ANTMAN: ADAPTIVE NEED TRACKING FOR MULTI-AGENT NAVIGATION IN LARGE INFORMATION SPACES

Anonymous authors
Paper under double-blind review

ABSTRACT

Information-seeking agents increasingly operate over information spaces that are too large to process exhaustively. Yet many multi-agent systems organize computation around static partitions of the available space, causing coordination to grow with how information is segmented rather than with what the query still requires. We introduce ANTMAN, an adaptive coordination framework that treats evolving unresolved information needs as the unit of runtime coordination. ANTMAN maintains a revisable Need Graph that tracks unresolved requirements, accumulated evidence, prior attempts, and search progress, and uses this state to control worker selection, routing, and task-local recovery as new evidence is discovered. By separating the coordination policy from substrate-specific search interfaces, the same need-conditioned mechanism can operate across different information spaces. Experiments across multi-document question answering, controlled long-context scaling, and realistic structured navigation show that ANTMAN remains effective across settings, including when execution is delegated to substantially smaller worker models. Under a $16\times$ increase in searchable context, ANTMAN increases active coordination by only $1.23\times$, compared with more than $15\times$ for partition-driven baselines, while preserving strong answer quality.

1 INTRODUCTION

Modern information-seeking agents increasingly operate over large, heterogeneous information spaces, gathering and reasoning over evidence across long interaction trajectories (Xi et al., 2026; Yao et al., 2026; Lee et al., 2026). Yet access to more information does not necessarily translate into more effective use of it. Performance can degrade as context grows even when relevant evidence is successfully identified, and can remain sensitive to where that evidence appears within the context (Du et al., 2025; Liu et al., 2024a). This raises a natural question: **How can agents search increasingly large information spaces without letting coordination grow with the space itself?**

Distributing information processing across multiple agents can reduce the context burden on any individual worker. Existing long-context multi-agent systems often follow this strategy by assigning different input partitions to different workers (Zhang et al., 2024). For example, LONGAGENT partitions the input into fixed-size chunks and assigns each chunk to a separate member agent (Zhao et al., 2024). This partition-based design reduces the context handled by any individual worker, but it also makes the size of the agent population grow with the number of partitions in the underlying information space. This coupling exposes a mismatch between the organization of the information space and the amount of coordination a query actually requires. When the underlying information need remains fixed, expanding the available context should not by itself require a proportionally larger agent population. Yet static partition-based designs instantiate workers according to how the space is segmented, causing coordination to grow mechanically with context size rather than query demand.As Figure 1 shows, under a $16\times$ increase in information-space size, ANTMAN's active coordination grows by only $1.23\times$, compared with more than $15\times$ for partition-driven baselines, resulting in substantially slower growth in model calls and inference cost. This motivates the question of what runtime state should determine the scope and direction of coordination. We organize coordination around the information requirements that remain unresolved as search progresses.

The relevant coordination state is itself dynamic. Classic work on information seeking has long argued that search is not simply a process of refining a fixed query; newly encountered information

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

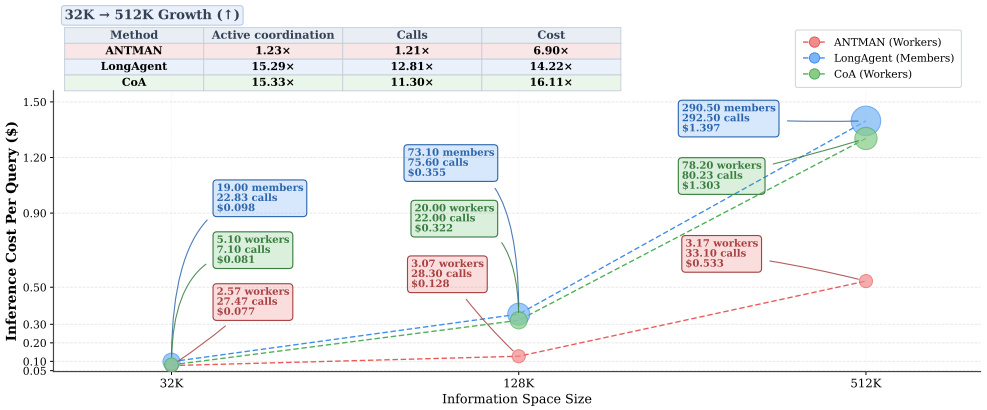


Figure 1: **ANTMAN decouples coordination from information-space growth.** Line height shows inference cost, bubble area shows active coordination, and annotations report model calls. ANTMAN maintains nearly constant coordination as the information space grows.

can change the searcher's understanding of the problem and redirect the search itself (Bates, 1989). For an information-seeking agent, new requirements may emerge only after intermediate evidence is discovered, previously identified needs may become resolved or require refinement, and unproductive search directions may need to be abandoned. Effective coordination therefore requires more than an initial decomposition of the query. It requires an explicit, revisable representation of what remains unresolved as evidence accumulates.

To address this challenge, we introduce ANTMAN, an adaptive coordination framework that treats evolving unresolved information needs, rather than input partitions, as the unit of runtime coordination. ANTMAN maintains a revisable *Need Graph* as a runtime control state that tracks unresolved requirements, accumulated evidence, prior attempts, and search progress. The current Need Graph determines which needs should be pursued and which workers should become active; as new evidence arrives, the graph can resolve, refine, or introduce needs and redirect subsequent search. When progress stalls, ANTMAN invokes task-local recovery by reframing or rerouting unresolved needs, or by falling back to an alternative resolution path. By separating the coordination policy from substrate-specific search interfaces, the same need-conditioned coordination mechanism can operate across different information spaces.

We evaluate ANTMAN across three settings that test complementary consequences of this design: (1) multi-document question answering, which tests whether need-conditioned coordination preserves answer quality on standard information-seeking tasks; (2) controlled information-space scaling, which tests whether active coordination remains decoupled from irrelevant growth in the searchable space; and (3) realistic structured navigation, which tests whether the same coordination abstraction transfers beyond flat long-context inputs. Our main contributions are:

- We introduce a revisable Need Graph as the runtime control state for multi-agent information seeking, allowing evolving unresolved requirements to govern worker activation, routing, and task-local recovery. Replacing this explicit state with graph-free adaptive replanning reduces performance by **23.82%** at 512K and **16.67%** on GAIA, demonstrating that the benefit extends beyond generic adaptive coordination.

- We show that need-conditioned coordination transfers across distinct information substrates without substrate-specific redesign. ANTMAN outperforms the strongest non-benchmark-specific baselines by **15.3%** on RepoProbe, **18.4%** on SWE-QA-Pro, and **27.9%** on GAIA. Also, the same coordination mechanism remains effective when execution is delegated to substantially smaller **8B worker models**. With Qwen3-8B for all non-orchestrator components, ANTMAN-H retains **89.2%**, **93.5%**, and **98.3%** of full ANTMAN's performance on RepoProbe, SWE-QA-Pro, and GAIA, respectively.

- We show that selective coordination preserves strong answer quality while substantially reducing sensitivity to evidence position. Across Early, Middle, and Late placements, ANTMAN has a middle-position gap of only $+.015$, compared with $-.181$ for direct full-context inference.

Table 1: **Comparison of representative information-seeking and coordination approaches.** ✓ denotes an explicit capability, △ a partial one, and "–" a non-central property.

Method	Large / Structured Information Space	Adaptive / Iterative Information Access	Multi-Agent Collaboration	Explicit Evolving Need / Gap State	Need-Driven Coordination Scope	Space-Decoupled Coordination
Adaptive Information Seeking						
Dense Retrieval (DPR)† (Karpukhin et al., 2020) EMNLP'20	✓	–	–	–	–	–
ChainRAG† (Zhu et al., 2025) ACL'25	–	✓	–	△	△	–
ReAct† (Yao et al., 2023) ICLR'23	△	✓	–	–	–	–
ReAct + RepoGraph† (Ouyang et al., 2025) ICLR'25	✓	✓	–	–	–	–
RepoDistill† (Yin et al., 2026) Findings ACL'26	✓	△	–	–	–	–
DRAGIN (Su et al., 2024) ACL'24	–	✓	–	△	△	–
KiRAG (Fang et al., 2025) ACL'25	–	✓	–	△	△	–
SelfRACG (Dong et al., 2025) EMNLP'25	–	✓	–	△	△	–
S2G-RAG (Li et al., 2026) ACL'26	–	✓	–	✓	△	–
Multi-Agent Navigation and Coordination						
LongAgent† (Zhao et al., 2024) EMNLP'24	✓	△	✓	–	–	–
CoA† (Zhang et al., 2024) NeurIPS'24	✓	△	✓	–	–	–
OWL† (Hu et al., 2025) NeurIPS'25	△	✓	✓	△	△	–
C-3PO (Chen et al., 2025) ICML'25	–	✓	✓	△	–	–
MAIN-RAG (Chang et al., 2025) ACL'25	–	△	✓	–	–	–
DyLAN (Liu et al., 2024b) COLM'24	–	–	✓	–	–	–
Ours						
ANTMAN	✓	✓	✓	✓	✓	✓

† denotes methods used in our experiments. Need-driven coordination means unresolved needs control the scope of active computation, not only the next retrieval step. Space-decoupled coordination refers to whether active worker participation is structurally tied to the number of available information-space partitions.

2 Background and Motivation

Information-seeking agents must decide not only how to reason over evidence, but also what information to access and how much computation to devote to finding it. Prior work addresses different parts of this problem through iterative retrieval, structured navigation, and multi-agent coordination. We review these directions before motivating ANTMAN's use of unresolved information needs as the runtime state for controlling search and coordination.

2.1 Adaptive Information Seeking

Retrieval methods reduce a larger corpus to a small set of candidate evidence. DPR (Karpukhin et al., 2020) provides a standard dense retrieval baseline, while more recent methods make information access iterative. ChainRAG progressively retrieves and rewrites across reasoning steps (Zhu et al., 2025), and ReAct interleaves reasoning with environment actions (Yao et al., 2023). For repository-level settings, RepoGraph exposes structural relations among code entities (Ouyang et al., 2025), while RepoDistill combines repository retrieval with learned context-budget allocation and compression (Yin et al., 2026). Recent systems further adapt retrieval according to what becomes necessary during execution. Several retrieval methods adapt information access according to evolving information needs or gaps during execution (Su et al., 2024; Fang et al., 2025; Dong et al., 2025). Most directly, S2G-RAG judges whether accumulated evidence is sufficient and, when it is not, generates structured gap items that become the next retrieval query (Li et al., 2026). These methods show that information access can adapt as evidence accumulates and that explicit needs or gaps can guide what information should be retrieved next. ANTMAN builds on this idea but uses evolving unresolved needs as a runtime control state for coordination. The same state governs not only what information should be sought next, but also which workers should become active and how search effort should be allocated.

2.2 Multi-Agent Navigation and Coordination

Multi-agent systems distribute information processing across workers. LongAgent partitions long inputs among member agents (Zhao et al., 2024), while Chain of Agents (CoA) processes segmented long contexts through a sequence of collaborating workers (Zhang et al., 2024). In both cases, the coordination footprint is closely tied to the partitioning of the available input. Other systems instead organize agents around specialized functions. C-3PO and MAIN-RAG use multiple agents for retrieval and evidence processing (Chen et al., 2025; Chang et al., 2025). OWL's Workforce combines hierarchical planning with coordinated specialized tool-using workers and failure-triggered replanning (Hu et al., 2025). Multi-agent organization can also adapt during execution through changes in team composition, communication, or routing based on task and runtime signals (Liu et al., 2024b; Wang et al., 2025b;a; Xiao et al., 2026). Dynamic multi-agent coordination is not itself the contribution

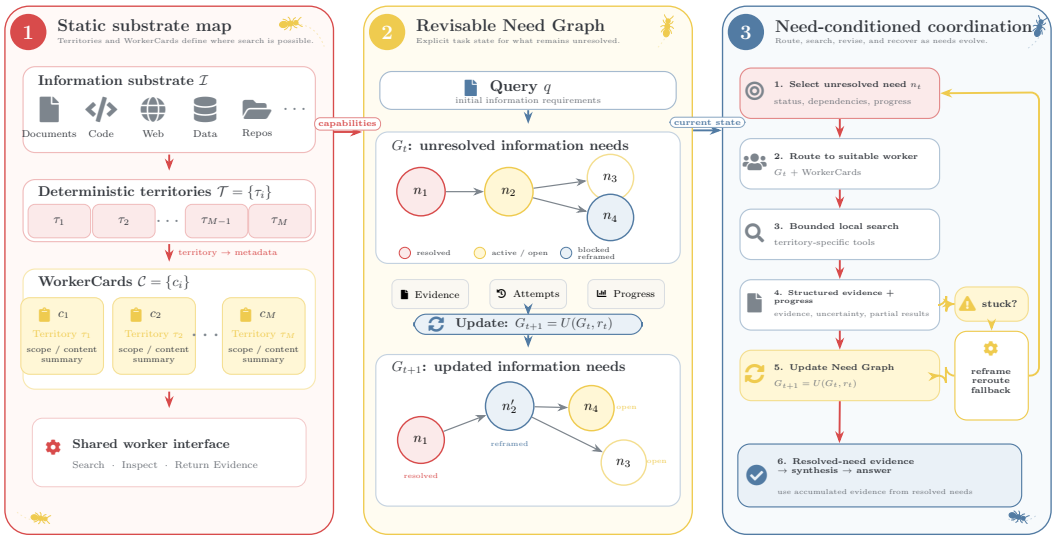


Figure 2: **Overview of ANTMAN. (1) Static substrate map:** the information substrate is partitioned into territories, each summarized by a WorkerCard describing its scope; workers operate through a shared substrate-specific interface. **(2) Revisable Need Graph:** unresolved information needs explicitly track evidence, prior attempts, and progress, and may be revised as new evidence arrives. **(3) Need-conditioned coordination:** the current Need Graph and WorkerCards determine worker routing; stalled needs can trigger reframing, rerouting, or fallback before final synthesis.

of ANTMAN. The distinction instead lies in what runtime state governs this adaptation. Prior work emphasizes either adaptive information seeking, which changes what information to access, or adaptive multi-agent coordination, which changes how computation is allocated. ANTMAN connects these two forms of adaptation through a revisable representation of unresolved information needs. Changes in what remains unknown can therefore alter information access, worker activation, routing, and recovery during execution. Table 1 summarizes this distinction.

3 Method

ANTMAN maintains a revisable representation of unresolved information needs and coordinates workers as those needs evolve. Figure 2 summarizes the workflow: **(1)** a static substrate map, **(2)** a runtime Need Graph, and **(3)** need-conditioned worker coordination.

3.1 Problem Formulation

We study information-seeking tasks in which answering a query q requires locating and integrating evidence from a potentially large information space $\mathcal{I}$. The space may correspond to a document collection, long-context corpus, software repository, web environment, or another tool-accessible substrate. The central difficulty is that the amount of information available in $\mathcal{I}$ can grow substantially while the information required by q remains comparatively small. A coordination strategy that expands workers with the size of $\mathcal{I}$ therefore couples computation to available information rather than to the requirements of the query. ANTMAN instead makes evolving unresolved information needs the runtime control state for coordination, allowing worker activation and search decisions to adapt as those needs change.

3.2 Static Substrate Map

Figure 2(1) describes *where* ANTMAN can search. The information space $\mathcal{I}$ is deterministically partitioned into M territories,

$$\mathcal{T} = P(\mathcal{I}) = \{\tau_1, \ldots, \tau_M\}, \qquad c_i = \mathrm{Card}(\tau_i), \qquad \mathcal{C} = \{c_1, \ldots, c_M\}, \tag{1}$$

where τ_i is a bounded searchable region, P is the substrate-specific partitioning procedure, c_i is the WorkerCard associated with τ_i, and $\mathcal{C}$ is the complete WorkerCard set. A WorkerCard is lightweight territory metadata which summarizes the scope and content represented by its assigned territory. Worker specialization is therefore primarily *scope-specialized*. ANTMAN does not require different workers to use different model architectures or tool sets. Within a substrate, workers may share the same interface for searching, inspecting, and returning evidence.

The icon strip in Figure 2(1) illustrates that the same abstraction can be instantiated over different substrates; it does not define a fixed set of worker personas. Importantly, Equation 1 defines the workers that are available, not the workers that must participate in every query.

3.3 Revisable Need Graph

Figure 2(2) describes *what* remains unresolved. Given query q, ANTMAN initializes a Need Graph G_0 and maintains its runtime state as

$$G_t = (N_t, \Delta_t), \qquad z_t(n) = \big(s_t(n), \mathcal{E}_t(n), \mathcal{H}_t(n), p_t(n)\big), \tag{2}$$

where N_t is the set of information-need nodes at step t and Δ_t contains dependencies between them. For a need $n \in N_t$, $s_t(n)$ denotes its resolution status, $\mathcal{E}_t(n)$ its accumulated evidence, $\mathcal{H}_t(n)$ its previous attempt history, and $p_t(n)$ its current progress state. These quantities correspond to the evidence, attempts, and progress signals shown in Figure 2(2).

Unlike a fixed decomposition, G_t is revised during execution. After a worker returns a structured report r_t, the coordinator updates the graph as $G_{t+1} = U(G_t, r_t)$, where U is the task-local graph-update operator. An update may resolve a need, preserve it as unresolved, reframe an unsuccessful need into n', or introduce additional dependencies or requirements revealed by newly collected evidence. The non-linear graph structure in Figure 2(2) reflects that information requirements may branch or depend on one another rather than forming a fixed sequential plan.

3.4 Need-Conditioned Coordination and Recovery

Figure 2(3) shows how the current graph controls computation. At coordination step t, ANTMAN selects an unresolved need n_t, routes it using the current Need Graph and WorkerCards, and invokes the corresponding territory-backed worker:

$$n_t = \mathrm{Select}(G_t), \qquad i_t = \mathrm{Route}(n_t, G_t, \mathcal{C}), \qquad r_t = \mathrm{Exec}(w_{i_t}, n_t, \tau_{i_t}). \tag{3}$$

where $i_t \in \{1, \ldots, M\}$ identifies the selected worker, w_{i_t} is the worker associated with territory τ_{i_t}, and r_t is its structured report containing evidence, progress, and remaining uncertainty. The report is then fed back, closing the runtime loop. Workers perform bounded local information seeking rather than reconstructing the entire global task. Global state remains in G_t, while a worker searches only within the scope assigned to the selected need. If repeated attempts fail to make progress, ANTMAN performs task-local recovery by reframing the need, rerouting it to another territory, or invoking a fallback resolution path. Worker definitions and the static substrate map remain unchanged. Once the required needs are resolved, evidence attached to those needs is used for final synthesis. The primary implementation prompts are provided in Appendix E.

Need-conditioned coordination and space decoupling. Our design separates the number of *available* territories from the amount of *active* coordination. Let M denote the total number of territories, $H(q)$ the number of information needs realized during execution, and ρ the maximum number of routing attempts per need. If T_q denotes the number of coordination steps for query q, then each step activates at most one territory-backed worker, so $|\mathcal{A}(q)| \le T_q$. Since each realized need can be routed at most ρ times, $T_q \le \rho H(q)$. Independently, $|\mathcal{A}(q)| \le M$, since no more than the M available territories can become active. These two constraints give the bound summarized in Box 1. When realized information demand remains bounded, enlarging the information space can increase M without forcing active coordination to grow with it. Appendix A formalizes this property, while Section 4.2.1 tests it by varying space size at approximately fixed information demand.

Box 1: Space-Decoupling Bound $\quad |\mathcal{A}(q)| \le \min\{M, \rho H(q)\}\,.$

Table 2: **Performance on standard multi-document QA.** We report EM and token-level F1 over 30 questions per benchmark, with macro averages across the three benchmarks.

Method	HotpotQA		2Wiki		MuSiQue		Avg.	
	EM	F1	EM	F1	EM	F1	EM	F1
Retrieval and RAG Baselines								
Iterative Sparse Retrieval (BM25)	.533	.684	.367	.515	.433	.582	.444	.594
Dense Retrieval (DPR) (Karpukhin et al., 2020) EMNLP'20	.533	.696	.267	.379	.200	.292	.333	.456
ChainRAG (Zhu et al., 2025) ACL'25	.567	.711	.733	.791	.500	.654	.600	.719
S2G-RAG (Li et al., 2026) ACL'26	.667	.810	**.800**	**.858**	.633	.730	.700	.799
Agentic and Multi-Agent Baselines								
ReAct (Yao et al., 2023) ICLR'23	.533	.734	.567	.781	.467	.626	.522	.714
LongAgent (Zhao et al., 2024) EMNLP'24	.433	.655	.467	.697	.400	.494	.433	.616
CoA (Zhang et al., 2024) NeurIPS'24	.567	.769	.600	.778	.533	.682	.567	.743
Ours								
ANTMAN-H†	**.700**	**.842**	.700	.796	.467	.651	.622	.763
ANTMAN	**.700**	**.842**	.767	.823	**.667**	**.777**	**.711**	**.814**

Best results are **bold**; second-best results are underlined; ties share formatting. † ANTMAN-H uses Qwen3-8B for all non-orchestrator components while retaining the same orchestrator as full ANTMAN.

4 Experiments

4.1 Preliminary Evaluation on Standard Multi-Document QA

Before studying how ANTMAN behaves as information spaces grow, we first evaluate its effectiveness in standard multi-document QA. We consider HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), and MuSiQue (Trivedi et al., 2022), which cover complementary forms of cross-document and multi-hop reasoning. We evaluate 30 questions per benchmark across nine methods, yielding 810 predictions scored with the same frozen clean-answer extraction and evaluation pipeline.

Model setting. All baselines and ANTMAN use GPT-4.1 (OpenAI, 2025) throughout; ANTMAN-H retains the GPT-4.1 orchestrator but uses Qwen3-8B (Team, 2025) for all non-orchestrator components.

Results. ANTMAN achieves the strongest aggregate performance despite being designed for large information spaces, remaining competitive with specialized retrieval systems such as S2G-RAG. It performs best on HotpotQA and MuSiQue and remains competitive on 2WikiMultiHopQA.

Smaller workers. ANTMAN-H remains competitive with all baselines except S2G-RAG despite using substantially smaller execution models. It matches full ANTMAN on HotpotQA and remains close on 2WikiMultiHopQA, with a larger gap only on the more demanding MuSiQue benchmark.

Preliminary takeaway. Need-driven coordination preserves strong multi-document QA effectiveness rather than trading answer quality for scalability, and remains effective even when execution is delegated to smaller worker models.

4.2 Scaling and Navigation in Large Information Spaces

We study how ANTMAN behaves in large and structured information spaces, asking whether need-driven coordination can remain efficient while still locating the evidence required for each query. We examine this through two research questions: **RQ1** tests whether active coordination remains decoupled from information-space size when the underlying information need is approximately fixed, while **RQ2** tests whether the same coordination principle remains effective in realistic structured environments that require navigation.

4.2.1 Controlled Information-Space Scaling

Setup. We follow the multi-needle setting of Needle-in-a-Haystack PLUS introduced by LongAgent (Zhao et al., 2024). We evaluate 10 underlying questions at Early, Middle, and Late evidence positions over 32K, 64K, 128K, and 512K contexts. For each question, the required evidence is preserved while additional filler expands the surrounding distractor space, yielding matched conditions that isolate information-space growth while keeping the underlying information need approximately fixed. We compare ANTMAN with LongAgent and CoA, two multi-agent systems whose designs most closely match our setting by distributing a large information space across multiple workers.

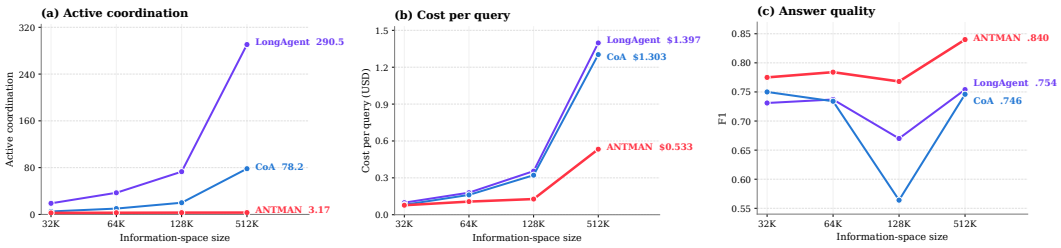


Figure 3: **Selective coordination as information space grows. (a)** Active coordination remains nearly constant for ANTMAN but grows with space size for LongAgent and CoA. **(b)** This yields slower cost growth. **(c)** Answer quality remains stable across scales.

Table 3: **Coordination under increasing information demand.** Searchable space is fixed at 512K while required evidence increases.

Required Evidence Units	Active Coordination	Calls / Query	Cost / Query	Complete Evidence	Answer Acc.
ANTMAN					
1	5.0	35.5	$0.107	100%	100%
4	4.3	43.6	$0.111	100%	100%
16	8.0	86.2	$0.293	100%	100%

Scaling behavior. Figure 3 shows the central scaling advantage of ANTMAN. When the information space grows by $16\times$, active coordination increases by only $1.23\times$, compared with over $15\times$ for LongAgent and CoA. This gap carries over to model calls and inference cost, indicating that ANTMAN avoids expanding coordination simply because more information is available.

This selectivity does not reduce answer quality. Importantly, ANTMAN remains stable as the information space grows because it activates workers according to unresolved needs rather than the size of the partitioned space. In contrast, LongAgent and CoA expand coordination as the underlying space is divided into more partitions. These results support the intended design principle that coordination should grow with query demand rather than information-space size. Full scaling results are reported in Appendix B.

4.2.2 Controlled Information-Demand Scaling

We complement the space-scaling experiment with the reverse intervention, holding the searchable space fixed at 512K while increasing required evidence from 1 to 4 to 16 units. Matched contexts preserve the surrounding distractors, evidence locations, worker organization, and execution setting.

Demand sensitivity. Table 3 shows that increasing required evidence from 1 to 16 units raises active coordination from 5.0 to 8.0 workers and model calls from 35.5 to 86.2 per query. Despite the higher demand, ANTMAN recovers all required evidence and answers all evaluated queries correctly. Additional construction details and trajectory diagnostics are provided in Appendix F.2.

Answer to RQ1. Coordination follows need, not space: ANTMAN keeps its coordination footprint nearly constant as the searchable space grows, while allocating substantially more computation when information demand increases at fixed space. Across both interventions, answer quality remains stable.

4.2.3 Navigation in Realistic Structured Information Spaces

We evaluate on two repository-level benchmarks and one general information-seeking benchmark. RepoProbe-Python (Yang et al., 2026) contains 108 questions across eight large Python repositories, while our frozen 80-question subset of SWE-QA-Pro (Cai et al., 2026) requires multi-file, agentic codebase exploration. GAIA (Mialon et al., 2024) extends the evaluation to heterogeneous tool-using information seeking; we use the fixed text-only GAIA-Text-103 subset. Within each repository benchmark, methods evaluated in our harness use the same frozen question set and model setting, while all methods evaluated on GAIA share the same tool interface (Appendix F.1). We additionally

Table 4: **Performance in realistic structured information spaces.** GAIA-Text-103 is broken down by difficulty level. Raw repository scores are reported in Appendix C.

Method	RepoProbe	SWE-QA-Pro	GAIA-Text-103			
			L1	L2	L3	Overall
Retrieval and Iterative Retrieval Baselines						
Direct	31.08	55.74	28.2	17.3	8.3	20.4
Iterative Sparse Retrieval	33.15	60.90	53.8	42.3	33.3	45.6
Dense Retrieval (Karpukhin et al., 2020)	31.51	58.30	43.6	26.9	25.0	33.0
S2G-RAG (Li et al., 2026) ACL'26	22.87	60.40	25.6	19.2	0.0	19.4
General Tool-Using Agents						
ReAct (Yao et al., 2023) ICLR'23	25.86	64.92	46.2	30.8	8.3	34.0
OWL (Hu et al., 2025) NeurIPS'25	–	–	53.8	38.5	8.3	40.8
Repository-Specific Agents						
ReAct + RepoGraph (Ouyang et al., 2025) ICLR'25	25.71	67.38	–	–	–	–
RepoDistill (Yin et al., 2026) Findings ACL'26	28.27	65.10	–	–	–	–
Benchmark-Specific Reference						
SWE-QA-Pro Agent (Cai et al., 2026)	–	77.58	–	–	–	–
Ours						
ANTMAN-H†	34.10	74.60	**69.2**	**55.8**	25.0	57.3
ANTMAN	**38.21**	**79.78**	**69.2**	53.8	**41.7**	**58.3**

Higher is better. † ANTMAN-H uses Qwen3-8B for all non-orchestrator components while retaining the same orchestrator as ANTMAN. "–" denotes settings outside the method-specific evaluation scope.

Table 5: **Ablation of runtime adaptive coordination.** Parentheses indicate relative quality drops from full ANTMAN.

Variant	512K F1	SWE-QA-Pro Score	GAIA Accuracy
Adaptive Coordination Ablations			
Static ANTMAN	40.32 ↓ 52.02%	68.58 ↓ 15.78%	50.00 ↓ 16.67%
Graph-free Adaptive	64.01 ↓ 23.82%	78.88 ↓ 3.13%	50.00 ↓ 16.67%
w/o Need Revision	74.44 ↓ 11.41%	75.32 ↓ 7.50%	42.50 ↓ 29.17%
w/o Adaptive Rerouting	82.20 ↓ 2.18%	72.13 ↓ 11.42%	60.00 (↔ 0.00%)
w/o Recovery	60.12 ↓ 28.45%	76.52 ↓ 6.03%	52.50 ↓ 12.50%
Ours			
Full ANTMAN	**84.03**	**81.43**	**60.00**

include OWL (Hu et al., 2025) as a general multi-agent tool-using baseline on GAIA; repository-specific systems are evaluated on the corresponding code-navigation benchmarks.

Structured navigation. Table 4 shows that the same need-driven coordination principle remains effective when evidence must be discovered through structured navigation. ANTMAN uses evolving unresolved needs to determine where search should continue and, in multi-worker substrates, which workers should participate. Its strong performance across both repository benchmarks shows that this coordination abstraction transfers beyond the controlled long-context setting. The clearest separation from competing methods appears on GAIA-Text-103, where ANTMAN operates over a distinct web-and-tool substrate. This provides further evidence that its effectiveness is not tied to repository-specific navigation machinery. Meanwhile, ANTMAN-H remains close to full ANTMAN, with the main degradation appearing on more difficult tasks. Most of the GAIA gap comes from Level 3, suggesting that full ANTMAN is most beneficial on harder cases, while ANTMAN-H achieves similar performance on less demanding tasks.

Answer to RQ2. Need-driven coordination transfers to realistic structured environments, allowing ANTMAN to selectively navigate repository and heterogeneous information spaces without requiring a substrate-specific coordination strategy.

4.3 Does the Evolving Need State Matter?

RQ3 asks whether ANTMAN benefits from using evolving unresolved needs as its runtime control state, beyond either a fixed initial plan or generic adaptive replanning. We compare full ANTMAN with Static ANTMAN, Graph-free Adaptive, and targeted ablations of Need revision, adaptive rerouting, and recovery. All variants share the same substrate organization, models, tools, and execution budget. Appendix D provides the variant definitions and complete execution statistics.

Evolving need state and adaptive coordination. Table 5 shows that full ANTMAN improves answer quality over both fixed coordination and graph-free adaptive replanning across all three settings. In controlled scaling, the Full–Graph-free gap persists despite comparable numbers of activated workers (Appendix D.2), suggesting that the benefit of explicit need tracking extends beyond simply engaging more workers. The targeted ablations further reveal complementary mechanisms. Recovery is particularly consequential in controlled scaling, while adaptive rerouting plays an important role in structured repository navigation. As expected, rerouting has no effect on GAIA, which uses a single territory-backed worker.

Answer to RQ3. Explicitly tracking evolving unresolved needs improves answer quality beyond fixed coordination and graph-free adaptation, with different runtime mechanisms contributing across information environments.

4.4 Position Robustness under Selective Coordination

Table 6: **Robustness to evidence position as the information space grows.** We report F1 when required evidence appears at Early (E), Middle (M), or Late (L) positions. The final column reports $M - \frac{1}{2}(E + L)$ after averaging each position across context lengths; negative values indicate a middle-position penalty.

Method	32K			64K			128K			Middle Gap
	E	**M**	L	E	**M**	L	E	**M**	L	
Direct Long-Context Reference										
Full Context	.911	.590	.813	.767	.667	.733	.733	.613	.870	−.181
Ours										
ANTMAN	.741	.783	.800	.747	.826	.780	.743	.748	.813	**+.015**

Position results use the same paired questions as the controlled scaling experiment. Full Context serves as the direct positional-sensitivity reference.

We further examine whether ANTMAN remains robust to where relevant evidence appears within the information space. Using the same paired Early, Middle, and Late conditions from Section 4.2.1, we compare ANTMAN with direct full-context inference. Table 6 shows that ANTMAN does not exhibit the strong positional degradation observed in direct full-context inference. While Full Context exhibits the classical "lost in the middle" pattern (Liu et al., 2024a), ANTMAN remains stable across Early, Middle, and Late evidence placements. This is consistent with ANTMAN's selective search design, which narrows the context considered for each unresolved need and thereby reduces sensitivity to the evidence's position in the global information space. Thus, ANTMAN retains positional robustness despite activating only a small fraction of the available workers.

Position robustness. Selective coordination substantially reduces sensitivity to evidence position, allowing ANTMAN to remain robust across Early, Middle, and Late placements.

5 Conclusion

We introduced ANTMAN, a multi-agent information-seeking framework that treats evolving unresolved information needs as the runtime control state for coordination. Rather than organizing computation around static partitions, ANTMAN maintains and revises a Need Graph that tracks what remains unresolved and guides information access, worker activation, routing, and task-local recovery. Across multi-document QA, controlled information-space scaling, and realistic structured navigation, ANTMAN maintains strong answer quality while selectively allocating coordination according to evolving information needs. In particular, active coordination remains nearly constant as the surrounding information space grows, while the same need-conditioned mechanism remains effective across different information-seeking settings and with substantially smaller worker models. Beyond these results, our findings suggest that coordination need not mirror the structure of the information space. By making unresolved needs the basis for runtime control, ANTMAN separates available search capacity from the computation activated for a query. Looking forward, we aim to extend ANTMAN to richer information substrates, develop more transferable representations of information needs, and refine policies for revising and routing those needs as search progresses.

AI USE STATEMENT

In this work, we used generative AI tools for language editing and code assistance. All AI-assisted content was reviewed and verified by the authors. We take responsibility for the final content of this work, including text, claims, experiments, code, and other artifacts.

REFERENCES

Marcia J. Bates. The design of browsing and berrypicking techniques for the online search interface. *Online Review*, 13(5):407–424, 05 1989. ISSN 0309-314X. doi: 10.1108/eb024320. URL https://doi.org/10.1108/eb024320.

Songcheng Cai, Zhiheng Lyu, Yuansheng Ni, Xiangchao Chen, Baichuan Zhou, Shenzhe Zhu, Yi Lu, Haozhe Wang, Chi Ruan, Benjamin Schneider, Weixu Zhang, Xiang Li, Andy Zheng, Yuyu Zhang, Ping Nie, and Wenhu Chen. SWE-QA-pro: A representative benchmark and scalable training recipe for repository-level code understanding. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 16958–16999, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.837. URL https://aclanthology.org/2026.findings-acl.837/.

Chia-Yuan Chang, Zhimeng Jiang, Vineeth Rakesh, Menghai Pan, Chin-Chia Michael Yeh, Guanchu Wang, Mingzhi Hu, Zhichao Xu, Yan Zheng, Mahashweta Das, and Na Zou. MAIN-RAG: Multi-agent filtering retrieval-augmented generation. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2607–2622, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.131. URL https://aclanthology.org/2025.acl-long.131/.

Guoxin Chen, Minpeng Liao, Peiying Yu, Dingmin Wang, Zile Qiao, Chao Yang, Xin Zhao, and Kai Fan. C-3PO: Compact plug-and-play proxy optimization to achieve human-like retrieval-augmented generation. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 8530–8551. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/chen25an.html.

Qian Dong, Jia Chen, Qingyao Ai, Hongning Wang, Haitao Li, Yiwu, Yao Hu, Yiqun Liu, and Shaoping Ma. SelfRACG: Enabling LLMs to self-express and retrieve for code generation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10694–10705, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.541. URL https://aclanthology.org/2025.emnlp-main.541/.

Yufeng Du, Minyang Tian, Srikanth Ronanki, Subendhu Rongali, Sravan Babu Bodapati, Aram Galstyan, Azton Wells, Roy Schwartz, Eliu A Huerta, and Hao Peng. Context length alone hurts LLM performance despite perfect retrieval. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 23281–23298, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.1264. URL https://aclanthology.org/2025.findings-emnlp.1264/.

Jinyuan Fang, Zaiqiao Meng, and Craig MacDonald. KiRAG: Knowledge-driven iterative retriever for enhancing retrieval-augmented generation. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 18969–18985, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.929. URL https://aclanthology.org/2025.acl-long.929/.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In Donia Scott, Nuria Bel, and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.580. URL https://aclanthology.org/2020.coling-main.580/.

Mengkang Hu, Yuhang Zhou, Wendong Fan, Yuzhou Nie, Ziyu Ye, Bowei Xia, Tao Sun, Zhaoxuan Jin, Yingru Li, Zeyu Zhang, Yifeng Wang, Qianshuo Ye, Bernard Ghanem, Ping Luo, and Guohao Li. OWL: Optimized workforce learning for general multi-agent assistance in real-world task automation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=MBJ46gd1CT.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

Jaewoo Lee, Archiki Prasad, Justin Chen, Zaid Khan, Elias Stengel-Eskin, and Mohit Bansal. PRInTS: Reward modeling for long-horizon information seeking. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 34120–34138, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1574. URL https://aclanthology.org/2026.acl-long.1574/.

Minghan Li, Junjie Zou, Xinxuan Lv, Chao Zhang, and Guodong Zhou. S2G-RAG: Structured sufficiency and gap judging for iterative retrieval-augmented QA. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 25846–25862, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1185. URL https://aclanthology.org/2026.acl-long.1185/.

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, 2024a. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. A dynamic LLM-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*, 2024b. URL https://openreview.net/forum?id=XII0Wp1XA9.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 9025–9049, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/25ae35b5b1738d80f1f03a8713e405ec-Paper-Conference.pdf.

OpenAI. Introducing gpt-4.1 in the api. https://openai.com/index/gpt-4-1/, 2025.

Siru Ouyang, Wenhao Yu, Kaixin Ma, Zilin Xiao, Zhihan Zhang, Mengzhao Jia, Jiawei Han, Hongming Zhang, and Dong Yu. Repograph: Enhancing ai software engineering with repository-level code graph. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 30098–30121, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/4a4a3c197deac042461c677219efd36c-Paper-Conference.pdf.

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. DRAGIN: Dynamic retrieval augmented generation based on the real-time information needs of large language models. In

Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12991–13013, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.702. URL https://aclanthology.org/2024.acl-long.702/.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475. URL https://aclanthology.org/2022.tacl-1.31/.

Qian Wang, Tianyu Wang, Zhenheng Tang, Qinbin Li, Nuo Chen, Jingsheng Liang, and Bingsheng He. MegaAgent: A large-scale autonomous LLM-based multi-agent system without predefined SOPs. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 4998–5036, Vienna, Austria, July 2025a. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.259. URL https://aclanthology.org/2025.findings-acl.259/.

Zhexuan Wang, Yutong Wang, Xuebo Liu, Liang Ding, Miao Zhang, Jie Liu, and Min Zhang. AgentDropout: Dynamic agent elimination for token-efficient and high-performance LLM-based multi-agent collaboration. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 24013–24035, Vienna, Austria, July 2025b. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1170. URL https://aclanthology.org/2025.acl-long.1170/.

Yunjia Xi, Jianghao Lin, Yongzhao Xiao, Zheli Zhou, Rong Shan, Te Gao, Jiachen Zhu, Weiwen Liu, Yong Yu, and Weinan Zhang. A survey of large language model-based search agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8244–8279, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.374. URL https://aclanthology.org/2026.acl-long.374/.

Yitao Xiao, Shaoyong Guo, Guoming Yang, Qingnan Wang, Yinlin Ren, Xuesong Qiu, and Qi Feng. RouterHGC: Optimized router for LLM-based multi-agent systems via heterogeneous graph contrastive learning. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 31759–31789, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.1589. URL https://aclanthology.org/2026.findings-acl.1589/.

Yuexi Yang, Alyssa Wu, Ji Luo, Richeng Xuan, Zhichao Hu, Yuhong Liu, and Zhen Qin. Repoprobe: Benchmarking architecture-aware repository comprehension with checklists. In *Proceedings of the 41st IEEE/ACM International Conference on Automated Software Engineering (ASE '26)*, New York, NY, USA, 2026. ACM.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. URL https://arxiv.org/abs/2210.03629.

Yilun Yao, Shan Huang, Elsie Dai, Zhewen Tan, Zhenyu Duan, Shousheng Jia, Yanbing Jiang, and Tong Yang. ARC: Active and reflection-driven context management for long-horizon information

seeking agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 18644–18659, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.930. URL https://aclanthology.org/2026.findings-acl.930/.

Xin Yin, Zixiang Ding, Yiang Zhang, Qiang Wang, Rui Wang, Chao Ni, and Zhe Cui. RepoDistill: Distilling repository knowledge through compression-aware budget allocation and policy optimization. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 4425–4443, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.217. URL https://aclanthology.org/2026.findings-acl.217/.

Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Sercan Ö. Arik. Chain of agents: Large language models collaborating on long-context tasks, 2024. URL https://arxiv.org/abs/2406.02818.

Jun Zhao, Can Zu, Xu Hao, Yi Lu, Wei He, Yiwen Ding, Tao Gui, Qi Zhang, and Xuanjing Huang. LONGAGENT: Achieving question answering for 128k-token-long documents through multi-agent collaboration. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 16310–16324, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.912. URL https://aclanthology.org/2024.emnlp-main.912/.

Rongzhi Zhu, Xiangyu Liu, Zequn Sun, Yiwei Wang, and Wei Hu. Mitigating lost-in-retrieval problems in retrieval augmented multi-hop question answering. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 22362–22375, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1089. URL https://aclanthology.org/2025.acl-long.1089/.

Algorithm 1 ANTMAN execution procedure. The substrate map remains fixed, while the Need Graph is revised during execution to select, route, and recover unresolved information needs.

Require: Query q, information space $\mathcal{I}$, execution budget B
Ensure: Final answer y

$\mathcal{T} \leftarrow P(\mathcal{I})$ ▷ fixed territories
$\mathcal{C} \leftarrow \{\mathrm{Card}(\tau_i)\}_{i=1}^{|\mathcal{T}|}$ ▷ static WorkerCards
$G_0 \leftarrow \mathrm{InitializeNeedGraph}(q)$
$t \leftarrow 0$
while not $\mathrm{Terminate}(G_t, B)$ **do**
  $n_t \leftarrow \mathrm{Select}(G_t)$ ▷ choose an unresolved need
  $i_t \leftarrow \mathrm{Route}(n_t, G_t, \mathcal{C})$ ▷ select territory-backed worker
  $r_t \leftarrow \mathrm{Exec}(w_{i_t}, n_t, \tau_{i_t})$ ▷ bounded local information seeking
  $G_{t+1} \leftarrow U(G_t, r_t)$ ▷ update evidence, status, and dependencies
  if $\mathrm{Stuck}(n_t, G_{t+1})$ **then**
    $a_t \leftarrow \mathrm{Recover}(n_t, G_{t+1}, \mathcal{C})$
    if $a_t = \mathrm{Reframe}$ **then**
      $G_{t+1} \leftarrow \mathrm{ReframeNeed}(n_t, G_{t+1})$
    else if $a_t = \mathrm{Reroute}$ **then**
      $\mathrm{MarkForRerouting}(n_t, G_{t+1})$
    else if $a_t = \mathrm{Fallback}$ **then**
      $G_{t+1} \leftarrow \mathrm{Fallback}(n_t, G_{t+1})$
    end if
  end if
  $t \leftarrow t + 1$
end while
$\mathcal{E}^* \leftarrow \mathrm{SelectResolvedEvidence}(G_t)$
$y \leftarrow \mathrm{Synthesize}(q, \mathcal{E}^*)$
return y

A ANTMAN EXECUTION AND SPACE-DECOUPLED COORDINATION

This appendix provides the complete execution procedure for ANTMAN and formalizes why its active coordination is governed by runtime information needs rather than directly by the size of the underlying information space.

A.1 FULL EXECUTION PROCEDURE

Let $\mathcal{I}$ denote the information space associated with query q. A substrate-specific partitioning procedure P produces a fixed set of M territories

$$\mathcal{T} = P(\mathcal{I}) = \{\tau_1, \ldots, \tau_M\}, \tag{4}$$

with corresponding WorkerCards

$$\mathcal{C} = \{c_1, \ldots, c_M\}, \qquad c_i = \mathrm{Card}(\tau_i). \tag{5}$$

Each WorkerCard summarizes the scope of its associated territory. The territory map and WorkerCards remain fixed during execution. Runtime adaptation occurs only through the Need Graph and the resulting routing decisions.

Given query q, the coordinator initializes a Need Graph $G_0 = (N_0, E_0)$. Each node $n \in N_t$ represents an information need and maintains task-local state including resolution status, accumulated evidence, attempt history, and progress. Algorithm 1 gives the complete execution loop.

The key architectural separation in Algorithm 1 is between the static candidate set $\mathcal{C}$ and the runtime state G_t. Increasing the information-space size may increase $|\mathcal{C}| = M$, but this does not by itself

require more workers to participate in a query. Workers become active only after an unresolved need has been selected and dispatched at Line 7.

In our implementation, Line 7 uses a two-stage routing procedure. A non-LLM retrieval stage ranks WorkerCards using sparse lexical matching, exact-term matching, and dense similarity, combined with reciprocal-rank fusion, and provides a bounded set of relevant candidates for the current routing decision. The orchestrator then decides whether and where to dispatch the current unresolved need using the runtime state in G_t.

Candidate discovery and runtime coordination play distinct roles. The retrieval stage limits the WorkerCard information considered within an individual routing decision, whereas the evolving Need Graph governs the sequence of routing decisions over the complete query trajectory, including whether further worker activation is required and when search can terminate. Our space-decoupling claim therefore concerns active multi-agent coordination, specifically whether worker participation is structurally coupled to the number of available territories, rather than claiming that all retrieval computation is independent of information-space size.

A.2 Runtime Coordination Demand

To distinguish available search capacity from actual coordination, we define the unresolved need set at step t as

$$\mathcal{U}_t = \{n \in N_t \mid s_t(n) \neq \text{resolved}\}, \tag{6}$$

where $s_t(n)$ denotes the resolution status of need n.

Because the Need Graph may be revised during execution, the set of needs that appear throughout a task can be larger than the initial node set N_0. We therefore define the task's realized need set as

$$\mathcal{H}(q) = \bigcup_{t=0}^{T_q} N_t, \qquad H(q) = |\mathcal{H}(q)|, \tag{7}$$

where T_q is the final coordination step. A reframed or newly introduced need is counted as a new realized need in $\mathcal{H}(q)$.

Let $a(n)$ denote the number of routing attempts associated with need n. This includes its initial routing and any subsequent rerouting caused by runtime recovery. Under the execution budget, let

$$a(n) \leq \rho \tag{8}$$

for all $n \in \mathcal{H}(q)$, where ρ is the maximum number of routing attempts permitted for a single need.

The total number of worker-dispatch events is therefore

$$K(q) = \sum_{n \in \mathcal{H}(q)} a(n). \tag{9}$$

Finally, let

$$\mathcal{A}(q) = \bigcup_{t=0}^{T_q - 1} \{i_t\} \tag{10}$$

denote the set of distinct territory-backed workers activated during execution.

A.3 Space-Decoupled Coordination

We now formalize the sense in which ANTMAN decouples active coordination from the total number of available territories.

Proposition 1. For any query q executed by Algorithm 1,

$$|\mathcal{A}(q)| \leq K(q) \leq \rho H(q). \tag{11}$$

Consequently,

$$|\mathcal{A}(q)| \leq \min\{M, \rho H(q)\}. \tag{12}$$

Thus, the number of workers activated by ANTMAN is bounded by realized task demand and the per-need routing budget, rather than directly by the number of available territories M.

Proof. Every worker activation in Algorithm 1 occurs through a routing decision on Line 7. Each such decision is associated with one currently selected information need.

For a realized need n, let $a(n)$ be the total number of times it is routed or rerouted. Summing these events over all needs encountered during execution gives

$$K(q) = \sum_{n \in \mathcal{H}(q)} a(n). \tag{13}$$

Since a single routing event can activate at most one territory-backed worker, the number of *distinct* workers activated cannot exceed the total number of routing events:

$$|\mathcal{A}(q)| \leq K(q). \tag{14}$$

By Equation 8, each need is routed at most ρ times. Therefore,

$$\begin{aligned} K(q) &= \sum_{n \in \mathcal{H}(q)} a(n) && (15)\\ &\leq \sum_{n \in \mathcal{H}(q)} \rho && (16)\\ &= \rho |\mathcal{H}(q)| && (17)\\ &= \rho H(q). && (18) \end{aligned}$$

Combining the two inequalities yields

$$|\mathcal{A}(q)| \leq K(q) \leq \rho H(q). \tag{19}$$

In addition, ANTMAN contains only M territory-backed workers, so trivially $|\mathcal{A}(q)| \leq M$. Together,

$$|\mathcal{A}(q)| \leq \min\{M, \rho H(q)\}. \tag{20}$$

Importantly, M appears only as the size of the *available* candidate pool. It does not appear as a multiplicative factor in the demand-dependent bound $\rho H(q)$. Therefore, enlarging the information space and increasing M does not by itself force additional workers to become active. □

Corollary 1. Consider a family of increasingly large information spaces $\{\mathcal{I}_M\}$ for the same class of queries. If the realized information demand remains bounded such that

$$H(q) \leq \bar{H} \tag{21}$$

and the per-need routing budget ρ does not depend on M, then

$$|\mathcal{A}(q)| \leq \rho \bar{H} = O(1) \qquad \text{with respect to } M. \tag{22}$$

Hence ANTMAN does not structurally require active coordination to grow linearly with information-space size.

Interpretation. The result does *not* claim that ANTMAN always activates a constant number of workers. A harder query may produce more unresolved needs, more revisions, or more recovery attempts, increasing $H(q)$ or $K(q)$. Instead, the proposition states that this growth is induced by *task demand*. Simply adding more searchable territories does not automatically add coordination.

A.4 Contrast with Partition-Driven Coordination

The distinction becomes clearer when compared with a partition-driven execution rule. Consider a method that assigns one worker to each of the M partitions of the information space and activates those workers as part of the initial decomposition. Its active worker count satisfies

$$|\mathcal{A}_{\text{partition}}(q)| = M, \tag{23}$$

or more generally

$$|\mathcal{A}_{\text{partition}}(q)| = \Theta(M) \tag{24}$$

when a constant fraction of partitions is activated.

ANTMAN instead satisfies

$$|\mathcal{A}_{\text{ANTMAN}}(q)| \leq \min\{M, \rho H(q)\}. \tag{25}$$

The difference is therefore not that ANTMAN ignores the size or structure of the information space. A larger space may produce more territories and hence a larger candidate worker pool. The difference is that this larger pool is *passive* until the runtime Need Graph creates demand for it.

Under fixed or comparable information requirements,

$$H(q) \not\propto M, \tag{26}$$

and therefore ANTMAN does not inherit the linear coordination growth of a partition-driven strategy.

This is the structural property evaluated in Section 4.2.1, where the searchable information space is increased while the underlying information requirements remain comparable.

B Detailed Controlled Scaling Results

This section provides additional results for the controlled information-space scaling experiments in Section 4.2.1. We first summarize endpoint growth from 32K to 512K, then report the complete per-length results and additional position-sensitivity analysis.

B.1 Endpoint Scaling

Table 7 summarizes how coordination, model calls, cost, and answer quality change from 32K to 512K.

Table 7: **Endpoint scaling from 32K to 512K.** Growth factors compare the smallest and largest information spaces in the controlled evaluation. ΔF1 reports the corresponding absolute change in answer quality.

Method	Coord. Growth	Calls Growth	Cost Growth	ΔF1
Multi-Agent Baselines				
LongAgent (Zhao et al., 2024)	15.29×	12.81×	14.22×	+.023
CoA (Zhang et al., 2024)	15.33×	11.30×	16.11×	−.004
Ours				
ANTMAN	**1.23**×	**1.21**×	**6.90**×	+.065

Table 8: **Complete results for controlled information-space scaling.** Active coordination denotes the average number of participating workers or members per query. Calls and cost are averaged per query. Lower values are better for coordination, calls, and cost; higher values are better for EM and F1.

Method	Length	Active Coord.	Calls / Query	Cost / Query	EM	F1
Multi-Agent Baselines						
LongAgent (Zhao et al., 2024)	32K	19.00	22.83	$0.0983	.533	.731
	64K	37.00	39.50	$0.1797	.533	.737
	128K	73.10	75.60	$0.3549	.500	.670
	512K	290.50	292.50	$1.3974	.567	.754
CoA (Zhang et al., 2024)	32K	5.10	**7.10**	$0.0809	.567	.750
	64K	10.00	**12.00**	$0.1613	.567	.734
	128K	20.00	**22.00**	$0.3216	.400	.564
	512K	78.20	80.23	$1.3027	.500	.746
Ours						
ANTMAN	32K	**2.57**	27.47	**$0.0773**	**.600**	**.775**
	64K	**2.87**	29.53	**$0.1069**	**.667**	**.784**
	128K	**3.07**	28.30	**$0.1276**	**.600**	**.768**
	512K	**3.17**	**33.10**	**$0.5332**	**.667**	**.840**

B.2 Complete Per-Length Results

Table 8 reports the complete results underlying the scaling curves in Figure 3. Each entry averages 30 evaluations at a given context length (10 questions × 3 evidence positions), spanning a 16× increase in information-space size from 32K to 512K.

C Raw Scores for Structured Navigation

Table 9 reports the original benchmark-scale scores underlying the normalized results in Table 4. RepoProbe-Python is reported on its original 10-point scale, while SWE-QA-Pro is reported on its original 50-point scale. The main-paper table linearly rescales these values to 0–100.

Table 9: **Raw scores for realistic structured navigation.** RepoProbe-Python scores are reported out of 10 and SWE-QA-Pro scores out of 50.

Method	RepoProbe-Python / 10	SWE-QA-Pro / 50
Retrieval and Iterative Retrieval Baselines		
Direct	3.108	27.87
Iterative Sparse Retrieval	3.315	30.45
Dense Retrieval (Karpukhin et al., 2020)	3.151	29.15
S2G-RAG (Li et al., 2026)	2.287	30.20
Agentic and Repository-Aware Baselines		
Matched ReAct (Yao et al., 2023)	2.586	32.46
ReAct + RepoGraph (Ouyang et al., 2025)	2.571	33.69
RepoDistill (Yin et al., 2026)	2.827	32.55
Benchmark-Specific Reference		
SWE-QA-Pro Agent (Cai et al., 2026)	–	38.79
Ours		
ANTMAN-H†	3.410	37.30
ANTMAN	**3.821**	**39.89**

Main-paper scores are obtained by linear normalization: RepoProbe-Python ×10 and SWE-QA-Pro ×2. † ANTMAN-H uses Qwen3-8B for all non-orchestrator components.

D ADAPTIVE-COORDINATION ABLATIONS

D.1 VARIANT DEFINITIONS

All variants preserve ANTMAN's territories, WorkerCards, models, tools, execution budget, and synthesis procedure. The Need-Graph variants additionally share the same initial decomposition and initial Need Graph. Graph-free Adaptive receives the same query, WorkerCards, accumulated evidence, and interaction history, but does not instantiate or maintain an explicit persistent representation of unresolved needs. Instead, the controller may adaptively select workers and revise its next search action directly from the available interaction history.

Table 10 summarizes the state representation and runtime mechanisms available to each variant.

Table 10: **Definition of state and adaptive-coordination ablations.** Checkmarks indicate enabled mechanisms; "–" denotes a mechanism that is not applicable without an explicit Need Graph.

Variant	Explicit Need State	Need Revision	Adaptive Rerouting	Recovery
Static ANTMAN	✓	×	×	×
Graph-free Adaptive	×	–	✓	✓
w/o Need Revision	✓	×	✓	✓
w/o Adaptive Rerouting	✓	✓	×	✓
w/o Recovery	✓	✓	✓	×
Full ANTMAN	✓	✓	✓	✓

Static ANTMAN freezes the initial Need Graph and routing plan, disabling Need revision, adaptive rerouting, and recovery during execution.

Graph-free Adaptive removes the explicit Need Graph while retaining adaptive control. At each step, the controller observes the query, accumulated evidence, and prior interaction history and may choose which worker to invoke next, redirect search based on new evidence, or recover from an unproductive search direction. It therefore tests whether an explicit evolving unresolved-need state provides value beyond generic evidence-conditioned adaptive replanning.

The w/o Need Revision variant retains the initial Need Graph but prevents changes to its structure or formulation while preserving evidence-dependent worker selection, rerouting, and recovery. The w/o Adaptive Rerouting variant allows the Need Graph to evolve but fixes each need's worker assignment after its initial routing. The w/o Recovery variant retains Need revision and adaptive rerouting but disables stuck detection, reframing, and fallback. Full ANTMAN retains the evolving Need Graph and all three runtime adaptive mechanisms.

We evaluate all variants under matched execution budgets on the 512K controlled condition, SWE-QA-Pro, and a fixed 40-question GAIA subset.

D.2 COMPLETE RESULTS AND ANALYSIS

Tables 11–13 report answer quality and execution statistics. Active coordination denotes the mean number of distinct territory-backed workers activated per query. It is reported for the multi-worker controlled and repository substrates; GAIA uses a single territory-backed worker. All variants receive the same maximum execution budget of 100 LLM calls per query, but may use that budget differently as a consequence of their runtime control policy. Realized calls therefore reflect when a variant continues, revises, or terminates search rather than an independently allocated resource.

Explicit need state and worker activation. In controlled scaling, full ANTMAN achieves substantially higher answer quality than Graph-free Adaptive while activating a comparable number of distinct workers. This suggests that the benefit of an explicit evolving Need Graph extends beyond simply engaging a larger worker set, highlighting the role of persistent unresolved-need tracking in directing subsequent search.

Need revision and adaptive rerouting. On SWE-QA-Pro, disabling Need revision lowers answer quality despite activating more distinct workers than full ANTMAN, indicating that broader worker

Table 11: **512K controlled scaling.** Answer quality is F1 score; cost and calls are per query.

Variant	Answer Quality	Active Coordination	Cost ($/query)	LLM Calls (/query)
Adaptive Coordination Ablations				
Static ANTMAN	40.32	1.97	0.156	28.7
Graph-free Adaptive	64.01	3.20	0.354	20.7
w/o Need Revision	74.44	3.10	0.418	21.0
w/o Adaptive Rerouting	82.20	2.00	0.451	28.2
w/o Recovery	60.12	2.07	0.353	25.9
Ours				
Full ANTMAN	**84.03**	**3.17**	0.533	33.1

Table 12: **SWE-QA-Pro ablation.** Results are reported on a 40-question subset sampled with seed 42. Answer quality is the direct score; cost and calls are per query.

Variant	Answer Quality	Active Coordination	Cost ($/query)	LLM Calls (/query)
Adaptive Coordination Ablations				
Static ANTMAN	68.58	1.35	0.273	36.5
Graph-free Adaptive	78.88	2.45	0.165	22.1
w/o Need Revision	75.32	3.50	0.298	32.4
w/o Adaptive Rerouting	72.13	2.20	0.407	48.6
w/o Recovery	76.52	2.15	0.522	56.5
Ours				
Full ANTMAN	**81.43**	**3.00**	0.504	54.3

Table 13: **GAIA ablation (40 questions).** Answer quality is accuracy; cost and calls are per query.

Variant	Answer Quality	Cost ($/query)	LLM Calls (/query)
Adaptive Coordination Ablations			
Static ANTMAN	50.00	0.196	42.8
Graph-free Adaptive	50.00	0.179	54.0
w/o Need Revision	42.50	0.115	28.7
w/o Adaptive Rerouting	60.00	0.221	47.4
w/o Recovery	52.50	0.325	69.5
Ours			
Full ANTMAN	**60.00**	0.206	45.4

activation alone does not ensure better evidence acquisition. Disabling adaptive rerouting also reduces answer quality, highlighting the value of revisiting worker assignments during repository navigation.

Recovery and execution cost. Removing recovery substantially reduces answer quality in controlled scaling. On GAIA, the same ablation lowers accuracy while increasing both model calls and inference cost. Thus, recovery contributes to answer quality and, in this setting, is also associated with lower realized execution cost.

E Primary Runtime Behaviors

This appendix summarizes the primary LLM-guided behaviors used during ANTMAN's runtime coordination loop. We focus on Need Graph construction and revision, need-conditioned worker execution, evidence selection, and need resolution.

E.1 Orchestrator

Need planning and selection. Given the current query, Need Graph, accumulated evidence, and execution history, identify the unresolved information requirements that should be pursued next. Revise or decompose needs when necessary, respect dependencies between needs, and select suitable workers for needs that are ready for execution.

Need Graph consolidation. Given the current Need Graph and a proposed new or revised need, determine how it should be incorporated into the graph. The proposal may be added as a new requirement, attached to an existing dependency, merged with an overlapping need, subsumed by an existing need, or discarded if redundant.

E.2 Worker Execution

Need-conditioned action planning. Given a selected unresolved need, the worker's assigned territory, and the available substrate-specific tools, choose and order a bounded sequence of actions for locating evidence relevant to the current need. The worker operates locally and returns evidence, remaining uncertainty, and progress to the coordinator.

E.3 Evidence and Need-State Update

Evidence selection. Given the evidence accumulated during execution, retain the evidence most relevant to resolving the query and discard redundant or unsupported material. The selected evidence is passed to subsequent coordination and final answer synthesis.

Need-resolution assessment. Given the current need and the evidence collected for it, determine whether the need is `resolved`, `partial`, or `unresolved`. A resolved need is closed; an unresolved need remains active; and a partial need is replaced or refined into a more specific unresolved requirement for subsequent coordination.

F Additional Experimental Details

F.1 Substrate-Specific Tool Interfaces

ANTMAN retains the same need-conditioned coordination abstraction across environments while adapting the information-access layer to each substrate. The controlled long-context setting uses lexical and semantic retrieval together with bounded document reading, whereas repository navigation additionally exposes structural code relations. GAIA uses a separate web-oriented interface for external information seeking and lightweight computation. Table 14 summarizes these capabilities at the interface level.

The controlled long-context and repository evaluations share the same local information-access implementation, with the repository setting additionally supporting structural code navigation. This allows workers to follow relations among program entities as well as retrieve and inspect relevant

Table 14: Information-access capabilities used across evaluation substrates.

Setting	Information access
Controlled long context	Lexical and semantic retrieval with bounded reading of selected document regions.
Repository navigation	Retrieval and bounded reading augmented with structural navigation over symbols, imports, references, inheritance, and call relations.
GAIA-Text-103	Web search, webpage text extraction, and isolated Python execution for tool-assisted information seeking.

content. These substrate-specific capabilities affect how evidence is accessed within a territory, while the Need Graph, worker organization, and need-conditioned coordination procedure remain unchanged.

For GAIA-Text-103, all evaluated methods receive the same tool interface. The shared interface supports text-oriented web search and webpage access together with isolated Python execution. Because GAIA-Text-103 contains no task attachments, attachment-oriented capabilities are not used in this evaluation. Web search is backed by Tavily.

F.2 Controlled Information-Demand Scaling Details

Task construction. We construct 10 matched 512K base contexts, each containing 16 planted evidence records distributed across a fixed pool of eight territories. Multiple records may occur in the same territory. For each base context, the document collection, distractors, evidence locations, territory assignment, and WorkerCards are held fixed across all demand conditions. Only the query and corresponding gold answer change.

Each evidence record contains an independently generated value $e_i \in \{0, \ldots, 7\}$. For required-evidence level $k \in \{1, 4, 16\}$, the answer is determined by

$$y_k = \left(\sum_{e_i \in D_k} e_i \right) \bmod 8,$$

where $D_1 \subset D_4 \subset D_{16}$ are nested required-evidence sets. The eight outcomes are mapped to neutral answer labels. Because each required value is independently generated, omitting any required evidence unit leaves the answer ambiguous over all eight classes.

We evaluate 30 matched conditions in total (10 base contexts $\times$ 3 demand levels). All conditions use the same model, tools, routing mechanism, worker organization, and execution budget, with a maximum of 10 coordination rounds. Answer quality is measured by exact-match accuracy, while *Complete Evidence* records whether all required evidence units are recovered. Evidence recovery is determined from the extracted evidence content rather than from merely accessing the corresponding document.

Low-demand exploration overhead. Although active coordination decreases slightly from 5.0 workers at one required unit to 4.3 at four units, trajectory analysis shows that this effect is driven by exploratory overhead rather than failed evidence acquisition. Table 15 decomposes trajectory-level worker activity according to whether it contributes to required evidence acquisition.

Table 15: **Trajectory diagnostics under increasing information demand.** Worker activity is decomposed according to whether it contributes to required evidence acquisition.

Required Evidence	Worker Activity	Relevant Activity	Irrelevant Activity
ANTMAN			
1 unit	5.0	0.9	4.1
4 units	4.3	3.2	1.1
16 units	8.2	8.0	0.2

At one required evidence unit, 4.1 of 5.0 worker activities are unrelated to the required evidence, accounting for 82% of the observed activity. At four units, 3.2 of 4.3 activities contribute to required evidence acquisition, while irrelevant activity falls to 1.1. At sixteen units, nearly all observed

worker activity contributes to required evidence acquisition. Thus, the slight low-demand inversion in active coordination is explained by proportionally greater exploration overhead at small information demand, while worker activity becomes increasingly aligned with required evidence as demand grows. Complete evidence recovery and answer accuracy remain at 100% across all three demand levels.

F.3 REPOPROBE-PYTHON EXECUTION DIAGNOSTICS

We further examine whether the RepoProbe-Python results can be explained by differences in execution budget or by the amount of repository interaction. All methods use the same frozen question set and model setting. Agentic methods are configured with the same maximum budget of 100 LLM calls per query. Table 16 reports realized LLM calls, tool calls, generation cost, and answer quality.

Table 16: **Execution diagnostics on RepoProbe-Python.** LLM calls, tool calls, and generation cost are averaged per query. RepoProbe-Python scores are reported on the same 0–100 scale as Table 4.

Method	LLM Calls	Tool Calls	Cost / Query	RepoProbe Score
Retrieval and Iterative Retrieval Baselines				
Direct	1.00	0.00	$0.0067	31.08
Iterative Sparse Retrieval	3.19	2.16	$0.0217	33.15
Dense Retrieval (Karpukhin et al., 2020)	1.00	1.00	$0.0124	31.51
S2G-RAG (Li et al., 2026)	37.12	24.00	$0.1788	22.87
Agentic and Repository-Aware Baselines				
Matched ReAct (Yao et al., 2023)	18.12	16.98	$0.1136	25.86
ReAct + RepoGraph (Ouyang et al., 2025)	26.62	25.09	$0.1922	25.71
RepoDistill (Yin et al., 2026)	4.18	35.22	$0.0801	28.27
Ours				
ANTMAN	38.13	26.81	$0.2987	**38.21**

Execution diagnostics. All agentic RepoProbe-Python methods use the same maximum budget of 100 LLM calls per query, while average realized usage remains below 40 calls for every method (Table 16). Realized interaction count also does not exhibit a simple monotonic relationship with performance. In particular, S2G-RAG and ANTMAN use similar numbers of LLM calls (37.12 vs. 38.13) and tool calls (24.00 vs. 26.81), yet obtain substantially different RepoProbe-Python scores (22.87 vs. 38.21). Thus, the observed performance differences cannot be explained by unequal externally imposed call limits or by interaction count alone.

Iterative sparse retrieval. The sparse-retrieval baseline is a lightweight iterative lexical retrieval method rather than a one-shot BM25 reader. It allows up to three retrieval rounds, with the LLM determining whether the accumulated evidence is sufficient and reformulating the lexical query when further retrieval is needed. Its average of 2.16 tool calls and 3.19 LLM calls per query reflects early termination on a subset of questions.