

FLASH-SEARCHER: FAST AND EFFECTIVE WEB AGENTS VIA DAG-BASED PARALLEL EXECUTION

Anonymous authors

Paper under double-blind review

ABSTRACT

Large language models (LLMs) have demonstrated remarkable capabilities in complex agent reasoning tasks when equipped with external tools. However, current frameworks predominantly rely on sequential processing, leading to inefficient execution particularly for tasks requiring extensive tool interaction. This paper introduces FLASH-SEARCHER, a novel parallel agent reasoning framework that fundamentally reimagines the execution paradigm from sequential chains to directed acyclic graphs (DAGs). FLASH-SEARCHER decomposes complex tasks into subtasks with explicit dependencies, enabling concurrent execution of independent reasoning paths while maintaining logical constraints. Through dynamic workflow optimization, our framework continuously refines the execution graph based on intermediate results, effectively integrating summary module. Comprehensive evaluations across multiple benchmarks demonstrate that FLASH-SEARCHER consistently outperforms existing approaches. Specifically, it achieves **67.7%** accuracy on BrowseComp and **83%** on xbench-DeepSearch, while reducing agent execution steps by up to **35%** compared to current frameworks. Furthermore, when distilling this parallel reasoning pipeline into single models, we observe substantial performance gains across diverse backbone architectures, underscoring the generalizability of our methodology. Our work thus represents a significant advance in agent architecture design, offering a more scalable and efficient paradigm for complex reasoning tasks.

1 INTRODUCTION

Recent advances in tool-augmented agents and multi-agent systems (MAS) (Dorri et al., 2018; Canese et al., 2021; Zhou et al., 2023c; 2024; Zhu et al., 2025a;b; Qiu et al., 2025; Roucher et al., 2025; Tang et al., 2025; Team, 2025) have demonstrated remarkable capabilities in complex problem-solving tasks, showcasing how collaborative agent frameworks can effectively address challenges requiring diverse reasoning abilities and tool manipulation. These systems leverage specialized agents with distinct roles, enabling sophisticated planning, reasoning, and tool utilization to solve tasks that would be challenging for single-agent approaches. Concurrently, research efforts have focused on Tool-Integrated Reasoning (TIR) (Jin et al., 2025a; Li et al., 2025c;d; Wu et al., 2025a; Sun et al., 2025; Zhang et al., 2025a; Zheng et al., 2025; Xue et al., 2025) approaches, which aim to incorporate the capabilities of tool execution or multi-agent systems into a single model through specialized training methodologies.

Despite their impressive performance, both MAS and TIR approaches face significant limitations when addressing general complex tasks. Multi-agent systems suffer from inefficient tool utilization, excessively long reasoning chains, and prolonged execution times due to sequential processing and redundant communication, while TIR methods encounter reasoning efficiency bottlenecks with chains frequently exceeding context window limitations. These challenges intensify in complex scenarios requiring deep research capabilities, where MAS and TIR methodologies incorporate additional verification mechanisms (reflection, self-critique, iterative refinement) to enhance reliability—at the cost of substantially increased computational overhead. When solving complex tasks, current agent frameworks typically require more than 20 interaction steps (Wang et al., 2025; Roucher et al., 2025; Hu et al., 2025), with execution times stretching to hours. This creates a sharp tension between solution quality and computational efficiency, severely limiting practical viability in user-responsive applications. *When confronted with complex tasks inducing unavoidable latency,*

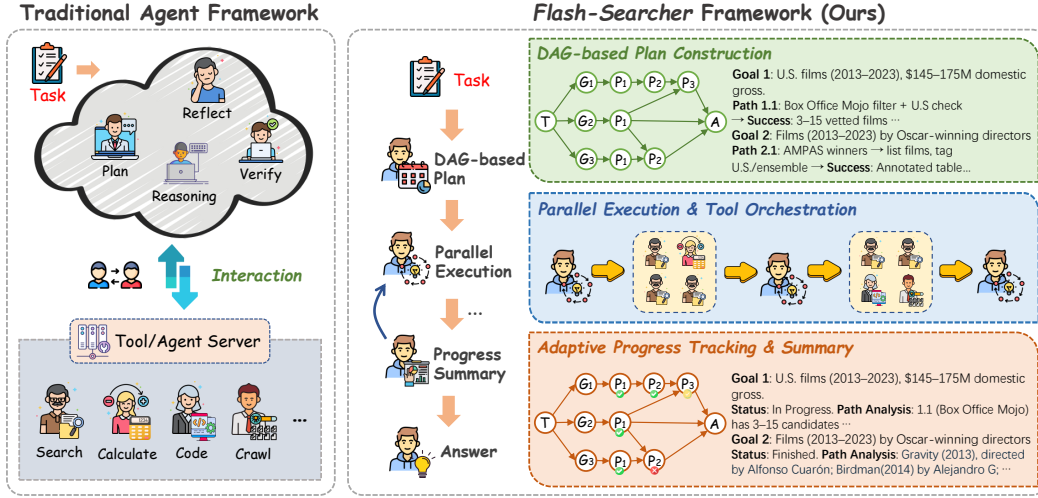


Figure 1: Overview of FLASH-SEARCHER: Framework and Key Components.

do users deem the better performance necessary enough to justify tolerating or paying for these delays?

To address these critical challenges, we introduce FLASH-SEARCHER, a novel parallel agent reasoning framework that fundamentally reimagines how agents collaborate to solve complex tasks. Building upon recent empirical advances in reasoning models, our approach leverages these models’ enhanced capabilities in simultaneously managing multiple cognitive threads. As illustrated in Figure 1, unlike traditional approaches that adhere to strict sequential processing, FLASH-SEARCHER decomposes the original task into multiple parallel execution paths, orchestrated via carefully designed agent workflows. This parallelization allows multiple reasoning paths to progress simultaneously while intelligently managing tool calls across different execution branches. The FLASH-SEARCHER framework redefines the efficiency-effectiveness frontier in complex task solving through key innovations: 1) adaptive decomposition and parallelization of tasks into concurrent subtasks with dynamic strategy adjustment, 2) dependency-aware reasoning graph management to model information dependencies and 3) optimize critical paths/information flow, and proactive information retrieval with knowledge sharing to anticipate downstream needs and reduce redundant interaction steps.

Our extensive evaluations demonstrate that FLASH-SEARCHER achieves state-of-the-art performance across multiple challenging benchmarks. our FLASH-SEARCHER (with GPT-5mini) reduces the average agent execution steps by **35%** (11.2 → 7.4 steps) and shortens the overall execution time by **~65%** compared to OAgents. Despite this dramatic efficiency improvement, FLASH-SEARCHER (with GPT-5) achieves an impressive average performance of **82.5%** on GAIA benchmark. Furthermore, on more challenging benchmarks such as xbench, HLE and BrowseComp, FLASH-SEARCHER achieves performance metrics of **83.0**, **44.0** and **67.7** respectively, surpassing current state-of-the-art methods. Furthermore, to validate the generalizability of our approach, we constructed FLASH-SEARCHER execution trajectories based on collected web agent data and conducted post-training on the Qwen-2.5 family of open-source models. This lightweight adaptation achieves **68.0** performance on the xbench-DeepSearch benchmark, a **29.3** improvement over WebDancer, verifying the effective transfer of the parallel agent paradigm to open-source models with minimal additional training.

In summary, our contributions are as follows:

- We present a novel parallel agent reasoning framework that substantially reduces execution steps while achieving SOTA performance across various benchmarks.
- High-quality parallel reasoning trajectories, systematically curated and constructed for model post-training, significantly boost performance on complex evaluation tasks.
- Experimental results demonstrate the effectiveness of lightweight post-training in propagating parallel agent strategies to open-source models - achieving comparable results to multi-agent systems.

- We fully open-source pipeline and datasets of FLASH-SEARCHER to catalyze research on search agents and models.

2 RELATED WORK

2.1 MULTI-AGENT SYSTEM.

Recent research has highlighted the effectiveness of multi-agent systems in addressing complex real-world challenges through collaborative agent frameworks. These systems typically employ multiple specialized agents with distinct roles, thereby supporting advanced planning, multi-turn reasoning, tool utilization, and environment interaction (Zhou et al., 2023c; 2024; Jin et al., 2025b; Zhu et al., 2025a;b; Mai et al., 2025; Hu et al., 2024; Tang et al., 2025; Shi et al., 2025; Tang et al., 2025; Zhou et al., 2023b). Early multi-agent systems such as CAMEL (Li et al., 2023) showed that dialog between agents can elicit stepwise reasoning through role-playing. Subsequent frameworks, including MetaGPT (Hong et al., 2024) and ChatDev (Qian et al., 2023), formalized this approach by implementing structured execution pipelines with dedicated roles such as manager, designer, and coder. Other approaches, like Magnetic-One (Fourney et al., 2024) and Smolagents (Roucher et al., 2025), incorporate a central planner that dynamically delegates subtasks to specialized tool-based agents. AgentVerse (Chen et al., 2023) refines collaborative reasoning via a recruitment–decision–execution–evaluation cycle, enhancing reflection and coordination. Workforce (Hu et al., 2025) decouples planning, coordination, and execution into modular agents, enabling efficient domain transfer through plug-and-play workers. Alita (Qiu et al., 2025) proposes autonomous tool exploration via iterative trial-and-error, expanding capabilities by transforming multi-attempt tasks into single-attempt ones. However, beyond performance, the latency in these complex multi-agent frameworks remains understudied.

2.2 EFFICIENT FRAMEWORK.

To address the efficiency bottlenecks inherent in existing agent frameworks, Tool-Integrated Reasoning (TIR) has recently emerged as a prominent research direction. Early efforts primarily adopted prompt-based strategies—such as Search-o1 (Li et al., 2025c)—which employ static templates to instantiate fixed *Thought–Action–Observation* loops, thereby enabling rudimentary tool-augmented reasoning. More recent work has pivoted toward post-training paradigms (Jin et al., 2025a; Li et al., 2025d; Wu et al., 2025a; Li et al., 2025a; Tao et al., 2025; Sun et al., 2025; Xue et al., 2025; Li et al., 2025b; Nguyen et al., 2025), where agents are refined via task-specific fine-tuning to enhance performance. Despite their empirical gains, these approaches typically enforce narrowly scoped execution workflows, which severely limit their adaptability and scalability in open-domain, real-world environments. These challenges have motivated a broader effort to improve the efficiency and scalability of reasoning-enabled agents. Recent advances have focused on two key directions: optimizing agent pipelines and parallelizing search processes. Efficient Agents (Wang et al., 2025) conducts a comprehensive analysis of core agent modules (workflow design, tool invocation, and memory architecture) to systematically balance performance and cost. Similarly, ParallelSearch (Zhao et al., 2025) trains models to detect parallelizable query structures, decomposing complex queries into independent sub-queries for retrieval tasks, resulting in significant performance gains in search-based tasks. However, existing systems remain constrained by isolated reasoning-execution loops or the prolonged cycles introduced by multi-step verification, highlighting the need for more efficient approaches to agent.

3 METHOD

3.1 PRELIMINARIES

Tool-Augmented Agents. Tool-augmented agents enhance the capabilities of LLMs by seamlessly integrating external tools to perform actions such as information retrieval, mathematical computation, and code execution. This paradigm mitigates the inherent limitations of parametric knowledge through a structured tool-calling pipeline. Formally, the agent–environment interaction is modeled as a Markov decision process, wherein each tool invocation induces a state transition driven by environmental feedback. At timestep t , the agent selects a tool-calling action $a_t \in \mathcal{A}$ —where \mathcal{A}

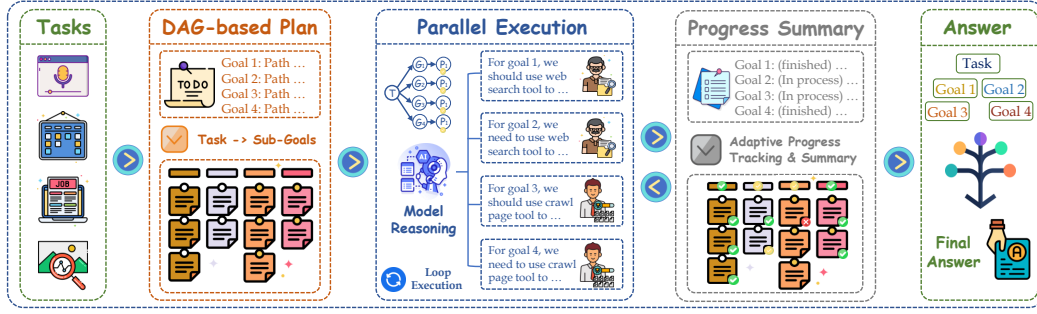


Figure 2: The pipeline of FLASH-SEARCHER.

denotes the action space comprising available tools—based on the current state s_t , and receives an observation $o_t \sim \mathcal{P}(\cdot | s_t, a_t)$ from the tool environment. The state transition function is defined as:

$$s_{t+1} = g(s_t, a_t, o_t), \quad (1)$$

where $g : \mathcal{S} \times \mathcal{A} \times \mathcal{O} \rightarrow \mathcal{S}$ represents a state update function incorporating task, action history, and structured tool outputs into the new state representation $s_{t+1} \in \mathcal{S}$.

Multi-Agent Systems. Consider a set of agents indexed by \mathcal{I} , where each agent is denoted as a_i for $i \in \mathcal{I}$. Each agent maintains a local state s_t^i and possesses specialized capabilities. The global system state at time t is defined as $S_t = \{s_t^1, s_t^2, \dots, s_t^n, c_t\}$, which aggregates all local states along with a shared context c_t . Agents coordinate through inter-agent communication protocols to optimize a common objective function $\mathcal{U}(S_t)$. The evolution of the global state follows:

$$S_{t+1} = f(\{a_t^i\}_{i \in \mathcal{I}}, S_t, O_t), \quad (2)$$

where $a_t^i \in \mathcal{A}$ denotes the action executed by agent i at timestep t , $O_t = \{o_t^i\}_{i \in \mathcal{I}}$ represents the collection of observations from all agents, and f integrates individual actions, the current global state, and observations to produce the next global state.

Existing approaches often adopt sequential execution with reflection and verification, prolonging task completion. Complex tasks may require 40+ interactions, introducing substantial latency. This sequential dependency creates a fundamental quality–efficiency trade-off, hindering real-world deployment.

3.2 FLASH-SEARCHER: PARALLEL AGENT REASONING FRAMEWORK

To overcome the inherent inefficiencies of sequential execution in conventional agent frameworks, we introduce FLASH-SEARCHER, a novel parallel reasoning framework that reformulates complex task solving as structured concurrency. Our approach transcodes the traditional linear workflow into a dynamic directed acyclic graph (DAG) plan, achieving substantial efficiency gains while preserving execution coherence. The full pipeline of FLASH-SEARCHER is illustrated in Figure 2.

DAG-based Plan Construction. Given a composite task T , FLASH-SEARCHER employs a decomposition function \mathcal{D} that identifies constituent subtasks and their interdependencies, yielding a DAG-based plan:

$$\mathcal{D}(T) = G_{\text{plan}} = (V, E), \quad (3)$$

where $V = \{t_1, t_2, \dots, t_n\}$ denotes subtasks and $E \subseteq V \times V$ captures prerequisite relations. Each directed edge $(t_i, t_j) \in E$ encodes that t_i must precede t_j .

Parallel Inferential Execution & Tool Orchestration. At execution step t , FLASH-SEARCHER selects candidate subtasks from the pending set $\mathcal{P}_t \subseteq V$:

$$\mathcal{E}(G_t, \mathcal{P}_t) = \{v_i \in \mathcal{P}_t \mid \varphi(v_i, G_t, s_t) = 1\}, \quad (4)$$

where $\varphi(\cdot)$ is a readiness predicate. Unlike strict topological scheduling, φ permits *aggressive parallelization*: a subtask v_i may be scheduled if either (i) all its prerequisites are complete, or (ii) partial execution can provide auxiliary signals for dependency verification. Thus, φ formalizes cross-validation as a hybrid criterion, blending dependency satisfaction and heuristic consistency

checks. During execution, multiple subtasks $\mathcal{E}(G_t, \mathcal{P}_t)$ are processed in parallel via tool or agent invocations. The system integrates observations into the reasoning state:

$$s_{t+1} = \left\{ \left(s_t, \{a_t^{(k)}\}_{k=1}^m, \{o_t^{(k)}\}_{k=1}^m \right) \right\}, \quad (5)$$

where $a_t^{(k)}$ and $o_t^{(k)}$ denote the action and observation of the k -th parallel execution, and $\{$ integrates the results via structured aggregation and performs state transitions based on the aggregated information.

Adaptive Progress Tracking & Summarization. To reflect execution progress, FLASH-SEARCHER periodically updates the DAG-based plan every Δ steps:

$$G_{\text{plan}}^{t+\Delta} = \mathcal{R}(G_{\text{plan}}^t, \mathcal{C}_t, \mathcal{P}_t, s_t), \quad (6)$$

where \mathcal{C}_t is the set of completed subtasks. The refinement rule \mathcal{R} eliminates resolved nodes, revalidates unresolved dependencies based on cross-validation outcomes, and dynamically inserts new decomposition nodes if needed. The interval Δ can be flexibly specified: a smaller Δ increases the frequency of plan updates, ensuring faster task adaptation and responsiveness; a larger Δ suppresses excessive optimization, reducing computational overhead in complex or stable tasks.

By integrating DAG-based decomposition, controlled aggressive parallelization, and periodic DAG optimization, FLASH-SEARCHER mitigates the sequential bottleneck of existing reasoning architectures. The framework strikes a balance between theoretical soundness and practical efficiency, ensuring reproducibility and scalability in complex real-world tasks. The full FLASH-SEARCHER pipeline is formally presented in Algorithm 1.

Algorithm 1 FLASH-SEARCHER Framework

Require: Composite task T

```

1:  $G_{\text{plan}} \leftarrow \mathcal{D}(T)$ 
2: Initialize  $s_0, \mathcal{P}_0 \leftarrow V, \mathcal{C}_0 \leftarrow \emptyset$ 
3:  $t \leftarrow 0$ 
4: while  $\mathcal{P}_t \neq \emptyset$  do
5:    $\mathcal{E}_t \leftarrow \{v \in \mathcal{P}_t \mid \varphi(v, G_t, s_t) = 1\}$ 
6:   Execute subtasks in  $\mathcal{E}_t$  in parallel
7:   Collect results  $\{o_t^{(k)}\}$  and update  $s_{t+1} = \{(s_t, \{a_t^{(k)}\}, \{o_t^{(k)}\})\}$ 
8:    $\mathcal{C}_{t+1} \leftarrow \mathcal{C}_t \cup \text{completed subtasks}$ 
9:    $\mathcal{P}_{t+1} \leftarrow \mathcal{P}_t \setminus \mathcal{C}_{t+1}$ 
10:  if  $t \bmod \Delta = 0$  then
11:     $G_{t+1} \leftarrow \mathcal{R}(G_t, \mathcal{C}_{t+1}, \mathcal{P}_{t+1}, s_{t+1})$ 
12:  end if
13:   $t \leftarrow t + 1$ 
14: end while
15: return Final state  $s_T$ 
```

4 EXPERIMENT

4.1 FLASH-SEARCHER FRAMEWORK

4.1.1 SETUP

Benchmarks. We evaluate FLASH-SEARCHER on four challenging benchmarks for information retrieval and reasoning:

- **GAIA** (Mialon et al., 2023): A comprehensive benchmark for evaluating complex task-solving capabilities. For this benchmark, we mainly use the text-only validation set (103 tasks), which requires deep information retrieval and complex reasoning. Notably, the full validation set is solely used in Figure 3 for fair comparison; all other evaluations are based on the text-only validation subset.

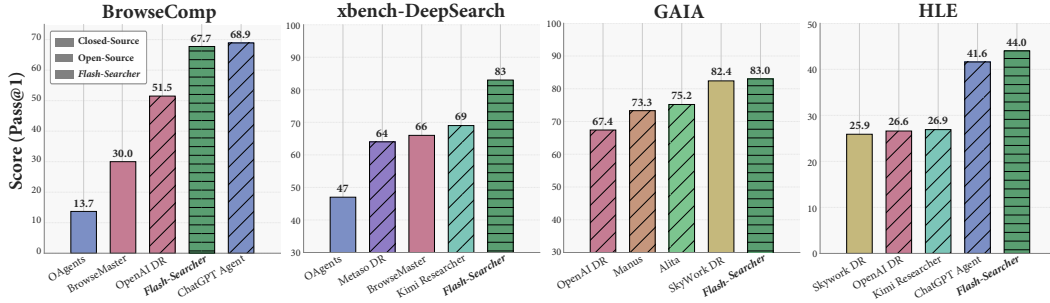


Figure 3: Performance comparison of agent frameworks on BrowseComp, xbench-DeepSearch, GAIA and HLE benchmarks. All results are reported using Pass@1 metric.

- **BrowseComp** (Wei et al., 2025): Large-scale benchmark comprising 1,266 tasks designed to test internet-scale information retrieval with hard-to-find information needs and sophisticated browsing strategies.
- **xbench-DeepSearch** (Xbench-Team, 2025): Professional benchmark with 100 tasks simulating real-world search scenarios, emphasizing multi-round refinement and cross-source information integration.
- **HLE** (Phan et al., 2025): A frontier benchmark covering over a hundred subjects, designed to address the limited difficulty of existing benchmarks. We follow the setting in AFM (Li et al., 2025b) and use HLE-500 for evaluations.

Framework Configuration. FLASH-SEARCHER employs a minimalist yet powerful tool configuration optimized for parallel execution. Our framework integrates two core components: a Search Tool implemented with the Serper API (Serper, 2025) for retrieving structured search results, and a Crawl Tool leveraging the Jina Reader (Jina, 2025) for content extraction. The crawl tool incorporates automatic summarization using the same backbone language model, ensuring consistent information representation while significantly reducing cognitive load. This streamlined design enables efficient parallel tool orchestration across reasoning branches while maintaining trajectory simplicity and operational coherence. More details can be found in Appendix D.

Metrics. We employ the LLM-as-Judge paradigm (Zheng et al., 2023; Wu et al., 2025a) for automated evaluation, utilizing GPT-4.1-mini as the judge model. Each agent output of different benchmarks receives a binary correctness assessment from the judge model. For final performance reporting, we default to presenting Pass@1 results: this metric quantifies the proportion of tasks where the agent generates a correct output on the first attempt. Specifically, the Pass@1 score for each individual benchmark is calculated based on its binary assessment results, and these per-benchmark Pass@1 scores are ultimately aggregated to report the overall performance. The standardized prompt for judgment is detailed in Appendix I.1.

4.1.2 MAIN RESULTS

We present a comprehensive evaluation of FLASH-SEARCHER against state-of-the-art closed-source and open-source agent frameworks across four challenging benchmarks: BrowseComp, xbench-DeepSearch, GAIA, and HLE. As illustrated in Figure 3, our method achieves highly competitive performance, matching or exceeding existing approaches while demonstrating superior efficiency and scalability. These results underscore the effectiveness of our DAG-based architecture in handling diverse task complexities.

In Figure 4, our FLASH-SEARCHER when integrated with GPT-5 achieves a competitive performance of **67.7%** on the BrowseComp benchmark. This result not only demonstrates a substantial advantage over state-of-the-art

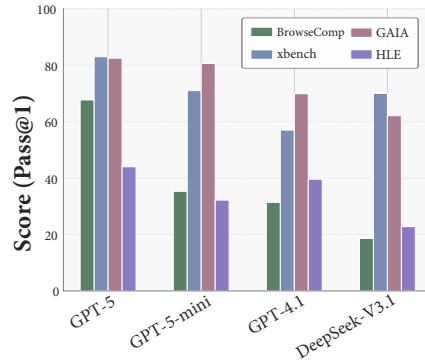


Figure 4: Performance of FLASH-SEARCHER with different backbones.

open-source frameworks (e.g., BrowseMaster (Pang et al., 2025), which attains 30.0%) but also approaches the performance of the leading closed-source solution, specifically the OpenAI Chat-GPT agent (68.9%). Even with less powerful backbone models such as GPT-5-mini, our framework achieves **35.3%**, demonstrating the effectiveness of our parallel reasoning approach regardless of the underlying model. For xbench-DeepSearch, FLASH-SEARCHER also shows remarkable performance, with our GPT-5 variant achieving **83%**, surpassing both BrowseMaster (66%) and Metaso DeepResearch (64%). This substantial improvement highlights the particular strength of our approach in deep research scenarios that demand extensive information gathering and complex reasoning. Besides, On the GAIA benchmark, FLASH-SEARCHER with lightweight, resource-efficient GPT-5-mini backbone achieves **80.6%**, exceeding even strong closed-source systems like Alita (75.2%) and Manus (73.3%). Additionally, our method demonstrates exceptional capability on the HLE benchmark, achieving a state-of-the-art **44.0%** with GPT-5, substantially outperforming all other frameworks.

These results demonstrate that our FLASH-SEARCHER, a parallel reasoning framework, effectively addresses the multifaceted challenges inherent to information retrieval tasks. The consistent performance of FLASH-SEARCHER across diverse backbone models spanning different architectures and capability levels further validates the robustness of our approach.

4.2 FLASH-SEARCHER LEARNING

4.2.1 SETUP

Dataset. To train our parallel reasoning agent, we construct a high-quality dataset derived from multiple sources including WebWalker (Wu et al., 2025b), ASearcher (Gao et al., 2025), WebShaper (Tao et al., 2025), and CoA (Li et al., 2025b). Our final dataset consists of **3354** effective DAG-based reasoning trajectories. Each trajectory incorporates periodic DAG workflow reviews and is formatted as a multi-turn dialogue, enabling effective context window extrapolation and long-range dependency modeling. This format specifically enhances the model’s ability to manage complex reasoning graphs while maintaining coherent conversation flow. More details can be found in Appendix G.1

Training Configurations. We maintain consistent evaluation metrics and benchmarks with the framework experiments in Section 4.1.1. All training is implemented using the Llama-Factory framework (Zheng et al., 2024). We employ supervised fine-tuning to develop robust parallel reasoning capabilities. Specifically, for all trained models, the maximum dialogue length is set to 131,072 tokens, the learning rate is set to 10^{-5} , and training is conducted for four epochs. The full training parameters and detailed data formatting specifications are comprehensively documented in Appendix G.2.

4.2.2 AGENT MODEL RESULTS

To validate the effectiveness of our parallel reasoning approach beyond framework implementation, we distilled FLASH-SEARCHER’s parallel reasoning capabilities into standalone agent models through lightweight supervised fine-tuning. Table 1 presents a comprehensive comparison of these agent models against existing state-of-the-art methods across four challenging benchmarks.

Our experimental analysis demonstrates that lightweight supervised fine-tuning effectively facilitates the transfer of FLASH-SEARCHER’s parallel reasoning capabilities to standalone agent models, consistently achieving state-of-the-art (SOTA) performance across diverse benchmarks and model backbone scales. Specifically, on the Qwen-2.5-32B backbone, FLASH-SEARCHER establishes a new performance ceiling. It outperforms the strongest prior method by 3.3% on BrowseComp, 5.0% on xBench-DeepSearch, and 2.0% on GAIA. Despite forgoing code interpreter tools, FLASH-SEARCHER achieves state-of-the-art performance at 19.4% on HLE, surpassing tool-augmented baselines and affirming the general effectiveness of FLASH-SEARCHER in handling general complex tasks. This result underscores FLASH-SEARCHER’s inherent reasoning robustness, as it delivers strong performance without relying on extensive tools.

Scaling FLASH-SEARCHER to 72B yields consistent and meaningful performance gains across all benchmarks, demonstrating that our parallel reasoning framework scales gracefully with model capacity. Notably, the most substantial improvements occur on complex, multi-step reasoning tasks

Table 1: Performance comparison of agent models on BrowseComp, xbench-DeepSearch, and GAIA benchmarks. All results are reported using Pass@1 metric. Gray-font values correspond to results reported in the associated reports.

Method	Backbone	BrowseComp	xbench-DeepSearch	GAIA	HLE
Cognitive Kernel-Pro	Qwen-3-8B	-	-	43.7	-
WebDancer	QwQ-32B	3.8	39.0	50.5	7.2
WebThinker-RL		2.8	24.0	48.5	-
SimpleDeepSearcher		-	-	50.5	-
WebShaper		-	-	53.3	12.2
SFR-DR		-	-	52.4	17.1
WebDancer	Qwen-2.5-32B	2.5	38.7	40.7	-
SimpleDeepSearcher		-	-	40.8	-
WebShaper		-	-	52.4	-
WebSailor		10.5	53.3	53.2	10.8
AFM-RL		11.1	58.0	55.3	18.0
FLASH-SEARCHER		14.4	63.0	57.3	19.4
WebSailor	Qwen-2.5-72B	12.0	55.0	55.4	-
WebShaper		-	-	60.1	-
FLASH-SEARCHER		18.9	68.0	61.2	20.2

such as BrowseComp and xbench-DeepSearch, with 5% gains, suggesting that increased parameter scale enhances the model’s ability to coordinate and refine reasoning steps. Even on HLE, the performance affirms that FLASH-SEARCHER internalizes structured reasoning without relying on external tools. This behavior confirms that our lightweight fine-tuning paradigm not only transfers reasoning capabilities effectively but also unlocks deeper potential as backbone capacity grows, making it suited for scalable, general-purpose agent deployment.

Notably, these results are achieved through lightweight supervised fine-tuning without RL or tool reliance. This confirms that parallel reasoning is a learnable and scalable inductive bias, efficiently transferred via minimal supervision. FLASH-SEARCHER thus emerges as a simple, robust, and parameter-efficient solution for real-world agents.

5 EFFICIENCY ANALYSIS

We present a comprehensive efficiency analysis of FLASH-SEARCHER using the GPT-5-mini backbone, evaluating its execution efficiency and framework improvements compared to existing agent systems. The distribution plot in Figure 5a demonstrates BrowseComp benchmark requiring the highest number of both metrics. This reflects the varying complexity demands across different benchmark types. Figure 5b reveals FLASH-SEARCHER’s operational efficiency through tool calls per execution step. The tight interquartile range, particularly evident in the GAIA benchmark, indicates consistent and predictable tool utilization patterns. This evidence directly supports FLASH-SEARCHER’s core claim: its DAG-based architecture optimizes tool utilization efficiency while cutting extraneous execution steps. Specifically, the architecture’s built-in parallel tool invocation enables concurrent execution of complementary tools, eliminating sequential bottlenecks that cause redundant steps in linear pipelines.

To fairly evaluate the execution efficiency of FLASH-SEARCHER, we compare FLASH-SEARCHER against OAgents (Zhu et al., 2025a) and OWL-Roleplaying (Hu et al., 2025) with their original configurations (Details in Appendix H). The experimental results are presented in Figure 6, which demonstrates significant efficiency improvements of our approach across four benchmarks.

As shown in Figure 6a, FLASH-SEARCHER outperforms OAgents on all four benchmarks, achieving higher task success rates and efficiency gains—with this advantage growing more pronounced as task complexity increases (*BrowseComp* > *xbench-DeepSearch* > *HLE* > *GAIA*). This validates FLASH-SEARCHER’s adaptability to complex scenarios, laying the foundation for subsequent efficiency analysis. Figure 6b further demonstrates that FLASH-SEARCHER (with GPT-5-mini back-

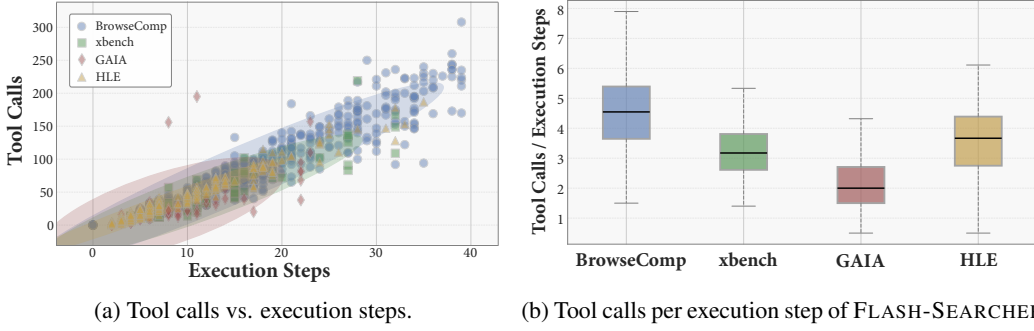


Figure 5: Efficiency analysis of FLASH-SEARCHER on four benchmarks: (a) shows the correlation between tool calls and steps; (b) characterizes the distribution of tool calls per step.

bone) reduces agent steps by **35%** versus OAgents and **30%** versus OWL-Roleplaying on GAIA benchmark, enabled by its parallel reasoning architecture. This efficiency gain stems from the DAG-based workflow’s ability to execute concurrent reasoning paths, which effectively mitigates the sequential bottleneck of traditional methods. Figure 5 illustrates the distribution of tool calls and steps for FLASH-SEARCHER: despite fewer total steps, our approach maintains higher per-step tool utilization efficiency (average **3.00** tool calls per step, compared to **0.83** for OAgents and **0.85** for OWL-Roleplaying), confirming more productive and effective reasoning iterations.

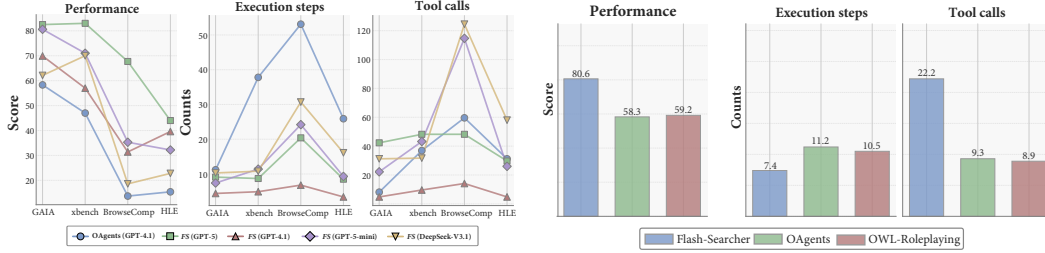


Figure 6: Efficiency comparison of agent frameworks on four benchmarks.

The core innovation lies in our DAG-based parallel execution mechanism, which directly addresses the fundamental limitation of redundant tool invocation cycles in sequential reasoning approaches. By coordinating information needs across parallel branches, we eliminate duplicate searches while maintaining reasoning diversity. As in Figure 6a, our framework simultaneously enhances both efficiency and performance, effectively resolving the longstanding efficiency-effectiveness trade-off in agent systems.

Although agent execution duration is inherently influenced by external factors such as API rate limits, FLASH-SEARCHER consistently achieves a **35%** reduction in execution steps under comparable environmental conditions. This reduction directly translates into lower end-to-end latency and improved throughput, offering a significant efficiency advantage. Such gains are especially critical in latency-sensitive applications and high-throughput deployment settings, where conventional sequential agent architectures encounter substantial scalability and responsiveness bottlenecks.

6 CONCLUSION

In this work, we introduce FLASH-SEARCHER, a novel parallel agent reasoning framework that overcomes the sequential bottlenecks of conventional tool-augmented agents through structured concurrency. By reformulating task solving as dynamic scheduling over DAGs, FLASH-SEARCHER enables fine-grained parallel execution while rigorously preserving logical coherence and correctness. Extensive experiments across BrowseComp, xbench-DeepResearch, GAIA, and HLE demonstrate that FLASH-SEARCHER achieves state-of-the-art performance, attaining a score of **67.7%** on BrowseComp, alongside substantial gains in computational efficiency through reduced latency and improved resource utilization. Our results, further corroborated by distilled agent variants, establish parallel reasoning as a foundational paradigm for building efficient, scalable, and robust AI systems capable of mastering complex real-world tasks.

7 ETHICS STATEMENT

Our research focuses on the development of agent frameworks and model architectures for web-based autonomous agents, aiming to create more effective systems that can assist users in completing complex tasks. We conduct rigorous evaluation on controlled benchmarks and ensure transparency in our experimental procedures. This work does not involve any risks related to ethics issues and is intended to advance research in web agent systems.

8 REPRODUCIBILITY STATEMENT

We have made significant efforts to ensure the reproducibility of FLASH-SEARCHER:

- **Training Dataset:** The complete FLASH-SEARCHER training dataset will be made publicly available upon publication. Detailed information about data collection, annotation guidelines, and quality control measures are provided in Appendix G.1.
- **Code:** We provide a comprehensive codebase including implementations of our framework, data generation procedures, and evaluation metrics in the supplementary materials.
- **Experimental Configuration:** We provide detailed experimental specifications, hyperparameter configurations, and procedural details are documented in Appendix D.
- **Limitations:** We note that exact reproduction of results for FLASH-SEARCHER frameworks and models may be challenging due to potential API changes or Inference uncertainty.

By providing these resources, we aim to facilitate reproduction of our results and encourage further research.

REFERENCES

- Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, 2024.
- Lorenzo Canese, Gian Carlo Cardarilli, Luca Di Nunzio, Rocco Fazzolari, Daniele Giardino, Marco Re, and Sergio Spanò. Multi-agent reinforcement learning: A review of challenges and applications. *Applied Sciences*, 11(11):4948, 2021.
- Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chen Qian, Chi-Min Chan, Yujia Qin, Yaxi Lu, Ruobing Xie, et al. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents. *arXiv preprint arXiv:2308.10848*, 2(4):6, 2023.
- Ali Dorri, Salil S Kanhere, and Raja Jurdak. Multi-agent systems: A survey. *Ieee Access*, 6:28573–28593, 2018.
- Adam Fourney, Gagan Bansal, Hussein Mozannar, Cheng Tan, Eduardo Salinas, Friederike Niedtner, Grace Proebsting, Griffin Bassman, Jack Gerrits, Jacob Alber, et al. Magentic-one: A generalist multi-agent system for solving complex tasks. *arXiv preprint arXiv:2411.04468*, 2024.
- Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous rl. *arXiv preprint arXiv:2508.07976*, 2025.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.
- Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, et al. Metagpt: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations, ICLR*, 2024.

- Mengkang Hu, Yuhang Zhou, Wendong Fan, Yuzhou Nie, Bowei Xia, Tao Sun, Ziyu Ye, Zhaoxuan Jin, Yingru Li, Qiguang Chen, Zeyu Zhang, Yifeng Wang, Qianshuo Ye, Bernard Ghanem, Ping Luo, and Guohao Li. Owl: Optimized workforce learning for general multi-agent assistance in real-world task automation, 2025. URL <https://arxiv.org/abs/2505.23885>.
- Xueyu Hu, Tao Xiong, Biao Yi, Zishu Wei, Ruixuan Xiao, Yurun Chen, Jiasheng Ye, Meiling Tao, Xiangxin Zhou, Ziyu Zhao, et al. Os agents: A survey on mllm-based agents for computer, phone and browser use, 2024.
- Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-rl: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025a.
- Yiyang Jin, Kunzhao Xu, Hang Li, Xueting Han, Yanmin Zhou, Cheng Li, and Jing Bai. Reveal: Self-evolving code agents via iterative generation-verification, 2025b. URL <https://arxiv.org/abs/2506.11442>.
- Inc. Jina. Jina reader, 2025. URL <https://jina.ai/reader/>.
- Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for "mind" exploration of large language model society. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
- Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, Weizhou Shen, Junkai Zhang, Dingchu Zhang, Xixi Wu, Yong Jiang, Ming Yan, Pengjun Xie, Fei Huang, and Jingren Zhou. Websailor: Navigating super-human reasoning for web agent, 2025a. URL <https://arxiv.org/abs/2507.02592>.
- Weizhen Li, Jianbo Lin, Zhuosong Jiang, Jingyi Cao, Xinpeng Liu, Jiayu Zhang, Zhenqiang Huang, Qianben Chen, Weichen Sun, Qiexiang Wang, et al. Chain-of-agents: End-to-end agent foundation models via multi-agent distillation and agentic rl. *arXiv preprint arXiv:2508.13167*, 2025b.
- Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-ol: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*, 2025c.
- Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yutao Zhu, Yongkang Wu, Ji-Rong Wen, and Zhicheng Dou. Webthinker: Empowering large reasoning models with deep research capability. *arXiv preprint arXiv:2504.21776*, 2025d.
- Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, Joydeep Biswas, and Peter Stone. Llm+ p: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023.
- Xinji Mai, Haotian Xu, Weinong Wang, Yingying Zhang, Wenqiang Zhang, et al. Agent rl scaling law: Agent rl with spontaneous code execution for mathematical problem solving. *arXiv preprint arXiv:2505.07773*, 2025.
- Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*, 2023.
- Xuan-Phi Nguyen, Shrey Pandit, Revanth Gangi Reddy, Austin Xu, Silvio Savarese, Caiming Xiong, and Shafiq Joty. Sfr-deepresearch: Towards effective reinforcement learning for autonomously reasoning single agents. *arXiv preprint arXiv:2509.06283*, 2025.
- Jiayi Pan, Xiuyu Li, Long Lian, Charlie Snell, Yifei Zhou, Adam Yala, Trevor Darrell, Kurt Keutzer, and Alane Suhr. Learning adaptive parallel reasoning with language models. *Conference on Language Modeling*, 2025.
- Xianghe Pang, Shuo Tang, Rui Ye, Yuwen Du, Yaxin Du, and Siheng Chen. Browsemaster: Towards scalable web browsing via tool-augmented programmatic agent pair. *arXiv preprint arXiv:2508.09129*, 2025.

- Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, et al. Humanity’s last exam. *arXiv preprint arXiv:2501.14249*, 2025.
- Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. Chatdev: Communicative agents for software development. *arXiv preprint arXiv:2307.07924*, 2023.
- Jiahao Qiu, Xuan Qi, Tongcheng Zhang, Xinzhe Juan, Jiacheng Guo, Yifu Lu, Yimin Wang, Zixin Yao, Qihan Ren, Xun Jiang, et al. Alita: Generalist agent enabling scalable agentic reasoning with minimal predefinition and maximal self-evolution. *arXiv preprint arXiv:2505.20286*, 2025.
- Aymeric Roucher, Albert Villanova del Moral, Thomas Wolf, Leandro von Werra, and Erik Kainismäki. ‘smolagents’: a smol library to build great agentic systems. <https://github.com/huggingface/smolagents>, 2025.
- Bilgehan Sel, Ahmad Al-Tawaha, Vanshaj Khattar, Ruoxi Jia, and Ming Jin. Algorithm of thoughts: Enhancing exploration of ideas in large language models. *arXiv preprint arXiv:2308.10379*, 2023.
- Inc. Serper. Serper api, 2025. URL <https://serper.dev/>.
- Dingfeng Shi, Jingyi Cao, Qianben Chen, Weichen Sun, Weizhen Li, Hongxuan Lu, Fangchen Dong, Tianrui Qin, King Zhu, Minghao Yang, et al. Taskcraft: Automated generation of agentic tasks. *arXiv preprint arXiv:2506.10055*, 2025.
- Shuang Sun, Huatong Song, Yuhao Wang, Ruiyang Ren, Jinhao Jiang, Junjie Zhang, Fei Bai, Jia Deng, Wayne Xin Zhao, Zheng Liu, et al. Simpledeepsearcher: Deep information seeking via web-powered reasoning trajectory synthesis. *arXiv preprint arXiv:2505.16834*, 2025.
- Xiangru Tang, Tianrui Qin, Tianhao Peng, Ziyang Zhou, Daniel Shao, Tingting Du, Xinming Wei, Peng Xia, Fang Wu, He Zhu, Ge Zhang, Jiaheng Liu, Xingyao Wang, Sirui Hong, Chenglin Wu, Hao Cheng, Chi Wang, and Wangchunshu Zhou. Agent kb: Leveraging cross-domain experience for agentic problem solving. In *ICML 2025 Workshop on Collaborative and Federated Agentic Workflows*, 2025.
- Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. Web-shaper: Agentically data synthesizing via information-seeking formalization, 2025. URL <https://arxiv.org/abs/2507.15061>.
- MiroMind AI Team. Miroflow: An open-source agentic framework for deep research. <https://github.com/MiroMindAI/MiroFlow>, 2025.
- Ningning Wang, Xavier Hu, Pai Liu, He Zhu, Yue Hou, Heyuan Huang, Shengyu Zhang, Jian Yang, Jiaheng Liu, Ge Zhang, et al. Efficient agents: Building effective agents while reducing cost. *arXiv preprint arXiv:2508.02694*, 2025.
- Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025.
- Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Yong Jiang, Pengjun Xie, et al. Webdancer: Towards autonomous information seeking agency. *arXiv preprint arXiv:2505.22648*, 2025a.
- Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, et al. Webwalker: Benchmarking llms in web traversal. *arXiv preprint arXiv:2501.07572*, 2025b.
- Xbench-Team. Xbench-deepsearch, 2025. URL <https://xbench.org/agi/aisharch>.
- Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Zejun Ma, and Bo An. Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. <https://simpletir.notion.site/report>, 2025. Notion Blog.

- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.
- Dingchu Zhang, Yida Zhao, Jialong Wu, Baixuan Li, Wenbiao Yin, Liwen Zhang, Yong Jiang, Yufeng Li, Kewei Tu, Pengjun Xie, et al. Evolvesearch: An iterative self-evolving search agent. *arXiv preprint arXiv:2505.22501*, 2025a.
- Shiqi Zhang, Xinbei Ma, Zouying Cao, Zhuosheng Zhang, and Hai Zhao. Plan-over-graph: Towards parallelable llm agent schedule. *arXiv preprint arXiv:2502.14563*, 2025b.
- Shu Zhao, Tan Yu, Anbang Xu, Japinder Singh, Aaditya Shukla, and Rama Akkiraju. Parallelssearch: Train your llms to decompose query and search sub-queries in parallel with reinforcement learning. *arXiv preprint arXiv:2508.09303*, 2025.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.
- Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyao Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*, 2024.
- Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*, 2025.
- Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning acting and planning in language models. *arXiv preprint arXiv:2310.04406*, 2023a.
- Wangchunshu Zhou, Yuchen Eleanor Jiang, Peng Cui, Tiannan Wang, Zhenxin Xiao, Yifan Hou, Ryan Cotterell, and Mrinmaya Sachan. Recurrentgpt: Interactive generation of (arbitrarily) long text, 2023b. URL <https://arxiv.org/abs/2305.13304>.
- Wangchunshu Zhou, Yuchen Eleanor Jiang, Long Li, Jialong Wu, Tiannan Wang, Shi Qiu, Jintian Zhang, Jing Chen, Ruipu Wu, Shuai Wang, et al. Agents: An open-source framework for autonomous language agents. *arXiv preprint arXiv:2309.07870*, 2023c.
- Wangchunshu Zhou, Yixin Ou, Shengwei Ding, Long Li, Jialong Wu, Tiannan Wang, Jiamin Chen, Shuai Wang, Xiaohua Xu, Ningyu Zhang, et al. Symbolic learning enables self-evolving agents. *arXiv preprint arXiv:2406.18532*, 2024.
- He Zhu, Tianrui Qin, King Zhu, Heyuan Huang, Yeyi Guan, Jinxiang Xia, Yi Yao, Hanhao Li, Ningning Wang, Pai Liu, Tianhao Peng, Xin Gui, Xiaowan Li, Yuhui Liu, Yuchen Eleanor Jiang, Jun Wang, Changwang Zhang, Xiangru Tang, Ge Zhang, Jian Yang, Minghao Liu, Xitong Gao, Wangchunshu Zhou, and Jiaheng Liu. Oagents: An empirical study of building effective agents, 2025a. URL <https://arxiv.org/abs/2506.15741>.
- King Zhu, Hanhao Li, Siwei Wu, Tianshun Xing, Dehua Ma, Xiangru Tang, Minghao Liu, Jian Yang, Jiaheng Liu, Yuchen Eleanor Jiang, Changwang Zhang, Chenghua Lin, Jun Wang, Ge Zhang, and Wangchunshu Zhou. Scaling test-time compute for llm agents, 2025b. URL <https://arxiv.org/abs/2506.12928>.