PROTOCOLS

For the benchmark comparisons, role names are normalized for case, whitespace, and explicit handoff suffixes. Agent and step accuracies are computed from their respective output fields. Every expected trajectory remains in the denominators; invalid fields count as errors for their respective targets.

All component conditions use the same reference labels and scoring rules. Role matching normalizes case and whitespace while retaining handoff suffixes; abstentions count as errors.

Component coverage is the percentage of trajectories with non-abstaining outputs that pass the validation checks for their condition. These checks cover recorded role names, root-step validity, explanatory-chain structure, and cited handoff evidence. Appendix H specifies the A3 control's changes to these checks and the A4 control's use of validity checks only for scoring.

For the full-benchmark comparison, we also report **source-weighted valid-output root MAE**. Let $\mathcal{V}_b$ be the trajectories in source b with a valid, in-range numeric root-step prediction, N_b the total number of trajectories from that source, and $N = \sum_b N_b$. The metric is

$$\frac{1}{N} \sum_b N_b \left(\frac{1}{|\mathcal{V}_b|} \sum_{i \in \mathcal{V}_b} |\hat{r}_i - r_i^*| \right).$$

We average absolute root-step errors over valid predictions within each source, then weight these source means by the original source proportions. Missing or invalid step outputs count as errors for both root-step accuracy metrics but are excluded from the source means used for MAE. Exact matching remains the primary localization metric. A fixed step tolerance represents different fractions of trajectories whose lengths differ by an order of magnitude.

Fidelity evaluates the fixed Mini QIDs using the same DS-V4-Flash predictions as the full benchmark. Appendix H records prompt requirements and validation rules; Algorithm 1 gives the field-preserving fallback sequence.

L.2 FULL-BENCHMARK SUPPLEMENTARY METRICS

Table 24: Full-benchmark results on all 1,140 trajectories. All methods use DS-V4-Flash. Agent Acc measures agreement with the reference responsible role. Root MAE is the source-weighted valid-output mean absolute step error. Higher values are better for the three accuracy metrics, whereas lower values are better for MAE.

Method	**Agent Acc↑**	**Step Acc↑**	**Step ±5↑**	**Root MAE↓**
All-at-once	26.2%	7.6%	19.9%	55.9
Step-by-step	22.2%	5.3%	16.9%	52.3
Binary search	23.0%	3.4%	13.3%	61.7
ECHO	27.5%	13.2%	24.7%	50.4
FALAT	19.0%	2.8%	12.5%	66.6
RCTA (ours)	**51.1%**	**24.1%**	**37.4%**	**38.6**

RCTA, as shown in Table 24, achieves 51.1% Agent Acc, 24.1% Step Acc, and 37.4% Step ±5, with a source-weighted valid-output root MAE of 38.6 steps. Relative to ECHO, the strongest evaluated baseline on all four metrics, its accuracy gains computed from unrounded values are 23.6, 11.0, and 12.6 percentage points, respectively, and its MAE decreases by 11.8 steps. Despite these gains, the exact reference root is recovered in fewer than one quarter of the trajectories.

Role priors. The majority-role baseline always predicts the most frequent reference role within each source, using the evaluated-set frequencies reported in Table 25. RCTA matches this baseline on VitaBench and exceeds it on the other four sources.

Table 25: Reference-role distributions and majority-role baselines on Full ($N = 1{,}140$). Role counts are listed in descending order. K is the number of distinct normalized roles in each history, reported as median [range] when variable. Majority and RCTA columns report Agent Acc (%); Overall weights trajectories equally. Source abbreviations follow Section 4.2.

Source	K	Reference roles (counts)	Majority	RCTA
SWE	5	`DiagnostAgent` (79), `ActionAgent` (41), `JudgeAgent` (4), `human` (3), `Computer_terminal` (1)	61.7	63.3
TB2	5	`DiagnostAgent` (20), `ActionAgent` (18), `JudgeAgent` (2), `Computer_terminal` (1), `human` (1)	47.6	57.1
TP	6 [3–7]	`stay_agent` (174), `manager_agent` (162), `Task_Planner` (103), `transport_agent` (84), `verifier_agent` (67), `food_attr_agent` (56), `Computer_terminal` (39)	25.4	41.2
Vita	2	`Action_Expert` (89), `Computer_terminal` (19)	82.4	82.4
WA	4	`Action_Expert` (97), `Task_Planner` (43), `Verification_Expert` (29), `Computer_terminal` (8)	54.8	60.5
Overall	–	–	40.3	51.1

L.3 Component analysis and diagnostics

Table 26: Component ablations on the fixed 200-trajectory sample. Rates are source-macro percentages. MAE uses non-abstaining outputs with parseable in-range steps (n=196/188/196/197/192); Cov. uses all trajectories as its denominator and requires non-abstaining outputs that pass the condition's validation checks (Appendix L.1). Δ is ablation Step Acc minus that of RCTA (all components). Intervals are pointwise; p_{Holm} controls the four-test Step Acc family.

Setting	Agent Acc↑	Step Acc↑	Step ±5 ↑	MAE↓	Cov.↑	Δ [95% CI]	p_{Holm}
RCTA (all components)	56.0	28.0	41.0	28.3	97.0	–	–
w/o candidate-information package (A1)	53.5	22.0	39.0	30.5	91.5	−6.0 [−11.0, −1.0]	.17
w/o handoff evidence (A2)	53.5	24.5	38.0	29.7	97.0	−3.5 [−9.0, +2.0]	.75
w/o backward attribution (A3)	51.0	25.0	41.0	31.4	97.5	−3.0 [−8.5, +2.5]	.75
w/o validation (A4)	54.0	26.5	39.5	29.2	89.0	−1.5 [−3.5, 0.0]	.75

Table 26 retains the complete metrics and pointwise intervals, which capture trajectory resampling and exclude inference-time variability. RCTA (all components) denotes the complete component configuration on the fixed sample (40 trajectories per source; sampling seed 20260821), evaluated under the ablation inference settings in Appendix H.

A1: Evidence retrieval and root citation. Base-candidate recall is 19.5% for the equal-size uniform control and 51.0% for RCTA (all components); recall in the handoff-augmented input excerpts is 37.5% and 63.0%, respectively. Root citation rates are 25.5% and 38.5%, compared with Step Acc of 22.0% and 28.0%. Root citation rate is the fraction of all 200 trajectories whose non-abstaining final output includes the annotated root step in a parseable evidence-ID list.

For RCTA (all components), the input excerpts contain the annotated root record in 126 trajectories. Of these, 52 receive correct root-step predictions, 71 receive valid predictions of a different step, and three have invalid or abstaining step outputs. The other four correct root-step predictions concern steps outside the input excerpt set. The final output cites the annotated root step in 77 trajectories: 56 select it as the root, while 21 select a different step. These joint counts distinguish input record coverage, output citation, and final localization.

The A1 comparison evaluates excerpt selection together with the accompanying location and provenance cues.

A2–A3: Handoff evidence and backward attribution. The diagnostic includes all 80 trajectories in the component sample with an explicit logged handoff: 40 from SWE-bench Pro and 40 from Terminal-Bench 2. The reference root, independently of model predictions, defines 41 handoff-root cases (25 SWE-bench Pro, 16 Terminal-Bench 2) and 39 non-handoff-root cases (15 and 24, respectively). Both groups contain handoffs; the other 120 trajectories are outside this diagnostic.

RCTA (all components) and the A2 and A3 ablations obtain 51.2%, 24.4%, and 31.7% Step Acc on handoff roots, respectively, and 28.2% each on non-handoff roots, as discussed in Section 4.3. Removing A2 omits the added handoff evidence; removing A3 retains the original excerpts while removing the backward-attribution package and yields more late predictions on handoff roots. The observed accuracy differences are concentrated in the handoff-root group within the shared diagnosis–execution–verification workflow. Section 4.3 summarizes these exploratory contrasts; Appendix L.1 specifies the scoring rules.

A4: Output validity. Validation changes the final prediction object on 18 trajectories and the parsed root on 9; exact-root correctness changes on 3 of 200, all favoring RCTA (all components). Valid-output coverage increases from 89% to 97%.

Appendix H reports the statistical protocol and the three-realization repeatability panel.