PANINI-DR: CONTROLLED DEEP RESEARCH THROUGH GENERATIVE SEMANTIC WORKSPACES

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep research requires connecting multiple steps of search, reasoning, and computation, with intermediate findings guiding what to investigate next. As investigations grow, agents must maintain research state across model calls and check findings before errors propagate into subsequent work. We introduce PANINI-DR, a multi-agent harness that separates coordination of the overall investigation from the research and verification of individual subquestions. A shared workspace, built on the Generative Semantic Workspace representation, records subquestions, their dependencies, findings, and supporting evidence. Researchers investigate individual subquestions, while a separate verification agent checks proposed answers against those subquestions and their evidence. Verification feedback guides further investigation, and recorded findings inform subsequent assignments. We evaluate PANINI-DR on BrowseComp-Plus, FRAMES, and MoNaCo against seven baseline implementations using a shared language-model backbone and frozen corpora. PANINI-DR achieves the strongest average performance across the three benchmarks, with strong gains on harder questions. On BrowseComp-Plus, PANINI-DR achieves **73.3%** accuracy, outperforming the next-best baseline by **16.0** percentage points, at higher inference cost. Its advantage is strongest on harder investigations, correctly answering **26 of 50** hard questions— nearly twice as many as the next-best baseline. Our ablation studies further show that structured planning and persistent state improve performance, as does verification combined with evidence checks. To assess whether correct answers are supported by the research performed, we also analyze the evidence retained by each system. PANINI-DR achieves the highest answer recoverability from saved evidence on all three benchmarks, while complementary claim-support audits show stronger evidence-backed answers. Accounting for this evidence changes how systems compare and further strengthens PANINI-DR's advantage over existing approaches.

1 INTRODUCTION

Answering questions over collections of documents is one of the most useful applications of language models. A common early approach was a simple retrieve-then-answer pipeline: a retriever selected passages relevant to a question, and a language model used those passages to generate an answer (Lewis et al., 2020; Izacard & Grave, 2021). As language models improved at reasoning and using tools, they became capable of answering increasingly complex questions (Yao et al., 2023b; Trivedi et al., 2023). One such class is deep research questions, which require gathering and connecting evidence across multiple sources (Zheng et al., 2025; Li et al., 2025). For these questions, the relevant documents may not be apparent at the outset, making them difficult to resolve with a single retrieve-then-answer step. One must first break down the problem into appropriate subquestions, search for evidence to answer them, and combine the findings into a final answer. This process can be iterative: answering one subquestion may reveal what to investigate next, requiring successive rounds of search and reasoning (Press et al., 2023; Trivedi et al., 2023).

Deep research agents carry out this process through repeated search, reading, and reasoning over source material (Zheng et al., 2025; Li et al., 2025). Consider the following question, taken from the FRAMES dataset (Krishna et al., 2024): "How old was the U.S. President's wife when the Camp David Accords were signed?" An agent can begin by establishing when the accords were signed and

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.

who was president at the time. Identifying *Jimmy Carter* then provides a starting point for finding his wife, *Rosalynn Carter*, and her birth date. Answering the question then requires reasoning over the retrieved facts: the agent must combine her birth date with the signing date to compute her age, accounting for whether her birthday had occurred that year.

Answering these questions thus requires connecting multiple steps of search, reasoning, and computation, with intermediate findings guiding what to investigate or calculate next. Over a longer investigation, these findings accumulate across multiple searches, model calls, and reasoning steps. Early approaches to agentic question answering, such as ReAct, used a single agent to decide what to investigate, invoke tools, and reason over the results, with progress retained in a growing interaction history (Yao et al., 2023b). When the state of the investigation is left implicit in this history, the model must recover earlier findings while carrying out further search, reasoning, and computation. As that history grows, relevant findings can become harder to use: long-context studies report difficulties using information in the middle of an input (Liu et al., 2024) and declining performance as input length increases (Hong et al., 2025).

These difficulties motivate a separation of concerns between **maintaining research state** and performing the search, reasoning, and computation needed to answer the question. The research state records the subquestions needed to answer the original query, their dependencies, the findings and supporting evidence obtained so far, and what remains unresolved. For example, identifying *Jimmy Carter* as the president resolves one subquestion and provides the context for the next (identifying his wife at the time). The need to track progress and dependencies also arises as LLM agents take on increasingly ambitious tasks in other domains, such as mathematical formalization. For example, Anthropic reports that early attempts to formalize Fermat's Last Theorem lost track of project state and struggled to coordinate. The successful effort followed a switch to Prove2Me, whose shared graph of theorem statements and dependencies helped agents maintain progress and coordinate parallel work (Anthropic, 2026; Chen et al., 2026). Within deep research, several methods similarly maintain explicit records of research progress. IterResearch reconstructs the context at each step using an evolving research report (Chen et al., 2025a), while SLEUTH organizes research progress into facts, hypotheses, and open questions (Liu, 2026). Dossier maintains a persistent research ledger of claims, contradictions, and information gaps, using it to guide parallel search branches and reconcile their findings (Chabra et al., 2026).

The dependencies between subquestions also allow errors to propagate. In the Camp David example, suppose the agent incorrectly identifies *Ronald Reagan* rather than *Jimmy Carter* as the president when the accords were signed. It could then retrieve accurate information about the wrong person's wife and correctly calculate her age, yet still produce an incorrect answer. **Verifying intermediate findings** against their supporting evidence and the questions they are intended to resolve provides an opportunity to identify and correct such errors before they guide subsequent work. Recent work reports substantial improvements in research-agent performance from verification feedback. DeepVerifier uses rubric-guided verification to provide feedback for further attempts (Wan et al., 2026), while Marco DeepResearch verifies candidate answers to guide continued research (Zhu et al., 2026a). Other methods integrate verification directly into the research state: Omni-Decision records evidence needs, dependencies, and critic judgments, and uses them to guide subsequent actions (Ma et al., 2026).

Building on these two guiding ideas and recent work, we introduce **PANINI-DR**, a multi-agent harness for deep research. PANINI-DR maintains a shared research workspace and coordinates separate agents to investigate subquestions and verify their proposed answers. The workspace uses the Generative Semantic Workspace (GSW) representation to express subquestions and their relationships (Rajesh et al., 2026a;b). A central agent uses the workspace to track overall progress and launch research assignments. Within each assignment, verification feedback guides further investigation. The workspace records the resulting findings, supporting evidence, and review outcomes, making them available to subsequent assignments. This enables us to allocate research and verification effort to individual subquestions while preserving their dependencies within the overall investigation.

We evaluate PANINI-DR on BrowseComp-Plus, FRAMES, and MoNaCo, comparing it with existing state-of-the-art deep research approaches. Among the evaluated systems, it achieves the strongest average performance across benchmarks, with strong gains on the hardest questions. This pattern is consistent with our motivation for maintaining research state across extended investi-

gations. Our ablation studies further show that structured planning and persistent state improve performance, as does verification combined with evidence checks.

To further assess whether correct answers are supported by the research performed, we analyze the evidence retained by each system. A model can answer correctly from its parametric knowledge even when its retrieved evidence is insufficient. Accounting for evidence support further widens PANINI-DR's advantage over existing approaches.

We summarize our contributions as follows: (1) **A shared workspace for research and verification.** We introduce PANINI-DR, a multi-agent harness that organizes subquestions, their dependencies, and supporting evidence in a shared workspace. This separates the coordination of the overall investigation from the research and verification of individual subquestions. (2) **Strong performance supported by ablation studies.** We show that PANINI-DR achieves the strongest average performance across the evaluated benchmarks, with particularly strong gains on harder questions. Through ablation studies, we examine how structured planning, persistent state, and verification with evidence checks contribute to this performance. (3) **Evaluation beyond answer correctness.** We further develop an evaluation of the evidence supporting each system's answers. We show that accounting for this evidence changes how systems compare and further strengthens PANINI-DR's advantage over existing approaches. We believe these contributions offer a path toward more reliable deep research systems.

2 THE PANINI-DR HARNESS

The PANINI-DR harness coordinates research and verification through a shared workspace that records the subquestions to resolve, their dependencies, and the findings obtained so far. We use the Camp David example in Figure 1 to illustrate how this workspace evolves during an investigation. The remainder of this section describes how the workspace is constructed and used to guide research, verification, and answer generation.

2.1 CONSTRUCTING THE RESEARCH WORKSPACE

Given a query q, the harness invokes an *Orchestrator* as its main agent. The Orchestrator oversees the creation and maintenance of research state, launches sub-agents to investigate subquestions, and submits the final answer. To initialize the workspace, it first calls a *Planner* agent to produce a structured research plan P_0 using the GSW representation introduced by Rajesh et al. (2026a;b). The plan records information supplied by the query, the subquestions to resolve, and the relationships and computations that establish dependencies between their answers. Our choice of GSW is motivated by its situational representation of entities, their roles and states, and their relationships. This structure provides explicit context for verifying whether a proposed answer satisfies the conditions of the subquestion it is intended to resolve.

The dependency structure encoded in P_0 forms a directed acyclic graph, which the Orchestrator uses to guide research. Its nodes correspond to the subquestions and a final answer target $v_\star$. An edge $u \to v$ indicates that the finding at u supplies information used to resolve v. For example, identifying the president provides the context for investigating his wife, whose identity then guides the search for her birth date. Nodes can be resolved through research or computation: in Figure 1, the signing date and birth date supply the inputs for computing the final answer, the wife's age.

The unknown answer at each node is represented as a *blank* in the GSW plan. The Planner expresses the corresponding subquestion in natural language as a *retrieval hint*, incorporating the blank's relations and relevant query context. For example, the relation "was born on" between the wife node u and birth-date blank v gives the dependency $u \to v$. The retrieval hint "When was [wife] born?" is attached to v, and the placeholder is replaced with her identity once established. Each blank also carries metadata to support verification, including an expected answer type, such as "date," and a verification claim describing what a proposed answer must satisfy, such as "the proposed date is the birth date of [wife]."

Before launching research, the harness checks the plan for structural errors, including invalid references, missing or multiple answer targets, and circular dependencies. It also checks whether retrieval hints are consistent with the plan's dependencies. For example, the birth-date hint may require the

wife's identity without the plan recording that dependency. A separate LLM call proposes corrections within a bounded repair procedure, and the harness checks these corrections before applying them.

After checking the plan, the harness initializes the shared research workspace by pairing the plan with per-node research records. These records store findings, supporting evidence, and review status. We denote the workspace at step t by $W_t = (P_t, S_t)$, where P_t is the current plan and S_t is the collection of research records linked to its nodes. The initial workspace is $W_0 = (P_0, S_0)$, with every blank marked as unresolved in S_0; information supplied by the query is already represented in P_0. The records are updated as research progresses. For example, a record for the wife subquestion could contain the answer *Rosalynn Carter*, references to source passages establishing that she was *Jimmy Carter*'s wife, and a verified review status. The Orchestrator uses these recorded findings to guide subsequent work. Appendix B.1 gives the schema and scheduling rules.

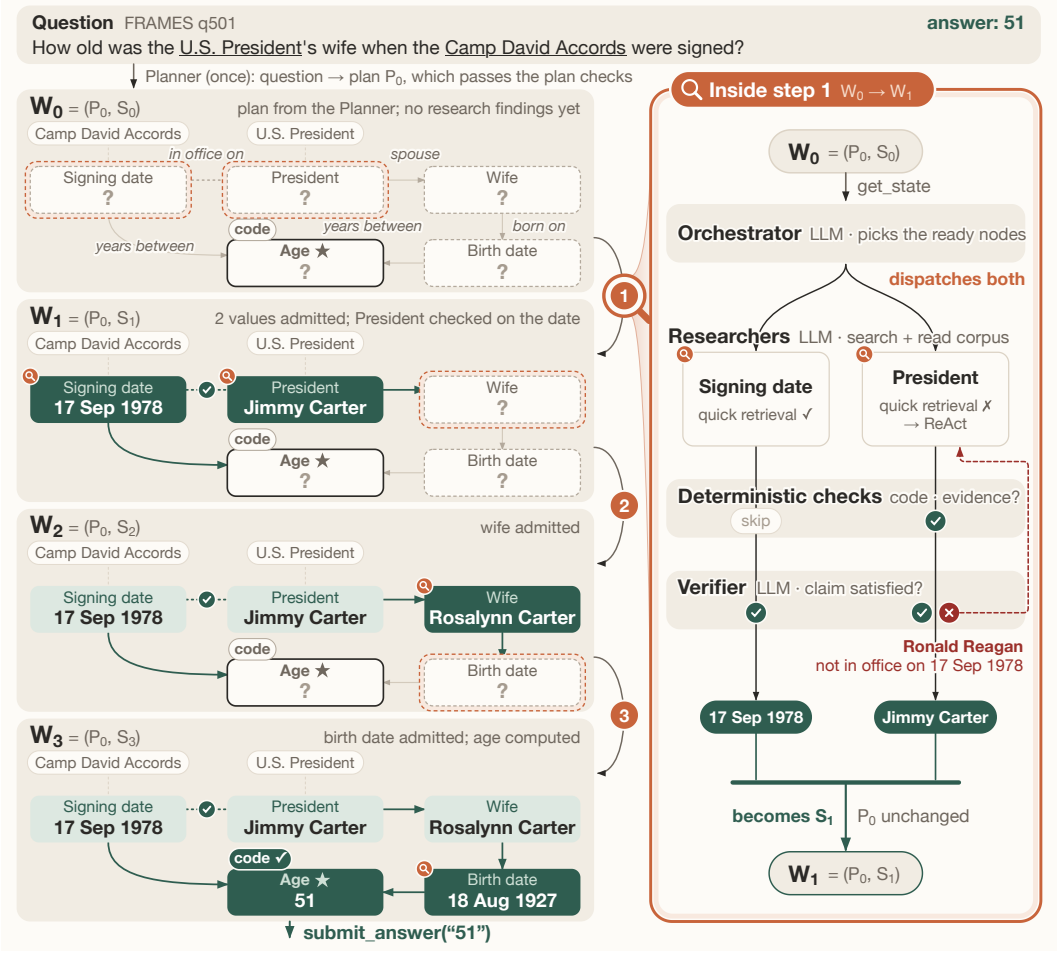

Figure 1: PANINI-DR illustrated on a sample query. The schematic stages separate the subquestions for explanation, rather than reproduce the recorded dispatch timing. Each $W_k = (P, S_k)$ pairs the plan with per-node research records; $\varnothing$ denotes that no findings have yet been recorded. Dark green marks additions, dashed boxes remain unresolved, and orange outlines indicate readiness. The expanded stage shows each Researcher's initial retrieval step (labelled "typed resolver") and, when needed, ReAct agent, followed by verification and admission; the Reagan rejection is hypothetical. Findings supply subsequent subquestions and the date calculation that produces 51. The recorded execution investigated all four factual subquestions in one dispatch.

2.2 Researching a Subquestion

The Orchestrator uses the current workspace W_t and the plan's dependencies to select subquestions to investigate, launching a *Researcher* agent for each assignment. Each Researcher receives the assigned subquestion, relevant plan context, and available findings. In Figure 1, the signing-date and president subquestions can be investigated in parallel.

Although the overall query may require multiple searches and reasoning steps, an individual subquestion may be answerable from evidence obtained in a single round of retrieval. The Researcher therefore begins with an initial retrieval step, substituting available findings into the retrieval hint and performing one search. For example, "When was [wife] born?" becomes "When was Rosalynn Carter born?" once her identity is established. The Researcher uses an LLM to propose a candidate answer with supporting citations from the retrieved passages. If the candidate passes type checking and verification (Section 2.3), the finding is recorded in the workspace. In Figure 1, this initial step establishes the signing date.

If the context required by the hint is unavailable, no candidate is found, or the candidate fails these checks, the Researcher continues with iterative search and reasoning using ReAct (Yao et al., 2023b), retaining any candidate, sources, and feedback already obtained. It chooses searches, reads sources, and submits proposals within its turn allowance. To limit source text accumulating in the Researcher's context, its reading tool uses a separate LLM agent to extract relevant information and supporting quotations. The president subquestion in Figure 1 illustrates this extended research path.

An assignment may end before its subquestion is resolved. To support later attempts, the harness retains candidates, retrieved sources, previous searches, and verification feedback for that subquestion. Relevant history is provided when the subquestion is assigned again, allowing the Researcher to build on earlier work. The Researcher submits candidate answers with supporting evidence for verification. The resulting feedback can prompt further research before a finding is recorded in the shared workspace.

2.3 Verifying Findings and Updating the Workspace

To verify a proposed answer, the harness combines deterministic checks with LLM-based assessment. It first checks the expected answer type and required evidence attachments. Establishing that the cited passages support the answer requires further interpretation—for example, whether a date refers to the birth of the intended person. A *Verifier* agent therefore evaluates the candidate against the subquestion's verification claim, using the cited evidence and available findings.

The Verifier checks whether the cited evidence supports the candidate as an answer to the subquestion and whether it distinguishes that candidate from plausible alternatives. Both judgments must be positive for the Verifier to endorse the answer. For the president subquestion in Figure 1, the evidence must establish who held office on the signing date. The hypothetical proposal of Ronald Reagan would fail because his dates in office do not match that date. When verification fails, the Verifier returns feedback identifying missing support or ambiguity. The Researcher can use this feedback to gather further evidence or revise its proposal, with a limited number of retries per assignment. Because each review concerns an individual subquestion, proposals from different Researchers can be checked in parallel.

The harness admits a finding by filling the corresponding blank in the workspace and recording its evidence and review status (Appendix B). In the $W_0 \rightarrow W_1$ transition in Figure 1, admitting the signing date and Jimmy Carter's identity provides a date for the final calculation and the context needed to investigate his wife.

2.4 Answering the Query

As findings are admitted, they guide subsequent research and provide inputs for computation. In Figure 1, identifying Rosalynn Carter at W_2 supplies the context needed to investigate her birth date. The harness executes each planned computation in code once its inputs are available. Here, the birth date of 18 August 1927 and signing date of 17 September 1978 yield 51 completed years: her birthday has already occurred in the signing year. The age fills the answer target at W_3. Once the harness checks that the target is resolved, the Orchestrator submits the answer.

The investigation in Figure 1 completes its original plan, but a Researcher may discover that answering its assigned subquestion requires an intermediate fact missing from the plan. It can then end its assignment and return a suggested additional subquestion, together with an explanation of why it is needed, to the Orchestrator. The Orchestrator can ask the Planner to revise the existing plan using the findings and the Researcher's suggestion. Revisions can add or change subquestions and reopen affected findings for further research, while retaining the remaining records. Research then continues on the revised plan. If the overall investigation ends without a final answer to the original query, for example after exhausting its research allowance, the harness can make a final LLM call to synthesize that answer from the collected evidence, without further search (Appendix B).

3 Experimental Setup

3.1 Datasets

We use three complementary question-answering benchmarks. BrowseComp-Plus (BCP) tests identifying an entity from detailed clues (Chen et al., 2025b); FRAMES tests combining facts across sources (Krishna et al., 2024); and MoNaCo includes collection-valued questions, where recovering the required items without adding incorrect ones is important (Wolfson et al., 2025). For BCP, we use the official 100K-document corpus and the Context-Folding CF-150 split, with 50 questions in each of the easy, medium, and hard groups (Sun et al., 2025). We evaluate on all 824 FRAMES questions and all 1,315 MoNaCo questions.

3.2 Evaluation Metrics

Answer quality. We report *official benchmark scores*: accuracy on BCP and FRAMES and mean item-level F1 on MoNaCo. All systems receive 3 `gpt-5.6-luna` judgments of each answer, aggregated by majority vote on BCP and FRAMES and maximum F1 over complete assessments on MoNaCo, since item matching can spuriously lower F1 in different assessments. Every question remains in the denominator; missing judgments score 0. Appendix A details scoring and a second-grader sensitivity check.

Observed support. Accuracy alone does not show whether the agent has evidence for its answer (Rashkin et al., 2023; Gao et al., 2023). *Observed support* (Obs.) keeps answer credit only when the credited content is recoverable from the system's saved evidence. For BCP and FRAMES, let c_i indicate that answer i is officially correct and s_i indicate that every checkable answer part is recoverable from its saved evidence. Over N questions,

$$\mathrm{Obs} = \frac{1}{N}\sum_{i=1}^{N} c_i s_i. \tag{1}$$

For MoNaCo, we recompute F1 using only judge-credited items that pass the support rule. Obs. measures recoverability through deterministic matching, and Appendix C.4 extends it to computed answers. Section 4.2 separately tests whether the evidence supports the requested claim. Appendix C details the support scorer.

Research cost. Locally served models reuse cached prefix states, so submitted-token totals overstate computation. We therefore also estimate computation from uncached input and generated tokens (Section 4.4).

3.3 Baselines

We compare against seven harnesses with different ways of organizing research and retaining findings. **PyRAG** (Sun et al., 2026) executes generated retrieval programs; **Re-TRAC** (Zhu et al., 2026b) compresses research trajectories; and **IterResearch** (Chen et al., 2025a) maintains an evolving research report. **SLEUTH** (Liu, 2026) tracks facts, ranked hypotheses, and open questions; **Omni-Decision** (Ma et al., 2026) maintains query-specific evidence needs; and **HyMem** (Wang et al., 2026) manages episode, working, and tool memories through folding and selective injection into later calls. **Open Deep Search** (Alzubi et al., 2025) provides a CodeAct search-and-tool agent.

We use released implementations where available. For the prompt-based SLEUTH, we follow the authors' published prompts and reasoning protocol; we reimplement Omni-Decision and HyMem from their published specifications. Appendix D.5 documents adaptations to the shared environment.

3.4 Implementation Details

All systems use `gpt-oss-120b` (OpenAI, 2025) and, within each benchmark, the same frozen corpus and retrieval service: BM25 and dense `Qwen3-Embedding-8B` retrieval, followed by `jina-reranker-v3.5`. Baselines access it through a shared search gateway that returns full documents on BCP and chunks on FRAMES and MoNaCo. SLEUTH keeps its published top-3 setting, chosen to avoid context dilution (Liu, 2026); other baselines use published retrieval breadth where specified and 6 hits otherwise, and PANINI-DR searches return 6 reranked snippets. Baselines keep their published or released-code research allowances; PANINI-DR allows each Researcher assignment 20 ReAct turns and up to 5 retries after failed checks (Appendix A).

4 Results and Discussion

We assess answer quality and retained evidence, then test the contributions of research structure and verification through ablations. Cost and allowance studies examine the resources needed for these gains.

4.1 Answer Quality

	BCP CF-150				FRAMES	MoNaCo	Avg.
Method	Easy	Med.	Hard	Overall	Accuracy	F1	
PyRAG	56	20	6	27.3	61.9	0.446	44.6
Re-TRAC	70	32	16	39.3	64.9	0.544	52.9
Omni-Decision	84	48	10	47.3	64.6	0.359	49.3
HyMem	82	52	12	48.7	70.8	**0.612**	60.2
Open Deep Search	82	50	<u>28</u>	53.3	76.0	<u>0.571</u>	62.1
IterResearch	84	<u>60</u>	20	54.7	70.5	0.564	60.5
SLEUTH	<u>86</u>	58	<u>28</u>	<u>57.3</u>	<u>77.2</u>	0.556	<u>63.4</u>
PANINI-DR	**92**	**76**	**52**	**73.3**	**79.1**	0.549	**69.1**

Table 1: Official accuracy (%) on BCP and FRAMES and mean F1 on MoNaCo. Avg. equally weights BCP Overall, FRAMES, and MoNaCo, with F1 rescaled to 0–100. BCP groups contain 50 questions each. Bold: best; underline: second best, including ties.

PANINI-DR has the best average across the three benchmarks, and its advantage is largest on the hardest questions (Table 1). Because all systems share the backbone and retrieval service, these differences come from the harnesses. On the BCP hard group, which ReAct agents solve least often, PANINI-DR nearly doubles SLEUTH's accuracy. The planning ablation points the same way, since adding the plan and persistent state doubles the hard questions answered (Appendix D.2.1). The 16-point BCP gain over SLEUTH holds when we remove the questions SLEUTH refuses for input length and is consistent across three runs per harness (Appendix D.4). On MoNaCo, HyMem has the highest F1, although our collection answers are the most precise. Once answers must be backed by saved evidence, PANINI-DR also leads on MoNaCo (Section 4.2).

4.2 Evidence Quality

Recording the evidence behind each finding should leave answers that readers can check against their sources. When correct answer content must be recoverable from saved evidence, PANINI-DR leads Obs. on all three benchmarks, and its margin over SLEUTH, the strongest baseline, widens compared with official scores (Table 2). A model-based claim-support audit tests whether the saved evidence supports the requested claims. It rebuilds each answer from the supported claims only. PANINI-DR remains ahead of SLEUTH on all three benchmarks under this audit (Appendix C.2).

Method	BCP	FRAMES	MoNaCo	Avg.
PyRAG	26.0	46.6	0.332	35.3
Re-TRAC	14.0	46.1	0.396	33.2
Omni-Decision	45.3	53.9	0.275	42.2
HyMem	46.7	36.7	0.388	40.7
Open Deep Search	51.3	47.6	0.325	43.8
IterResearch	51.3	45.9	0.390	45.4
SLEUTH	54.0	55.2	0.396	49.6
PANINI-DR	**70.7**	**66.1**	**0.458**	**60.9**

Table 2: Observed support (Obs.): official answer credit recoverable from saved evidence, as defined in Section 3.2. BCP and FRAMES report percentages; MoNaCo reports mean F1. Avg. equally weights the three benchmarks, with F1 rescaled to 0–100. Best scores are bold; second-best scores are underlined, including ties.

We also examine how research finds and retains relevant sources. On BCP, PANINI-DR reaches most of the annotated source pages, while SLEUTH reaches fewer than half, and more of the pages it cites are annotated. Both gains also hold on FRAMES and MoNaCo (Appendices D.1.1 and D.1.3). A matched trajectory shows how findings guide further research. PANINI-DR identifies Jim Bridger from biographical clues and then uses the name to find the expedition Bridger joined, while SLEUTH never establishes the identity. Finally, we test whether the retained evidence is useful beyond our own answer generation. In a common-reader test, the same `gpt-oss-120b` reader answers from the evidence of four systems. Our evidence yields the highest answer quality on all three benchmarks, whether the reader is told an answer is present or may abstain (Appendix C.3).

4.3 ABLATIONS

We first test whether structured planning and persistent research state improve answers beyond prose notes. Both reduced harnesses retain Researchers and the Verifier: OFF keeps notes and Researcher reports, while ON adds a plan, findings linked to its subquestions, and plan revision (Appendix D.1.2). Under the same configured limits, ON raises BCP accuracy from 41.3% to 58.0%. The 16.7-point gain persists when both use the same final answer synthesis (56.7% versus 73.3%). To check whether findings guide later work, we track newly recorded entities absent from the question and followed by another research assignment. With planning and state, 77.4% appear in a later search, compared with 32.1% without them: earlier findings more often become inputs to subsequent research (Appendix D.2.1).

Because errors in these findings can propagate, we next test verification before reuse. With the workspace and research tools fixed, the full verification policy with up to 5 retries adds 16.0 points to BCP accuracy and Obs. compared with removing the Verifier and the citation and evidence-attachment checks. Assignment and answer-type checks remain in both conditions. Gains also hold on the matched FRAMES and MoNaCo subsets (Appendix D.2.2). The proposal histories show how feedback changes the investigation: 77 of 101 proposals endorsed after an earlier rejection change the answer or cited sources. For example, rejecting an editor's surname leads to a full name supported by credits from both requested films (Appendix D.2.3). Verification thus helps improve findings through further evidence gathering before they guide later work.

4.4 COMPUTATION AND RESEARCH ALLOWANCES

PANINI-DR sends more prompt tokens than most baselines, though fewer than Re-TRAC (Table 14 in Appendix D.3.1). Most of these tokens go to the Researcher's ReAct search and its reading tool, and many calls resend context from earlier calls. The vLLM server caches recently processed prompt prefixes, so it does not recompute resent context. In separate BCP runs where we recorded its cache counters, 66.5% of our input tokens came from this cache. Our estimated computation is therefore $7.3\times$ SLEUTH's, although we send $19.2\times$ its prompt tokens (Appendix D.3.1).

We use this estimate to test whether repeated research with similar computation can close the accuracy gap. We run SLEUTH 8 times per question and take its most common non-empty answer. This

uses $1.12\times$ our computation but reaches 64.7% BCP accuracy, versus 73.3% for PANINI-DR (Appendix D.3.2). Instead of more independent runs, we also give each investigation more turns to build on its findings. Raising the Researcher's turn allowance from 20 to 50 and 80 turns improves BCP accuracy from 73.3% to 74.7% and 76.0%, while SLEUTH shows no steady gain (Appendix D.3.3). However, 80 turns uses $2.8\times$ the default's prompt tokens for 2.7 more points, so we keep the default of 20 turns.

5 Related Work

Research state and executable plans. Chain-of-thought prompting makes intermediate reasoning explicit within a model call (Wei et al., 2022). Self-Ask and IRCoT connect question decomposition to retrieval (Press et al., 2023; Trivedi et al., 2023). ReAct interleaves reasoning and tool use (Yao et al., 2023b), while ReWOO and LLMCompiler make tool dependencies explicit in execution plans (Xu et al., 2023; Kim et al., 2024). Several of our baselines preserve progress in evolving reports, compressed trajectories, or memories (Chen et al., 2025a; Zhu et al., 2026b; Wang et al., 2026); SLEUTH and Omni-Decision also maintain explicit evidence state with checked updates (Liu, 2026; Ma et al., 2026). PANINI-DR builds on the GSW representation (Rajesh et al., 2026a;b); its contribution is the execution protocol connecting typed research assignments, subquestion-scoped review, and admitted values used by subsequent work, rather than a dependency graph alone.

Verification during research. Process supervision and verifier-guided search use intermediate checking to improve reasoning (Lightman et al., 2023; Snell et al., 2024). Chain-of-Verification revises drafts using separate checking questions (Dhuliawala et al., 2024). Self-RAG learns retrieval and generation critiques (Asai et al., 2024); CRAG evaluates retrieved material to guide corrective retrieval (Yan et al., 2024). LLM-as-a-Verifier studies fine-grained, decomposed model-based evaluation (Kwok et al., 2026). PANINI-DR connects this checking role to persistent research subquestions: the Verifier evaluates candidates against their subquestions' verification claims and cited evidence, and the harness determines admission while retaining the review status.

Attribution and support. Attribution work evaluates whether sources support generated claims (Rashkin et al., 2023; Gao et al., 2023; Malaviya et al., 2024). Observed support complements this with deterministic answer recoverability from saved evidence. Appendix E gives an extended discussion of related work.

6 Limitations and Concluding Remarks

In this work, we introduce PANINI-DR, a multi-agent harness for deep research built around two ideas: maintaining explicit research state and verifying intermediate findings before they guide subsequent work. We implement these ideas using the GSW representation to organize subquestions, their dependencies, findings, and supporting evidence in a shared workspace. The workspace allows the Orchestrator to coordinate the investigation while separate agents research and verify individual subquestions. Across three deep research benchmarks, PANINI-DR achieves the strongest average answer quality, with particularly large gains on harder BrowseComp-Plus questions. Ablations support the value of both structured research state and intermediate verification. Our observed-support evaluation further shows that this advantage widens when correct answer content must be recoverable from the retained evidence.

However, we identify several limitations to our work. Verification depends on the subquestions identified by the Planner, the candidates considered during research, and the evidence available to the Verifier. Model-based judgments can be incorrect, and checking individual findings cannot guarantee that the plan covers everything needed to answer the original query. Our primary comparisons use one research backbone and fixed benchmark corpora; generalization to other models and open-web research remains untested. Observed support measures answer recoverability from saved evidence, whose coverage and format vary across systems (Appendix C). It does not establish that every claim is supported, and our complementary claim-support audits also rely on model judgments. The harness requires substantial computation, while increasing the verification allowance does not consistently improve answer quality (Section 4.4; Appendix D.2.3). In future work, we plan to expand our study to a wider range of models and open-web datasets. Developing reliable, computationally efficient methods for scaling verification is therefore an important direction for future research.

Reproducibility Statement

Appendix A records the frozen corpora, shared backbone, retrieval service, run limits, and judging protocol for every comparison; Appendix D.5 records all baseline compatibility changes. The observed-support scorer deterministically combines each system's saved traces, source references, and cached judgments, as specified in Appendix C. The release will contain the harness, gateway, scorer, evaluation configurations, and analysis scripts in a public GitHub repository after the anonymous review process.

Ethics Statement

This work aims to improve the reliability of deep research systems and evaluates the proposed methods on existing benchmark datasets. We do not identify additional ethical concerns specific to this study beyond those associated with language models and their applications.

AI Use Statement

Language models were used as components of the systems studied and as graders, as described in Section 3. The authors also used AI assistance to edit the draft and to catch grammatical errors. The authors take full responsibility for the contents of the paper.

References

Salaheddin Alzubi, Creston Brooks, Purva Chiniya, Edoardo Contente, Chiara von Gerlach, Lucas Irwin, Yihan Jiang, Arda Kaz, Windsor Nguyen, Sewoong Oh, Himanshu Tyagi, and Pramod Viswanath. Open deep search: Democratizing search with open-source reasoning agents, 2025. URL https://arxiv.org/abs/2503.20201.

Hong-Yu An, Yun-Jian Zhang, Chen-Wei Liang, Tian-Yi Zhang, Jian Ding, Yi-Lun Wu, Ao-Bo Li, Wei-Cong Su, Saifullah, and Mujiangshan Wang. HyCE-RAG: Hypergraph chain-of-evidence retrieval-augmented generation for explainable multi-hop question answering, 2026. URL https://arxiv.org/abs/2607.22597.

Anthropic. Formalizing Fermat's last theorem. Anthropic Research, 2026. URL https://www.anthropic.com/research/formalizing-fermats-last-theorem. September 4.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=hSyW5go0v8.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17682–17690, 2024. doi: 10.1609/aaai.v38i16.29720. URL https://ojs.aaai.org/index.php/AAAI/article/view/29720.

Om Chabra, Noah Ziems, Meng Jiang, Omar Khattab, and Hari Balakrishnan. Dossier: Deep research via ledger-driven branching search and query encoding learning. In *Proceedings of the ACM Conference on AI and Agentic Systems*, pp. 263–275. Association for Computing Machinery, 2026. doi: 10.1145/3786335.3813122. URL https://people.csail.mit.edu/hari/papers/p193.pdf.

Guoxin Chen, Zile Qiao, Xuanzhong Chen, Donglei Yu, Haotian Xu, Wayne Xin Zhao, Ruihua Song, Wenbiao Yin, Huifeng Yin, Liwen Zhang, Kuan Li, Minpeng Liao, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. IterResearch: Rethinking long-horizon agents with interaction scaling, 2025a. URL https://arxiv.org/abs/2511.07327.

Shuze Chen, Kunal Marwaha, Xiaoyang Lu, Henry Yuen, and Tianyi Peng. Prove2Me: An open collaborative platform for scaling math formalization. *arXiv preprint arXiv:2608.28433*, 2026. URL https://arxiv.org/abs/2608.28433.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, Sahel Sharifymoghaddam, Yanxi Li, Haoran Hong, Xinyu Shi, Xuye Liu, Nandan Thakur, Crystina Zhang, Luyu Gao, Wenhu Chen, and Jimmy Lin. Browsecomp-plus: A more fair and transparent evaluation benchmark of deep-research agent. *arXiv preprint arXiv:2508.06600*, 2025b.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Łukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-verification reduces hallucination in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3563–3578. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.212. URL https://aclanthology.org/2024.findings-acl.212/.

Dhruv Dixit and Paritosh Pandey. R2VC: Modular fact-checking with retrieval, verification, and confidence calibration, 2026. URL https://arxiv.org/abs/2609.11955.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 6465–6488, 2023. doi: 10.18653/v1/2023.emnlp-main.398. URL https://aclanthology.org/2023.emnlp-main.398/.

Kelly Hong, Anton Troynikov, and Jeff Huber. Context rot: How increasing input tokens impacts LLM performance. Technical report, Chroma, July 2025. URL https://trychroma.com/research/context-rot.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 874–880, 2021. doi: 10.18653/v1/2021.eacl-main.74. URL https://aclanthology.org/2021.eacl-main.74/.

Yuelyu Ji, Rui Meng, Zhuochun Li, and Daqing He. Memory-aware and uncertainty-guided retrieval for multi-hop question answering, 2025. URL https://arxiv.org/abs/2503.23095.

Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W. Mahoney, Kurt Keutzer, and Amir Gholami. An LLM compiler for parallel function calling. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 24370–24391. PMLR, 2024. URL https://proceedings.mlr.press/v235/kim24y.html.

Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqui. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation, 2024. URL https://arxiv.org/abs/2409.12941.

Jacky Kwok, Shulu Li, Pranav Atreya, Yuejiang Liu, Yixing Jiang, Chelsea Finn, Marco Pavone, Ion Stoica, and Azalia Mirhoseini. LLM-as-a-verifier: A general-purpose verification framework. *arXiv preprint arXiv:2607.05391*, 2026. URL https://arxiv.org/abs/2607.05391.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven C. H. Hoi. CodeRL: Mastering code generation through pretrained models and deep reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 21314–21328, 2022. URL https://arxiv.org/abs/2207.01780.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://papers.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. WebThinker: Empowering large reasoning models with deep research capability. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-4011. URL https://papers.nips.cc/paper_files/paper/2025/hash/ae03bdef276132fae089692445725635-Abstract-Conference.html.

Junli Liang, Pengfei Zhou, Wangqiu Zhou, Wenjie Qing, Qi Zhao, Ziwen Wang, Qi Song, and Xiangyang Li. SentGraph: Hierarchical sentence graph for multi-hop retrieval-augmented question answering, 2026. URL https://arxiv.org/abs/2601.03014.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023. URL https://arxiv.org/abs/2305.20050.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Ning Liu. Track, rank, crack: Epistemic working memory scales multi-hop reasoning in language agents, 2026. URL https://arxiv.org/abs/2607.12267.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. Faithful chain-of-thought reasoning. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 305–329. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.ijcnlp-main.20. URL https://aclanthology.org/2023.ijcnlp-main.20/.

Ming Ma, Yi Zhu, Yiran Zhong, Feida Zhu, Weigao Sun, Junhan Shi, Lingrui Mei, Tianming Yang, and Steven Hoi. Omni-Decision: A progressive evidence-state agent system for omni-modal QA, 2026. URL https://arxiv.org/abs/2607.11433.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46534–46594, 2023. URL https://arxiv.org/abs/2303.17651.

Chaitanya Malaviya, Subin Lee, Sihao Chen, Elizabeth Sieber, Mark Yatskar, and Dan Roth. ExpertQA: Expert-curated questions and attributed answers. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3025–3045, 2024. doi: 10.18653/v1/2024.naacl-long.167. URL https://aclanthology.org/2024.naacl-long.167/.

Thang Nguyen, Peter Chin, and Yu-Wing Tai. MA-RAG: Multi-agent retrieval-augmented generation via collaborative chain-of-thought reasoning, 2025. URL https://arxiv.org/abs/2505.20096.

OpenAI. gpt-oss-120b & gpt-oss-20b model card, 2025. URL https://openai.com/index/gpt-oss-model-card/.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023. URL https://arxiv.org/abs/2310.08560.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023. doi: 10.1145/3586183.3606763. URL https://doi.org/10.1145/3586183.3606763.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-emnlp.378. URL https://aclanthology.org/2023.findings-emnlp.378/.

Jingxi Qiu, Zeyu Han, and Cheng Huang. SURE-RAG: Sufficiency and uncertainty-aware evidence verification for selective retrieval-augmented generation, 2026. URL https://arxiv.org/abs/2605.03534.

Shreyas Rajesh, Pavan Holur, Chenda Duan, David Chong, and Vwani Roychowdhury. Beyond fact retrieval: Episodic memory for RAG with generative semantic workspaces. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 32782–32790, 2026a. doi: 10.1609/aaai.v40i39.40557. URL https://ojs.aaai.org/index.php/AAAI/article/view/40557.

Shreyas Rajesh, Pavan Holur, Mehmet Yigit Turali, Chenda Duan, and Vwani Roychowdhury. Panini: Continual learning in token space via structured memory, 2026b. URL https://arxiv.org/abs/2602.15156.

Hannah Rashkin, Vitaly Nikolaev, Matthew Lamm, Lora Aroyo, Michael Collins, Dipanjan Das, Slav Petrov, Gaurav Singh Tomar, Iulia Turc, and David Reitter. Measuring attribution in natural language generation models. *Computational Linguistics*, 49(4):777–840, 2023. doi: 10.1162/coli_a_00486. URL https://aclanthology.org/2023.cl-4.2/.

Yuling Shi, Maolin Sun, Zijun Liu, Mo Yang, Yixiong Fang, Tianran Sun, and Xiaodong Gu. Reasoning in trees: Improving retrieval-augmented generation for multi-hop question answering, 2026. URL https://arxiv.org/abs/2601.11255.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024. URL https://arxiv.org/abs/2408.03314.

Jiashuo Sun, Jimeng Shi, Yixuan Xie, Saizhuo Wang, Jash Rajesh Parekh, Pengcheng Jiang, Zhiyi Shi, Jiajun Fan, Qinglong Zheng, Peiran Li, Shaowen Wang, Ge Liu, and Jiawei Han. Retrieval is cheap, show me the code: Executable multi-hop reasoning for retrieval-augmented generation, 2026. URL https://arxiv.org/abs/2605.12975.

Weiwei Sun, Miao Lu, Zhan Ling, Kang Liu, Xuesong Yao, Yiming Yang, and Jiecao Chen. Scaling long-horizon llm agent via context-folding, 2025. URL https://arxiv.org/abs/2510.11967.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10014–10037, 2023. doi: 10.18653/v1/2023.acl-long.557. URL https://aclanthology.org/2023.acl-long.557/.

Yuxuan Wan, Tianqing Fang, Zaitang Li, Yintong Huo, Wenxuan Wang, Haitao Mi, Dong Yu, and Michael R. Lyu. Inference-time scaling of verification: Self-evolving deep research agents via test-time rubric-guided verification. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 24822–24835. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1243. URL https://aclanthology.org/2026.findings-acl.1243/.

XinQi Wang, Jinwei Xiao, Sijia Cui, Hongming Zhang, Yanna Wang, Qingyang Zhang, and Bo Xu. HyMem: Hierarchical context management for long-horizon agents via information isolation. *arXiv preprint arXiv:2608.15703*, 2026. URL https://arxiv.org/abs/2608.15703.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Tomer Wolfson, Harsh Trivedi, Mor Geva, Yoav Goldberg, Dan Roth, Tushar Khot, Ashish Sabharwal, and Reut Tsarfaty. MoNaCo: More natural and complex questions for reasoning across dozens of documents, 2025. URL https://arxiv.org/abs/2508.11133.

Xixi Wu, Kuan Li, Yida Zhao, Liwen Zhang, Litu Ou, Huifeng Yin, Zhongwang Zhang, Yong Jiang, Pengjun Xie, Fei Huang, Minhao Cheng, Shuai Wang, Hong Cheng, and Jingren Zhou. ReSum: Unlocking long-horizon search intelligence via context summarization. *arXiv preprint arXiv:2509.13313*, 2025. URL https://arxiv.org/abs/2509.13313.

Binfeng Xu, Zhiyuan Peng, Bowen Lei, Subhabrata Mukherjee, Yuchen Liu, and Dongkuan Xu. ReWOO: Decoupling reasoning from observations for efficient augmented language models, 2023. URL https://arxiv.org/abs/2305.18323.

Ke Xu, Han Xu, Xinran Chen, Yuqian Wang, Zhixuan Li, Xiaojian Liu, Changwo Wu, Jianqiang Xia, and Yuchen Li. STAMP: Provenance-guided credit assignment for deep search agents, 2026. URL https://arxiv.org/abs/2607.11172.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. Corrective retrieval augmented generation, 2024. URL https://arxiv.org/abs/2401.15884.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023a. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract-Conference.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=WE_vluYUL-X.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. DeepResearcher: Scaling deep research via reinforcement learning in real-world environments. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 414–431, 2025. doi: 10.18653/v1/2025.emnlp-main.22. URL https://aclanthology.org/2025.emnlp-main.22/.

Bin Zhu, Qianghuai Jia, Tian Lan, Junyang Ren, Feng Gu, Feihu Jiang, Longyue Wang, Zhao Xu, and Weihua Luo. Marco deepresearch: Unlocking efficient deep research agents via verification-centric design, 2026a. URL https://arxiv.org/abs/2603.28376.

Jialiang Zhu, Gongrui Zhang, Xiaolong Ma, Lin Xu, Miaosen Zhang, Ruiqi Yang, Song Wang, Kai Qiu, Zhirong Wu, Qi Dai, Ruichun Ma, Bei Liu, Yifan Yang, Chong Luo, Zhengyuan Yang, Linjie Li, Lijuan Wang, Weizhu Chen, Xin Geng, and Baining Guo. RE-TRAC: Recursive trajectory compression for deep search agents, 2026b. URL https://arxiv.org/abs/2602.02486.

Kunlun Zhu, Zijia Liu, Bingxuan Li, Muxin Tian, Yingxuan Yang, Jiaxun Zhang, Pengrui Han, Qipeng Xie, Fuyang Cui, Weijia Zhang, Xiaoteng Ma, Xiaodong Yu, Gowtham Ramesh, Jialian Wu, Zicheng Liu, Pan Lu, James Zou, and Jiaxuan You. Where LLM agents fail and how they can learn from failures. *arXiv preprint arXiv:2509.25370*, 2025. URL https://arxiv.org/abs/2509.25370.

Yikai Zhu, Kunfeng Chen, Qihuang Zhong, Juhua Liu, and Bo Du. ConRAG: Consensus-driven multi-view retrieval for multi-hop question answering, 2026c. URL https://arxiv.org/abs/2605.28093.

APPENDIX CONTENTS

Section	Contents	Page
A	Evaluation Protocol Details	17
A.1	Shared Evaluation Setup	17
A.2	Answer Generation and Grading	17
A.3	Sensitivity to the Grading Model	18
B	Implementation Through Recorded Investigations	18
B.1	From the Question to a Research Plan	18
B.2	Brazuca: Findings, Feedback, and Evidence Completion	19
B.3	When Research Takes a Different Path	20
B.4	Fort Santiago: Answering from the Retained Research	21
C	Evidence Quality and Validation	22
C.1	Checking the Scorer and Evidence Record	22
C.2	Model-Based Grounded-Only Audit	23
C.3	Answering from Each System's Evidence	23
C.4	Checking Computed Answers	24
D	Additional Experimental Results	25
D.1	Research Evidence and Answer Completion	25
D.1.1	Finding Sources and Connecting Findings	25
D.1.2	Revising the Plan and Recovering an Answer	25
D.1.3	Answering with Annotated Sources	26
D.2	Planning, Verification, and Recovery Controls	26
D.2.1	Structured Planning and Persistent State	26
D.2.2	Verification Before Reuse	27
D.2.3	Responding to Verification Feedback	28
D.3	Computation and Research Allowances	28
D.3.1	Recorded Tokens and Cache Reuse	28
D.3.2	Repeated Sampling at Comparable Computation	29
D.3.3	Extending Individual Investigations	30
D.4	Run Stability, Paired Comparisons, and Answer Structure	31
D.5	Baseline Adaptations	31
E	Extended Related Work	33
E.1	Reasoning Structure, Plans, and Semantic Workspaces	33
E.2	Deep-Research Harnesses and Persistent Memory	34
E.3	Verification, Feedback, and Corrective Retrieval	36
E.4	Attribution and Evidence-Based Evaluation	37

A Evaluation Protocol Details

A.1 Shared Evaluation Setup

To compare complete research harnesses on a common foundation, we share the backbone, fixed corpus, and retrieval service within each benchmark. The CF-150 difficulty groups in Section 3.1 are based on ReAct accuracy over 8 runs. FRAMES questions draw on 2 to 15 articles; Table 20 separates MoNaCo's single-item and collection questions.

The shared backbone is `gpt-oss-120b`, served by vLLM 0.24 with MXFP4 weights, temperature 0, seed 42, and medium reasoning effort unless a system sets its own. All systems use BM25 and Qwen3-Embedding-8B retrieval followed by `jina-reranker-v3.5`. The baseline search gateway returns full documents on BCP and chunks on FRAMES and MoNaCo; on BCP, this adapts SLEUTH's original paragraph retrieval (Liu, 2026). PANINI-DR returns 6 snippets per search, with source reading handled separately (Appendix B).

Across all 3 benchmarks, PANINI-DR uses medium effort for the Planner, Researcher's ReAct stage, Reader, and Verifier, with 20 ReAct turns per Researcher assignment, 12 Orchestrator turns, and 5 plan-update requests. Its shared review cap of 5 limits failed-check re-attempts, not total Researcher turns or Verifier calls. Baselines retain their own budgets (Table 3): Re-TRAC, Omni-Decision, SLEUTH, and HyMem follow paper-reported values; IterResearch and Open Deep Search use released-code defaults. Retrieval breadth follows published values where specified and otherwise defaults to 6. SLEUTH retains its published top-3 setting to avoid diluting useful evidence with additional retrieved text (Liu, 2026). These comparisons do not equalize computation; Appendix D.3.2 tests additional sampling at comparable computation. Appendix D.5 documents code adaptations and paper-based reimplementations.

System	Protocol
PyRAG	1 executable retrieval program; no read tool.
Re-TRAC	8 recursive search cycles, each compressed into the next cycle's starting state.
IterResearch	25 turns; sampling temperature is hard-coded at 0.6.
Omni-Decision	15 iterations behind an evidence-state readiness gate.
Open Deep Search	15 CodeAct steps; search returns model-written summaries.
SLEUTH	Published default of 20 turns and retrieval top-$k = 3$; maintains ranked hypotheses.
HyMem	12 planner turns and 10 executor turns per task, with top-$k = 6$.

Table 3: Baseline protocols used for the reported runs.

A.2 Answer Generation and Grading

The headline comparisons use 1 research run per configuration, producing 1 answer per question. We grade that same answer 3 times with `gpt-5.6-luna`, replacing the benchmark judge model while retaining the BrowseComp-Plus criteria for BCP and FRAMES and the official MoNaCo criteria. BCP and FRAMES use majority voting over the 3 judgments of full-answer correctness.

For MoNaCo, a grading pass can overlook valid item matches. We therefore select the highest-F1 complete judgment of the 3 passes, identically for every system, and use its item-level credits for both official F1 and support accounting. Every question remains in the denominator; N counts questions, not grading passes.[1]

The sampling control uses these same grading rules to test whether repeated SLEUTH runs can match PANINI-DR at comparable computation (Appendix D.3.2, Table 17). For each question, we select the most frequent non-empty answer across runs. This answer-selection vote is separate from the 3 grading judgments; accuracy and Obs. use the selected run's verdict and saved evidence.

[1] Blank FRAMES answers score 0 without a grader call. Missing judgments score 0; unreadable judgments do not count as correct. Otherwise, we keep the official judgment, even when it gives credit to an empty answer or an abstention.

A.3 Sensitivity to the Grading Model

Because answer correctness is assessed by an LLM judge, we test whether PANINI-DR's BCP lead depends on the cost-focused `gpt-5.6-luna` grader. We re-grade the same saved answers for all 150 questions from 5 systems with `gpt-5.6-terra`, which is designed to balance capability and cost,[2] keeping all other grading settings fixed (Table 4). Four systems receive identical verdicts on every question; PANINI-DR gains 1 correct answer. The BCP ranking therefore survives the grader change, with the lead over the strongest baseline increasing from 24 to 25 questions.

System	Primary grader	Second grader	Δ	Agreement	κ
PANINI-DR (cap 5)	110 (73.3)	111 (74.0)	+0.7	99.3	0.983
SLEUTH	86 (57.3)	86 (57.3)	0.0	100	1.000
HyMem	73 (48.7)	73 (48.7)	0.0	100	1.000
Omni-Decision	71 (47.3)	71 (47.3)	0.0	100	1.000
Re-TRAC	59 (39.3)	59 (39.3)	0.0	100	1.000

Table 4: Grader sensitivity on BCP CF-150 (150 questions): `gpt-5.6-luna` (primary) versus `gpt-5.6-terra`, with the same answers, official prompt, and 3-pass majority. Scores are correct-answer counts with accuracy (%) in parentheses; Δ is in percentage points. Agreement is the percentage of matching verdicts; κ is Cohen's kappa.

B Implementation Through Recorded Investigations

BCP q628 follows an investigation from its initial plan to the answer *Brazuca*, showing how Researchers obtain intermediate findings, respond to feedback, and use earlier findings in later assignments. We then examine collection answers, retries, plan revision, and recovery through other recorded investigations.

B.1 From the Question to a Research Plan

The full question is:

> There is a short film, produced in 2017, with a runtime of 19 minutes, that revolves around a young boy and his pursuit to win a special ball and rise up in the eyes of his friends. The director of this film was born before 1990 and after 1980, he studied Public Administration, and this film is his third. The film's editor was also an editor for a 2016 film about murder solving. The production designer for the film in question was also the art director for the same film.

Finding a plausible title is therefore only part of the investigation: the research must connect that title to the people, dates, and credits in the question.

Making the clues researchable. The Planner connects the film to its year, runtime, director, editor, and designer, then connects those people to the required biography and credits. In this cap-5 execution, the plan has 9 given anchors, 13 relations, and 12 plan nodes with unknown values (*blanks*): the answer target, 7 intermediate facts, and 4 Boolean outputs. Four deterministic constraints check runtime equality, third-film equality, and the 2 birth-year bounds. Code evaluates these comparisons once their inputs are available; they need no research assignments.

Omitting step subscripts, write $P = (\mathcal{A}, V, E, \mathcal{C}, v_\star)$: anchors, nodes, dependency edges, computations, and answer target. $G = (V, E)$ is Section 2.1's dependency graph. Each node specifies its role, value type, relations, retrieval hint, and verification claim; each computation names its function, inputs, and output. Before research, the harness requires a nonempty plan, one target, valid references, and no dependency cycle; LLM-proposed repairs must pass the same checks. Advisory tags flag possible omissions or ambiguity without blocking dispatch or changing the plan. These checks do not establish that the plan can produce an answer or uses consistent units.

[2] OpenAI's model descriptions: GPT-5.6 Luna and GPT-5.6 Terra.

The workspace $W = (P, S)$ starts with no admitted findings in S. The Orchestrator chooses assignments using the frontier $F(W)$: unresolved plan nodes whose dependencies are given or admitted. Research outside this frontier is logged, not blocked: a Researcher may investigate unresolved prerequisites while answering its assigned subquestion. Computations require available inputs. Here the plan stays unchanged through 3 dispatches requesting 4, 3, and 1 research nodes.

B.2 BRAZUCA: FINDINGS, FEEDBACK, AND EVIDENCE COMPLETION

First dispatch: obtain the runtime and identify the people. The initial assignments request the runtime, editor, production designer, and director; the film target remains unresolved. Each Researcher first uses a typed-resolver stage: focused retrieval, one extraction call, and one verification attempt. Missing retrieval-hint inputs skip focused retrieval. If no candidate is found or verification fails, the same Researcher continues with a ReAct agent, retaining the candidate, sources, and feedback.

The Researchers assigned runtime and director propose 19 minutes and Faidon Gkretsikos at the typed-resolver stage. Their *Brazuca* programme page gives the production year, duration, director, birth year, Public Administration studies, and third-short-film description. Both proposals receive positive satisfaction and uniqueness judgments. Admission stores $S(v) = (x, \mathcal{E}_v, \sigma)$: value, evidence references, and verification status. These VERIFIED findings become available to later work, and the runtime comparison evaluates to true. Admission changes S, not P.

Continue research when the first lookup is insufficient. For editor and designer, the typed-resolver stage returns `unsupported` without candidates; both Researchers continue in ReAct within the same dispatch. The designer Researcher asks the separate Reader whether the programme establishes both jobs. Its extraction with supporting quotations reports "Sets: Eleni Kariori" but no art-director credit. Reading Kariori's credits page then finds separate *Brazuca* (2017) entries under Production Designer and Art Director: film-specific credits, not occupations inferred from the heading. The Researcher submits Eleni Kariori with both passages. The Verifier endorses satisfaction and uniqueness, and the harness admits the finding.

Search and reading supply material but do not resolve a plan node. Before proposing a value, the ReAct Researcher can attach passages to relations. Attachment checks scope and identifiers, not support for the claim. The deterministic checks require attachments for checkable factual relations whose partners are given or resolved; the Verifier assesses whether the finding satisfies the assigned claim. The initial retrieval stage sends its cited proposal directly to type checking and LLM review, skipping the relation-attachment checks used in ReAct, as shown in Figure 1.

Use verification feedback to resolve an ambiguous identity. The editor Researcher submits Georgopoulos, citing the programme's "Editing: Yorgos Georgopoulos" and the editor list for *The Other Me*. The Verifier rejects the surname as non-unique, pointing to another listed editor. The Researcher retrieves separate *Brazuca* credits naming Giorgos Georgopoulos and resubmits that full name with both films' passages, now using the same spelling. Both judgments are positive: feedback leads to a more specific proposal and sources establishing the shared-editor link.

The Verifier receives the plan node's ID and type, candidate, resolved context, Researcher note, and passages. Known values fill the Planner's verification claim; absent or unfilled claims fall back to relations involving this node. ReAct supplies deduplicated proposal and relation-bound citations, so evidence can be present even with no proposal citations; the typed-resolver stage supplies its cited chunks. Endorsement requires satisfaction and uniqueness; rejection explains missing support or ambiguity. Missing decisions, unreadable replies, and failed calls do not endorse the proposal. The Verifier does not search for rivals, treats snapshot dates as given, and can defer clauses assigned elsewhere or awaiting unresolved values. Its judgment covers the currently checkable claim.

Second dispatch: use the admitted identities. With the people identified, Researchers seek the other film, the director's film number, and his birth year. Their typed-resolver stages propose *The Other Me*, 3, and 1988. The other-film passage names Giorgos Georgopoulos as editor and describes a criminology professor investigating 5 murders; the programme biography supplies the director values. All 3 receive positive satisfaction and uniqueness judgments. The remaining comparisons evaluate to true: $3 = 3$, $1988 < 1990$, and $1988 > 1980$. All 7 intermediate facts and 4 DERIVED

comparisons are now stored, but the target remains unresolved. Computations use admitted values without another LLM check; the plan identifies inputs without a versioned history.

Bindings carry findings; retained research context helps later assignments continue. ReAct keeps recent tool results in context and older ones as retrievable pointers. Across assignments, a dossier for each researched subquestion retains candidates, sources, feedback, and searches. Dossiers, conversations, and archived results do not admit values: later research can use this context without treating every note as an admitted finding.

Third dispatch: complete the target's evidence. The target Researcher's typed-resolver stage returns `unsupported`, so it continues in ReAct. It binds the programme to the target's production-year, runtime, director, editor, and production-design connections, then submits *Brazuca*. The deterministic checks refuse the update because the art-director connection has not been bound. The Researcher reads Kariori's credits again and attaches that passage to the missing connection. Its second submission reaches the Verifier but is rejected: the supplied programme and designer passages do not establish the connection to *The Other Me*. After another read of that film's editor credits, the Researcher submits *Brazuca* with all 3 passages. The Verifier now endorses satisfaction and uniqueness, and the harness admits the target.

The workspace finishes with 8 factual bindings endorsed by the Verifier and 4 DERIVED Boolean outputs. The Orchestrator submits *Brazuca*. A separate LLM synthesis check assesses whether the final answer follows from retrieved facts; its repair mode can replace the final answer text with a high-confidence local correction. Here it accepts *Brazuca*, without warned admission or a Finalizer call. Ordinary submission requires a resolved target, not a separate check that all dependencies are resolved.

Separately, an LLM *post-loop grounding check* assesses admitted relations against cited passages or bounded article windows without changing bindings. The plan's 2016 year matches *The Other Me (2016)*, but conflicts with the cited 19 January 2017 release date, which this check marks unsupported. The plot clue guides retrieval but has no dedicated plan relation and is absent from the final target Verifier's input.

Verification guides further research. Across 3 target proposals in 1 dispatch, the title and plan stay fixed while feedback prompts further reading: first to attach the art-director evidence, then to establish the shared-editor link. Earlier findings guide later assignments, and verification helps complete the evidence for the target's checked conditions.

B.3 When Research Takes a Different Path

Brazuca returns one film, passes verification, and keeps its original plan. Other investigations may seek several answers, exhaust verification allowances, or reveal a missing research step.

Answering from a collection of findings. MoNaCo `monaco_1157` stores 3 government positions as one admitted list, then computes the answer 3 with `count`. Like Brazuca's comparisons, it stores a finding and then computes from it. Evidence and verification cover the whole list, not each member; later admission replaces the list rather than taking a union. Lists are not automatically complete: the Verifier records completeness and missing items, but these fields do not independently gate endorsement.

Continuing after unsuccessful checks. Failed checks return feedback so the Researcher can gather more evidence, as in Brazuca. The deterministic checks and the Verifier share a ReAct review cap of 5 failed-check re-attempts, separate from the typed-resolver stage's single attempt. At exhaustion, the harness can admit a WARNED value marked `refuted_best_effort`. The plan node must remain known and assigned, the value valid and correctly typed, and citation identifiers valid. This admits without endorsement: incomplete evidence attachments skip the Verifier, or admission follows a negative verdict.

The Orchestrator can retry unresolved research assignments with new hints and retained context. To revisit an admitted finding, including a warned one, it must request a supported plan revision that reopens the plan node, then dispatch it again. Warnings do not automatically schedule retries. Until reopened, verified and warned findings remain usable in research and computation. Their

verification flags remain in bindings, not ordinary state or dispatch summaries, and DERIVED outputs do not inherit them. Appendix D.2.2 examines feedback and outcomes.

An assignment ends when its plan node resolves, its allowance runs out, or the Researcher suggests revision, not through a separate self-reported finish. Repeat assignments can have smaller allowances. The Orchestrator can continue with assignments or revision until its overall allowance runs out or repeated rounds without grounded progress trigger a stop.

Adding a missing research step. BCP q1266 initially plans only the person being sought. Before dispatch, the Orchestrator adds a subquestion for the series associated with that person's 2016 breakthrough performance. This resolves on turn 2, the target on turn 3, and turn 4 submits the correct answer, *Maryam Tanveer Ali*. Revision turns a missing intermediate fact into a research assignment (Appendix D.1.2).

Only the Orchestrator requests updates; a Researcher's suggestion ends its assignment without changing the plan. The Planner uses separate prompts for initial plan creation and revision. The revision call receives the question, current plan, bindings, reason, and evidence, and proposes edits to the existing plan. The prompt asks for at most 2 requests; the runtime permits 5, counting rejections, with up to 3 edit attempts per request. At the 5-request limit, best-effort submission can bypass target resolution. This occurred on 1 MoNaCo question and is separate from Finalizer recovery.

Revision also determines which findings need reopening. A first pass initializes new plan nodes, removes deleted ones, and resets surviving outputs of added, removed, or changed computations. It follows `derived`, `argmax`/`argmin`, and comparison dependencies, not `rank_n`, `filter`, or `count`. A second pass reopens preserved resolved nodes whose definitions changed and their resolved descendants. First-pass resets are not starting points for this second pass, so some affected findings remain resolved. Reopening clears values, evidence, and verification records; a failed final structural check restores the previous plan and state. Ordinary value replacement does not reset descendants, and computations update only unresolved outputs. In BCP q591, later edits leave an incorrect target resolved, preserving the wrong answer (Appendix D.1.2).

B.4 FORT SANTIAGO: ANSWERING FROM THE RETAINED RESEARCH

When research ends before resolving the target, the harness can make one final LLM call to answer the original question from the retained record without a new search. We refer to this fallback as the *Finalizer*. BCP q500 asks for a sixteenth-century structure's English name, using its capital-city location, destruction, rebuilding, and later damage as clues. Retained articles include *Fort Santiago – Wikipedia*, *Welcome to Fort Santiago!*, and *About us*. The last page's retrieved text begins "Welcome to Fort Santiago!" and describes Legazpi re-establishing Rajah Soliman's Manila in 1571: research has reached sources naming the structure and describing its historical setting.

The difficulty is admitting the target. Across assignments, the Researcher's typed-resolver stage proposes `1580` on 5 attempts; the Verifier rejects a year in place of a name. ReAct also proposes *Fort Santiago* with citations, but 5 recorded updates stop at the deterministic checks: links to construction year, location, destruction, attacker, and rebuilding lack attached passages. Citing the name does not complete these bindings or establish the timeline, including the recorded construction year 1580 and derived destruction year 1583.

After 5 consecutive rounds without grounded progress, the loop stops with no answer. This triggers recovery: if read articles exist, the Finalizer receives the question, resolved values, leading unresolved candidates, and the 4 latest distinct articles, newest first. Limits are 45,000 characters per article, 130,000 total, and 4,000 generated tokens, with no search or read tools. If the answer remains empty, the *stored-candidate fallback* returns an eligible target candidate with evidence without another model call. Neither step changes workspace bindings or verification judgments.

Here the Finalizer returns *Fort Santiago*; all 3 grading judgments accept it, and it passes Obs. The correct answer is recoverable from the retained sources even though the target's relation-specific bindings remain incomplete. Recovery makes that research useful without reversing the earlier checks or marking the target verified (Appendix D.1.2).

C Evidence Quality and Validation

Evidence bindings should leave a research record that supports checking and reusing an answer. We first define how Obs. measures recoverable answer credit, then test the matching procedure, contextual claim support, answering from the collected evidence, and recorded calculations.

What counts as evidence. The scorer uses saved search, read, and find outputs, source-validated Reader quotations, and recorded inputs to the typed-resolver stage and post-loop grounding check (Appendix B.2). MoNaCo also includes recorded Reader windows. IterResearch contributes page-summarizer inputs with HTML removed; Open Deep Search contributes model-written search summaries. The agent's own investigative prose and gold-document annotations are excluded. The headline score also allows source text reconstructed from recorded identifiers, including PANINI-DR's references and Re-TRAC's self-citations. Thus Obs. measures recoverability from the assembled record, not confirmed receipt of every passage; Appendix C.1 tests the effect of removing identifier-only evidence.

Combining correctness and evidence. On BCP and FRAMES, we test the judge-extracted answer, splitting lists, dictionaries, and numeric/date compounds into checkable parts. Matching uses fixed text, number, date, and name variants with token boundaries; an all-but-one-token match requires at least 4 content tokens. Bare numbers and short words require no additional proximity to question words, and Boolean parts do not block support on these benchmarks. For official score o_i, nonnegative credit $c_i = \max(0, o_i)$, answer parts $\mathcal{P}_i$, and evidence $\mathcal{E}_i$, Equation 1 uses $s_i = \prod_{p \in \mathcal{P}_i} M(p, \mathcal{E}_i)$, where M is the fixed matching rule. A parsed answer with no items receives zero support.

MoNaCo item-level credit. MoNaCo preserves partial answer credit. We select the judge pass with highest official F1, breaking ties by Obs. and then pass order; its credited items, prediction count, and support labels are used together. Let k_i^{obs} count credited items that pass the support rule, g_i count gold items, and n_i be the prediction count. We use the positive judge count or, otherwise, the parsed-answer count, never below the credited count. For multi-item questions,

$$P_i^{\text{obs}} = \frac{k_i^{\text{obs}}}{\max(1, n_i, k_i^{\text{obs}})}, \qquad R_i^{\text{obs}} = \frac{\min(k_i^{\text{obs}}, g_i)}{g_i}.$$

Let $H(P, R) = 2PR/(P + R)$, or 0 when $P + R = 0$. For a single-item question, s_i tests the judge-extracted answer. Then

$$\widetilde{o}_i = \begin{cases} c_i s_i, & g_i = 1, \\ \min\{c_i, H(P_i^{\text{obs}}, R_i^{\text{obs}})\}, & g_i > 1,\ n_i > 0, \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

$$\text{Obs}_{\text{MoNaCo}} = \frac{1}{N} \sum_{i=1}^{N} \widetilde{o}_i.$$

This caps recoverable credit at nonnegative official credit. MoNaCo's numeric official scores can be negative; only Obs. floors them at zero. Empty and unjudged answers receive zero support, with all 1,315 questions retained.

Matching MoNaCo claims. We preserve the credited claim's numbers before normalizing its text. Deterministic matching may use corresponding gold-item text to resolve alternative wording, but cannot substitute a different number or require unrelated gold attributes. A bare value is paired with its entity when recoverable. Multi-part claims must match in one document, within 600 characters of a common anchor or with a text part identifying the document through its title or first 400 normalized characters. Boolean-only claims do not pass this base test.

C.1 Checking the Scorer and Evidence Record

We test whether matching errors or reconstructed source text account for the observed-support advantage. In a balanced development sample, separate human and model reviews agreed on all 96 matching decisions: 8 passing and 8 failing items from each combination of 3 systems (PANINI-DR, SLEUTH, Open Deep Search) and 2 benchmarks (BCP, MoNaCo). Review used the matching

source window, or the nearest window for a failed match. MoNaCo consistency checks also reconcile credited and supported items with the counts from the selected judge pass.

The fixed parser can discard credited answers: FRAMES has 6 such records across systems—4 non-Latin answers, 1 Morse-code answer, and 1 null answer. The normalization issue affects 2 PANINI-DR questions (0.24 points); reported scores retain the fixed rules.

Using only recorded evidence. Removing document bodies reconstructed solely from identifiers leaves all 7 assessable systems' BCP Obs. scores unchanged. The retained pool contains saved tool outputs, source-validated Reader spans, and validated passages shown to the typed-resolver stage and post-loop grounding check. PANINI-DR remains at 70.7% versus SLEUTH's 54.0%, a paired gain of 16.7 points (95% interval $[+8.7, +24.7]$). The advantage therefore persists without identifier-only evidence. Re-TRAC cannot be assessed because it saves citations but no tool outputs.

Numerical-matching sensitivity. As a secondary check, Obs.$^{+}$ allows unmatched numerical/Boolean parts while still requiring text matches. On MoNaCo, it additionally credits items mapped to numerical/Boolean gold values and recomputes F1, keeping the selected pass and prediction count fixed. Denominators and official judgments are unchanged. PANINI-DR remains highest on all 3 benchmarks among available results; HyMem's MoNaCo Obs.$^{+}$ is unavailable. Appendix C.4 complements this matching analysis by checking recorded inputs and replaying calculations.

C.2 Model-Based Grounded-Only Audit

To check whether sources support the requested claims, not merely contain the answer text, we rebuild answers using only claims accepted by a common model auditor. We compare PANINI-DR at cap 5, SLEUTH, and HyMem on all 150 BCP, 824 FRAMES, and 1,315 MoNaCo questions, using each system's own saved evidence.

`gpt-5.6-luna` first splits answers into claims without judging support. `gpt-5.6-terra` then checks them against the question and evidence, using the same prompt and 80,000-character windows with 8,000 characters of overlap. The auditor sees neither gold answers nor official verdicts. It accepts paraphrases but requires the requested attribute to be supported in at least one window. We reconstruct answers from accepted claims and re-grade them with `gpt-5.6-luna` under the shared official protocol (Appendix A.2); empty reconstructions score zero. Reconstructed answers containing all split claims provide a separate formatting control in the analysis artifacts. Grounded-only reports the official benchmark score of the reconstructed answer.

Method	BCP-150 Accuracy (%)	FRAMES-824 Accuracy (%)	MoNaCo-1315 Mean-F1
HyMem	48.0	43.6	0.352
SLEUTH	55.3	67.1	0.417
PANINI-DR	**72.0**	**73.2**	**0.443**

Table 5: Answer quality after a common model auditor filters unsupported claims. Rebuilt answers are re-graded under each benchmark's official protocol. Every question is included, with empty answers scoring zero. Bold marks the best score.

HyMem can answer without retrieval, contributing to official scores but not adding retrieved evidence for this audit (154 FRAMES, 139 MoNaCo, and 4 BCP questions). All questions nevertheless remain in the comparison. PANINI-DR leads both baselines on all 3 benchmarks (Table 5): its advantage therefore extends from recoverable answer text to answers rebuilt from claims judged supported in context.

C.3 Answering from Each System's Evidence

To test whether retained research remains useful beyond the harness's own answer generation, we give evidence from 4 systems to a common `gpt-oss-120b` evaluation reader, with no new research or retrieval. It generates new answers; the internal Reader extracts source evidence, while Grounded-only filters original answers.

For each evidence set, the *answer-present* prompt tells the reader that the documents contain the answer and to use only those documents. The *uncertain-evidence* prompt instead warns that evidence may be incomplete or irrelevant and allows abstention. Inputs and reader settings are unchanged between conditions (temperature 0, seed 42). Each cell is one run over all 150 BCP, 824 FRAMES, or 1,315 MoNaCo questions, scored with `gpt-5.6-luna` under the shared 3-judgment protocol; empty answers score zero. Each system supplies its collected evidence without matching evidence volume or reading cost.

	BCP-150 (%)		FRAMES-824 (%)		MoNaCo-1315 (F1)	
Evidence source	Present	Uncertain	Present	Uncertain	Present	Uncertain
PANINI-DR	**72.0**	**62.7**	**76.7**	**69.4**	**0.601**	**0.547**
SLEUTH	59.3	53.3	70.0	59.3	0.506	0.476
HyMem	54.7	50.0	52.2	43.9	0.470	0.419
Omni-Decision	58.7	55.3	71.2	62.5	0.538	0.515

Table 6: Answer quality of a common evaluation reader using each system's saved evidence. Present assumes the answer is available; Uncertain allows abstention. All questions are included; empty answers score zero. Bold marks the best score.

PANINI-DR's evidence yields the highest answer quality on all 3 benchmarks under both prompts (Table 6). With abstention allowed, the lead over the strongest alternative is 62.7% versus 55.3% on BCP, 69.4% versus 62.5% on FRAMES, and 0.547 versus 0.515 on MoNaCo. The MoNaCo ordering against HyMem reverses: HyMem leads original-answer F1 (0.612 versus 0.549), but its evidence yields 0.419 versus our 0.547 here. The retained material therefore remains useful beyond our own readout, even when the reader is not promised that an answer is present.

C.4 Checking Computed Answers

A computed answer need not appear in a source passage. We therefore test whether the saved inputs and operations reproduce the answer, using Obs-D to supplement text matching. This is a PANINI-DR-only audit: baseline traces do not expose equivalent calculation chains.

For each target with positive official credit, we follow its recorded calculation to given literals and retrieved inputs. Each retrieved input must be nonempty and recoverable from received text in a document it cites; empty collections do not establish support. Recorded quotations and inputs to the typed-resolver stage and post-loop grounding check are source-validated; citation identifiers alone cannot supply text. Matching pools recorded text by cited document, rather than requiring the binding's exact chunk. We then replay the recorded operations and compare the result with the graded answer under fixed text, date, Boolean, and numeric matching rules, without introducing an unrecorded conversion. A warned input excludes the calculation; missing Verifier records are recorded as unknown, not accepted.

The Camp David example in Figure 1 illustrates the difference. The answer, 51, is absent from the 3 opened pages, so Obs. gives no credit. Obs-D checks the recorded birth and signing dates against received evidence and replays the completed-years operation to recover 51; both date inputs have accepted Verifier records.

For a passing calculation, let a_i be the nonnegative official credit not already retained by Obs. Then

$$\text{Obs-D} = \text{Obs} + \frac{1}{N}\sum_{i=1}^{N} a_i. \tag{3}$$

FRAMES additions are binary; MoNaCo retains partial credit for the graded derived item. Each passing target in these exports is its answer's sole graded item. Fully covered answers receive no addition, and all questions remain in the denominator. This automated audit checks whether recorded calculations reproduce the graded answer from source-matched inputs.

Replay adds credit on 51 FRAMES and 55 MoNaCo questions, raising scores from 66.1% to 72.3% and from 0.458 to 0.498, respectively (Table 7). Every passing calculation has accepted Verifier records for all retrieved inputs, so restricting additions to that group leaves the gains unchanged.

	FRAMES-824	MoNaCo-1315
Arithmetic/Boolean target records	192	335
Passing calculations	118	184
Answers receiving additional credit	51	55
Obs.	66.1	0.458
Obs-D	**72.3**	**0.498**

Table 7: PANINI-DR-only calculation audit. FRAMES scores are percentages; MoNaCo scores are mean-F1. All questions remain in the denominator. Only official credit not already retained by Obs. is added.

These answers extend the inspectable record from facts found in text to results reproduced from those facts.

D Additional Experimental Results

D.1 Research Evidence and Answer Completion

D.1.1 Finding Sources and Connecting Findings

Persistent findings should help research reach relevant sources and retain them for later use. We compare PANINI-DR and SLEUTH on matched questions, measuring the fraction of annotated pages found and the coverage and precision of pages recorded as support (Table 8). PANINI-DR leads all three measures on every benchmark: broader discovery is accompanied by a higher annotated share of retained support. On BCP, its record contains every annotated page on 121 questions versus 46 for SLEUTH. Page presence includes snippets and recorded typed-resolver and post-loop grounding inputs, not just full-page reads.

Benchmark	n	Found recall	Cited recall	Cited precision (%)
BCP	150	.860 / .455	.485 / .248	43.3 / 37.0
FRAMES	823	.945 / .820	.774 / .586	91.9 / 84.4
MoNaCo	1,315	.515 / .368	.281 / .250	73.3 / 67.0

Table 8: Source coverage: PANINI-DR / SLEUTH. Recall is the fraction of annotated pages found or recorded as support; precision is the annotated share of support pages. Values average per-question fractions, with zero precision when no citation is recorded. Source measures omit one FRAMES question without annotations; answer scores retain all 824. SLEUTH source titles are normalized and matched to recorded pages.

A matched trajectory shows how a finding can direct the next search. BCP q276 asks which expedition an entrepreneur joined. PANINI-DR identifies him as Jim Bridger, then uses his name to find a biography connecting him to the Ashley–Henry expedition. It answers correctly with observed support. SLEUTH never establishes the identity and returns no answer. Here, persistent research state connects successive searches: an intermediate answer supplies the next search subject.

D.1.2 Revising the Plan and Recovering an Answer

Research can reveal a missing step in the initial plan. In the default BCP and FRAMES executions, accepted revisions add 59 and 162 subquestions; 30 and 82 of these hold a value at the end. These counts describe new research tasks and their completion, not a separate revision-on/off accuracy effect. Appendix B.3 follows an added intermediate through to an answer and explains the rules for reopening earlier findings.

When the target remains unfinished, answer recovery uses Finalizer synthesis or stored-candidate fallback (Table 9). On BCP, these paths supply 40 of the 110 correct answers, with 38 passing Obs. Recovery therefore preserves useful research that would otherwise end without an answer; the Fort Santiago trace in Appendix B.4 shows this without changing the workspace or reversing

earlier verification judgments. Final answer quality and workspace completion remain distinct. At the default cap, warned findings comprise 2/666 (0.3%), 31/2,638 (1.2%), and 235/3,619 (6.5%) resolved nodes on BCP, FRAMES, and MoNaCo.

Benchmark	Recovery answers	Official	Obs.
BCP	60	66.7	63.3
FRAMES	103	60.2	52.4
MoNaCo	250	0.430	0.358

Table 9: Scores over all identified recovery answers, including incorrect ones: percentages on BCP/FRAMES and mean F1 on MoNaCo. Finalizer outputs number 50/89/241 and stored-candidate fallback outputs 10/14/9, respectively. One additional MoNaCo output has an unclassified origin.

D.1.3 Answering with Annotated Sources

To test whether the same backbone can answer difficult questions once relevant sources are supplied, we give `gpt-oss-120b` the annotated documents without a search, planning, or verification loop. The reference reaches 93.3% BCP accuracy, including 45/50 hard questions versus PANINI-DR's 26/50 (Table 10). This locates substantial room for improvement in finding and using evidence, rather than answering ability alone.

	BCP				FRAMES	MoNaCo
	Easy	Med.	Hard	All	Acc.	F1
PANINI-DR	92	76	52	73.3	79.1	0.549
Annotated-source reference	96	94	90	93.3	79.5	0.550

Table 10: Official scores with research or supplied sources: accuracy (%) on BCP/FRAMES and mean F1 on MoNaCo. Denominators are 150/824/1,315; each BCP difficulty group contains 50 questions.

Source packaging follows document volume. BCP includes up to 140,000 characters directly on 137 questions; the remaining 13 use an index and up to nine 12,000-character windows. All 462 document entries resolve. FRAMES includes up to 10 articles, each capped at 16,000 characters; 743 questions have clipped articles, retaining 26.7% of selected text. MoNaCo reads the first 60,000 characters of each resolvable page into 1,200-character notes, compressing notes above 40,000 characters before answering. It resolves 40,606/56,734 entries and reads 71.5% of their text. Each cell is one execution, with all questions retained and the same three-judgment aggregation as the headline scores. These different input procedures make the reference a diagnostic, not a cost-matched comparison or universal accuracy ceiling.

The reference also answers 14 of 17 BCP questions where PANINI-DR found all annotated documents in returned tool records but answered incorrectly. In q192, for example, research reaches the sources but connects the target to the wrong artist's death. Finding the pages is therefore only the first step: their facts must remain connected to the right question through answer submission.

D.2 Planning, Verification, and Recovery Controls

D.2.1 Structured Planning and Persistent State

To test whether explicit research structure improves answers beyond prose memory, we compare two reduced harnesses on all 150 BCP questions. Off retains prose notes and Researcher reports; On adds a plan, findings attached to its subquestions, and revision. Both retain ReAct Researchers, direct source reading, and LLM verification with five non-endorsement retries. Both omit the typed resolver, dedicated Reader, executable calculations, relation-level deterministic checks, and auxiliary full-system checks. On retains plan construction/repair and up to five updates. The backbone, seed 42, temperature 0, and configured limits are shared: 12 Orchestrator turns, 20 turns per Researcher, up to eight Researchers per dispatch, and 32,000/16,384 prompt/completion tokens. The intervention is this planning-and-state package, not a single component.

	Recovery off		Recovery on		Prompt (M)
Arm	Official	Obs.	Official	Obs.	All roles, mean
OFF: prose memory	41.3	39.3	56.7	53.3	1.38
ON: planning/state	58.0	56.0	73.3	68.7	1.78
Full system	46.7	45.3	73.3	70.7	1.43

Table 11: Planning/state and common-recovery controls on all 150 BCP questions; scores are percentages. Prompt means cover the recorded reference runs: they exclude post-hoc Finalizer calls on the controlled arms and include live recovery in the full system. Full-system rows restore multiple components and are not isolated planning/state controls.

ON raises accuracy by 16.7 points: 37 answers become correct and 12 become incorrect (95% paired interval $[+8.0, +25.3]$). Hard-question successes double from 10 to 20. To check whether answer recovery explains this advantage, we apply the same tool-free Finalizer policy to each arm's empty-answer states. It receives the question, up to four recently read articles, and workspace findings/candidates on ON or Researcher reports on OFF. No ON output qualifies for stored-candidate fallback. The full system's recovery-off cell withholds its 60 recovery outputs from the recorded run, without rerunning research. Common recovery adds 23 correct answers to each controlled arm, preserving the 16.7-point advantage ($[+9.3, +24.7]$). Thus, the gain persists when both versions can retain notes, verify findings, and recover answers from their research.

With recovery, ON matches the full system's 73.3% BCP accuracy, but the full system records fewer prompt tokens (1.43 M versus 1.78 M) and higher Obs. (70.7 versus 68.7). Its Reader supplies focused source extracts instead of placing full documents in the Researcher's context. The typed resolver and computation together supply about half of the full system's correct FRAMES answers and MoNaCo answers receiving credit. The full package therefore offers benefits beyond matching BCP accuracy; this comparison does not isolate the contributions of individual components.

We next check whether retained findings become inputs to later searches. Among recorded entities absent from the question and followed by another assignment, later searches reuse 192/248 (77.4%) on ON versus 176/548 (32.1%) on OFF. In q59, for example, ON records Maitreyi Raman and then searches for her award; OFF encounters the name but never searches with it. These entity matches show earlier findings more often supplying the subject of subsequent research. A stratified contextual check of 40 excerpts per arm labels 37 versus 28 genuine reuse; the original ON annotations and two OFF annotations were manual, and an author reviewed the other 38 assistant-drafted OFF labels. This check validates sampled matches, not the causal contribution of reuse to accuracy.

D.2.2 VERIFICATION BEFORE REUSE

An incorrect intermediate finding can misdirect later research. We therefore test the verification policy while retaining the workspace and research tools. Off removes LLM review and the deterministic checks for valid citations and relation-specific source attachments. Assignment and type checks and evidence logging remain. The on condition uses the default cap of five. Both conditions use the same 150 BCP questions and matched stratified samples of 200 FRAMES and 200 MoNaCo questions; the MoNaCo on arm is the matching subset of the full cap-5 run. Backbone, retrieval, seed, temperature, and scoring are shared.

Benchmark	Verification	Official	Obs.	Prompt (k)
BCP (150)	On	73.3	70.7	1,042
	Off	57.3	54.7	453
FRAMES (200)	On	79.5	69.0	125
	Off	69.5	57.0	35
MoNaCo (200)	On	0.583	0.479	288
	Off	0.517	0.412	47

Table 12: Verification ablation on matched questions. Off removes LLM review and source-evidence checks but retains assignment and type checks. Scores are percentages on BCP/FRAMES and mean F1 on MoNaCo. Prompt use is the per-question median in thousands, using trajectory accounting.

Full verification improves official and observed-support scores on all three matched sets, adding 16.0 points to both BCP measures. The gain includes research prompted by feedback, with median BCP prompt use rising from 0.453 M to 1.042 M. The next analysis follows what Researchers change before a previously rejected finding is accepted.

D.2.3 Responding to Verification Feedback

The review cap controls failed-check re-attempts for an assigned plan node within each dispatch, separately from the Researcher's turn allowance. With the other full-system limits fixed, caps 1, 2, 5, and 10 on the same 150 BCP questions favor bounded review (Table 13). Cap 5 reaches 73.3% accuracy with only two warned resolved findings; cap 10 removes warnings but falls to 66.0%. Cap 2 is close to cap 5, whose paired accuracy advantage is 1.3 points with interval $[-5.3, +8.0]$. More opportunities for verification therefore do not consistently produce better answers.

Cap	Acc.	Obs.	Hard / 50	Warned / resolved	Prompt (M)
1	67.3	65.3	19	128 / 773	1.249 (0.817)
2	72.0	68.7	23	27 / 636	1.147 (0.854)
5	**73.3**	**70.7**	**26**	2 / 666	1.200 (1.042)
10	66.0	63.3	20	0 / 673	1.317 (1.115)

Table 13: Shared review-cap sensitivity on BCP CF-150. Accuracy and Obs. are percentages; Hard counts correct answers out of 50. Warning counts use resolved bindings in the final workspace snapshots, including computed bindings. Prompt use is the per-question mean in millions, with the median in parentheses, using the trajectory accounting in Table 14. These are benchmark averages for the supplied executions, not averages across research repetitions.

At cap 5, 77 of 101 Verifier-endorsed proposals following rejection change the value or citation set relative to earlier rejected proposals: 40 introduce a new value and 37 pair an earlier value with a new citation set. These are proposal events, excluding warned admissions and evidence-attachment calls. The Brazuca editor sequence (Appendix B.2) shows the mechanism: feedback leads to a full name and evidence connecting that person to both films. Verification can thus guide further evidence gathering before reuse.

D.3 Computation and Research Allowances

D.3.1 Recorded Tokens and Cache Reuse

Method	**Prompt (M)**	**Completion (k)**
PyRAG	0.002 (0.017)	3.2 (6.5)
SLEUTH	0.037 (0.074)	2.3 (7.3)
Open Deep Search	0.10 (0.11)	10.0 (10.9)
Omni-Decision	0.13 (0.16)	9.0 (10.1)
IterResearch	0.17 (0.17)	9.8 (10.6)
HyMem	0.29 (0.31)	51.0 (65.9)
Re-TRAC	5.12 (5.34)	33.6 (42.7)
PANINI-DR	1.04 (1.20)	36.1 (44.2)

Table 14: BCP CF-150 token use: per-question medians (means), with prompt tokens in millions and completion in thousands. All rows use `gpt-oss-120b`. Counting details appear in Appendix D.3.1.

To explain the token expenditure in Table 14, we break down prompt use by role (Table 15). The Researcher's ReAct stage and Reader account for 85–89% across benchmarks, versus 4–6% for the Orchestrator and 2–3% for the LLM Verifier.

On BCP, a question averages 109.45 ReAct model calls at 10.40k prompt tokens each and 13.56 Reader requests at 9.92k prompt tokens per request. A Reader request can include multiple model calls or a cache hit. Deterministic checks require source attachments; the Verifier checks the assigned

	BCP	FRAMES	MoNaCo
Questions	150	824	1,315
Reported trajectory prompt total (M)	180.04	270.38	775.12
Sum of 10 recorded roles (M prompt tokens)	213.94	340.79	1,040.13
Role sum / trajectory total	1.188	1.260	1.342
Selected roles: mean prompt tokens per question, k (share of role sum)			
Orchestrator	58.4 (4.1%)	23.1 (5.6%)	32.8 (4.1%)
Researcher (ReAct)	1,138.0 (79.8%)	303.7 (73.4%)	550.8 (69.6%)
Reader	134.6 (9.4%)	48.8 (11.8%)	156.2 (19.7%)
LLM Verifier	40.5 (2.8%)	7.5 (1.8%)	13.7 (1.7%)
ReAct + Reader share (%)	89.2	85.2	89.4

Table 15: PANINI-DR cap-5 role accounting from saved traces. Role telemetry covers every question; shares use summed role prompt tokens across all 10 roles, including the 6 not listed individually. ReAct denotes the Researcher's second stage, excluding the typed resolver. k and M denote thousands and millions of tokens. Combined shares use unrounded counts. Table 14 instead reports trajectory totals.

claim and conditions. In the Brazuca trace, these checks trigger further reading before admitting an unchanged title (Appendix B.2).

Table 14 uses recorded usage totals for IterResearch and Re-TRAC, and counting proxies otherwise. Role sums aggregate calls differently and do not partition trajectory totals.

To account for prefix-cache reuse, we measure engine counters in separate runs of each harness over all 150 BCP questions: `vLLM`, two RTX PRO 6000 Blackwell GPUs, tensor parallelism 2, concurrency 8, seed 42, temperature 0, and one harness at a time after requests drain (Table 16).

Harness	Input (k)	Cache (%)	Uncached (k)	Generated (k)	PFLOP
PANINI-DR	1,422	66.5	477	78	5.68
SLEUTH	74	7.3	69	7	0.78
Omni-Decision	157	7.1	146	10	1.59
HyMem	309	45.0	170	66	2.41

Table 16: Engine measurements over all 150 BCP questions. Tokens and analytically estimated model-core PFLOPs are per-question means; cache percentage is the share of submitted input reused.

PANINI-DR submits 19.2 times SLEUTH's prompt tokens, but reuses 66.5% through the prefix cache. Uncached input plus generation gives a 7.3-fold model-core computation ratio under $\widehat{F} = 2N_{\text{active}}(T_{\text{uncached input}} + T_{\text{generated}})$, with $N_{\text{active}} = 5.12 \times 10^9$. It excludes attention and runtime overhead and is not GPU time or energy. The following control compares repeated SLEUTH runs under this accounting.

D.3.2 Repeated Sampling at Comparable Computation

We give SLEUTH independent attempts to test whether repeated sampling can match the complete harness. Each research run uses the headline configuration at temperature 0.7 (our change from its greedy default), with seeds 42–49. We select the most frequent non-empty normalized answer, breaking ties by the lowest seed; both official and Obs. scores use the lowest-seed run producing that answer and its evidence. If every answer is empty, the question scores zero. Sampling runs use concurrency 16; engine counters measure their actual cache reuse and cumulative computation.

At eight research runs per question, SLEUTH uses 1.12 times our model-core computation but reaches 64.7% accuracy versus 73.3%; the paired advantage is 8.7 points ($[+1.3, +16.0]$) for both official and Obs. scores. The rule fixed in advance continues past eight runs only if voted accuracy is still increasing; it peaks at seven. Even selecting any correct run would answer only 70.0% of questions, although that is an oracle choice, not the voting procedure. Under this tested sampling policy, additional baseline attempts do not reproduce the harness's answer quality.

Runs	PFLOP/q	vs. ours	Vote acc.	Any correct	Vote Obs.
1	0.76	0.13	61.3	61.3	58.7
2	1.66	0.29	62.0	66.0	59.3
4	3.22	0.57	63.3	69.3	60.7
6	4.79	0.84	64.7	69.3	61.3
7	5.51	0.97	65.3	70.0	62.7
8	6.34	1.12	64.7	70.0	62.0
PANINI-DR: 5.68 PFLOP/q			73.3	—	70.7

Table 17: Repeated SLEUTH research runs on all 150 BCP questions at cumulative model-core computation. Vote selects the most frequent non-empty answer; "Any correct" credits a question if at least one run answers correctly. Scores are percentages. Individual runs average 55.5% accuracy versus the greedy reference's 57.3%.

D.3.3 Extending Individual Investigations

Instead of restarting research, longer assignments can build on findings already collected. We compare three allowance policies on all 150 BCP questions (Table 18). Increasing each PANINI-DR Researcher's ReAct allowance from 20 to 50 and 80 turns, with review cap 5 and the 12-turn Orchestrator limit fixed, raises accuracy incrementally from 73.3% to 74.7% and 76.0%. The 50-turn run gains 12 correct answers and loses 10; the 80-turn run gains 14 and loses 10. The largest allowance uses 2.8 times the prompt tokens for 2.7 additional points, favoring the lower-cost default for these executions.

Allowance	Accuracy	Obs.	Mean prompt (M)	Empty
PANINI-DR: turns per Researcher, review cap 5				
20 (default)	73.3	70.7	1.20	4
50	74.7	71.3	2.54	2
80	76.0	74.0	3.33	6
PANINI-DR: shared question budget, review cap 5				
20	31.3	28.7	0.115	71
50	49.3	46.7	0.279	38
80	57.3	54.0	0.440	34
SLEUTH: its own turn allowance				
20	57.3	54.0	0.074	26
50	59.3	56.0	0.106	22
80	57.3	54.0	0.111	28

Table 18: Separate allowance sweeps on all 150 BCP questions, including unanswered cases. Scores are percentages; prompt means use trajectory accounting. Turn limits govern different activity across panels and are not matched computation. Each row is a separate execution, not a continuation of the shorter run.

The traces show additional turns completing support for existing candidates. In q1204's 80-turn run, Stephan Weyte is proposed at turn 20; citation repairs and further verification lead to admission at turn 26 without changing the candidate. Across the separate 50- and 80-turn executions, 24 and 26 questions move from a correct recovery answer at the default to a correct answer with a resolved workspace target. This shift is larger than the net accuracy gain: longer research can complete evidence that recovery already used to answer.

The shared-budget panel instead limits combined Orchestrator/Researcher turns. Remaining turns are split across dispatched Researchers with a minimum of one each, so a final dispatch can exceed the nominal cap. The default has separate limits, not a shared question budget; its higher accuracy accompanies higher expenditure. SLEUTH's own allowance sweep shows no sustained gain: 57.3%, 59.3%, and 57.3%, despite mean prompt use rising from 74k to 111k. Input-length refusals affect 19, 18, and 26 of 150 questions, respectively, while total unanswered counts are 26, 22, and 28. From 20 to 80 turns, budget exhaustion falls from 7 to 1 question, so answer coverage decreases by only 2 questions despite 7 additional input-length refusals. All unanswered questions remain in the

scores. These turn policies cover different activity; only the preceding sampling control compares model-core computation.

D.4 Run Stability, Paired Comparisons, and Answer Structure

Stability across research runs. To check stability, we evaluate three research runs per harness under the headline BCP settings, including the headline run (Table 19). PANINI-DR averages 72.7% accuracy, versus 58.0% for both SLEUTH and Open Deep Search—a 14.7-point lead. Its lowest-scoring run exceeds every baseline run in this comparison.

Method	Table 1	Run 2	Run 3	Mean $\pm$ SD	Mean (%)
PANINI-DR	**110**	**107**	**110**	$\mathbf{109.0 \pm 1.7}$	**72.7**
SLEUTH	86	91	84	87.0 ± 3.6	58.0
Open Deep Search	80	88	93	87.0 ± 6.6	58.0
IterResearch	82	83	80	81.7 ± 1.5	54.4
HyMem	73	73	74	73.3 ± 0.6	48.9
Omni-Decision	71	68	59	66.0 ± 6.2	44.0
Re-TRAC	59	52	52	54.3 ± 4.0	36.2
PyRAG	41	48	43	44.0 ± 3.6	29.3

Table 19: BCP stability across three research runs per method, each on the same 150 questions under the headline settings. Run columns and mean $\pm$ sample SD count correct answers; the final column converts the mean count to accuracy. The first run is the one reported in Table 1.

Paired question-level comparisons. To check whether reported gains depend on the sampled questions, Table 21 gives 95% intervals from 10,000 paired bootstrap resamples of unrounded question-level differences. All questions keep their benchmark weight, including failures with zero credit. These intervals describe question-sampling uncertainty within the recorded executions, not variation across repeated research runs. The listed Obs. advantages over baselines remain positive, while several official-score differences include zero or favor another harness.

On FRAMES, the accuracy advantage over SLEUTH has a paired 95% interval that includes zero ($[-1.0, +4.9]$; Table 21). On MoNaCo, PANINI-DR trails SLEUTH and HyMem in overall F1; on collection questions, its highest precision (0.690) does not compensate for lower recall (0.462 versus HyMem's 0.566; Table 20).

Because SLEUTH's input-length refusals could partly explain the BCP gap, we exclude the same 19 questions from both systems. On the remaining 131, accuracy is 74.8% versus 65.6%, a gain of 9.2 points ($[+0.8, +17.6]$); Obs. is 71.8% versus 61.8%, a gain of 9.9 points ($[+1.5, +18.3]$). The advantage therefore persists on this common subset. This removes recorded refusals, rather than predicting performance with a different context window.

MoNaCo precision, recall, and answer structure. To explain the MoNaCo accuracy trade-off, we separate finding one answer from enumerating a collection. Table 20 groups the 720 single-item and 595 collection questions by the official judge's gold-item counts. Precision and recall use the same selected maximum-F1 pass as the headline score. Each class retains all its questions, with missing predictions scoring zero.

On collections, PANINI-DR has the highest precision (0.690), but its 0.462 recall leaves F1 at 0.495: finding fewer wrong items does not compensate for missing valid ones. Against SLEUTH, it scores higher on single-item F1 (0.593 versus 0.579) but lower on collection F1 (0.495 versus 0.528). PANINI-DR leads overall and on single-item observed support; its collection Obs. is 0.474.

D.5 Baseline Adaptations

Table 22 separates paper-based reimplementations from released-code adaptations. SLEUTH, Omni-Decision, and HyMem had no available code when experiments were established; we follow their published specifications rather than reproducing released implementations.

Omni-Decision leaves the zero-novelty termination limit unspecified. We use three, the least restrictive setting tested: limits one, two, and three lose six, two, and zero answers on FRAMES-50.

	Single item (720)		Collection (595)			
Harness	F1	Obs.	P	R	F1	Obs.
PANINI-DR	0.593	**0.446**	**0.690**	0.462	0.495	0.474
SLEUTH	0.579	0.338	0.678	0.503	0.528	0.467
Open Deep Search	0.598	0.269	0.652	0.519	0.538	0.394
IterResearch	0.583	0.317	0.617	0.546	0.540	**0.479**
Omni-Decision	0.410	0.282	0.399	0.281	0.298	0.267
Re-TRAC	0.572	0.360	0.591	0.507	0.511	0.441
PyRAG	0.474	0.297	0.581	0.380	0.413	0.375
HyMem	**0.651**	0.360	0.654	**0.566**	**0.566**	0.423

Table 20: MoNaCo answer structure for the main-table executions. Columns average over all questions in each group using the same maximum-F1 pass. Mean F1 is not the harmonic mean of aggregate precision and recall. Bold marks the largest value.

			Official		Obs.	
Dataset	Comparator	n	Δ	95% CI	Δ	95% CI
Headline comparisons: PANINI-DR *(cap 5) minus baseline*						
BCP	SLEUTH	150	$+16.0$	$[+8.0, +24.0]$	$+16.7$	$[+8.7, +24.7]$
FRAMES	SLEUTH	824	$+1.9$	$[-1.0, +4.9]$	$+10.9$	$[+7.8, +14.1]$
MoNaCo	SLEUTH	1,315	-0.007	$[-0.029, +0.015]$	$+0.062$	$[+0.041, +0.083]$
BCP	HyMem	150	$+24.7$	$[+16.0, +34.0]$	$+24.0$	$[+14.7, +32.7]$
FRAMES	HyMem	824	$+8.4$	$[+5.0, +11.7]$	$+29.5$	$[+25.7, +33.1]$
MoNaCo	HyMem	1,315	-0.064	$[-0.086, -0.042]$	$+0.070$	$[+0.046, +0.095]$
MoNaCo	Re-TRAC	1,315	$+0.004$	$[-0.021, +0.029]$	$+0.062$	$[+0.038, +0.086]$
MoNaCo	Open Deep Search	1,315	-0.022	$[-0.044, -0.001]$	$+0.133$	$[+0.110, +0.156]$
MoNaCo	IterResearch	1,315	-0.015	$[-0.040, +0.009]$	$+0.068$	$[+0.043, +0.094]$
MoNaCo	PyRAG	1,315	$+0.102$	$[+0.079, +0.127]$	$+0.126$	$[+0.105, +0.148]$
BCP	SLEUTH, non-refused	131	$+9.2$	$[+0.8, +17.6]$	$+9.9$	$[+1.5, +18.3]$
BCP cap sweep: cap 5 minus comparator						
BCP	Cap 1	150	$+6.0$	$[-1.3, +13.3]$	$+5.3$	$[-2.0, +12.7]$
BCP	Cap 2	150	$+1.3$	$[-5.3, +8.0]$	$+2.0$	$[-4.7, +9.3]$
BCP	Cap 10	150	$+7.3$	$[+0.7, +14.0]$	$+7.3$	$[+0.7, +14.0]$
BCP shared budget at cap 5: hierarchical reference minus shared budget						
BCP	Shared 80	150	$+16.0$	$[+8.0, +24.0]$	$+16.7$	$[+8.7, +24.7]$
BCP	Shared 50	150	$+24.0$	$[+15.3, +32.7]$	$+24.0$	$[+15.3, +32.7]$
BCP	Shared 20	150	$+42.0$	$[+33.3, +50.7]$	$+42.0$	$[+33.3, +50.7]$
Verification ablation: verification on (cap 5) minus verification off						
BCP	Verification off	150	$+16.0$	$[+8.0, +24.0]$	$+16.0$	$[+8.0, +24.0]$
FRAMES	Verification off	200	$+10.0$	$[+3.5, +16.5]$	$+12.0$	$[+5.5, +18.5]$
MoNaCo	Verification off	200	$+0.066$	$[+0.015, +0.118]$	$+0.067$	$[+0.019, +0.116]$
Planning and state: ON *minus* OFF						
BCP	Planning/state OFF	150	$+16.7$	$[+8.0, +25.3]$	$+16.7$	$[+8.0, +25.3]$
Recovery control: +R marks the full system's Finalizer applied to a controlled arm						
BCP	ON+R $-$ OFF+R	150	$+16.7$	$[+9.3, +24.7]$	$+15.3$	$[+7.3, +23.3]$
BCP	ON+R $-$ ON	150	$+15.3$	$[+10.0, +21.3]$	$+12.7$	$[+7.3, +18.0]$
BCP	OFF+R $-$ OFF	150	$+15.3$	$[+10.0, +21.3]$	$+14.0$	$[+8.7, +20.0]$
BCP	Full $-$ ON+R	150	$+0.0$	$[-6.7, +7.3]$	$+2.0$	$[-4.7, +9.3]$
Matched computation: PANINI-DR *minus 8-sample SLEUTH at 1.12 times our PFLOP*						
BCP	8-sample SLEUTH	150	$+8.7$	$[+1.3, +16.0]$	$+8.7$	$[+1.3, +16.0]$

Table 21: Paired score differences on the matched question sets, with bootstrap 95% confidence intervals from 10,000 resamples paired by question. Positive differences favor the first-named arm in each subtraction. Values are percentage points on BCP and FRAMES and mean-F1 units on MoNaCo; n is the number of matched questions. Differences use unrounded scores. Each comparison resamples the same question set in both arms, so an interval describes the difference rather than either score separately.

System	Implementation or adaptation
SLEUTH	Paper/prompt-based: published reasoning, memory, commitment, and top-3 retrieval rules; dense and BM25 routes.
Omni-Decision	Paper-based: 15 iterations, top-6 retrieval, zero-novelty termination after 3 iterations.
HyMem	Paper-based: 4 tools, published tool wording, and per-layer budgets.
Open Deep Search	Released code: search timeout raised from 10 to 300 s; local-link gateway.
PyRAG	Output limit raised from 1,024 to 4,096 tokens; JSON-schema output.
IterResearch	Endpoint/return-value compatibility; assistant prefill, XML repair, JSON lift, action seeding, answer extraction; authors' hard-coded temperature 0.6.
Re-TRAC	Repository prompting configuration; endpoint/return-value compatibility and a persistent driver event loop.

Table 22: Baseline implementations and changes for the shared evaluation environment. Model-facing changes are listed separately from interface compatibility within each row.

The rule fires on 433/1,315 MoNaCo questions; its effect there is unmeasured. Re-TRAC uses its repository entry point and unchanged configuration file with the shared backbone, not its specialized fine-tuned checkpoint. On BCP it produces 76 empty outputs and seven abstentions: its final node returns the last trajectory message, which can be a tool call rather than answer text. These 83 outcomes score zero in the reported comparison.

E Extended Related Work

The methods below address complementary parts of a research investigation: how work is decomposed, what persists between calls, how findings are checked, and how answers are evaluated. We first situate the representation, then explain the research harnesses individually and connect their mechanisms to verification and evidence assessment.

E.1 Reasoning Structure, Plans, and Semantic Workspaces

Reasoning chains and search. Chain-of-thought prompting elicits intermediate steps before an answer (Wei et al., 2022). Self-consistency samples diverse reasoning paths and selects an answer by aggregating across them (Wang et al., 2023). Tree of Thoughts explores alternative reasoning paths, and Graph of Thoughts supports more general combinations of intermediate results (Yao et al., 2023a; Besta et al., 2024). These methods organize how reasoning is generated, searched, or combined. PANINI-DR addresses its continuation across research calls: admitted intermediate values determine what subsequent assignments can use.

Follow-up questions and retrieval. Self-Ask explicitly asks and answers follow-up questions before answering the original query, and can use a search engine to supply the intermediate answers (Press et al., 2023). IRCoT lets intermediate reasoning guide retrieval and retrieved passages inform subsequent reasoning (Trivedi et al., 2023). ReAct interleaves reasoning, actions, and observations to update a course of action through tool interaction (Yao et al., 2023b). These methods connect decomposition to evidence gathering. In PANINI-DR, the research subquestion persists beyond an individual interaction: it specifies the assignment, the claim to check, and where an admitted answer enters the shared state.

Executable intermediate reasoning. Faithful Chain-of-Thought translates a natural-language problem into a symbolic reasoning chain, then uses a deterministic solver to execute that chain and produce the answer (Lyu et al., 2023). The answer is thus generated from the represented computation. PANINI-DR similarly executes defined constraints over admitted inputs, but also tracks how those inputs were researched and reviewed across calls. Its workspace exposes both the lookup dependencies and the calculation; the evidence and local review outcomes remain attached to the researched values.

ReWOO and LLMCompiler. ReWOO separates planning, tool execution, and answer synthesis (Xu et al., 2023). Its Planner emits interdependent instructions with placeholders for evidence; Researchers populate those placeholders, and a Solver combines the plan and observations. LLMCompiler generates a graph of function calls, dispatches tasks as their dependencies resolve, and

executes independent calls in parallel (Kim et al., 2024). It also supports replanning from intermediate results. These are direct precedents for explicit executable dependencies. PANINI-DR attaches a research claim, evidence bindings, and a local review outcome to the values used in such dependencies, with an admission policy governing their entry into state.

Reasoning trees and modular agent pipelines. RT-RAG decomposes a multi-hop question into an explicit reasoning tree, choosing among candidate trees by consensus and separating the core query from its known and unknown entities, then traverses the tree bottom-up with iterative query rewriting and further decomposition when the accumulated evidence is insufficient (Shi et al., 2026). MA-RAG assigns planning, step definition, evidence extraction, and answering to separate agents that exchange intermediate reasoning, with the extractor filtering retrieved passages before synthesis (Nguyen et al., 2025). These methods organize retrieval around explicit intermediate tasks. PANINI-DR connects those tasks to evidence-linked findings and an admission policy: a subquestion carries the claim to be checked, an admitted value enters the shared state with its evidence and review outcome attached, and a checked revision can add or reformulate subquestions while specified bindings survive.

Graph-organized evidence. SentGraph builds a hierarchical sentence graph offline, distinguishing nucleus from satellite sentences and linking topic-level subgraphs through cross-document entity bridges, then performs graph-guided selection and path expansion at query time (Liang et al., 2026). HyCE-RAG represents entities, relations, and contextual evidence as hyperedges, propagates confidence over the entity–hyperedge incidence structure, and assembles ranked evidence chains before generation (An et al., 2026). These structures are built over the corpus, and their purpose is to present the model with a coherent chain in place of flat passages. PANINI-DR instead structures the subquestions: relations connect plan nodes to given anchors and to one another, and retrieved text is attached as evidence to subquestions and proposed values. The admission policy governs which values become available to subsequent assignments. The two are complementary, since graph-organized retrieval could supply the passages a research assignment cites.

Slot-bound retrieval. ConRAG combines corpus-side evidence links with query-side slot binding: intermediate answers fill dependency placeholders for later retrieval, while relation, entity, and text views contribute to a common evidence ranking (Zhu et al., 2026c). PANINI-DR shares the use of intermediate values to guide subsequent work, but centers the execution protocol on persistent subquestions with claim-scoped review, bounded admission, and evidence-linked shared state.

Generative Semantic Workspaces. GSW represents episodes through entities, roles, and relations to preserve context beyond isolated facts (Rajesh et al., 2026a;b). PANINI-DR uses this relational vocabulary to express a query's given anchors and typed blanks at plan nodes. The research harness supplies the execution behavior: assignments seek values for research nodes, review assesses proposals against their claims, and admitted bindings support later research or computation. This separates the inherited representation from the protocol studied here.

E.2 Deep-Research Harnesses and Persistent Memory

DeepResearcher and WebThinker study agents that conduct extended research through search and tool interaction (Zheng et al., 2025; Li et al., 2025). Maintaining continuity over these interactions is a broader memory problem: MemGPT manages information across memory tiers, Generative Agents retrieves and reflects on accumulated experience, and ReSum and context folding compress long interaction contexts (Packer et al., 2023; Park et al., 2023; Wu et al., 2025; Sun et al., 2025). The following harnesses make different choices about the form and execution role of that retained information. We describe their published mechanisms; Appendices A and D.5 specify the implementations and settings evaluated in this paper.

Memory-aware retrieval. MIND uses token entropy and attention to trigger retrieval, filters entities for refined subqueries, and retains retrieved facts across reasoning steps (Ji et al., 2025). Its focus on retrieval timing and memory selection is complementary to PANINI-DR's subquestion-level interface, which connects an assignment, a proposed value's evidence and review, and its subsequent use in the investigation.

SLEUTH. SLEUTH maintains source-linked confirmed facts, ranked answer hypotheses, and prioritized open questions (Liu, 2026). Hypotheses reference supporting and contradicting facts, and

the agent rewrites the memory after tool calls to guide its next action. Runtime checks can enforce the memory format, while a budget-dependent commitment mechanism encourages synthesis. PANINI-DR also uses persistent state to guide research, but Researchers submit local candidate values and evidence for typed subquestions. The harness's review and admission policy determines how these proposals become bindings available to subsequent assignments.

Omni-Decision. Omni-Decision initializes query-specific evidence needs and fact or computation slots, tracks their dependencies, and combines critic judgments with a deterministic reducer that updates shared state (Ma et al., 2026). Its planner and critics use a state digest; dependency completion determines which needs become actionable. It is therefore a close evidence-state approach, including both explicit evidence needs and controlled updates. PANINI-DR realizes a particular combination of subquestions represented through GSW, dispatched research assignments, review scoped to each candidate's claim and evidence, bounded admission with recorded status, and explicit binding-reset rules when the plan changes.

HyMem. HyMem separates a persistent planner from temporary execution and isolated reasoning contexts (Wang et al., 2026). Structured returns carry findings, evidence, sources, status, and remaining gaps; episode, working, and tool memories preserve selected information across context boundaries. Folding and selective memory injection limit what each role receives. PANINI-DR likewise separates local work from shared state. Its focus is the research subquestion: scoped review and admission determine whether a proposed value becomes an evidence-bound input to later work.

Open Deep Search. Open Deep Search combines an Open Search Tool with an agent that decides how to use search and other tools (Alzubi et al., 2025). The search component rephrases queries, processes search-result metadata, and scrapes, chunks, and reranks linked pages. The paper describes ReAct and CodeAct reasoning-agent variants. PANINI-DR places a subquestion-level execution protocol around retrieval: assignments specify what to establish, evidence is attached to relations and proposed values, and admitted bindings provide inputs for further work.

PyRAG. PyRAG decomposes a question into subqueries and synthesizes an executable Python program over retrieval and answering operations (Sun et al., 2026). Intermediate answers occupy variables with explicit data dependencies, and execution records queries, documents, and intermediate results. Runtime errors can trigger program repair, while insufficient-answer signals can expand retrieval for affected steps. PANINI-DR also exposes dependencies, but represents them in a relational research plan linked to candidate review and admitted evidence bindings. Its revision protocol specifies which bindings are reset or retained when the plan's subquestions change.

IterResearch. IterResearch reconstructs its working context from the original question, an evolving report, and the preceding action and tool response (Chen et al., 2025a). Each interaction produces the next report alongside reasoning and an action, preserving selected findings while replacing older interaction history. PANINI-DR instead retains research subquestions and admitted values with evidence and local review status. Both preserve task-relevant information across calls, but the evolving report and the subquestion-bound state provide different interfaces for choosing and carrying out subsequent work.

Re-TRAC. Re-TRAC compresses completed research rollouts into structured state for subsequent rollouts (Zhu et al., 2026b). Its compression retains partial answers and conclusions, evidence provenance and source-verification information, unresolved hypotheses, unexplored branches, and failed attempts. This memory supports exploration across attempts while reducing repeated work. PANINI-DR maintains subquestions and bindings within the investigation, using admitted values to enable dependent assignments and recording local review at admission. This contrasts rollout-level exploration memory with bindings that guide individual research assignments.

Answer completion. Harnesses differ in how they finish an investigation. SLEUTH uses a commitment nudge to synthesize from accumulated facts (Liu, 2026), while Omni-Decision can terminate with an insufficient status when no remaining action can reduce an open need (Ma et al., 2026). PANINI-DR recovers an answer from retained articles when the loop ends without an admitted target, leaving the workspace unchanged. These policies govern how accumulated research becomes a delivered answer; Appendix D.1.2 measures recovery's contribution.

E.3 Verification, Feedback, and Corrective Retrieval

Learned verifiers and execution feedback. Cobbe et al. (2021) train verifiers on generated mathematical solutions labeled by final-answer correctness and use their scores to select among sampled solutions. Process supervision instead supplies judgments of intermediate steps (Lightman et al., 2023). CodeRL uses a critic and execution feedback to improve code generation (Le et al., 2022), while verifier-guided search studies how to allocate test-time computation using such evaluations (Snell et al., 2024). These approaches use verification to guide solution selection, generation, or further search.

Iterative feedback and separated checking. Self-Refine uses the same underlying LLM to generate an output, give feedback, and revise it iteratively without additional training (Madaan et al., 2023). Reflexion retains verbal feedback as memory for later attempts (Shinn et al., 2023). Chain-of-Verification first drafts an answer, generates checking questions, answers those questions, and revises the draft (Dhuliawala et al., 2024). Its factored variant isolates each verification question from the draft and other verification answers. These are precedents for feedback-driven revision and scoped checking.

LLM-as-a-Verifier. Kwok et al. (2026) develop fine-grained scoring from scoring-token probabilities, repeated evaluation, and decomposed criteria for assessing agentic solutions without additional training. This work develops the checking capability itself; PANINI-DR studies its role in a research protocol rather than adopting that scoring algorithm. Analyses of agent failures further motivate tracking how a local decision affects subsequent actions (Zhu et al., 2025).

Self-RAG and CRAG. Self-RAG trains a model to decide when to retrieve and to generate reflection tokens assessing passage relevance, support for generated text, and response utility (Asai et al., 2024). These signals guide retrieval and the selection of generated segments. CRAG evaluates the relevance of retrieved documents, then uses the resulting confidence to refine retrieved content, seek additional web evidence, or combine both (Yan et al., 2024). Both use checking to guide retrieval-augmented generation.

Sufficiency and calibrated abstention. SURE-RAG treats evidence sufficiency as a property of the retrieved set rather than of individual passages: a shared claim–evidence verifier scores each pair, and the local distributions are aggregated into coverage, relation-strength, uncertainty, and retrieval features that yield a support, refute, or insufficient decision, with the system answering only when support is established (Qiu et al., 2026). R2VC separates retrieval, generation, verification, and calibration, using an external natural-language-inference model to select among structured verdict candidates and a sequence-level calibrator for selective abstention (Dixit & Pandey, 2026). In both, the decision governs the final answer, which is either produced or withheld. PANINI-DR applies its check to an intermediate value instead: a negative verdict returns the proposal to the Researcher under the shared review cap rather than ending the question. The admission policy retains the review outcome in the binding record while making the admitted value available to later work. A calibrated sufficiency score could inform that admission decision, which currently rests on a structured verdict carrying no confidence estimate.

Dependency-guided formal verification. The reported formalization of Fermat's Last Theorem illustrates how an explicit dependency structure can organize work around local checks: Prove2Me coordinates parallel proof tasks through a dependency graph, while Lean checks the resulting formal proofs (Anthropic, 2026; Chen et al., 2026). The relevant connection is the coordination of dependent assignments and their checking outcomes. Deep research does not have an equivalent formal proof checker for open-ended source claims; PANINI-DR instead records scoped model judgments and the harness's admission decisions alongside the evidence.

Across these lines of work, feedback affects which results are selected, revised, or supported through further retrieval. PANINI-DR connects these decisions to the research state. The Verifier assesses a candidate value against its persistent research subquestion's local claim and evidence; feedback can prompt further research, while the harness separately applies its admission policy and preserves the outcome in the binding record. The Orchestrator can request a plan revision to reopen a subquestion for further investigation.

E.4 Attribution and Evidence-Based Evaluation

Attributable-to-Identified-Sources evaluation asks whether a generated statement is supported by identified material (Rashkin et al., 2023). ALCE evaluates cited generation, including citation correctness and completeness (Gao et al., 2023), and ExpertQA studies attributed answers to expert-authored questions (Malaviya et al., 2024). These approaches assess semantic support. Our observed-support measure instead retains official answer credit when the answer is recoverable from the available execution evidence record. For computed answers, Obs-D additionally checks cited inputs and replays the recorded calculation. The measures thus address evidence availability and recorded computation alongside, rather than in place of, question-conditioned attribution.

Provenance as a training signal. STAMP uses provenance during training rather than evaluation. A reference-based verifier checks whether each cited document supports an entity or relation in a training-time evidence graph, first-exposure attribution traces a supported citation back to the action that first surfaced it, and the resulting step-level credit is applied by advantage modulation that leaves the trajectory-level reward unchanged (Xu et al., 2026). PANINI-DR is training-free and uses its records for measurement and control rather than for credit assignment. The correspondence is nonetheless direct: an admitted binding already names the proposal, the evidence attached to it, and the review outcome that admitted it, which is the provenance a signal of this kind requires.