Plan/Search: Structured Planning as a Training Signal for Retrieval-Augmented Reasoning

Anonymous ACL submission

Abstract

Recent work has shown that reinforcement learning (RL) can train language models to effectively use retrieval tools for multi-hop question answering. However, existing approaches rely on implicit reasoning within free-form chain-of-thought, leaving the model to discover effective search strategies on its own. We propose Plan/Search, which introduces explicit planning scaffolds—structured templates that decompose reasoning into goal-setting, progress tracking, and action planning—as an on-demand action learned through RL. Our approach outperforms Search-R1 baselines across four multi-hop QA benchmarks and demonstrates strong zero-shot transfer to the GAIA benchmark. Analysis of training dynamics reveals that models learn qualitatively different strategies: Plan/Search develops "short and frequent" interactions that track explicit subgoals and enable course correction through environmental feedback, while Search-R1 favors longer, monolithic reasoning chains that risk error compounding by filling knowledge gaps through internal inference.Ablations show that explicit progress tracking is the most critical component, and that on-demand invocation outperforms mandatory structure. Our findings suggest that structured scaffolds act as an inductive bias that shapes how models learn to coordinate reasoning with external tools.

1 Introduction

Recent work has demonstrated that training language models with outcome-based rewards on free-form reasoning traces achieves remarkable performance on mathematical and coding benchmarks (OpenAI et al., 2024; Guo et al., 2025). Subsequent work has extended this paradigm to retrieval-augmented settings (Jin et al., 2025; Song et al., 2025), where models interleave reasoning with search actions. However, whether the same free-form reasoning approach remains equally effective when coordinating with external tools is less clear: unlike self-contained math problems, multi-hop retrieval requires tracking partial progress across multiple search steps and distinguishing between different types of relationships (e.g., PhD advisor vs. postdoc mentor, as illustrated in Figure 1).

We hypothesize that the intermediate thinking channel in tool-augmented reasoning serves a different role than in pure reasoning tasks. In mathematical problem-solving, extended thinking helps explore solution paths; in retrieval settings, the answer cannot be derived through reasoning alone, but must be gathered through interactions with external tools. This suggests a different strategy may be optimal: rather than thinking longer, models should perhaps think shorter and interact more often. This motivates our central question:

> **Research Question.** In retrieval-augmented reasoning, where knowledge must be gathered rather than derived, can structured scaffolds shift models toward shorter reasoning and more frequent tool engagement?

To test this hypothesis, we propose **Plan/Search**, an RL training framework that introduces structured planning as an explicit, on-demand action. Rather than letting models reason freely between searches, Plan/Search constrains intermediate thinking to three fields: (i) **Goal** (the current sub-question), (ii) **Status** (established facts and remaining gaps), and (iii) **Next-step** (the intended next action). This lightweight scaffold encourages focused, incremental progress rather than extended internal deliberation. Figure 1 illustrates how explicit status tracking prevents cascading errors by distinguishing resolved sub-goals from open questions. Crucially, planning is *on-demand*: the policy learns when structured thinking is useful rather than following a mandatory template.

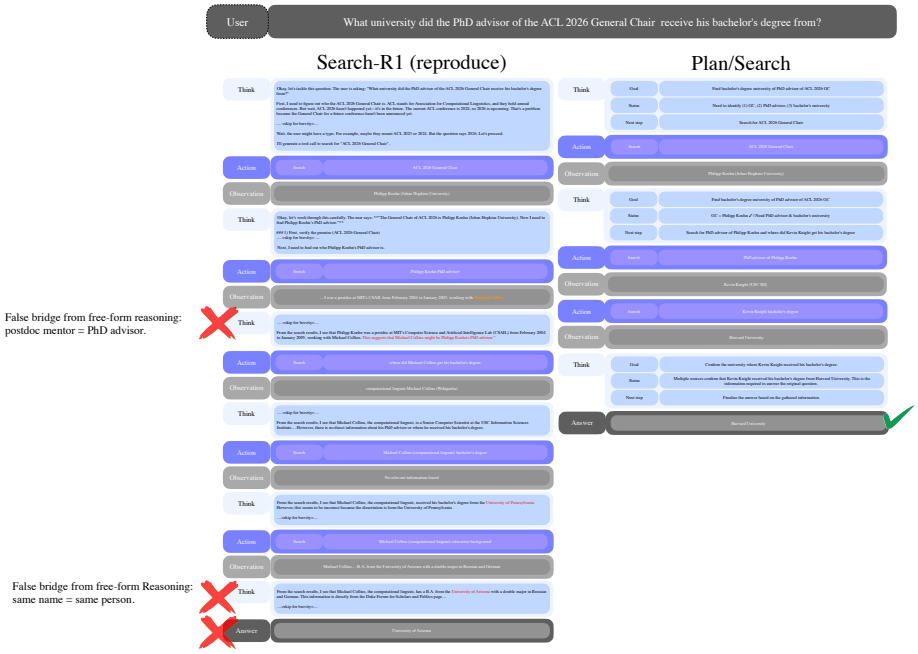

Figure 1: **Comparison of Search-R1 and Plan/Search on a 3-hop question.** Without explicit status tracking, Search-R1 conflates a postdoc mentor with a PhD advisor, leading to cascading errors. Plan/Search maintains structured state (Goal/Status/Next-step) that tracks resolved and pending sub-goals, preventing such confusions.

Our experiments reveal a surprising asymmetry between prompting and training. Enforcing the same scaffold at inference time degrades performance—even for frontier models like GPT-5.1 (Table 4). Yet when learned through RL, the scaffold improves performance across four multi-hop QA benchmarks (Table 1) and transfers zero-shot to GAIA despite a shift from retrieval APIs to web browsing. This suggests that explicit structure functions not as a prompting trick, but as an inductive bias that shapes how models learn to coordinate reasoning with external tools.

Analysis of training dynamics (Figure 2) reveals that Plan/Search and Search-R1 converge to qualitatively different strategies. Plan/Search develops iterative, focused retrieval patterns (more turns, fewer tokens per turn), while Search-R1 favors longer, monolithic reasoning chains (fewer turns, more tokens per turn). Ablations identify two key factors: (i) explicit progress tracking via the **Status** field yields the largest component contribution (Table 2), and (ii) on-demand invocation outperforms both mandatory structure and free-form thinking (Table 3), indicating that *agency over when to think* is itself a learnable skill.

Contributions.

- We introduce Plan/Search, which treats structured planning (Goal/Status/Next-step) as an on-demand action learned through RL. The scaffold provides an effective training signal while requiring no additional supervision beyond outcome-based rewards.

- We show that structured scaffolds lead to qualitatively different reasoning strategies: Plan/Search learns "short and frequent" interactions that enable error correction through environmental feedback, while free-form thinking develops longer chains prone to error compounding.

- We demonstrate strong out-of-domain generalization, including zero-shot transfer from HotpotQA to GAIA despite a shift from retrieval APIs to web browsing, suggesting that learned planning skills generalize beyond the training distribution and tool interface.

2 Related Work

Interleaved Reasoning and Action via Prompting. A line of work treats tool use as an inference-time prompting problem. ReAct (Yao et al., 2023b) interleaves reasoning traces with environment actions to improve robustness in QA and interactive tasks. Self-Ask (Press et al., 2022) decomposes

Category	Method	In-domain	Out-of-domain		
		HotpotQA†	2wiki⋆	Musique⋆	Bamboogle⋆
Direct Reasoning (No Retrieval)					
	Qwen3-8B (Thinking)	23.0	28.5	6.4	37.6
Pipeline-based Retrieval (No Training)					
	Qwen3-8B + Naive RAG	36.4	32.0	13.4	40.0
	Qwen3-1.7B + Plan/Search (Prompting)	18.6	20.4	6.8	20.8
	Qwen3-4B + Plan/Search (Prompting)	23.9	18.2	10.1	36.8
RL-trained Retrieval					
	Qwen2.5-3B-Instruct (Search-R1)	32.4	31.9	10.3	26.4
	Qwen3-1.7B (Search-R1)	32.8	33.0	10.5	31.2
	Qwen3-1.7B + Plan/Search (Ours)	**34.6**	**37.5**	**13.1**	**32.8**
	Qwen3-4B (Search-R1)	44.7	49.2	**21.4**	48.0
	Qwen3-4B + Plan/Search (Ours)	**46.5**	**49.6**	21.1	**49.6**

Table 1: **Main results.** We compare three paradigms: direct reasoning, pipeline-based retrieval, and RL-trained retrieval. Within the RL-trained category, our Plan/Search approach consistently outperforms the Search-R1 baseline. Highlighting indicates best result within each model size. † / ⋆ represents in-domain/out-of-domain datasets.

complex questions into follow-up queries that can be executed by a search engine. IRCoT (Trivedi et al., 2023) explicitly interleaves retrieval with stepwise chain-of-thought for multi-hop QA. These methods demonstrate the value of interleaving reasoning with action, but primarily impose structure at inference time, leaving open how such structures should be *learned*.

Structured Prompting for Reasoning. Prior work injects inductive biases into chain-of-thought via decomposition (Wang et al., 2023; Zhou et al., 2022; Press et al., 2022), structured representations (Chen et al., 2022), or explicit search structures (Yao et al., 2023a; Besta et al., 2024). More recently, structured "thinking tools" have shown promise in agentic settings.[1] However, these approaches rely on prompting; whether such structures remain beneficial when learned through RL has received less attention.

RL for Retrieval-Augmented Reasoning. Search-R1 (Jin et al., 2025) pioneered applying outcome-based RL to train models that interleave free-form thinking with search actions. Subsequent work has scaled this paradigm along various axes: data and model size (Chen et al., 2025; Song et al., 2025), web browsing interfaces (Li et al., 2025b; Zheng et al., 2025; Wu et al., 2025), and multi-turn interactions (Team et al., 2025; Li et al., 2025a). These efforts primarily retain free-form thinking as the intermediate representation, focusing on scaling rather than the structure of thinking itself.

What Should Be in the Thinking? While most RL-for-search work treats thinking as an unconstrained text channel, recent analysis suggests this may be suboptimal. Wang et al. (2025) show that only a minority of "high-entropy" tokens drive effective RL for reasoning, raising questions about the efficiency of long free-form traces. For self-contained tasks like mathematics, extended exploration may be necessary; for tool-augmented settings, thinking must additionally maintain state across interleaved tool calls—tracking what has been retrieved and what remains. Our work investigates whether replacing free-form thinking with structured, on-demand planning can provide a better learning signal for retrieval-augmented reasoning.

3 Method

3.1 Overview

Plan/Search replaces free-form intermediate reasoning with structured, on-demand thinking. The policy can invoke an optional THINK action that emits **Goal**, **Status**, and **Next-step** fields, externalizing the agent's intermediate state for subsequent decisions.

Figure 1 illustrates why this matters: without explicit status tracking, Search-R1 conflates a post-

[1] https://www.anthropic.com/engineering/claude-think-tool reports improvements on Tau-bench (Yao et al., 2024) using a think tool, though without systematic ablation of the tool's arguments.

Method	HotpotQA	Δ
Full Plan/Search	34.6	–
w/o *goal*	32.1	-7.2%
w/o *status*	29.5	**-14.7%**
w/o *next-step*	30.2	-12.7%
Search-R1 (no scaffold)	32.8	-5.2%

Table 2: **Component ablation on Qwen3-1.7B.** We evaluate the contribution of each scaffold component. Δ indicates relative change from the full model. Removing *status* causes the largest drop, suggesting progress tracking is critical for coherent multi-hop reasoning.

doc mentor with a PhD advisor, leading to cascading errors. Plan/Search maintains a running record of resolved sub-goals (e.g., "GC = Philipp Koehn ✓"), preventing such confusions across multi-hop reasoning chains.

3.2 Problem Formulation

We formulate multi-hop question answering with tool use as a sequential decision process. Given a question x and a retrieval tool $\mathcal{R}$, the agent interacts for multiple steps to produce a final answer $\hat{y}$.

At step t, the agent observes a state s_t consisting of the original question and the history of actions and tool observations:

$$s_t = \big(x, a_{<t}, o_{<t}\big),$$

where $a_{<t}$ are previous actions and $o_{<t}$ are retrieved passages returned by the tool. The agent selects an action

$$a_t \in \{\textsc{Search}(q),\ \textsc{Plan}(g, s, n),\ \textsc{Answer}(\hat{y})\}.$$

$\textsc{Search}(q)$ queries the retriever and returns passages $o_t \leftarrow \mathcal{R}(q)$. $\textsc{Answer}(\hat{y})$ terminates the episode. $\textsc{Plan}(g, s, n)$ is an internal action that appends a structured planning record to the context without accessing external information.

We use an outcome-based terminal reward:

$$r = \mathrm{EM}(\hat{y}, y),$$

where EM is exact match and y is the ground-truth answer.

3.3 Plan/Search: On-demand Structured Planning

The core design of Plan/Search is to expose planning as an explicit action that produces a lightweight scaffold. When the agent invokes PLAN, it outputs three fields:

- **Goal** (g_t): the current sub-question or information need.
- **Status** (s_t^{plan}): a brief summary of established evidence and remaining gaps.
- **Next-step** (n_t): the next intended action (e.g., a search query or answering).

We denote the resulting planning record as

$$p_t = (g_t,\ s_t^{plan},\ n_t),$$

which is appended to the context and becomes part of the next state. Crucially, **planning is optional**: the agent may call PLAN only when it finds explicit planning useful. This "on-demand" design lets the policy learn when to externalize its intermediate state, rather than forcing a fixed template at every step. As illustrated in Figure 1, the scaffold serves as a compact interface that connects observations (retrieved passages) to subsequent actions.

3.4 Training (Standard GRPO)

We adopt GRPO as in Cui et al. (2025) for all methods. For each training instance, we sample a group of G rollouts, compute terminal rewards, normalize rewards within the group to form group-relative advantages, and apply a clipped policy update with KL regularization to a fixed reference policy. Full training details and hyperparameters are provided in Appendix A.

4 Experiments

4.1 Experimental Setup

Datasets. We train exclusively on HotpotQA (Yang et al., 2018) and evaluate on three out-of-domain benchmarks: 2WikiMultiHopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). For zero-shot transfer evaluation, we use 103 text-only samples from the GAIA benchmark (Mialon et al., 2023), which requires web browsing rather than retrieval APIs. This setup tests whether learned retrieval strategies generalize beyond the training distribution and tool interface. Details on the tool configuration are provided in Appendix C.

Models. We experiment with Qwen3-1.7B, Qwen3-4B-Instruct.

Baselines. We compare against: (1) direct reasoning without retrieval, (2) pipeline-based retrieval including naive RAG and prompting-based Plan/Search, and (3) RL-trained retrieval including Search-R1 (Jin et al., 2025).

Evaluation. We report exact match accuracy following standard practice.

Training budget. Due to compute constraints, we train for 60 optimization steps (early stopping) rather than completing a full pass over the training set. With a prompt batch size of 512 and $G = 12$ rollouts per prompt, this corresponds to 30,720 training prompts in total.

Retrieval System. For HotpotQA training and multi-hop QA evaluation, we use dense retrieval over the Wikipedia corpus. We use E5-base-v2 (Wang et al., 2022) as the retrieval encoder with FAISS (Douze et al., 2024) for efficient similarity search on GPU. The corpus and pre-built index are from `PeterJinGo/wiki-18-e5-index`.[2] Each search query returns the top-3 passages.

5 Results and Analysis

5.1 Structured Thinking Improves Multi-hop Retrieval

Table 1 presents our main results across three paradigms: direct reasoning, pipeline-based retrieval, and RL-trained retrieval.

RL-trained retrieval outperforms alternatives. Direct reasoning with Qwen3-8B achieves only 23.0% on HotpotQA, while even naive RAG improves this to 36.4%, demonstrating the necessity of external knowledge for multi-hop QA. RL-trained approaches substantially outperform both, with our Plan/Search achieving 46.5% on Qwen3-4B.

Plan/Search consistently outperforms Search-R1. Within the RL-trained category, Plan/Search outperforms Search-R1 across model sizes. The improvements are particularly notable on out-of-domain benchmarks: +4.5%p on 2WikiMultiHopQA and +2.6%p on MuSiQue for Qwen3-1.7B. This suggests that explicit planning improves generalization beyond the training distribution.

Learned retrieval compensates for model scale. The 4B RL-trained model (46.5%) substantially outperforms the 8B direct reasoning model (23.0%), demonstrating that effective retrieval strategies can compensate for limited model capacity.

[2] https://huggingface.co/PeterJinGo/wiki-18-e5-index

Method	Structure	Invocation	HotpotQA
Plan/Search (Ours)	Structured	On-demand	**34.6**
Forced structure	Structured	Every step	26.9
Search-R1	Free-form	Every step	32.8

Table 3: **Structure vs. Agency ablation on Qwen3-1.7B.** Forcing structured thinking at every step (26.9%) performs worse than free-form thinking (32.8%). However, making structured thinking an *optional tool call* (34.6%) outperforms both.

5.2 Scaffolds Require Training to Be Beneficial

A striking finding is that the same scaffold that helps during RL training *hurts* performance when used only at inference time.

Prompting-based scaffolds degrade performance. Pipeline-based Plan/Search with prompting (23.9%) performs *worse* than direct reasoning (23.0%) for Qwen3-4B (Table 1). This suggests that explicit scaffolds impose cognitive overhead that untrained models cannot effectively manage.

Frontier models also struggle with structured tools. Table 4 reveals that this pattern extends to much larger models. GPT-5.1 with a structured thinking tool (26.2%) performs dramatically worse than GPT-5.1 with free-form web search (54.4%)—a drop of 28 percentage points. Similarly, Claude Sonnet 4.5 drops from 52.4% to 33.9% (-18.5%p) when given a thinking tool. This model-agnostic pattern confirms that structured scaffolds are not inherently beneficial; they require training to become effective.

RL bridges the gap. Our RL-trained 4B model (25.2% on GAIA) achieves comparable performance to the untrained GPT-5.1 with thinking tool (26.2%), despite the massive scale difference. This demonstrates that RL training transforms a harmful scaffold into an effective reasoning tool.

5.3 What Makes the Scaffold Work?

We conduct ablation studies to identify which aspects of Plan/Search drive improvements.

Status tracking is critical. Table 2 shows the contribution of each scaffold component. Removing **Status** causes the largest drop (-14.7%), followed by **Next-step** (-12.7%) and **Goal** (-7.2%). This suggests that explicit progress tracking—maintaining a record of what has been established

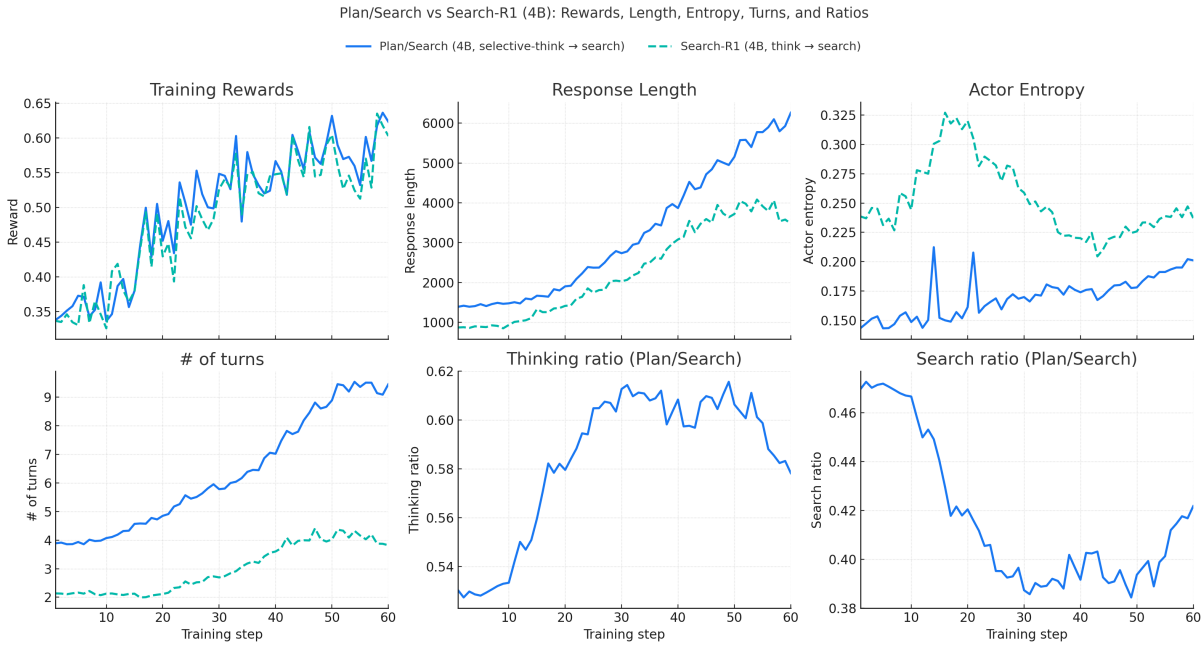

Figure 2: **Training dynamics comparison between Plan/Search and Search-R1.** Both methods achieve similar final rewards, but exhibit distinct behavioral patterns throughout training.

and what remains unknown—is the most critical component for coherent multi-hop reasoning.

Components work synergistically. Intriguingly, partial scaffold removal (e.g., w/o Status: 29.5%) performs worse than complete removal (Search-R1: 32.8%). This indicates that the scaffold components work synergistically; an incomplete scaffold may provide conflicting signals that hurt more than having no structure at all.

On-demand invocation outperforms mandatory structure. Table 3 disentangles the effect of structure from the effect of treating thinking as an action. Forced structure at every step (26.9%) performs substantially *worse* than free-form thinking (32.8%), suggesting that mandatory planning imposes overhead that outweighs its benefits. However, when the model can *choose* when to invoke structured thinking (34.6%), it outperforms both alternatives.

Thinking as action. This finding reframes our contribution: Plan/Search succeeds not because it adds structure, but because it treats structured thinking as an *action* in the model's action space. The model learns when explicit planning is beneficial (e.g., after ambiguous retrieval results) versus when it would be redundant overhead. This "thinking as action" paradigm enables flexible deployment of cognitive resources rather than rigid patterns.

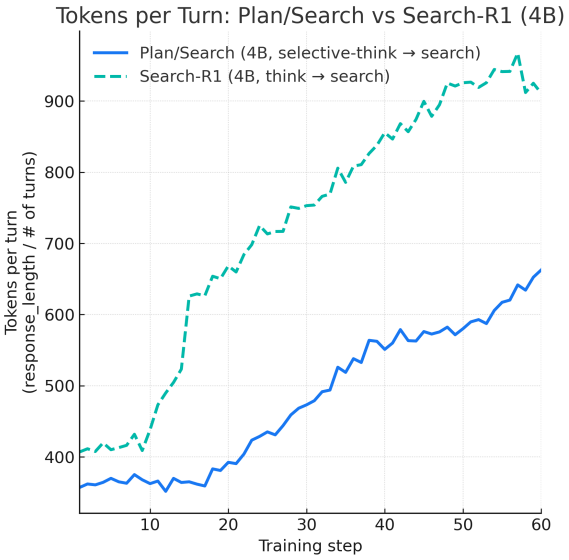

Figure 3: **Tokens per turn comparison.** Plan/Search generates fewer tokens per turn than Search-R1, despite using more turns overall. This suggests Plan/Search learns a more focused, iterative strategy.

5.4 Structured Thinking Changes Learning Dynamics

Beyond final accuracy, Plan/Search and Search-R1 converge to qualitatively different problem-solving strategies during training (Figure 2).

Iterative vs. monolithic strategies. Plan/Search learns to use substantially more turns (increasing

from 4 to 9 throughout training), while Search-R1 remains relatively stable (2 to 4 turns). Conversely, Search-R1 generates more tokens per turn (400→920) compared to Plan/Search (350→660). This reveals fundamentally different strategies: Search-R1 attempts to solve problems in fewer, longer reasoning chains, whereas Plan/Search decomposes problems into multiple focused interactions.

Shift from search-heavy to think-heavy behavior. The thinking ratio in Plan/Search steadily increases from 0.53 to 0.61 over training, while the search ratio decreases from 0.47 to 0.38. This suggests the model learns to allocate more computation to planning and becomes more selective in retrieval, rather than searching indiscriminately.

Why Plan/Search may generalize better. Figure 1 illustrates how these different strategies lead to different failure modes. When search results lack the desired information (e.g., PhD advisor is not directly stated), Search-R1's long free-form reasoning attempts to fill the gap using internal knowledge—in this case, inferring that a postdoc mentor might also be the PhD advisor. Such commonsense inferences, while plausible, are often incorrect and cause errors to compound through subsequent searches (searching for the wrong person's bachelor's degree). In contrast, Plan/Search's "short and frequent" strategy maintains explicit subgoals in the **Status** field and interacts with the environment more frequently, allowing the model to detect and correct errors before they propagate. This may explain why Plan/Search shows stronger out-of-domain generalization: rather than relying on potentially unreliable internal knowledge to bridge information gaps, it decomposes the problem into verifiable retrieval steps.

5.5 Transfer Beyond Training Domain

To test whether learned planning skills generalize beyond multi-hop QA, we evaluate on GAIA without any task-specific training (Table 4).

Planning skills transfer to new tasks. The RL-trained Plan/Search model achieves 25.2% average accuracy on GAIA, a 4.3× improvement over Qwen3-4B direct reasoning (5.8%). The largest gains appear on Level 2 questions (1.9% → 26.8%), which require multi-step reasoning.

Transfer despite tool interface shift. Notably, GAIA uses web browsing rather than the retrieval API used during training. The fact that Plan/Search still improves suggests that what transfers is not a dataset-specific heuristic, but a higher-level strategy: decomposing complex information needs and tracking progress across multiple interactions.

6 Discussion

> **Takeaway 1.** Free-form reasoning excels when answers can be derived internally; structured scaffolds excel when answers must be gathered through interaction.
>
> **Takeaway 2.** Frequent environment interaction reduces error compounding by enabling course correction before mistakes propagate.
>
> **Takeaway 3.** The benefit of structure comes from learned, on-demand deployment, not from imposing structure itself.

When to Think Long vs. Interact Often. Our results suggest that the optimal reasoning strategy depends on where knowledge resides. For reasoning-intensive tasks like mathematics and coding, extended internal deliberation helps explore solution paths. For retrieval-augmented reasoning and agentic tasks like GAIA, knowledge must be gathered from the environment rather than derived internally. In such settings, shorter reasoning and more frequent tool engagement prove more effective (Table 1, Table 4). Plan/Search operationalizes this insight by encouraging incremental verification through environmental feedback.

Why Frequent Interaction Helps. Training dynamics reveal that Plan/Search and Search-R1 converge to qualitatively different strategies. As shown in Figure 2, Plan/Search learns to use substantially more turns (4→9) while Search-R1 remains stable (2→4). Conversely, Figure 3 shows that Plan/Search generates fewer tokens per turn (350→660) compared to Search-R1 (400→920). This "short and frequent" pattern allows models to verify partial progress and correct errors before they compound. In contrast, Search-R1's longer reasoning chains attempt to fill knowledge gaps through internal inference, risking cascading failures when early retrievals are ambiguous (Figure 1).

Thinking as Action. The ablation in Table 3 reveals that the benefit of Plan/Search comes not from structure itself, but from treating structured

Method	GAIA (text-only, 103 samples)			
	Level 1	Level 2	Level 3	Average
Direct Reasoning (No Retrieval)				
Qwen3-4B-Instruct	12.8	1.9	0.0	5.8
Claude Sonnet 4.5	28.2	22.6	9.1	23.3
GPT-5.1	30.8	22.6	9.1	24.3
Pipeline-based Retrieval (No RL Training)				
Claude Sonnet 4.5 + Search	**61.5**	**50.9**	27.3	**52.4**
GPT-5.1 + Web Search	66.7	49.1	**36.4**	54.4
Claude Sonnet 4.5 + Thinking Tool	43.6	28.3	27.3	33.9 (-18.5)
GPT-5.1 + Thinking Tool	30.8	26.4	9.1	26.2 (-28.2)
RL-trained Retrieval (HotpotQA only)				
Qwen3-4B (Search-R1)	20.5	20.8	9.1	19.4
Qwen3-4B + Plan/Search (Ours)	**28.2**	**26.8**	9.1	**25.2** (+5.8)

Table 4: **Zero-shot generalization on GAIA text-only benchmark.** Adding a structured thinking tool *without RL training* hurts performance for both Claude Sonnet 4.5 and GPT-5.1 (-18.5 and -28.2 points respectively). This model-agnostic pattern confirms that structured scaffolds require training to be beneficial. Our RL-trained 4B model (25.2%) approaches the performance of untrained frontier models with thinking tools.

thinking as an optional action. Forcing structure at every step (26.9%) performs worse than free-form thinking (32.8%), suggesting that mandatory scaffolds impose overhead that outweighs their benefits. However, when models can choose when to invoke explicit planning (34.6%), they learn to deploy this cognitive tool strategically. This finding extends to frontier models: even GPT-4.1 suffers a 28-point drop when given a structured thinking tool without training (Table 4), confirming that on-demand deployment requires learning.

7 Conclusion

We investigated whether the free-form reasoning paradigm that excels at reasoning-intensive tasks like mathematics also applies to agentic settings where knowledge must be gathered from external tools. Our findings suggest it does not: for retrieval-augmented reasoning, models benefit more from short, frequent interactions with the environment than from extended internal deliberation.

Plan/Search operationalizes this insight through a lightweight scaffold (Goal/Status/Next-step) that encourages incremental progress over monolithic reasoning. Crucially, this behavioral shift emerges from RL with only exact-match rewards, requiring no reward engineering. The resulting strategy reduces error compounding by verifying progress through environmental feedback rather than filling knowledge gaps through internal inference.

Our results point to a broader principle: the optimal reasoning structure may depend on where knowledge resides. When answers must be derived, thinking longer helps; when answers must be gathered, interacting more often may be the better strategy.

Limitations

Scale. Our experiments are limited to models up to 4B parameters. It remains unclear whether explicit scaffolds provide benefits at larger scales, where models may have sufficient capacity to discover effective strategies without structural guidance.

Scaffold Design. We explored one specific scaffold design (goal/status/next-step). Other decompositions may be more effective, and the optimal scaffold may depend on the task domain.

Potential risks. As a tool-augmented method, our approach may inherit risks from the underlying retrieval/browsing tools, such as surfacing incorrect or biased information. We do not introduce new user-facing data collection or deployment components beyond standard benchmark evaluation.

References

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and 1 others. 2024. Graph of thoughts:

Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 17682–17690.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, and 1 others. 2025. Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. 2022. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. 2025. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Rui Ye, Yida Zhao, Liwen Zhang, Litu Ou, Dingchu Zhang, Xixi Wu, Jialong Wu, and 1 others. 2025a. Websailor-v2: Bridging the chasm to proprietary agents via synthetic data and scalable reinforcement learning. *arXiv preprint arXiv:2509.13305*.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. 2025b. Webthinker: Empowering large reasoning models with deep research capability. *arXiv preprint arXiv:2504.21776*.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. Gaia: a benchmark for general AI assistants. *Preprint*, arXiv:2311.12983.

Aaron Jaech OpenAI, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2022. Measuring and narrowing the compositionality gap in language models. *Preprint*, arXiv:2210.03350.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711.

Guangming Sheng, Chi Cao, Zilingfeng Hou, Xibin Dai, and 1 others. 2024. Verl: A unified and flexible library for reinforcement learning for llms. *arXiv preprint*.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025. R1-searcher: Incentivizing the search capability in LLMs via reinforcement learning. *Preprint*, arXiv:2503.05592.

MiroMind Team, Song Bai, Lidong Bing, Carson Chen, Guanzheng Chen, Yuntao Chen, Zhe Chen, Ziyi Chen, Jifeng Dai, Xuan Dong, and 1 others. 2025. Mirothinker: Pushing the performance boundaries of open-source research agents via model, context, and interactive scaling. *arXiv preprint arXiv:2511.11793*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics (TACL)*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. *arXiv preprint arXiv:2305.04091*.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, and 1 others. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*.

Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Gang Fu, Yong Jiang, and 1 others. 2025. Webdancer: Towards autonomous information seeking agency. *arXiv preprint arXiv:2505.22648*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2024.

$$\backslash$$

tau $-bench$: $Abenchmarkfortool-agent-userinteractioninreal-worlddomains.$ *arXiv preprint arXiv:2406.12045*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023b. React: Synergizing reasoning and acting in language models. *Preprint*, arXiv:2210.03629.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. 2025. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and 1 others. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

A Training and Implementation Details

We train both Qwen3-1.7B and Qwen3-4B models using identical hyperparameters and training procedures. All experiments are conducted using the VERL framework (Sheng et al., 2024) with GRPO advantage estimation. We did not perform an extensive hyperparameter sweep; most hyperparameters follow Cui et al. (2025) with small pilot adjustments.

Infrastructure. All RL training runs were executed on a single node with 8× NVIDIA H100 GPUs. Rollouts were generated using `sglang` with multi-turn tool interactions enabled.

Addressing entropy collapse with KL-Cov. During RL training, policy entropy tends to collapse rapidly, leading to overconfident predictions and performance saturation (Cui et al., 2025). To mitigate this issue, we adopt the **KL-Cov** method proposed by Cui et al. (2025), which selectively applies KL penalty to tokens exhibiting high covariance between advantage and log-probability. Specifically, for tokens where this covariance is largest, the gradient update is regularized via a KL divergence term against the reference policy. This approach maintains higher entropy throughout training while preserving stable learning dynamics. Key parameters include:

- `loss_mode=kl_cov`: Enables the KL-Cov loss computation instead of standard PPO loss.
- `kl_cov_ratio=0.002`: The fraction of tokens (top 0.2%) with highest covariance that receive KL regularization.
- `ppo_kl_coef=1.0`: The coefficient for the KL penalty term applied to high-covariance tokens.

PPO clipping. We use asymmetric clipping bounds for the importance sampling ratio in the policy gradient loss:

- `clip_ratio_low=0.2`: Lower bound, i.e., $\max(r_t, 1 - \epsilon_{\text{low}})$ where $\epsilon_{\text{low}} = 0.2$.
- `clip_ratio_high=0.2`: Upper bound, i.e., $\min(r_t, 1 + \epsilon_{\text{high}})$ where $\epsilon_{\text{high}} = 0.2$.

Following standard practice, we disable KL penalty in the reward function (`use_kl_in_reward=False`) and do not apply a separate actor-side KL loss (`use_kl_loss=False`), relying solely on the KL-Cov mechanism for regularization.

Setting	Value
Models	Qwen3-1.7B, Qwen3-4B
Hardware	8× NVIDIA H100 (single node)
Framework	VERL + GRPO + sglang
Steps	60
Train batch size	512
Rollouts per prompt (G)	12
Max assistant turns	60
Max prompt / response length	4,096 / 28,672
Learning rate	5×10^{-6}
Warmup ratio	0.285
Loss aggregation	token-mean
`clip_ratio_low / high`	0.2 / 0.2
`loss_mode`	`kl_cov`
`kl_cov_ratio`	0.002
`ppo_kl_coef`	1.0
KL in reward	False

Table 5: Key hyperparameters for GRPO training with KL-Cov regularization, applied to both Qwen3-1.7B and Qwen3-4B.

Rollout and optimization. We sample G=12 rollouts per prompt and use token-mean loss aggregation. The learning rate is set to 5×10^{-6} with a warmup ratio of 0.285. Due to compute constraints, we run RL for 60 optimization steps (early stopping) rather than completing a full pass over the training set. We use a prompt batch size of 512. Hyperparameters were selected via limited pilot runs rather than an extensive sweep; the final values are reported in Table 5.

B Qualitative Analysis Case Study

We present a detailed case study comparing the baseline **Search-R1** against our proposed **Plan/Search** method on a challenging task from the GAIA benchmark. This example demonstrates how explicit planning scaffolds enable the model to handle precise information retrieval tasks where unstructured reasoning fails.

Question

A paper about AI regulation that was originally submitted to arXiv.org in June 2022 shows a figure with three axes, where each axis has a label word at both ends. Which of these words is used to describe a type of society in a Physics and Society article submitted to arXiv.org on August 11, 2016?

Search-R1 (Baseline)

The model struggles with unguided exploration and eventually hallucinates an incorrect answer.

Think
Okay, let's figure this out. Key points: 1) A 2022 arXiv paper with three axes. 2) Answer describes a society type from a 2016 Physics and Society article.
Let me search for the 2016 paper...

Tool Call
`search {query: "arXiv Physics and Society 2016-08-11"}`

Output
Title: Exa Search... Total of 1162 entries... (truncated)

Think
Search result doesn't give the title directly. Let me open the link...

Tool Call
`open {id: 0}`

Output
Title: Physics Aug 2016... viewing lines [4-243] of 343...

Think
Entry [286] "Choosing Collaboration Partners..." has Physics and Society subject. That's a clue!
In absence of full text, I'll guess. A 2016 paper might discuss "complex societies"... Another possibility is "sustainable"...
I think the answer is "sustainable".

Answer: sustainable
Model hallucinates without finding the actual paper.

Plan/Search (Ours)

Our model uses explicit goal/status/next-step scaffold to formulate precise queries.

Think [Structured]
Goal: Identify word describing a society type from Aug 11, 2016 article.
Status: Need to find the specific article...
Next: Search for Physics and Society article.

Tool Call
`search {query: "Physics and Society article arXiv August 11 2016"}`

Output
[0] Exa Search: Physics and Society article arXiv August 11 20... (truncated)

Tool Call
`select {target: "1"}`

Output
Title: [1608.03637] Phase transition from **egalitarian** to **hierarchical** societies...
URL: arxiv.org/abs/1608.03637

Think [Structured]
Goal: Identify the word...
Status: Found article mentioning 'egalitarian' and 'hierarchical' societies.
Next: Search for AI regulation paper from June 2022...

Tool Call
`search {query: "AI regulation arXiv June 2022 figure three axes"}`

Error
HTTP 403 Failed to fetch...

Think [Structured]
Goal: Verify the answer...
Status: Cannot access AI paper, but found 'egalitarian' in Physics article title.
Next: Conclude with 'egalitarian'.

Answer: egalitarian
Structured planning finds correct paper and extracts answer.

C Tool Configurations

C.1 HotpotQA Training Environment

During training on HotpotQA, models have access to a single retrieval tool:

search(query: *string*) → List[Passage]

Retrieve relevant passages from the corpus.

C.2 GAIA Evaluation Environment

For GAIA evaluation, models use a web browsing interface with five tools:

search

`query`: *string*
search query

`topn`: *int*
max number of results (default: 10)

open

`id`: *int* | *string*
link ID or URL to open

`cursor`: *int*
cursor position in history

`loc`: *int*
line number to start from

`num_lines`: *int*
number of lines to display

find

`pattern`: *string*
text pattern to search for

`cursor`: *int*
cursor position in history

select

`target`: *int* | *string*
link ID or label substring

`cursor`: *int*
cursor position in history

`loc`: *int*
line number to start from

`num_lines`: *int*
number of lines to display