SEARCHPROOF: CHECKABLE NEGATIVE EVIDENCE FROM ZERO-RESULT AGENT SEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Search tools tell language-model agents what they found, but rarely what they actually covered. This asymmetry makes a zero-result call unusually difficult to reason about: an empty result can justify absence inside a searched directory while saying nothing about the rest of a repository. We introduce SEARCHPROOF, an inference-time interface that compiles a proposed negative claim into a claim-relative coverage obligation. Each search returns a compact receipt recording the relevant universe, executed scope, result set, coverage, and minimal completion set. The agent can pass this checked set directly to ordinary search before licensing a global negative conclusion. We evaluate this capability with SCOPEBENCH: 72 controlled dependency-graph tasks and 144 query-matched tasks mined from 8 repositories spanning 5 language ecosystems and 2,625 source files. Every identifier appears once in each case type, and query-only balanced accuracy remains near chance. Across 4 agents and 1,872 analyzed model–task outcomes, agents recognize incomplete coverage reliably, yet exact controlled completion planning is only 25.0% from raw results and 100.0% with SearchProof. In interactive use, SEARCHPROOF raises grounded success from 94.8% to 100.0% in the two-model ablation grid. A preregistered full-corpus confirmation across DeepSeek and GLM-5.3 then moves 267/288 paired outcomes (92.7%) to 288/288 (100.0%): 21 wins, no losses. The direction remains positive at every nonzero session, query-group, and repository aggregation. Removing the executable completion field reduces DeepSeek to 95.1% (7–0 restored outcomes), isolating the operative mechanism. A source-backed deterministic executor reaches 144/144 tasks with exact completion and no follow-up language-model decision. Across 720 saved agent scopes, direct file execution agrees with the frozen backend on 100.0% of outcomes. SearchProof reads 62.0% fewer files and 60.6% fewer bytes than full-universe restart, reducing median scan latency by 54.3%. On the discriminating incomplete-absence stratum, the two-model gain is 79.2% to 100.0%. A third-agent stress replication moves from 41.7% to 85.4% (paired +43.8 points, 95% CI [25.0, 62.5]), with positive effects after holding out every repository. Median receipt construction plus checking is 4.53 ms at 4,096 components. These results make coverage a first-class evidence type and turn "not found" into a precise, executable observation for tool-using agents.

1 INTRODUCTION

Consider an agent preparing to add a class named `PaymentResolver`. It searches the current package and receives `No matches found`. The result is factually correct, yet the decision that matters is repository-wide: does the class already exist somewhere relevant? Declaring absence can create a duplicate; searching the whole repository discards useful work and can be expensive; and choosing the wrong next directory consumes the remaining tool budget. The missing object is a coverage certificate: an explicit account of the region over which an empty result has evidential force.

Modern agents interleave reasoning and external actions through interfaces such as ReAct (Yao et al., 2023), Toolformer (Schick et al., 2023), and repository agents (Yang et al., 2024; Zhang et al., 2024). Search research has rapidly improved retrieval, ranking, multi-turn policies, and system efficiency (Reddy et al., 2026; Wang et al., 2026; Yang et al., 2026). These interfaces are optimized for positive evidence: they return documents, snippets, or ranked code. A zero hit is normally rendered as a

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.

short string. The agent must recover the search scope from its trajectory, relate that scope to the downstream claim, infer which regions remain, and decide whether more search is warranted. Strong models can often make the first binary decision, but our pilot reveals a sharper bottleneck: after recognizing incomplete coverage, they select the exact dependency frontier in only 14.6–35.4% of raw-search cases.

We formulate negative-evidence use as *claim completion*. A negative claim names both a predicate and the universe over which it applies. A search execution covers a subset of that universe. The difference between the two is a concrete obligation, not an invitation to produce more chain-of-thought. SEARCHPROOF computes this difference in the tool layer, emits it as a structured coverage receipt, and preserves already completed work. When coverage is complete and the hit set is empty, the claim is licensed. When it is incomplete, the receipt exposes the smallest additional component set needed to settle the claim. This design is model-agnostic, requires no training, and adds millisecond-scale overhead.

We build SCOPEBENCH to measure this capability directly. Controlled dependency graphs separate complete absence, incomplete absence, and a hidden positive match. Public- repository tasks instantiate the same distinctions on frozen Python, JavaScript, Go, Rust, and Java codebases. Every task contains an exact manifest and executable search ground truth; evaluation therefore needs no model judge. Crucially, every identifier appears once in each case type, so spelling carries zero label information by construction. Four matched interfaces separate a generic warning, coverage accounting, and the executable completion set. We hold model, query multiset, task order, batch composition, output schema, and tool budget fixed within each comparison. The resulting evidence contains 1,872 saved model–task outcomes, genuine tool calls, a full 144-task replication, and a query-only leakage control.

Our contributions are:

- **A measured agent bottleneck.** Agents reliably detect incomplete coverage while failing to name the exact remaining frontier; across four model endpoints, this failure concentrates precisely on premature absence conclusions.
- **An inference-time method and checkable artifact.** SEARCHPROOF attaches a coverage receipt to ordinary search and converts unresolved claims into precise completion actions without changing the underlying language model.
- **A broad, judge-free evaluation.** SCOPEBENCH contributes 144 query-matched public-repository tasks across 8 projects and 5 language ecosystems, 1,872 model–task outcomes from 4 agents, full-corpus cross-model and field confirmations, a hard-stratum stress replication, source replay, dependence-aware sensitivity analysis, and a 144/144 construction audit.
- **A practical result.** In full-corpus interactive execution, SEARCHPROOF improves grounded end-to-end success by 7.3 points over raw search while real file-backed execution reads 62.0% fewer files, 60.6% fewer bytes, and takes 54.3% less median scan time than full-universe restart; median receipt construction and verification together take 4.53 ms at 4,096 components.

2 NEGATIVE EVIDENCE IS A COVERAGE PROBLEM

Search and claims. Let a repository manifest be $M = (V, E, w)$, where V is a set of searchable components, E records a structural relation such as dependency or containment, and $w(v) > 0$ is a cost proxy (files or bytes). A search operator $S(q, X)$ evaluates query q over $X \subseteq V$ and returns all hits H_X. A negative claim is a pair $c = (q, U_c)$: "query q has no match in claim universe U_c." The universe may be a repository root, a package subtree, or the transitive dependency closure of an entry component. It is determined by the task, not by whichever scope happened to be searched.

After an execution over X, define the covered claim set $C = X \cap U_c$ and the coverage ratio

$$\rho(c, X) = \frac{\sum_{v \in C} w(v)}{\sum_{v \in U_c} w(v)}. \tag{1}$$

An empty hit set is conclusive exactly when $C = U_c$. Otherwise it supports the scoped statement "no match in C" and leaves a completion obligation

$$\Delta(c, X) = U_c \setminus X. \tag{2}$$

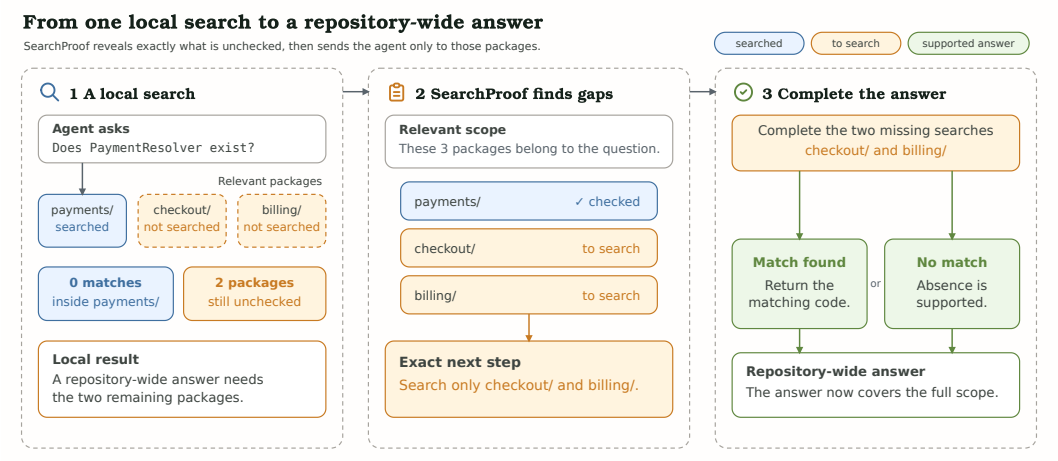


Figure 1: SEARCHPROOF converts a zero-result tool call into a checkable, actionable coverage object. Blue marks executed scope, amber marks the exact completion set, and green marks settled evidence.

For graph-defined universes, the controller may expose the next dependency frontier $F = \{v \in \Delta : \exists u \in C, (u, v) \in E\}$, then update the receipt after execution.

Externalizing exact coverage accounting. An agent can in principle reconstruct Equation 2 from a manifest and its trace. In practice, this mixes language reasoning with exact set accounting. Components may number in the hundreds, scope arguments may have appeared many turns earlier, and already searched regions should not be paid for twice. A generic warning signals incompleteness; exact accounting identifies the remaining region. We therefore move accounting to the layer that already possesses the manifest and execution log.

3 SEARCHPROOF

3.1 CLAIM COMPILATION

The claim compiler maps a task goal and proposed conclusion to $c = (q, U_c)$. Our implementation supports two deterministic universe constructors. *Root scope* includes every indexed source component below a declared repository root. *Dependency scope* computes the transitive closure from an entry component. The compiler receives relevance from the task adapter through the universe rule and entry/root, while the query comes directly from the executed search. This separation makes the receipt reproducible and lets applications add new universe constructors without retraining the agent.

3.2 COVERAGE RECEIPTS

Each call returns the ordinary result plus

$$R = \langle q, U_c, X, H_X, A, \rho, \sigma, \Delta \rangle, \tag{3}$$

where A accounts for every claim component as searched or unsearched, $\sigma \in \{\texttt{COMPLETE}, \texttt{INCOMPLETE}\}$, and Δ is present for an incomplete claim. Counts and weights are recomputed from the frozen manifest and the search execution record. A checker confirms that searched and completion sets are disjoint, their union equals U_c, weights sum correctly, and status agrees with the sets. This compact object is both human-readable and directly consumable by an agent controller.

3.3 CLAIM COMPLETION

When $H_X \neq \emptyset$, the positive evidence settles existence. When $H_X = \emptyset$ and $\sigma = \texttt{COMPLETE}$, the controller returns the licensed negative result. For $\texttt{INCOMPLETE}$, it executes $S(q, \Delta)$ or successive

Table 1: SCOPEBENCH exposes both controlled mechanisms and real repository structure. Every split balances complete absence, incomplete absence, and an omitted positive.

Split	Tasks	Projects	Components	Files	Ecosystems
Controlled pilot	72	synthetic	12–34/task	6–75/component	dependency DAGs
Public repositories	144	8	6–195/project	2,625 total	Py, JS, Go, Rust, Java
Scale benchmark	5,000	synthetic	16–4,096/task	weighted	dependency DAGs

frontiers under a tool budget, merges the execution record, and emits a new receipt. Already covered components are never re-searched. A budgeted deployment can stop early with an explicitly scoped negative statement; our benchmark instead supplies enough budget to settle every task so that accuracy and cost are directly comparable.

Guarantees under exact component search. Assume S enumerates all matches in each executed component and the checker accepts R. If $H_X = \emptyset$ and $\sigma = \texttt{COMPLETE}$, then no match for q exists in U_c: completeness gives $U_c \subseteq X$, and exact component search would otherwise place a match in H_X. Further, for worst-case certification of absence, $\Delta = U_c \setminus X$ is the unique component-minimal completion that preserves prior work. Any sufficient additional set must contain every member of Δ; searching exactly Δ adds none outside the claim universe and none already covered. A positive instance may settle earlier if an adaptive search encounters a hit. The minimality statement concerns black-box exact component search without auxiliary logical evidence. These statements are simple by design: SEARCHPROOF turns an implicit reasoning burden into an auditable set relation.

4 SCOPEBENCH

4.1 CONTROLLED AND PUBLIC TASKS

Controlled dependency graphs. We generate acyclic component graphs with 12–34 nodes, 1–3 incoming dependency edges, and heterogeneous file weights. An entry component defines U_c by transitive closure. One third of tasks are fully searched and truly absent; one third are partially searched and absent; and one third place a unique target in the missing closure. Identifiers, graphs, and weights are generated from a frozen seed. This design isolates whether an agent stops correctly and whether it selects the exact next frontier.

Public repositories and query matching. We freeze eight permissively available repositories at exact commits. A language- specific parser extracts top-level function, class, type, trait, and interface identifiers from Python, JavaScript/TypeScript, Go, Rust, and Java; positive targets exclude test and benchmark paths. Components are real source directories. We form 48 matched triples. For each identifier, one task places its unique definition in a component omitted from the initial search, one uses the same spelling for incomplete absence in another repository, and one uses it for complete absence in a third. Thus every query appears exactly once in every case type, and absent queries are also verified not to occur as source tokens anywhere in their target repository. Opaque hashed case IDs remove ordering cues. The corpus contains 2,625 source files and 6–195 components per project. Task construction stores the source path and line for positives, but model prompts reveal neither. Automated invariant checks pass all 144 tasks; a deterministic 24-task sample spans every repository–case-type pair for source-level inspection.

4.2 INTERFACES AND MODELS

We compare four matched interactive interfaces: **Raw** returns hits and searched components; **Caveat** adds one sentence that a repository-level claim may require more search; **Receipt without completion** provides the claim universe, accounted set, coverage, and status; and **SearchProof** additionally provides the exact minimal completion. A controlled experiment also includes **Manifest reasoning**, which asks the model to derive missing components itself. We compute a **full-universe restart** cost reference by summing every file in the frozen claim universe; this row provides a cost endpoint for the interactive interfaces. A fifth, model-free policy test executes SearchProof's checked Δ directly and answers only from the returned exact-search hits. This **execute-Δ controller** isolates the value

of the receipt from the language model's decision to copy or modify it; it makes no follow-up model call and never reads the case label when choosing an answer.

The controlled and primary interactive experiments use DeepSeek V4 Flash and GLM-5.2 Fast. A frozen balanced 48-task slice drives genuine tool-calling sessions for both agents; it contains 16 matched query triples and spans all eight repositories. Each condition uses the same twelve four-task batches and query multiset. We then preregister and execute a full-corpus confirmation on all 144 tasks: DeepSeek under Raw, receipt-without-completion, and SearchProof, and GLM-5.3 Flash under matched Raw and SearchProof. This yields 180 genuine tool-calling sessions with identical four-task batches across matched arms. The originally frozen GLM-5.2 endpoint had been retired and returned no experimental answers; a dated amendment substituted the current same-family GLM-5.3 endpoint before any cross-model outcome was observed, while leaving every task, condition, metric, and stopping rule frozen. Within a session, the agent has one successful batched follow-up search that can name every remaining component for all four cases, followed by one answer commit. In every condition, the backend executes exactly the component set emitted by the model; the receipt never substitutes an oracle action. For deterministic scoring during model runs, the backend answers exact-identifier queries from the frozen task index. A separate replay rebuilds indexes from actual repository files and checks every executed query against the resulting definitions. The prompt, action schema, matched batch composition and order, temperature policy, and search-call budget are fixed within each model. Condition names, target locations, and correctness feedback are hidden, while raw responses and API metadata are preserved.

The primary grid revealed that complete negatives and hidden positives saturate every interface. We therefore preregistered the discriminating *incomplete-absence* stratum and ran all 48 such tasks with a third agent, GPT-5.6 Luna, under matched Raw and SearchProof interfaces. The 24 four-task sessions add 96 model–task outcomes at a recorded cost of $0.016. The tool validates every component before execution; one Raw request named an out-of-manifest component and was retained as a zero-execution protocol failure rather than rerun to success.

4.3 Metrics and statistics

The primary metric is *grounded end-to-end success*: a hidden positive is found, or a true negative is concluded after full claim coverage. We also report answer accuracy, exact completion, hidden-target recall, added files, tool calls, tokens, agent/controller wall-clock latency, and API cost. Accuracy intervals are Wilson intervals; headline paired differences use 10,000 task-level bootstrap samples clustered by repository. We additionally report paired wins/losses, exact McNemar tests, and leave-one-repository-out effect ranges. Because the same tasks recur across agents and four-task calls, a dependence sensitivity pass also aggregates deltas by model–session, by query group while pooling agents, and by repository while pooling agents. At each unit we report positive–negative–tied cluster counts, an exact sign test over nonzero clusters, and a cluster-resampled difference interval. Rule-based ground truth removes judge variance. A query-only control withholds repository identity, manifests, and search observations, then measures balanced accuracy with query-group bootstrap and a 50,000-draw within-group label-permutation test. The artifact tests every construction invariant, independently checks definitions with GNU ctags plus Python AST, and records the stratified source audit.

5 Results

RQ1: receipts turn coverage into exact action. Both controlled agents select whether to continue search correctly, yet raw output leaves the completion frontier implicit. Their pooled exact-frontier rate is 25.0% with raw results and 41.7% with a scoped caveat (Table 3). Showing the complete manifest and asking the model to reconstruct the difference reaches 30.2%. SearchProof supplies the checked set difference and reaches 100.0% for both agents. This reveals the operative bottleneck: recognizing incomplete coverage is easier than executing exact set accounting over the remaining components.

RQ2: claim completion produces fully supported answers. Raw agents achieve 100.0% binary answer accuracy, but grounded success is 94.8%: several correct labels are unsupported by either a located definition or complete coverage. SearchProof preserves answer accuracy and raises grounded

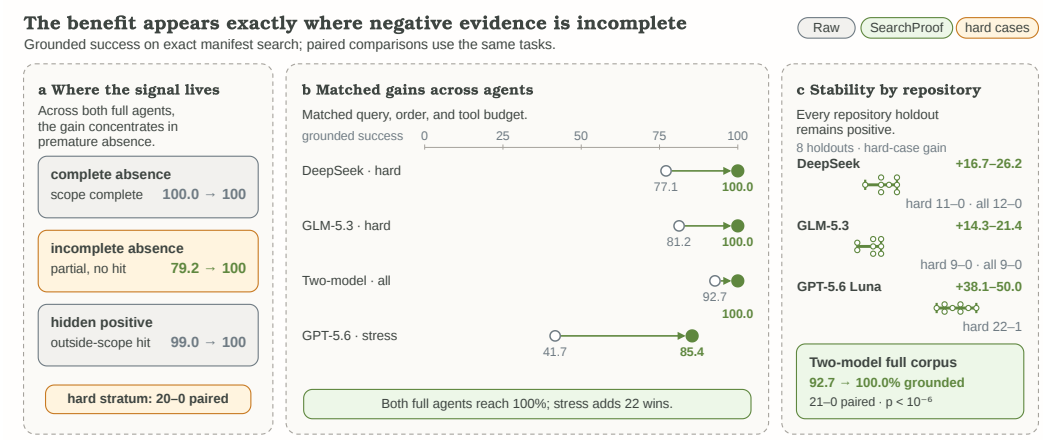


Figure 2: The initial 5.2-point result contains two saturated task types; the discriminating incomplete-absence stratum exposes the execution gap. (a) Only premature absence requires additional coverage. (b) SearchProof improves both full-corpus agents and the third-agent stress test. (c) Every leave-one-repository-out effect remains positive; hollow points are the eight held-out-repository estimates and whiskers show their range. Across the two complete 144-task model runs, SearchProof moves 267/288 grounded outcomes to 288/288, with 21 wins and no losses.

Table 2: Grounded success under genuine tool execution. Panel (a) is the matched 48-task interface ablation; SearchProof improves 91/96 supported outcomes to 96/96 (paired +5.2 points, repository-clustered 95% CI [0.8, 11.8]). Panel (b) executes the complete 144-task corpus for each agent. Every arm executes the model-requested scope; success requires a correct answer backed by an observed target or complete claim coverage.

(a) Matched 48-task interface ablation

Agent	Raw	Caveat	Coverage, no Δ	SearchProof	Exact request Raw/SP	Gain
DeepSeek V4 Flash	93.8%	95.8%	95.8%	**100.0%**	87.5%/100.0%	+6.2
GLM-5.2 Fast	95.8%	95.8%	95.8%	**100.0%**	85.4%/100.0%	+4.2
Pooled	94.8%	95.8%	95.8%	**100.0%**	86.5%/100.0%	+5.2

(b) Complete 144-task corpus per agent

Agent	n/arm	Raw	Coverage, no Δ	SearchProof	Wins–losses	McNemar p
DeepSeek V4 Flash	144	91.7%	95.1%	**100.0%**	12–0	0.000488
GLM-5.3 Flash	144	93.8%	–	**100.0%**	9–0	0.003906
Pooled	288	92.7%	–	**100.0%**	21–0	$< 10^{-6}$

success to 100.0%, an absolute 5.2-point gain (clustered 95% CI [0.8, 11.8]). DeepSeek gains 6.2 points (CI [1.6, 12.9]), and GLM gains 4.2 points (CI [0.0, 11.1]). The backend executes each model request verbatim; SearchProof agents themselves request the exact completion set on 100.0% of tasks, compared with 86.5% under raw search. The receipt therefore converts a plausible label into a conclusion supported by the agent's actual tool trajectory.

The discriminating stratum reveals the full effect. Complete negatives and hidden positives reach 100% grounded success under every primary interface, so the aggregate comparison deliberately includes 64 saturated model–task pairs. Among the 32 incomplete-absence pairs, SearchProof moves 84.4% to 100.0% (+15.6 points, clustered 95% CI [2.2, 40.0]); leaving out any repository yields gains in the range [10.0, 19.2] points. The two complete-corpus model runs provide 96 paired hard-stratum outcomes: 79.2% to 100.0% (+20.8 points, 95% CI [9.4, 34.4]), with 20 paired wins and 0 losses (exact McNemar $p = 0.000002$); every held-out-repository estimate remains positive ([15.5, 23.8] points). The third agent sharpens the same result on all 48 hard tasks: 41.7% to 85.4%, +43.8 points (95% CI [25.0, 62.5]), with 22 wins and 1 loss. Its leave-one-repository-out gains remain 38.1–50.0 points. Figure 2 therefore localizes the method's value rather than asking a saturated average to carry the mechanism.

Table 3: Exact next-frontier selection on 72 controlled tasks per model. A warning and the full manifest help model reasoning; the executable receipt makes every frontier exact.

Agent	Raw	Caveat	Manifest	SearchProof
DeepSeek V4 Flash	35.4	56.3	45.8	**100.0**
GLM-5.2 Fast	14.6	27.1	14.6	**100.0**
Pooled	25.0	41.7	30.2	**100.0**

Table 4: Query-only leakage control on the 48-task interactive split. Each query spelling occurs once in every task type. Balanced accuracy remains statistically indistinguishable from the 50% baseline; inference is clustered by query group.

Agent	Balanced acc.	95% CI	Permutation p
DeepSeek V4 Flash	48.4%	[42.2, 54.7]	0.788
GLM-5.2 Fast	42.2%	[32.8, 53.1]	0.971

RQ3: the completion field drives the effect. A generic caveat reaches 95.8% grounded success, and a coverage receipt without Δ reaches 95.8%. Adding the executable completion set raises success by 4.2 points in the initial grid. The complete-corpus DeepSeek ablation makes the mechanism decisive: coverage without Δ reaches 95.1%, while SearchProof reaches 100.0% (+4.9 points, 95% CI [1.4, 9.0]; 7–0 paired, exact McNemar $p = 0.015625$). Exact completion requests rise from 88.9% without Δ to 99.3% with it. Binary answer accuracy without Δ is 100%, so the effect is not label prompting: coverage status explains why more search is needed, while Δ supplies the exact action that settles the claim. The seven restored outcomes span four positive and four tied repositories (repository sign $p = .125000$), while the session and query-group direction counts are 6–0 and 7–0. By case type, raw grounded success is 100.0% for complete negatives, 84.4% for incomplete negatives, and 100.0% for hidden positives; SearchProof reaches 100.0% in every type.

The source-backed execute-Δ policy independently confirms that the receipt is an action, not merely explanatory text. It scans exactly the checked source components, answers only from returned definitions, and reaches 144/144 grounded tasks with 144/144 exact completions, 124.6 added files per task, and 0 follow-up LM decisions (Appendix Table 8). The two agent endpoints reach 288/288 grounded outcomes while requesting 99.0% exact completions and searching 124.7 files per task. Direct execution therefore preserves perfect success while isolating the small residual cost of model-modified completion sets.

RQ4: the effect survives full-corpus tool execution. The v2 corpus assigns every identifier once to complete absence, incomplete absence, and hidden presence. Query-only predictions remain near chance for both agents (Table 4), while all 144 construction checks pass. An independent ctags/AST validator confirms all 96 absent cases and 45/48 positive definitions; direct source inspection confirms the three language constructs outside its coverage. On all 144 tasks, the preregistered matched DeepSeek execution moves 91.7% to 100.0% (12–0, $p = 0.000488$), and the frozen cross-model confirmation moves GLM-5.3 from 93.8% to 100.0% (9–0, $p = 0.003906$). Pooled across 288 paired outcomes, grounded success rises from 92.7% to 100.0%, a 7.3-point change (repository-clustered 95% CI [3.1, 11.8]; 21–0; exact McNemar $p < 10^{-6}$). This is observed end-to-end tool use rather than proposed action scoring: all 180 sessions commit answers, and source replay agrees on 100.0% of 720 model–task outcomes. The direction holds separately in both full-corpus model endpoints and spans all eight repositories and five language ecosystems. It also survives the dependence audit (Appendix Table 7): the nonzero direction count is 20–0 across model–session pairs, 16–0 across query groups pooling both models, and 7–0 across repositories pooling both models; the respective cluster-resampled intervals are [4.5, 10.1], [4.2, 10.8], and [3.1, 11.8] points.

RQ5: completion reduces measured source-search work. Direct file-backed execution over all 288 full-corpus SearchProof outcomes reads 62.0% fewer files and 60.6% fewer bytes than a measured full-universe restart; median paired scan latency falls 54.3%. On the 192 outcomes that genuinely require follow-up, the reductions remain 43.0% in files, 40.8% in bytes, and 36.0% in latency. All 720 saved full-corpus scopes reproduce the frozen backend's hit/no-hit decision (100.0%

agreement). The interactive grid costs $0.21 in model calls, and the full-corpus confirmation costs $0.06. Tool-side accounting stays inexpensive at larger scales: median construction and checking for a 4,096-component receipt take 2.67 and 1.87 ms, respectively (4.53 ms combined; p95 5.18 ms) (Table 5).

Table 5: Receipt construction and checking over 1,000 generated graphs per size. Times are milliseconds on one CPU process; all 5,000 receipts pass verification.

Components	Build med.	Build p95	Check med.	Check p95
16	0.006	0.009	0.006	0.007
64	0.026	0.031	0.021	0.026
256	0.123	0.139	0.093	0.106
1,024	0.567	0.599	0.410	0.440
4,096	2.667	3.023	1.867	2.156

6 Related Work

Tool-using language-model agents. ReAct interleaves reasoning and environment actions (Yao et al., 2023); Toolformer learns when to invoke external APIs (Schick et al., 2023); and Self-RAG jointly retrieves and critiques evidence (Asai et al., 2024). Repository agents and benchmarks extend this pattern to long-horizon software tasks (Jimenez et al., 2024; Yang et al., 2024; Zhang et al., 2024). These methods decide what action to take or which positive context to consume. SearchProof is complementary: it represents what an executed action did *not* establish and compiles that gap into the next action.

Search agents and retrieval evaluation. Search Arena studies large-scale human preferences for search-augmented models (Miroyan et al., 2026); InfoMosaic-Bench evaluates multi-source tool use (Du et al., 2026); and recent work improves multi-turn information-gain policies and end-to-end efficiency (Wang et al., 2026; Yang et al., 2026). Embedding retrieval has representational limits even for simple realistic queries (Weller et al., 2026), while GraphRAG-Bench identifies settings in which graph structure helps retrieval (Xiang et al., 2026). SearchProof uses graph or containment structure for a different purpose: defining the universe required by a downstream negative claim. Its central metric is therefore not recall@k, but whether the agent's absence conclusion is supported by complete claim-relative coverage.

Data provenance and completeness. Database research represents how outputs depend on input records through provenance semirings (Green et al., 2007) and uses explicit completeness statements to decide when a query answer is complete (Darari et al., 2013). SearchProof translates this foundation into an agent-facing execution contract. Its unit is a proposed natural-language claim over a tool-specific universe; the receipt binds that claim to the executed scope, and the controller turns missing coverage into the next tool action. ScopeBench measures this action-level consequence across model agents.

Code understanding and localization. CodeSense provides fine-grained semantic reasoning tasks grounded in real projects (Roy et al., 2026), and SWERank improves positive issue-to-code localization (Reddy et al., 2026). Building on this line, ScopeBench contributes the complementary decision boundary after a search returns no code: stop, search an exact remaining set, or pay for full-universe restart.

7 Demonstrated Scope and Extensions

Our experiments establish manifest-bound, exact-identifier claim completion. The task adapter supplies U_c, and SearchProof performs exact accounting over that universe. The three-agent hard stratum, two-model full-corpus confirmation, and uniformly positive leave-one-repository-out effects demonstrate the mechanism across difficulty, models, and repositories. The full-corpus comparison reuses frozen tasks to isolate model dependence, while the interactive protocol isolates execution

fidelity with at most one batched follow-up. Fuzzy semantic retrieval adds threshold and index-recall terms; new domains add their universe adapter. Longer trajectories can update the same receipt after each call and stop when a positive match appears.

8 CONCLUSION

Search results are evidence about both matches and coverage. SEARCHPROOF makes the latter explicit: a negative claim becomes a universe, a search becomes an accounted subset, and the missing difference becomes an executable next step. Across controlled graphs and public multi-language repositories, this small interface change yields a large improvement in exact completion and manifest-bound search success while avoiding full-universe restart. Checkable coverage receipts provide a practical foundation for agents that reason precisely from "not found."

REPRODUCIBILITY STATEMENT

The anonymous supplementary package contains frozen task manifests and source commits, generation and analysis scripts, model prompts and identifiers, scored per-task outputs, provider-free per-session tool-event chronology, search and commit records, seeds, receipt checks, aggregate cost logs, and commands for every reported table and figure. Raw provider JSONL is retained for author audit but excluded from the anonymous release because it contains provider/session and machine-specific metadata. All headline numbers are generated from saved JSON artifacts. The included sanitized audit script replays protocol counts and ordering without raw provider traces, and the evaluation requires no model judge.

AI USE STATEMENT

Language-model agents are the object of study and were used to execute the frozen evaluation tasks. Language-model assistance also supported literature retrieval and discovery, research ideation and execution, implementation, drafting, editorial revision, and pre-submission critique under author direction. The authors defined the research question and evaluation, inspected generated tasks and outputs, executed the experiments, and verified all cited relationships and quantitative claims against the saved artifacts. The anonymous code, configurations, sanitized session records, and derived results are available at `https://anonymous.4open.science/r/searchproof/`. The authors take full responsibility for the submission.

REFERENCES

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.

Fariz Darari, Werner Nutt, Giuseppe Pirrò, and Simon Razniewski. Completeness statements about RDF data sources and their use for query answering. In *The Semantic Web–ISWC 2013*, pp. 66–83. Springer, 2013. doi: 10.1007/978-3-642-41335-3_5.

Yaxin Du, Yuanshuo Zhang, Xiyuan Yang, Yifan Zhou, Cheng Wang, Gongyi Zou, Xianghe Pang, Wenhao Wang, Menglan Chen, Shuo Tang, and Siheng Chen. Infomosaic-bench: Evaluating multi-source information seeking in tool-augmented agents. In *International Conference on Learning Representations*, 2026. URL `https://openreview.net/forum?id=AhQFDBBIRZ`.

Todd J. Green, Grigoris Karvounarakis, and Val Tannen. Provenance semirings. In *Proceedings of the Twenty-Sixth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pp. 31–40, 2007. doi: 10.1145/1265530.1265535.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024.

Mihran Miroyan, Tsung-Han Wu, Logan King, Tianle Li, Jiayi Pan, Xinyan Hu, Wei-Lin Chiang, Anastasios Angelopoulos, Trevor Darrell, Narges Norouzi, and Joseph E. Gonzalez. Search arena: Analyzing search-augmented LLMs. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=MMGRlDnhtI.

Revanth Gangi Reddy, Tarun Suresh, JaeHyeok Doo, Ye Liu, Xuan-Phi Nguyen, Yingbo Zhou, Semih Yavuz, Caiming Xiong, Heng Ji, and Shafiq Joty. SWERank: Software issue localization with code ranking. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=OnkRqbNhe3.

Monoshi Kumar Roy, Simin Chen, Benjamin Steenhoek, Jinjun Peng, Gail Kaiser, Baishakhi Ray, and Wei Le. Codesense: A real-world benchmark and dataset for code semantic reasoning. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ehXVDJm0PS.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, 2023.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=qkWP6phrvZ.

Orion Weller, Michael Boratko, Iftekhar Naim, and Jinhyuk Lee. On the theoretical limitations of embedding-based retrieval. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=k9CzIvzfaA.

Zhishang Xiang, Chuanjie Wu, Qinggang Zhang, Shengyuan Chen, Zijin Hong, Xiao Huang, and Jinsong Su. When to use graphs in RAG: A comprehensive analysis for graph retrieval-augmented generation. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=i9q9xDMjG7.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, 2024.

Tiannuo Yang, Zebin Yao, Bowen Jin, Lixiao Cui, Yusen Li, Gang Wang, Xiaoguang Liu, and Willie Neiswanger. Demystifying and enhancing the efficiency of large language model based search agents. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=BtWBi17eVi.

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*, 2023.

Kechi Zhang, Jia Li, Ge Li, Xianjie Shi, and Zhi Jin. Codeagent: Enhancing code generation with tool-integrated agent systems for real-world repo-level coding challenges. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 13643–13658, 2024. doi: 10.18653/v1/2024.acl-long.737.

A Receipt and execution schema

Each receipt serializes the query and claim universe, the executed and accounted component sets, weighted coverage, result count, status, and Δ. The checker recomputes all set and weight relations. In the interactive harness, the agent calls `start` once, emits one batched component map to `search`, and commits one answer map. The search tool deduplicates and validates the proposed components and then executes that same set in every condition.

Table 6: Grounded success (%) pooled across the two full-corpus agents. Each repository contributes 36 paired model–task outcomes per arm (18 tasks per agent).

Repository	Raw	SearchProof
Express	97.2	100.0
FastAPI	80.6	100.0
Flask	100.0	100.0
Gin	91.7	100.0
Pytest	91.7	100.0
Requests	97.2	100.0
Spring Petclinic	97.2	100.0
Tokio	86.1	100.0

Table 7: The 7.3-point headline effect remains one-directional after aggregation at three dependence units. Query-group and repository rows pool both model endpoints; sign tests discard tied clusters. The mechanism contrast is SearchProof versus the same receipt with Δ withheld.

Contrast	Unit	Clusters	$+/-/=$	95% CI (points)	Sign p
Headline	Session	72	20–0/52	[4.5, 10.1]	.000002
Headline	Query group	48	16–0/32	[4.2, 10.8]	.000031
Headline	Repository	8	7–0/1	[3.1, 11.8]	.015625
Mechanism	Session	36	6–0/30	[1.4, 9.0]	.031250
Mechanism	Query group	48	7–0/41	[2.1, 8.3]	.015625
Mechanism	Repository	8	4–0/4	[1.4, 9.0]	.125000

B Replication by repository

The pooled paired change is 7.3 points with a repository-clustered 95% CI of [3.1, 11.8]. Each repository contributes 36 outcomes per arm and every query belongs to a three-way matched group, so the table exposes both scale and concentration of the remaining execution errors.

C Dependence-aware paired sensitivity

All nonzero cluster effects favor SearchProof. The headline comparison is positive in 20–0 nonzero session pairs, 16–0 nonzero query groups, and 7–0 nonzero repositories. The completion-field mechanism is positive in 6–0, 7–0, and 4–0 nonzero clusters at the same three units. The repository sign test for the narrower mechanism has only four nonzero clusters; we therefore use that row as a concentration diagnostic rather than as a standalone significance claim.

D Deterministic completion policy

The source-backed deterministic policy scans the exact checked component sets and settles every task with 0 follow-up language-model decisions, versus 72 four-task decision sessions for the two agents. It never reads the frozen target location when choosing an answer. This baseline separates the receipt's executable content from model adherence: agent-directed execution is already perfect in grounded success, while direct execution makes every completion exact and removes the follow-up decision.

E Interactive protocol audit

The protocol-enforced ablation grid contains 96 sessions and 384 model–task outcomes. Every session has exactly one successful `start`, one successful batched `search`, and one successful `commit`. The full-corpus confirmation and completion-field ablation add 180 protocol-compliant sessions; 2 all-complete batches correctly commit without an unnecessary follow-up. Invalid or repeated requests are rejected without additional execution and retained in the audit. No workspace,

Table 8: SearchProof supports both agent-directed and deterministic execution. The controller uses the returned hits, not the task label, to choose PRESENT or ABSENT.

Policy	Units	Grounded	Exact Δ	Added files (mean)	LM decisions
Two SearchProof agents	288 model–tasks	288/288	99.0%	124.7	72 sessions
Deterministic execute-Δ	144 tasks	144/144	144/144	124.6	0

shell, web, or auxiliary agent tool is available. Across arms, task IDs, within-batch order, and the one-search budget are identical.

The released provider-free audit is independently runnable over the sanitized records. It reconstructs the run manifest, terminal start/search/commit chronology, per-batch task maps, committed answers, and rejected calls for all 180 full-corpus sessions. All 180 pass; the audit retains 8 rejected nonexecuting evaluation calls and finds 0 auxiliary tool calls. Thus checking the session protocol does not require the excluded provider trace.

The third-model stress replication contains 24 additional sessions over the same frozen tool contract. 22/24 are fully regular. One Raw session names an out-of-manifest component and receives no successful search; one SearchProof session commits a misspelled task ID, leaving the intended answer missing. Both remain failures in the primary result. A matched sensitivity removes both affected batches from both arms and still moves 16/40 Raw successes to 34/40 SearchProof successes (18–0, exact sign $p = .000008$). Two other rejected Raw requests are followed by one valid search and remain protocol-compliant.

F Outcome accounting and exploratory failure

The 1,872 analyzed outcomes comprise 576 controlled, 384 primary interactive, 96 third-model stress, 96 query-only, 432 full-corpus DeepSeek outcomes (three conditions), and 288 full-corpus GLM-5.3 outcomes (two conditions). An earlier Qwen3.7-Plus controlled pilot produced 288 responses with zero schema parse coverage; it was excluded before comparative analysis and is not part of the reported 1,872. Its aggregate failure record is retained in the supplement.

G Corpus validity checks

Every query spelling appears in one task of each case type. Opaque hashed IDs replace repository and type-bearing identifiers, and absent queries must be absent as source tokens throughout the target repository. All 144 deterministic construction checks pass. GNU ctags plus Python AST independently confirms all 96 absent cases and 45/48 positive definitions. The three remaining positives are a Java controller method, a Rust unsafe trait, and a Rust async method that appear at the recorded source lines but are outside this validator's tag coverage. The query-only control, prompts, permutation procedure, and task-level outputs are included in the artifact.

H Actual-source execution

The Round 7 executor reads and parses only the files named by each component set. The source-backed deterministic controller settles 144/144 tasks without consulting frozen target locations. Across 720 saved full-corpus model–task scopes, actual hit/no-hit outcomes agree with the frozen backend at 100.0%. Table 9 reports measured work.

Table 9: Direct source execution on saved SearchProof scopes. Reductions compare each scope with a measured full-universe restart for the same task. Bytes are MB; latency is the median wall-clock scan time in ms on this machine.

Scope	n	Files	MB	ms	File ↓	Byte ↓
All SearchProof	288	124.7	0.79	3.44	62.0%	60.6%
Follow-up required	192	187.0	1.19	6.04	43.0%	40.8%
Full restart	288	328.1	2.00	8.15	—	—