HiRA: Decoupling Planning and Execution with Hierarchical Reasoning in Deep Search

Jiajie Jin
Xiaoxi Li
Yuyao Zhang
jinjiajie@ruc.edu.cn
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China

Guanting Dong
Zhao Yang
Yutao Zhu
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China

Zhicheng Dou*
dou@ruc.edu.cn
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China

Abstract

Complex information needs in real-world search scenarios demand deep reasoning and knowledge synthesis across diverse sources, which traditional retrieval-augmented generation (RAG) pipelines struggle to address effectively. Current reasoning-based approaches face a key architectural challenge: they employ a single model to handle both high-level planning and detailed execution, resulting in inefficient reasoning and limited scalability. In this paper, we introduce HiRA, a hierarchical framework that separates strategic planning from specialized execution. Our approach decomposes complex search tasks into multiple subtasks, assigns each subtask to a domain-specific agent equipped with external tools and reasoning capabilities, and coordinates the results through a structured integration mechanism. This separation prevents execution details from disrupting high-level reasoning while enabling the system to leverage specialized expertise for different types of information processing. Experiments on four complex, cross-modal deep search benchmarks show that HiRA significantly outperforms state-of-the-art RAG and agent-based systems, highlighting the effectiveness of decoupled planning and execution for multi-step information seeking tasks. The code is available at https://github.com/RUC-NLPIR/HiRA.

CCS Concepts

• **Information systems** → **Retrieval models and ranking**; *Web applications*.

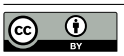
Keywords

Deep Search, Hierarchical Reasoning, Multi-Agent Systems, Retrieval-Augmented Generation

ACM Reference Format:

Jiajie Jin, Xiaoxi Li, Yuyao Zhang, Guanting Dong, Zhao Yang, Yutao Zhu, and Zhicheng Dou. 2026. HiRA: Decoupling Planning and Execution with Hierarchical Reasoning in Deep Search. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '26), July 20–24, 2026, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3805712.3809666

*Corresponding author.



SIGIR '26, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-2599-9/2026/07
https://doi.org/10.1145/3805712.3809666

1 Introduction

The information explosion on the Internet has made it increasingly difficult to find answers to complex queries, leading to the rapid development of deep search tasks that require understanding complex information needs and synthesizing accurate answers from multiple sources [19, 44]. However, traditional search engines only return ranked web pages based on keyword matching, requiring users to filter and collect information manually.

While large language models (LLMs) equipped with web search can provide direct answers, they typically utilize direct information from search results, lacking deep reasoning and comprehensive analysis capabilities [11, 23]. This has motivated the development of specialized AI agents for deep search tasks, such as OpenAI DeepSearch and Grok DeepSearch [12, 25], which aim to bridge the gap between simple search-enhanced models and deep information seeking systems.

Conventional approaches typically employ retrieval-augmented generation (RAG) techniques with predefined workflows [23, 32, 35], which incorporate components like query decomposition, document summarization, and self-reflection to improve generation quality. Recently, large reasoning models (LRMs) such as OpenAI-o1 [29] and DeepSeek-R1 [4] have introduced new opportunities by integrating web search and browsing capabilities within their reasoning processes [16, 24, 33]. As shown in Figure 1(b), these search-augmented reasoning methods can autonomously plan and acquire external knowledge for complex information retrieval in an end-to-end manner, significantly improving deep search performance.

However, existing approaches suffer from architectural limitations due to their reliance on a single reasoning model to handle all tasks. Current methods typically work by prompting reasoning models to generate special tokens [6, 16, 24, 27], which are used to trigger corresponding tool activations during their thinking process. While the broader agent community has explored decoupling planning from execution [41, 46], applying this paradigm effectively to deep search remains challenging due to the unique demands of context management, dynamic tool routing, and multi-step information synthesis. Specifically, the single-model approach introduces two critical deficiencies: (1) *Limited capability extensibility*: In these

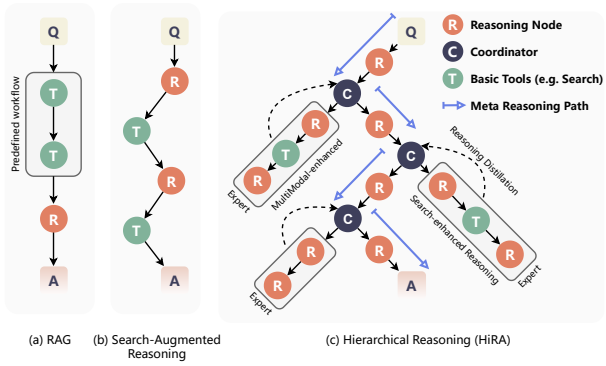


Figure 1: Comparison of current approaches for deep search tasks: (a) Direct Reasoning with LRMs, (b) Search-Augmented Reasoning that enables LRMs to use a search engine during reasoning, and (c) Hierarchical Reasoning that autonomously interacts with expert agents and tools in a continuous thinking process.

systems, adding new tools often requires re-engineering the token system itself, including designing new, tool-specific special tokens (e.g. <|begin_search_query|>, <|begin_code|>) and retraining or extensively re-prompting the reasoning model to learn their precise usage and context. This process is brittle and often requires re-engineering token systems or extensive training to ensure reliable token generation and tool coordination. (2) *Reasoning disruption*: As shown in Figure 1, external execution results are directly injected into the main reasoning chain, introducing noise that may disturb core reasoning processes. This disruption weakens the model's logical thinking and occupies limited context windows with irrelevant operational details. These limitations stem from the assumption that one agent is required to handle all aspects of complex reasoning tasks. We argue that an effective agent execution should follow a hierarchical structure, as shown in Figure 1(c): including a meta-agent for high-level planning, a coordinator for task reasoning transfer, and several specialized execution agents for specific operations. Each execution agent only needs to complete a single subtask through internal multi-step reasoning and iterative tool usage, allowing for deeper analysis without contaminating the overall planning process.

Based on this insight, we propose a **Hi**erarchical **ReA**soning (HiRA) model, a framework that builds upon the established planning-execution separation paradigm [46] and optimizes it for deep search tasks. This architecture consists of three components: a meta reasoning planner, an adaptive reasoning coordinator, and a group of domain-specialized executors. The meta reasoning planner breaks down complex tasks into subtasks through a reasoning process. These subtasks are then assigned to specialized agents by the adaptive reasoning coordinator. The assignments are based on the task complexity and required expertise. Each domain-specialized executor leverages specific reasoning models and external tools to execute the assigned subtask, with reasoning result distillation into the planner's process via the coordinator.

Our hierarchical design effectively decouples the strategic planning from the execution details, allowing for scalable, coherent reasoning in complex tasks. By integrating specialized expertise at the execution level and maintaining a coordinated flow across the hierarchy, HiRA ensures both the flexibility and efficiency necessary for tackling advanced reasoning challenges. We conduct experiments on four complex, cross-modal, multi-scenario deep search tasks. The results demonstrate that our framework significantly outperforms existing methods across several aspects. Overall, the key contributions of this work are threefold:

- **A Hierarchical Planning and Execution Framework (HiRA)**: Building on the established planning-execution separation paradigm in agent research, we introduce HiRA, a hierarchical framework that extends this paradigm for deep search tasks. Our architecture separates high-level strategic planning from specialized execution through a novel Coordinator layer, preventing execution-level details from disrupting the core reasoning chain and enabling the flexible, plug-and-play integration of diverse expert capabilities without complex re-engineering.
- **A Coordinated Communication and Knowledge Fusion Mechanism**: To efficiently achieve reasoning interaction between different levels, we propose a novel coordination mechanism to enable efficient, bidirectional communication in our decoupled framework. It leverages Reasoning Distillation to refine executor feedback for coherent high-level planning, and a Dual-Channel Memory to facilitate inter-agent knowledge fusion and enhance collaborative efficiency.
- **Superior Empirical Performance:** Experiments across four complex cross-modal search tasks demonstrate significant improvements over traditional RAG and current agent-based methods, validating the effectiveness of hierarchical reasoning augmentation for complex information retrieval.

2 Related Work

From Retrieval-Augmented Generation to Deep Search. Retrieval-Augmented Generation (RAG) combines external knowledge with LLMs' parametric knowledge [2, 9, 23]. Early approaches used single-step retrieval mechanisms [11, 20], while iterative RAG pipelines later incorporated query decomposition [3, 7, 35, 47], document refinement [17, 21, 42], and multi-round search [32, 35]. However, RAG methods rely on predefined workflows that limit adaptive decision-making for complex queries. Recent work with LRMs integrate retrieval directly into reasoning processes [16, 18, 24], but still requires inserting documents into reasoning chains or using auxiliary models for summarization [27]. These limitations motivate our exploration of hierarchical reasoning augmentation, where specialized agents serve as cognitive extensions while expanding additional capabilities.

Planning-Execution Separation and Multi-Agent Systems. The idea of separating planning from execution has a rich history in both classical AI planning and recent LLM-based agent research. The ReAct framework [46] established a foundational paradigm by interleaving reasoning traces with action execution within a single model, demonstrating the value of combining thought and action.

Building on this, multi-agent system (MAS) frameworks such as AutoGen [41] and MetaGPT [13, 26] explored orchestrating multiple LLM-based agents with specialized roles. More recently, dedicated planning-execution separation approaches have emerged [1, 8]. *Action-level separation* assigns executors to single-step tasks, as in Plan-Act [10] for HTML manipulation and CoAct [14]. However, CoAct relies on a "Plan-Act-Replan" paradigm, where local agents typically execute static plan segments, which struggles in deep search scenarios where the next strategic step depends on information dynamically revealed during execution. *Query-level separation* decomposes problems at higher granularity: REMA [5, 36] trains RL-based planners for mathematical reasoning, while LLMCompiler [22] and Query Compiler [47] break down QA tasks into parallel execution graphs. These methods often suffer from rigid task decomposition. Our work builds upon and extends the ReAct and MAS paradigms by introducing a *Dynamic Delegation & Distillation* framework specifically designed for deep search. Unlike ReAct, which suffers from context pollution as raw tool outputs accumulate in its reasoning chain, HiRA isolates execution from planning through a Coordinator layer that filters and distills execution results. Unlike prior MAS approaches that focus on general-purpose agent collaboration, HiRA targets the specific challenges of complex information seeking with domain-specialized executors and structured reasoning transfer.

3 Methodology

3.1 Problem Formulation

Given a complex question q that required information seeking and a predefined external environment $\mathcal{E}$, our objective is to design a framework that generates a final solution containing an answer $\mathcal{A}$ and the corresponding reasoning process $\mathcal{R}$. The generation process can be represented as:

$$P(\mathcal{R}, a \mid q, \mathcal{E}) = \prod_{t=1}^{T_{\mathcal{R}}} P(\mathcal{R}_t \mid \mathcal{R}_{<t}, q, \mathcal{E}_{<t}) \cdot P(a \mid q, \mathcal{R}),$$

where $T_{\mathcal{R}}$ represents the token generation steps for the reasoning process, $\mathcal{A}$ denotes the final answer, and $\mathcal{E}_{<t} = \{\mathcal{E}(R_{<s})\}_{s<t}$ denotes the collection of all environment interaction results prior to timestep t. While existing agentic reasoning approaches typically use tools directly as the external environment, our work introduces a higher-level abstraction where the environment $\mathcal{E}$ consists of a collection of expert agents, each capable of reasoning and utilizing specific tools to accomplish specialized tasks.

3.2 Overview of the HiRA Framework

The HiRA framework is a hierarchical reasoning system that enhances deep search effectiveness through the separation of planning and execution. As shown in Figure 2, the framework consists of three modules: (1) a **meta reasoning planner** that decomposes complex tasks into subtasks through step-by-step reasoning, (2) an **adaptive reasoning coordinator** that assigns subtasks to appropriate domain-specialized executors (expert agents) based on task complexity and capabilities, and (3) a **domain-specialized executors** that execute subtasks using specialized reasoning models and external tools. Results obtained by the executors are integrated back into the planner's reasoning process through the coordinator. This hierarchical design decouples strategic planning from execution details, enabling scalable and coherent reasoning for complex information seeking tasks.

In our framework, the meta reasoning planner operates at a high level of abstraction, decomposing the overarching task into high-level subtasks while deliberately omitting low-level implementation details. This design prevents the reasoning process from being biased or disrupted by fine-grained subtask specifics, thus enhancing the robustness and generalizability of the generated plans. A subtask is defined as a comprehensive, self-contained instruction that includes both the objective and the contextual requirements for accomplishing a specific component of the overall task. Unlike simple queries that typically request single pieces of information, subtasks are composite directives that may involve multiple steps of reasoning or tool calling , and include detailed requirements for information processing and result formatting.

3.3 Hierarchical Reasoning Paradigm

3.3.1 Meta Reasoning Planner. The meta reasoning planner serves as the core module of the framework, responsible for planning, reasoning, and answer generation. Unlike conventional tool-augmented approaches that require models to directly invoke tools with specific parameters, our meta planner decouples the task into several high-level subtasks containing strategic instructions for expert agents. This design enables natural collaboration between meta and expert agents, ensuring smooth information transfer while eliminating the noise and overhead associated with direct tool invocation and execution-level decision making.

To enable dynamic subtask generation, we employ a straightforward and unified prompting mechanism where the reasoning model uses a single, generic set of special tokens for all subtask dispatches. The overview of the process is shown in Figure 2. In the reasoning process, the planner will automatically generate special tokens and place the description and requirements of subtasks in the middle, which is similar to the process of humans issuing tasks. As shown in Equation (1), the generated subtask description s_k is based on previous task execution results $\{\mathcal{E}(s_j)\}_{j<k}$ and the current reasoning progress $O_{<t}$, naturally enabling reflection, correction, supplementation, and continuity generation of prior tasks. We impose no explicit constraints on subtask scope, difficulty, size, or relationships with previous subtasks to preserve overall reasoning flexibility.

$$P_M(s_k) = P_M(s_k \mid q, O_{<t}, \{\mathcal{E}(s_j)\}_{j<k}). \tag{1}$$

Then, the coordinator layer assigns s_k to corresponding expert agents for execution. Execution results $\mathcal{A}_k(s_k)$ are wrapped in special tokens, and then integrated into the model's reasoning process for continued generation. Notably, the subtask execution results incorporated into the reasoning process contain essential execution procedures and final conclusions, rather than vanilla tool invocation results that may contain noise and require subsequent processing.

During the generation process, the model incrementally conditions on the original query q, the prior decoding history $O_{<t}$, and the set of executed subtask results $\mathcal{A}_j(s_j)_{j\leq K}$ to derive the final

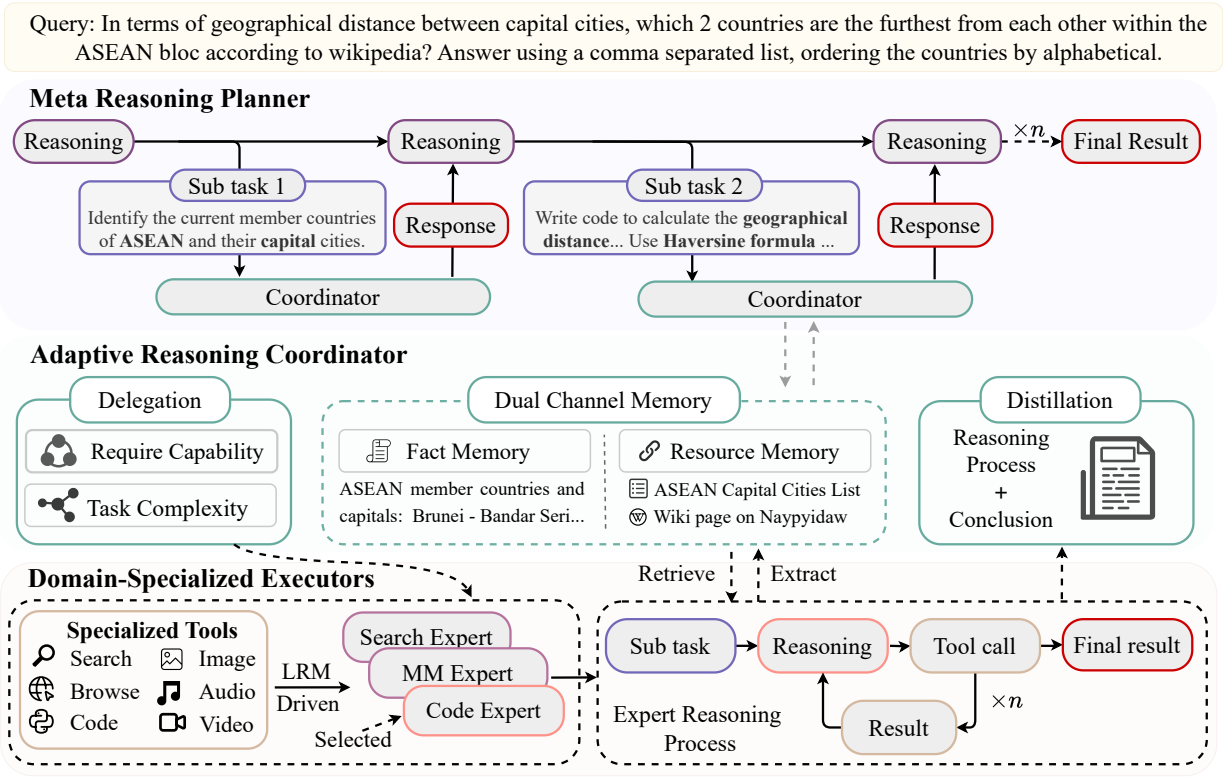


Figure 2: Overview of the HiRA Framework.

answer a, formalized as:

$$P_M(a) = P_M(a \mid q, O_{<t}, \mathcal{A}_j(s_j)_{j \le K}), \tag{2}$$

where K denotes the total number of subtasks. This design differs fundamentally from monolithic models in two ways:

- High-Level Abstraction: Our planner uses a unified token set for delegating comprehensive, high-level tasks, not invoking specific, low-level tools. This is an established and effective prompting strategy that relies on the model's ability to follow strategic instructions.
- Superior Scalability: The framework is highly extensible. New expert agents can be integrated without any modification to the Meta Planner's tokens or training. The Coordinator assumes the responsibility of routing tasks to new capabilities. This modular design avoids the need to constantly re-engineer the token system and retrain the model for each new tool, a critical bottleneck in monolithic architectures.

3.3.2 Adaptive Reasoning Coordinator. While separating execution from planning provides clear scalability advantages and reduces computational noise, it introduces the risk of information loss between components. To mitigate this challenge, we design an adaptive reasoning coordinator that incorporates bidirectional, task-specific reasoning transfer and a dual-channel memory mechanism. This coordinator facilitates seamless reasoning delegation from the meta agent to expert agents while enabling reasoning distillation in the reverse direction, thus preserving efficient inter-agent communication and maintaining the architectural benefits of separation. The coordinator has three core functions as follows.

(1) **Reasoning transfer process.** The coordinator is designed to interpret subtasks provided by the meta planner and identify the most suitable expert agent for task execution. Given the current subtask s_k and detailed information about all experts $I_{\mathcal{E}} = \{\mathcal{I}_{\mathcal{A}}\}_{\mathcal{A} \in \mathcal{E}}$, the coordinator first analyzes the subtask requirements, then evaluates agent capabilities across two key dimensions before making the optimal selection. Our instruction framework $\mathcal{I}_{\text{select}}$ encompasses: (a) Required capabilities: the domain knowledge and tool utilization abilities necessary for task completion, and (b) Task complexity: the computational difficulty of the subtask and the required depth of analysis. For tasks within the same category (e.g., information retrieval), target information often resides at varying depths across data sources. While deploying the most sophisticated agent can ensure problem resolution, it may introduce unnecessary computational overhead and analytical redundancy. Therefore, the coordinator prioritizes selecting the most efficient agent for each specific task to optimize overall system performance.

Formally, we model this selection process as a classification problem:

$$\mathcal{A}_k^* = \text{argmax}_{\mathcal{A} \in \mathcal{E}} P_C(O_{\text{dele}}^{(k)}, \mathcal{A} \mid s_k, \mathcal{I}_{\mathcal{E}}, \mathcal{I}_{\text{select}}),$$

where the coordinator generates selection reasoning $O^{(k)}_{\text{dele}}$ and identifies the optimal expert agent $\mathcal{A}^*_k$ through chain-of-thought reasoning.

(2) **Reasoning distillation process.** The coordinator is responsible for understanding and refining the expert agent's reasoning process before integrating the results into the meta reasoning planner's cognitive flow to maintain overall reasoning coherence. Unlike traditional tool-augmented reasoning approaches that primarily focus on final execution outputs, our framework considers both the expert agent's intermediate reasoning steps and the final conclusions. This dual consideration enables the meta planner to comprehend the underlying reasoning logic without being encumbered by low-level execution details, thereby facilitating autonomous reflection and critical evaluation of execution outcomes.

Specifically, given subtask s_k and the expert agent's reasoning process $O^{(k)}_{\text{expert}}$, the distilled reasoning process $O^{(k)}_{\text{dist}}$ and refined conclusion $\mathcal{R}^{(k)}_{\text{dist}}$ are generated through:

$$P_C\left(O^{(k)}_{\text{dist}}, \mathcal{R}^{(k)}_{\text{dist}} \mid s_k, O^{(k)}_{\text{expert}}\right) = \underbrace{P_C\left(O^{(k)}_{\text{dist}} \mid O^{(k)}_{\text{expert}}, \cdot\right)}_{\text{Reasoning Refinement}} \cdot \underbrace{P_C\left(\mathcal{R}^{(k)}_{\text{dist}} \mid O^{(k)}_{\text{dist}}, O^{(k)}_{\text{expert}}, \cdot\right)}_{\text{Conclusion Extraction}}.$$

Subsequently, $O^{(k)}_{\text{dist}}$ and $\mathcal{R}^{(k)}_{\text{dist}}$ are concatenated and seamlessly integrated as reasoning distillation feedback into the meta reasoning planner's ongoing cognitive process.

(3) **Dual-channel memory mechanism.** To enable effective information sharing and knowledge transfer among expert agents, we design a dual-channel memory mechanism tailored specifically for deep search task scenarios. The comprehensive memory repository $\mathcal{M}$ encompasses two distinct types of memory: fact memory $\mathcal{M}_f$ and resource memory $\mathcal{M}_r$.

• *Fact Memory ($\mathcal{M}_f$):* This module archives factual discoveries and insights extracted from expert agents' reasoning processes. To ensure traceability and verification, each memory entry comprises a factual assertion paired with its corresponding source attribution (e.g., URL, file name, or webpage identifier). Furthermore, to maintain memory efficiency and reduce redundancy, multiple similar factual statements originating from identical sources are intelligently aggregated into unified entries.

• *Resource Memory ($\mathcal{M}_r$):* This module maintains a repository of informational resources encountered during expert agent execution processes. Each entry contains a descriptive summary alongside the corresponding access path. This design provides subsequent agents with valuable exploration insights from previous interactions (e.g., knowing which URL contains specific data types), thereby preventing redundant exploration steps and enhancing overall system efficiency.

To strictly control computational latency, we avoid treating memory updates as an isolated post-processing step. Instead, the *Memory Construction* phase is implemented directly within the same instruction cycle as the reasoning distillation process. As the coordinator distills the expert's reasoning $O^{(k)}_{\text{expert}}$, it simultaneously identifies key findings ($\mathcal{M}_f$) and useful paths ($\mathcal{M}_r$), systematically updating the global memory repository $\mathcal{M}$ in a single pass. During the *Reasoning Transfer* phase, the coordinator utilizes $\mathcal{M}$ by retrieving pertinent entries based on semantic relevance between memory content and the current subtask requirements. These retrieved memories are then provided as contextual "warm-start" information to the selected expert agent, closing the knowledge loop.

3.3.3 Domain-Specialized Executors. To cover the diverse capabilities required for deep search tasks, we design three orthogonal agent capability dimensions, ensuring the system can handle complex and varied deep search scenarios:

- **Information acquisition:** This dimension is responsible for acquiring and integrating information from the web.
- **Cross-Modal understanding:** This dimension handles the understanding and fusion of multimodal information, capable of processing data from different modalities such as images, videos, and audio.
- **Computational reasoning:** This dimension handles mathematical computation, file processing, and other computational reasoning tasks, capable of transforming abstract problems into executable code solutions.

Based on these three dimensions, we implement four reasoning model-driven, specialized agents. For **information acquisition**, we design two search agents with different exploration depths: one based on a simple RAG pipeline that performs single retrieval for subtasks followed by reasoning, and another based on the WebThinker implementation [27] that can perform deep search and information acquisition on the internet. The combination of these two approaches enables flexible solutions for both simple and complex tasks. For **cross-modal understanding**, we embed multimodal models as tools within the reasoning model's inference process to achieve dynamic understanding of information in multimodal data. For **computational reasoning**, we embed code interpreters into the reasoning process.

For the aforementioned reasoning-driven agents, their reasoning process follows the tool-augmented reasoning execution flow, capable of dynamically outputting special tokens during reasoning to trigger corresponding tools. The reasoning process is:

$$P(O^{(k)} \mid s_k, \mathcal{T}, \mathcal{M}_k) = \sum_{t=1}^{T_k} P(O^{(k)}_t \mid O^{(k)}_{<t}, \{\mathcal{T}_j\}_{<t}, \cdot),$$

where $O^{(k)}$ represents the reasoning process of expert agent for subtask s_k, $\mathcal{M}_k$ represents the memory related to s_k, $\mathcal{T}$ represents the tools used in the process (e.g., code interpreter), and $\{\mathcal{T}_j\}_{<t}$ represents all tool invocation results before time step t. Based on this process, we can embed arbitrary tools into the reasoning model's inference process. More details can be found in the appendix.

3.4 Inference Process of HiRA

The inference process of HiRA follows an agentic reasoning approach. For a given question, the inference process begins with reasoning by the meta agent. During reasoning, the meta planner decodes special token pairs, wrapping the subtasks to be executed between them. The coordinator then processes and distributes the subtasks to expert agents for execution. After the expert agents perform reasoning and multiple rounds of tool invocation, subtask execution results are obtained.

This reasoning process and results are then processed by the coordinator through the reasoning distillation process to obtain refined results, which are integrated into the meta planner's reasoning chain to continue generation. During this process, the meta planner dynamically adjusts its plan and corresponding subtasks to be distributed based on execution results, until all information has been collected and the final answer is provided. The detailed example can be found in the appendix.

3.5 Theoretical Analysis: An Intuitive Framework for Decoupling

In this section, we provide an intuitive conceptual framework to illustrate why decoupling planning and execution is beneficial. We emphasize that this analysis serves as an *intuitive motivation* rather than a formal computational complexity proof; the empirical results in Section 5 provide the rigorous validation. The reasoning process of an agent in a deep search task can be conceptualized as a search for an optimal reasoning path within a vast information space. In this context, tools are essential instruments for acquiring external knowledge to prune this space and refine the path. However, conventional monolithic agents are constrained by an intrinsic *Cognitive Load*, which limits their performance on complex, multi-step tasks.

3.5.1 Cognitive Load in Naive Monolithic Agents. For a monolithic agent, we define its **Cognitive Capacity** as C_M, representing its finite resources for planning and processing information. At any reasoning step t, the agent's **Cognitive Load**, L_t, is a function of the entire interaction history $\mathcal{E}_{<t}$. We can model this load as:

$$L_t = \underbrace{\tau_{\text{plan}}}_{\text{Global Planning}} + \underbrace{\tau_{\text{exec}}(R_t)}_{\text{Action Execution}} + \underbrace{\sum_{s<t} \beta |\mathcal{E}(R_{<s})|}_{\text{Accumulated History Processing}} \tag{3}$$

Here, τ_{plan} is the constant overhead of maintaining a high-level strategy for question q. $\tau_{\text{exec}}(R_t)$ is the effort required to generate the current reasoning token R_t, especially when it involves formulating a complex tool call. The final term represents the cumulative burden of processing the raw, and often verbose, outputs from all past environment interactions, $\mathcal{E}_{<t}$.

In a monolithic agent, this entire load is handled by a single model. The probability of generating a correct and effective reasoning step R_t is therefore dependent on the agent's residual cognitive capacity:

$$P(R_t | R_{<t}, q, \mathcal{E}_{<t}) \propto \sigma\left(C_M - L_t\right), \tag{4}$$

where σ is a monotonically increasing function. As t increases, the size of $\mathcal{E}_{<t}$ grows, causing L_t to increase, which progressively consumes the agent's cognitive capacity, leading to a decay in its ability to generate high-quality reasoning steps.

3.5.2 HiRA: A Test-Time Enhancement of Reasoning Capacity. The HiRA framework directly addresses this limitation through its architectural **decoupling** of planning and execution, which effectively functions as a form of *test-time capability scaling*.

1. Offloading Execution Burden. In our framework, the Meta Reasoning Planner is shielded from the low-level details of execution. Its cognitive load, $L^{(t)}_{\text{planner}}$, is significantly reduced because it does not process the raw environment outputs. Instead, it operates on a distilled history $\mathcal{E}'_{<t}$:

$$L^{(t)}_{\text{planner}} = \tau_{\text{plan}} + \sum_{s<t} \beta' |\mathcal{D}_s|. \tag{5}$$

Here, $\mathcal{D}_s$ is the concise summary produced by the Adaptive Reasoning Coordinator's **Reasoning Distillation** process. Since the distilled information $|\mathcal{D}_s|$ is much smaller and less noisy than the raw output $|\mathcal{E}(R_{<s})|$, the cumulative load grows at a much slower rate ($\beta' \ll \beta$). This preserves the planner's capacity for robust, long-horizon strategic thinking.

2. Specialized and Stateless Execution. Each Domain-Specialized Executor is invoked to handle a self-contained subtask. Its cognitive load, L_{executor}, is *stateless* with respect to the global reasoning chain and does not accumulate over time t:

$$L_{\text{executor}} = \tau_{\text{exec}}(R_{\text{subtask}}) + \beta |\mathcal{E}(R_{\text{subtask}})|. \tag{6}$$

Furthermore, because each executor is an agent equipped with specialized capabilities and employs a highly optimized workflow, its effective cognitive capacity C_E can be considered greater than or equal to that of a generalist agent ($C_E >= C_M$). This results in a higher success probability for each delegated subtask. In summary, HiRA's hierarchical architecture enhances the system's overall **Effective Reasoning Capacity**. By separating the high-level planning from the details of execution, our framework prevents the cognitive overload and reasoning attenuation that plague monolithic agents. We note that the above analysis provides an intuitive characterization of the benefit; exploring rigorous formal guarantees remains an interesting direction for future work.

4 Experimental Settings

4.1 Datasets and Evaluation

To comprehensively evaluate our method on deep search tasks, we utilize four diverse benchmarks. (1) **GAIA** [28] is a general-purpose AI assistant benchmark for multi-step reasoning and retrieval; we use all its validation samples, covering text-only, multimodal, and file-based scenarios. (2) **WebWalkerQA** [40] tests web navigation and information extraction in both English and Chinese. (3) **SimpleQA** [39] is a challenging factual dataset requiring in-depth knowledge acquisition, suitable for evaluating performance on problems with fewer reasoning steps. (4) **Humanity's Last Exam (HLE)** [31] is a high-difficulty benchmark requiring complex reasoning and external retrieval. To reduce computational costs while maintaining statistical significance, we sample 500 questions from the HLE validation set and 200 questions each from WebWalkerQA and SimpleQA. To ensure fair evaluation, we adopt the LLM-as-Judge approach using Qwen2.5-72B-Instruct to assess the consistency between generated answers and golden answers. The evaluation instruction follows WebThinker [27].

4.2 Baselines

We compare HiRA with three categories of baselines. To ensure a fair comparison, **all baselines utilize the same backbone model and identical tool definitions** where applicable.
(1) **Direct Reasoning**: We evaluate models' native reasoning capabilities without external tools, including open-source models

Table 1: Overall performance on various deep search tasks, with accuracy results for each dataset obtained using llm-as-judge. For 32B models, the best results are indicated in bold, and the second-best results are underlined. Results from larger or closed-source models are presented in gray for reference.

Method	General AI Assistant				WebWalkerQA				Humanity's Last Exam				SimpleQA
	Level 1	Level 2	Level 3	Avg.	Easy	Med.	Hard	Avg.	NS	CE	SF	Avg.	Acc
Direct Reasoning													
Qwen3-32B-no-thinking	14.3	6.1	10.5	9.5	4.4	2.1	0.8	2.8	7.1	6.0	3.1	6.2	6.5
Qwen3-32B-thinking	26.2	12.1	0	14.9	6.9	1.1	2.9	3.1	14.6	9.8	8.4	12.6	10.5
DeepSeek-R1-32B	21.5	13.6	0.0	14.2	7.5	1.4	4.2	3.8	6.6	5.1	6.5	6.4	5.5
QwQ-32B	30.9	6.5	5.2	18.9	7.5	2.1	4.6	4.3	11.5	7.3	5.2	9.6	6.5
GPT-4o	23.1	15.4	8.3	17.5	6.7	6.0	4.2	5.5	2.7	1.2	3.2	2.6	39.0
DeepSeek-R1-671B	40.5	21.2	5.2	25.2	5.0	11.8	11.3	10.0	8.5	8.1	9.3	8.6	42.4
o1-preview†	-	-	-	-	11.9	10.4	7.9	9.9	12.9	8.1	6.6	11.1	42.7
Single-Capability Enhanced													
Vanilla RAG	40.5	21.2	5.2	25.2	57.4	44.6	40.0	46.0	10.6	3.7	11.6	9.6	72.5
Search-o1	45.3	25.8	5.3	29.1	**70.2**	44.6	40.0	49.0	13.0	8.5	12.6	12.2	74.0
WebThinker	50.0	34.9	10.5	36.2	55.3	53.0	50.0	52.5	13.9	9.7	12.6	13.0	78.0
CodeAct	26.2	15.1	0.0	16.5	6.4	4.8	4.3	5.0	9.9	6.1	10.5	9.4	7.5
Multimodal Enhanced	23.8	9.1	0.0	12.6	4.3	0	4.3	4.0	9.3	9.8	8.4	9.2	10.5
Multi-Capability Enhanced													
Plan-and-Solve	28.6	18.2	0.0	18.9	44.7	33.7	24.3	33.0	10.2	4.9	7.3	8.8	57.5
ReAct	45.3	28.8	5.2	30.7	46.8	31.3	31.4	35.0	12.7	**11.0**	**20.0**	13.8	73.5
HiRA (ours)	**61.9**	**37.9**	**15.8**	**42.5**	59.6	**54.2**	**51.4**	**54.5**	**15.2**	**11.0**	13.7	**14.2**	**81.5**

(Qwen3-32B [45], QwQ-32B [34], DeepSeek-R1-32B [4]) and commercial SOTA models (GPT-4o [15], DeepSeek-R1 [4], o1-preview [30]).
(2) **Single-Capability Enhanced**: These methods augment reasoning with a specialized tool. We compare against **Search-o1** [24] and **WebThinker** [27] for search capabilities, **CodeAct** [38] for code execution, and a Multimodal Enhanced baseline. *Note regarding WebThinker*: To address potential concerns about implementation transparency, we utilize the **official repository** for WebThinker and Search-o1. While WebThinker supports report generation, we strictly evaluate its QA performance using the same query set, ensuring a direct comparison.
(3) **Multi-Capability Reasoning**: We include **Plan-and-Solve** [37] and **ReAct** [46] as representative agentic workflows. These baselines are implemented with the **exact same toolset** (Search, Code, Multimodal) as HiRA to isolate the impact of our hierarchical architecture.

4.3 Implementation Details

We primarily use **QwQ-32B** as the base model for both the Meta Reasoning Planner and Domain-Specialized Executors, and a same-sized Qwen2.5-Instruct as the Adaptive Reasoning Coordinator. To demonstrate the generalizability of our framework across model families, we explicitly include additional experiments using the **Qwen3 family** (specifically Qwen3-30B-A3B-Thinking). For all methods, we enforce a consistent context window of **128k tokens** to eliminate length bias. Inference parameters are set to temperature 0.7, top_p 0.95, and top_k 20.

For tool environments, we ensure that the prompt engineering for expert agents in our method remains consistent with the baselines, avoiding any specific optimizations or few-shot examples that would bias the results. Our toolset includes: (1) **Search**: The Bing Web Search API (US-EN region, top-10 results). (2) **Multimodal**: Qwen2.5-Omni-7B [43] invoked dynamically for image/audio understanding. (3) **Code**: A secure Python sandbox environment for computational tasks and file processing.

4.4 Designs of Domain-Specific Agents

We designed four specialized agents to address distinct functional requirements:

(1) Simple Search Agent: Optimized for efficient information gathering and fact verification. It operates by generating query plans, executing searches, and synthesizing answers. **(2) Deep Search Agent**: Designed for complex web exploration, enabling search engine utilization and link interaction. We adopt the WebThinker [27] implementation for this role. **(3) Computational Reasoning Agent**: Autonomousy invokes a sandboxed Python interpreter for code-assisted reasoning and data acquisition (e.g., file I/O, web scraping). Following Search-o1 [24], we utilize the QwQ-32B model to drive the Python environment. **(4) Multimodal Agent**: Integrated with multimodal capabilities to process diverse data formats. We employ Qwen-omni-7B as the embedded tool, enabling the handling of multimodal inputs directly within the reasoning chain.

Table 2: Ablation studies of HiRA, showing results for each layer. GAIA-B refers to GAIA queries without associated files, while GAIA-F refers to the subset with files.

Method	GAIA-B	GAIA-F	Web.	HLE	Simp.	Avg.
HiRA	**42.5**	**42.1**	**54.5**	**14.2**	**81.5**	**44.9**
Coordinator Layer						
w/o Reasoning Trans.	33.9	36.8	44.5	10.4	76.5	40.4
w/o Memory	37.8	31.6	52.0	11.8	79.0	42.4
Executor Layer						
w/o Search	15.7	31.6	4.0	12.4	9.5	14.6
w/o Code	33.9	28.9	51.5	12.8	76.5	40.7
w/o Multimodal	36.2	36.8	55.0	13.6	81.0	44.5

5 Experimental Results

5.1 Main Results

As shown in Table 1, we evaluate our method on the deep search tasks against several baselines that integrate reasoning with tool usage. We have the following observations: (1) **Overall performance superiority**: Our method consistently outperforms all baseline methods. It significantly surpasses direct reasoning models without tool usage and achieves notable improvements over existing tool-augmented approaches. Compared to the strongest search agent baseline WebThinker, our approach demonstrates substantial advantages on both complex tasks (GAIA and HLE). (2) **Hierarchical reasoning design advantages**: Results show that our hierarchical design achieves better performance when using the same set of tools. For Plan-and-Solve which relies on a fixed plan executed sequentially, performs poorly (e.g., 18.9 in GAIA), highlighting the necessity of dynamic planning during reasoning. While ReAct enables dynamic planning by integrating multiple tools into the reasoning chain, its performance suffers in multi-tool scenarios such as GAIA due to the overhead of tool selection and noisy intermediate tool outputs. In contrast, our method outperforms WebThinker on GAIA, which requires diverse capabilities, while also achieving superior results on general web search tasks. (3) **Robustness across task complexity**: Our framework shows moderate gains on simpler tasks (e.g., SimpleQA and WebWalkerQA), but exhibits much larger improvements on more complex tasks, demonstrating its strength in handling complex reasoning scenarios.

5.2 Ablation Study

We conduct ablation studies to investigate the contribution of each component within the framework by removing modules from the Coordinator Layer and the Executor Layer. Results are shown in Table 2.

In the Coordinator Layer, removing the Reasoning Transfer mechanism results in a substantial performance decline, particularly on complex reasoning tasks, with nearly 30% decrease in performance. In contrast, the Memory mechanism has a more limited effect, except on file-related tasks (approx. 10% drop), indicating that the resource component in memory effectively facilitates information propagation.

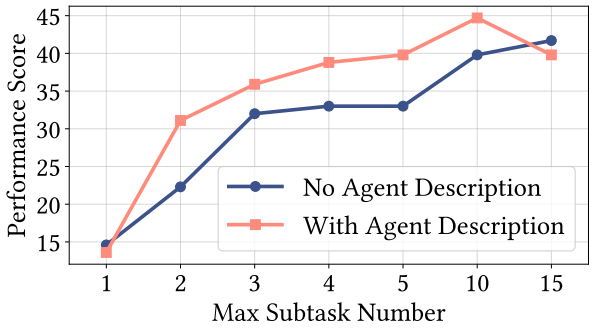


Figure 3: Performance comparison on whether the expert agent description is provided to the meta planner and maximum number of sub-tasks limit.

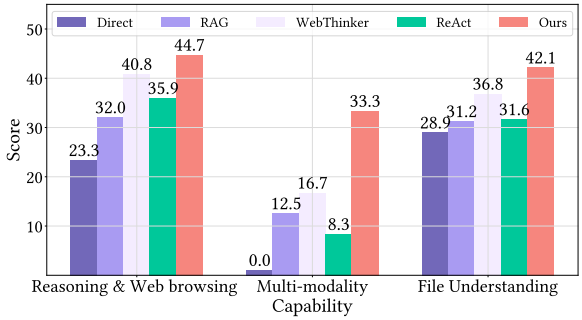


Figure 4: Comparison of our method with baselines on three GAIA subsets, evaluating performance across different dimensions of capability.

In the Executor Layer, removing individual expert agents leads to substantial performance degradation. Among them, removing the Search Agent causes the most severe drop across all datasets, highlighting the essential role of information seeking on web. Similarly, removing the Code Agent significantly impacts performance on multi-functional datasets such as GAIA, showing its importance for general-purpose tasks. Removing the Multimodal Agent results in a slight drop on GAIA, where some cases require multimodal capabilities.

5.3 Generalization and Effectiveness of Meta Planner

In our framework, the meta-planner receives expert capability information for better subtask planning, while also limiting the number of subtasks to control computational cost. These two factors affect both generalization (supporting more agents) and effectiveness (achieve higher performance). To evaluate both aspects, we introduce an additional setting where no expert information is provided to the meta-planner, and we vary the maximum number of allowed subtasks. Results are shown in Figure 3, leading to two key observations: (1) **Decoupled meta-planner design**: Even without expert information, the meta-planner achieves comparable performance. This suggests that, within our architecture, the meta-planner and

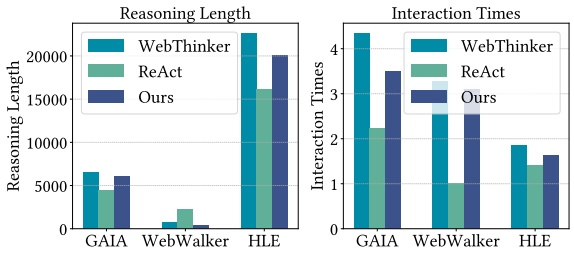


Figure 5: Comparison of different methods in terms of reasoning length (number of output tokens during model inference) and interaction times (number of interactions with the environment during inference) in three datasets.

Table 3: Comparison of the operational efficiency of various methods on GAIA-B, including inference time, number of environment interactions, and final score.

Method	Avg Time (s)	Avg Inter.	Score
Direct Gen	12.3	0	18.9
RAG	25.7	1.0	25.2
WebThinker	47.3	4.3	36.2
ReAct	31.4	2.2	30.7
HiRA	39.7	3.5	42.5

expert agents are relatively decoupled—the generated subtasks primarily depend on the task semantics rather than the specific agents, enabling better scalability to new agents. (2) **Subtask scaling trade-off**: As the number of subtasks increases, performance first improves then degrades. This resembles an inference-time scaling effect: allowing more subtasks enables deeper reasoning chains, while overly restricting subtasks may prevent sufficient exploration. However, excessive subtasks may introduce inefficient plans, eventually hurting performance.

5.4 Capability Analysis

Beyond the main experiments, we further evaluate our system on additional subsets of GAIA to assess its capabilities beyond web search (e.g., multimodal understanding, file understanding), which are also critical for more comprehensive information acquisition. As shown in Figure 4, our method achieves the best performance across all capability dimensions, demonstrating its broad applicability. While baselines like ReAct perform well in purely textual reasoning and web browsing, our integration of the DeepSearch agent yields superior results. Moreover, due to our multimodal architecture, we achieve additional gains in non-textual tasks. It is worth noting that, although ReAct can invoke multimodal and code tools, its planning model struggles to coordinate multiple tools simultaneously, often leading to weak performance (e.g., 8.3 vs. 12.5 in multimodal tasks), in some cases even underperforming pure search-based approaches.

Table 4: Generalization performance using Qwen3-30B-A3B as the backbone model.

Method	GAIA	HLE	WebWalker	SimpleQA
Plan-and-Solve	21.6	9.4	38.0	64.0
ReAct	29.1	11.6	40.5	77.5
WebThinker	36.2	12.2	50.5	79.0
HiRA (Ours)	**39.4**	**13.6**	**57.5**	**82.5**

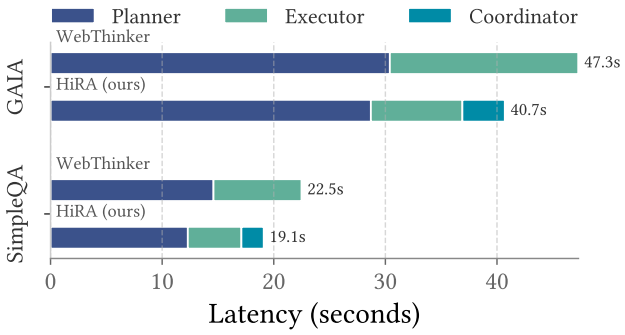


Figure 6: Time consumption of different stages during the execution of WebThinker and HiRA. The results represent the average of five runs.

5.5 Efficiency Analysis

We evaluate HiRA against baselines in terms of inference tokens, interactions, and time. As shown in Figure 5 and Table 3, HiRA optimizes the cost-performance trade-off. It outperforms the strongest baseline, WebThinker (42.5 vs. 36.2), while reducing environment interactions (3.5 vs. 4.3) and inference time (39.7s vs. 47.3s). While ReAct offers lower latency, its shallow reasoning significantly harms performance (30.7 vs. 42.5 on GAIA). Conversely, HiRA yields substantial gains over RAG with only a marginal time increase, effectively balancing latency and depth. A module-wise latency breakdown (Figure 6) reveals that introducing the Coordinator reduces total latency ($\sim$ 16% speedup vs. WebThinker). This efficiency stems from two factors. First, information distillation: the Coordinator filters execution noise into summaries, maintaining a lean context window to accelerate the Planner's decoding (28.7s vs. 30.4s). Second, hierarchical routing: the *Executor* phase is significantly faster (8.2s vs. 16.9s on GAIA) because HiRA dynamically routes simple tasks to a lightweight RAG agent, reserving heavy Code or Deep Search agents only for complex sub-tasks.

5.6 Generalization Across Model Families

To verify that HiRA is model-agnostic, we evaluate the framework using another reasoning model Qwen3-30B-A3B. As shown in Table 4, HiRA maintains its performance advantage over ReAct and WebThinker, confirming the benefits of decoupled planning and execution across different underlying reasoning models.

Table 5: Coordinator routing accuracy comparing GAIA and SimpleQA datasets.

Metric	GAIA	SimpleQA
Overall Accuracy	75.6% (273/361)	58.8% (77/131)
Breakdown by Case:		
Correct Answer	67.4% (95/141)	58.8% (60/102)
Incorrect Answer	80.9% (178/220)	58.6% (17/29)

5.7 Coordinator Accuracy and Bottleneck

To identify the source of errors in HiRA, we employ GPT-4o to evaluate the Coordinator's routing accuracy. As shown in Table 5, the Coordinator achieves a routing accuracy of 75.6% on GAIA . However, on the relatively simpler SimpleQA dataset, GPT-4o assign marginally lower accuracy scores. Manual inspection reveals that this discrepancy primarily stems from the underestimation of query complexity by both the Coordinator and the GPT-4o judge, leading to routing assignments to simpler agents. This validates the rationality of our architecture but suggests the Coordinator requires further alignment to discern subtle variations in task difficulty.

Notably, routing accuracy is significantly higher for incorrect responses than for correct ones. This counter-intuitive finding confirms the Coordinator's robustness: failures in HiRA are rarely due to misrouting. Instead, errors primarily arise from Executor limitations or synthesis failures by the Meta Planner. Thus, the main bottleneck lies in the execution capabilities of experts, not the coordination mechanism.

We acknowledge that the current framework lacks a dedicated verification mechanism at the Meta Planner level to detect plausible but incorrect Executor outputs. While the Reasoning Distillation process provides partial mitigation by enabling the Planner to review the distilled reasoning logic (not just final answers) and autonomously reflect on the results, incorporating an explicit verification step—such as cross-referencing results across multiple executors or employing a dedicated fact-checking agent—is an important direction for future work.

6 Conclusion

In this paper, we propose HiRA, a hierarchical reasoning framework that overcomes limitations of monolithic models in deep search tasks. Our multi-agent architecture combines a Meta reasoning planner, Adaptive Reasoning Coordinator, and Domain-Specialized Executors to enable scalable, modular reasoning through specialized cognitive components rather than rigid pipelines. Using dual-channel memory and coordinated task routing, HiRA supports coherent knowledge synthesis and dynamic integration of diverse tool capabilities. Experiments across four complex, multi-modal deep search tasks show HiRA significantly outperforms conventional RAG systems and existing agent-based methods, highlighting the effectiveness of our decoupled design.

Acknowledgements

This work was supported by National Natural Science Foundation of China No. 62272467 and No. 625B2178. The work was partially done at Engineering Research Center of Next-Generation Intelligent Search and Recommendation, MOE.

References

[1] Ahsan Bilal, Muhammad Ahmed Mohsin, Muhammad Umer, Muhammad Awais Khan Bangash, and Muhammad Ali Jamshed. 2025. Meta-Thinking in LLMs via Multi-Agent Reinforcement Learning: A Survey. arXiv:2504.14520 [cs.AI] https://arxiv.org/abs/2504.14520

[2] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2022. Improving Language Models by Retrieving from Trillions of Tokens. In *Proc. of ICML*. 2206–2240.

[3] Chi-Min Chan, Chunpu Xu, Ruibin Yuan, Hongyin Luo, Wei Xue, Yike Guo, and Jie Fu. 2024. RQ-RAG: Learning to Refine Queries for Retrieval Augmented Generation. *CoRR* abs/2404.00610 (2024). https://doi.org/10.48550/ARXIV.2404.00610 arXiv:2404.00610

[4] DeepSeek-AI, Daya Guo, and et al. Dejian Yang. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *CoRR* abs/2501.12948 (2025). https://doi.org/10.48550/ARXIV.2501.12948 arXiv:2501.12948

[5] Guanting Dong, Licheng Bao, Zhongyuan Wang, Kangzhi Zhao, Xiaoxi Li, Jiajie Jin, Jinghan Yang, Hangyu Mao, Fuzheng Zhang, Kun Gai, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. 2025. Agentic Entropy-Balanced Policy Optimization. *arXiv preprint arXiv:2510.14545* (2025).

[6] Guanting Dong, Yifei Chen, Xiaoxi Li, Jiajie Jin, Hongjin Qian, Yutao Zhu, Hangyu Mao, Guorui Zhou, Zhicheng Dou, and Ji-Rong Wen. 2025. Tool-Star: Empowering LLM-Brained Multi-Tool Reasoner via Reinforcement Learning. *arXiv preprint arXiv:2505.16410* (2025).

[7] Guanting Dong, Jiajie Jin, Xiaoxi Li, Yutao Zhu, and Zhicheng Dou. 2025. RAG-Critic: Leveraging Automated Critic-Guided Agentic Workflow for Retrieval Augmented Generation. *arXiv preprint* (2025).

[8] Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. 2025. Agentic Reinforced Policy Optimization. *arXiv preprint arXiv:2507.19849* (2025).

[9] Guanting Dong, Yutao Zhu, Chenghao Zhang, Zechen Wang, Zhicheng Dou, and Ji-Rong Wen. 2024. Understand What LLM Needs: Dual Preference Alignment for Retrieval-Augmented Generation. *CoRR* abs/2406.18676 (2024). https://doi.org/10.48550/ARXIV.2406.18676 arXiv:2406.18676

[10] Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2025. Plan-and-Act: Improving Planning of Agents for Long-Horizon Tasks. In *Proceedings of the 42nd International Conference on Machine Learning (Proceedings of Machine Learning Research)*. PMLR. https://proceedings.mlr.press/v267/erdogan25a.html

[11] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2024. Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv:2312.10997 [cs.CL]

[12] Grok. 2025. Grok 3 Beta — The Age of Reasoning Agents. https://x.ai/news/grok-3.

[13] Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net. https://openreview.net/forum?id=VtmBAGCN7o

[14] Xinming Hou, Mingming Yang, Wenxiang Jiao, Xing Wang, Zhaopeng Tu, and Wayne Xin Zhao. 2024. CoAct: A Global-Local Hierarchy for Autonomous Agent Collaboration. arXiv:2406.13381 [cs.CL] https://arxiv.org/abs/2406.13381

[15] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276* (2024).

[16] 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* (2025).

[17] Jiajie Jin, Xiaoxi Li, Guanting Dong, Yuyao Zhang, Yutao Zhu, Yongkang Wu, Zhonghua Li, Qi Ye, and Zhicheng Dou. 2025. Hierarchical Document Refinement for Long-context Retrieval-augmented Generation. *arXiv preprint arXiv:2505.10413* (2025).

[18] Jiajie Jin, Yanzhao Zhang, Mingxin Li, Dingkun Long, Pengjun Xie, Yutao Zhu, and Zhicheng Dou. 2026. LaSER: Internalizing Explicit Reasoning into Latent

Space for Dense Retrieval. *arXiv preprint arXiv:2603.01425* (2026).

[19] Jiajie Jin, Yuyao Zhang, Yimeng Xu, Hongjin Qian, Yutao Zhu, and Zhicheng Dou. 2025. FinSight: Towards Real-World Financial Deep Research. *arXiv preprint arXiv:2510.16844* (2025).

[20] Jiajie Jin, Yutao Zhu, Zhicheng Dou, Guanting Dong, Xinyu Yang, Chenghao Zhang, Tong Zhao, Zhao Yang, and Ji-Rong Wen. 2025. FlashRAG: A Modular Toolkit for Efficient Retrieval-Augmented Generation Research. In *Companion Proceedings of the ACM on Web Conference 2025*. ACM, 737–740. https://doi.org/10.1145/3701716.3715313

[21] Jiajie Jin, Yutao Zhu, Yujia Zhou, and Zhicheng Dou. 2024. BIDER: Bridging Knowledge Inconsistency for Efficient Retrieval-Augmented LLMs via Key Supporting Evidence. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, 750–761. https://doi.org/10.18653/V1/2024.FINDINGS-ACL.42

[22] Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W. Mahoney, Kurt Keutzer, and Amir Gholami. 2024. An LLM Compiler for Parallel Function Calling. In *Proceedings of the 41st International Conference on Machine Learning (Proceedings of Machine Learning Research)*. PMLR, 24370–24391. https://proceedings.mlr.press/v235/kim24y.html

[23] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html

[24] Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic Search-Enhanced Large Reasoning Models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 5420–5438. https://doi.org/10.18653/v1/2025.emnlp-main.276

[25] Xiaoxi Li, Wenxiang Jiao, Jiarui Jin, Guanting Dong, Jiajie Jin, Yinuo Wang, Hao Wang, Yutao Zhu, Ji-Rong Wen, Yuan Lu, and Zhicheng Dou. 2025. DeepAgent: A General Reasoning Agent with Scalable Toolsets. *arXiv preprint arXiv:2510.21618* (2025).

[26] Xiaoxi Li, Wenxiang Jiao, Jiarui Jin, Shijian Wang, Guanting Dong, Jiajie Jin, Hao Wang, Yinuo Wang, Ji-Rong Wen, Yuan Lu, and Zhicheng Dou. 2026. OmniGAIA: Towards Native Omni-Modal AI Agents. *arXiv preprint arXiv:2602.22897* (2026).

[27] Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yutao Zhu, Yongkang Wu, Ji-Rong Wen, and Zhicheng Dou. 2025. WebThinker: Empowering Large Reasoning Models with Deep Research Capability. *CoRR* abs/2504.21776 (2025). https://doi.org/10.48550/ARXIV.2504.21776 arXiv:2504.21776

[28] Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2024. GAIA: a benchmark for General AI Assistants. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net. https://openreview.net/forum?id=fibxvahvs3

[29] OpenAI. 2024. Learning to Reason with LLMs. https://openai.com/index/learning-to-reason-with-llms.

[30] OpenAI. 2024. Learning to Reason with LLMs. https://openai.com/index/learning-to-reason-with-llms.

[31] Long Phan and et al. Alice Gatti. 2025. Humanity's Last Exam. *CoRR* abs/2501.14249 (2025). https://doi.org/10.48550/ARXIV.2501.14249 arXiv:2501.14249

[32] Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing Retrieval-Augmented Large Language Models with Iterative Retrieval-Generation Synergy. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. Association for Computational Linguistics, 9248–9274. https://doi.org/10.18653/v1/2023.findings-emnlp.620

[33] 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. *arXiv preprint arXiv:2503.05592* (2025).

[34] Qwen Team. 2024. Qwq: Reflect deeply on the boundaries of the unknown. *Hugging Face* (2024).

[35] 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 (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, 10014–10037. https://doi.org/10.18653/V1/2023.ACL-LONG.557

[36] Ziyu Wan, Yunxiang Li, Xiaoyu Wen, Yan Song, Hanjing Wang, Linyi Yang, Mark Schmidt, Jun Wang, Weinan Zhang, Shuyue Hu, and Ying Wen. 2025. ReMA: Learning to Meta-think for LLMs with Multi-Agent Reinforcement Learning. arXiv:2503.09501 [cs.AI] https://arxiv.org/abs/2503.09501

[37] 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. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2609–2634. https://doi.org/10.18653/v1/2023.acl-long.147

[38] Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. 2024. Executable code actions elicit better llm agents. In *Forty-first International Conference on Machine Learning*.

[39] Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. 2024. Measuring short-form factuality in large language models. arXiv:2411.04368 [cs.CL] https://arxiv.org/abs/2411.04368

[40] Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, and Fei Huang. 2025. WebWalker: Benchmarking LLMs in Web Traversal. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 10290–10305. https://aclanthology.org/2025.acl-long.508/

[41] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. 2023. AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation Framework. *CoRR* abs/2308.08155 (2023). https://doi.org/10.48550/ARXIV.2308.08155 arXiv:2308.08155

[42] Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2023. RECOMP: Improving Retrieval-Augmented LMs with Compression and Selective Augmentation. *CoRR* abs/2310.04408 (2023). https://doi.org/10.48550/ARXIV.2310.04408 arXiv:2310.04408

[43] Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, Bin Zhang, Xiong Wang, Yunfei Chu, and Junyang Lin. 2025. Qwen2.5-Omni Technical Report. arXiv:2503.20215 [cs.CL] https://arxiv.org/abs/2503.20215

[44] Renjun Xu and Jingwen Peng. 2025. A Comprehensive Survey of Deep Research: Systems, Methodologies, and Applications. arXiv:2506.12594 [cs.AI] https://arxiv.org/abs/2506.12594

[45] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 Technical Report. arXiv:2505.09388 [cs.CL]

[46] Shunyu Yao, Jeffrey Zhao, Dian Yu, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. ReAct: Synergizing Reasoning and Acting in Language Models. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*. https://openreview.net/forum?id=tvI4u1ylcqs

[47] Yuyao Zhang, Zhicheng Dou, Xiaoxi Li, Jiajie Jin, Yongkang Wu, Zhonghua Li, Ye Qi, and Ji-Rong Wen. 2025. Neuro-Symbolic Query Compiler. In *Findings of the Association for Computational Linguistics: ACL 2025*. Association for Computational Linguistics, 12138–12155. https://aclanthology.org/2025.findings-acl.628/