FLASHRESEARCH: REAL-TIME AGENT ORCHESTRATION FOR EFFICIENT DEEP RESEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep research agents, which synthesize information across diverse sources, are significantly constrained by their sequential reasoning processes. This architectural bottleneck results in high latency, poor runtime adaptability, and inefficient resource allocation, making them impractical for interactive applications. To overcome this, we introduce **FlashResearch**, a novel framework for efficient deep research that transforms sequential processing into parallel, runtime orchestration by dynamically decomposing complex queries into tree-structured sub-tasks. Our core contributions are threefold: **(1)** an **adaptive planner** that dynamically allocates computational resources by determining research breadth and depth based on query complexity; **(2)** a **real-time orchestration layer** that monitors research progress and prunes redundant paths to reallocate resources and optimize efficiency; and **(3)** a **multi-dimensional parallelization framework** that enables concurrency across both research breadth and depth. Experiments show that FlashResearch consistently improves final report quality within fixed time budgets, and can deliver up to a $5\times$ speedup while maintaining comparable quality.

1 INTRODUCTION

Deep research tasks, which involve synthesizing information from diverse sources and navigating complex, interdependent concepts, pose significant challenges for existing AI systems. These tasks often demand knowledge retrieval, advanced reasoning, sophisticated tool use, and dynamic planning over multiple steps under structural uncertainty and evolving objectives (Du et al., 2025). Applications include literature review (Haman & Školník, 2025), open-domain question answering, and policy analysis (Gambrell, 2025), where the ability to evaluate conflicting perspectives, explore hypotheses, and revise beliefs as new evidence emerges is essential. However, current systems often take tens of minutes to respond. This latency can break users' cognitive flow (Iqbal & Horvitz, 2007), incur high context-switching costs (Mark et al., 2008), and degrade overall experience.

Much of this inefficiency stems from poor orchestration. Existing systems typically rely on sequential processing (Xu & Peng, 2025), leading to unnecessary latency when subproblems are independent, leaving parallelizable evidence gathering, hypothesis branching, and speculative exploration underexploited. Moreover, static planning strategies in these systems also fail to adapt to the dynamic nature of research (Zheng et al., 2025b). The value of subqueries or deeper investigation often becomes clear only during execution, but current systems rarely prune low-value paths or reallocate effort during runtime.

To address these challenges, we propose **FlashResearch**, a framework designed for efficient deep research that integrates adaptive planning, real-time orchestration, and concurrent execution. FlashResearch treats deep research as a dynamic, tree-structured traversal, where a complex query is decomposed into concurrent subqueries that dynamically populate the research tree. The objective is to maximize response quality under a time budget by adjusting the tree structure and reallocating effort across promising research directions.

At the core of FlashResearch is an adaptive planner that decides, at each step, how many subqueries to open and whether to explore further depth, based on the complexity of the input query. It weighs the expected marginal utility of each branch, expanding breadth when broad exploration is valuable and deepening paths selectively when the information gain justifies further effort. This allows

FlashResearch to flexibly allocate resources depending on whether the query is on a broad topic that demands diverse perspectives, or a specific one that requires deeper investigation.

However, planning alone is insufficient. Since research is inherently iterative and non-linear, newly emerged evidence may reshape priorities mid-execution. FlashResearch incorporates a real-time orchestration layer that monitors ongoing research outputs, evaluates them against goal satisfaction and quality metrics, and makes live adjustments. This mechanism allows the system to terminate low-value or redundant branches early and reallocate computational resources toward more promising paths. More importantly, it allows speculative execution by launching child tasks before parent-level planning decisions are finalized, thereby reducing idle time and accelerating throughput.

This tight feedback loop between planning and execution produces a highly dynamic research tree that evolves in response to both query structure and emergent information. To handle this, FlashResearch employs a multi-dimensional parallelization framework to schedule tasks across breadth and depth through a unified task pool. Built on a fully asynchronous infrastructure with thread-safe state management, the system enables simultaneous exploration of multiple research paths and allows non-blocking orchestration to adapt the tree structure as new findings emerge.

To evaluate the effectiveness of FlashResearch, we conduct experiments on two recent deep research benchmarks, DeepResearchGym and DeepResearch Bench (Coelho et al., 2025; Alzubi et al., 2025). Compared to the baseline, FlashResearch can consistently accomplish deeper and wider research within fixed time constraints, producing research reports of better comprehensiveness and insights.

The key contributions of this work are:

- A formal formalization of deep research tasks as a tree-structured optimization problem.
- An adaptive planning module for real-time, context-aware decisions on task branching, recursion, and termination.
- A real-time orchestration framework for dynamic task monitoring, speculative execution, and intelligent resource reallocation.
- A fully asynchronous and parallelized execution architecture enabling concurrent research execution across multiple dimensions.
- A comprehensive empirical evaluation demonstrating FlashResearch's superior improvements in terms of throughput, quality, and efficiency on complex research tasks.

2 Related Works

2.1 Deep Research Agents

Building on earlier tool-use frameworks like WebGPT (Nakano et al., 2021) and ReAct (Yao et al., 2023b), recent deep research agents – including GPT-Researcher (Elovic, 2023), Open Deep Search (Alzubi et al., 2025), and LangChain's Open Deep Research (Langchain, 2025) – decompose complex queries into tool-augmented subtasks. To standardize evaluation, emerging benchmarks like DeepResearchGym (Coelho et al., 2025) and DeepResearch Bench (Du et al., 2025) introduce LLM-as-a-judge protocols tailored for complex research questions.

Despite these advances, current systems typically rely on fixed, pre-specified parameters for controlling the research structure. Their orchestration strategies are dominated by sequential execution or coarse-grained parallelism (Xu & Peng, 2025), which limits adaptability. As a result, when information quality shifts during execution, these systems either waste compute or incur unnecessary latency. FlashResearch addresses these limitations by introducing a real-time orchestration layer that couples multi-dimensional parallelism across both depth and breadth. Unlike prior static approaches, it adaptively expands or prunes subqueries in real time based on intermediate evidence, enabling more efficient and responsive deep research.

2.2 Agentic Workflow Orchestration

Recent work compiles high-level goals into executable agent graphs via MCTS-guided code search (Zhang et al., 2024), evolutionary populations of heterogeneous workflows (Niu et al., 2025), and modular activity-on-vertex graphs (Zhang et al., 2025). These systems mainly optimize *offline* and then execute largely fixed graphs, and their runtime control over partially executed graphs remains

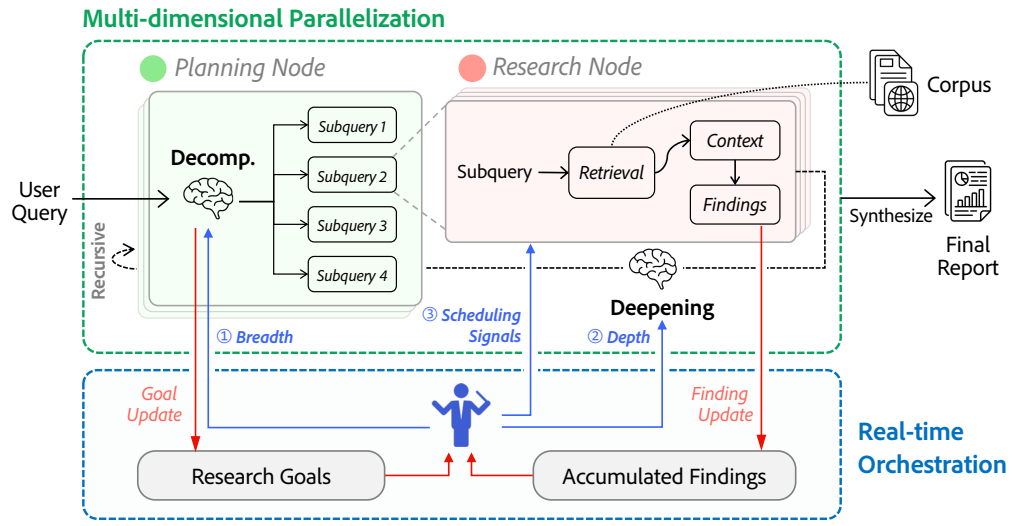


Figure 1: Overview of FlashResearch: the Planning Nodes adaptively decompose queries into parallel subqueries executed by Research Nodes for findings, which may recursively trigger deeper planning. The **adaptive planner** expands (1) *breadth* to explore prior-research and regulates (2) *depth* to pursue promising paths post-research. The **real-time orchestration layer** monitors progress and reallocates resources through (3) *scheduling signals* mid-research. A **multi-dimensional parallelization framework** enables flexible concurrency across both breadth and depth.

limited. Production frameworks such as AutoGen (Wu et al., 2023), LangGraph (LangChain, 2024), DSPy (Khattab et al., 2024), and OpenAI Swarm (OpenAI, 2024) provide valuable abstractions for multi-agent pipeline control. However, they lack support for *real-time* replanning and cross-branch compute reallocation. FlashResearch addresses this gap with an *real-time* orchestrator that continuously monitors task states and reallocates resources on the fly, enabling suspension, escalation, and replanning during execution rather than only before execution.

2.3 Parallel and Speculative Reasoning

Token- and action-level acceleration methods such as speculative decoding (Leviathan et al., 2023; Miao et al., 2023; Cai et al., 2024) and speculative reasoning for fast inference (Pan et al., 2025; Yang et al., 2025) reduce latency via draft-and-verify or multi-token prediction. At the reasoning level, Dynamic Parallel Tree Search (Ding et al., 2025) accelerates Tree-of-Thoughts by expanding and pruning nodes in parallel, while ParaThinker (Wen et al., 2025) and Parallel-R1 (Zheng et al., 2025a) instill native parallel reasoning. These improve efficiency and accuracy but still rely on static branching. Inspired by these works, FlashResearch advances parallelism to the workflow level: it not only reallocates compute and prunes branches dynamically, but also supports speculative execution—allowing branches to expand without delay and later discarding them if evidence shows they are unnecessary.

3 Background

3.1 Formulating Deep Research

Deep research tasks involve tackling complex, open-ended queries by multi-step reasoning, gathering diverse information, and synthesizing knowledge into comprehensive responses. We formalize such a task as follows: a user query $q \in \mathcal{Q}$, where $\mathcal{Q}$ is the space of natural language queries. The goal is to produce a response $r \in \mathcal{R}$ by integrating knowledge from retrieved context $C = \{c_1, c_2, \ldots, c_n\}$ sourced from a corpus D (e.g., web searches or local documents). During this process, research findings $F = \{f_1, f_2, \ldots, f_m\}$, comprising reasoning artifacts and key insights,

are iteratively derived from the context. Consequently, the response is generated as

$$r = \sigma(q, C, F), \tag{1}$$

where $\sigma : \mathcal{Q} \times 2^C \times 2^F \to \mathcal{R}$ is a synthesis function that aggregates and refines the inputs to maximize factual accuracy, comprehensiveness, and relevance. Here, 2^C and 2^F denote the power sets of C and F, representing all possible subsets of contexts and findings, respectively. In practice, these are subsets selected based on relevance constraints, and σ is typically realized by an LLM agent.

To solve such tasks scalably, a deep research framework must balance the thoroughness of exploration with computational overheads. Conventional sequential pipelines—such as linear chains of retrieval-augmented generation (RAG) steps—often falter under intricate queries, incurring high costs from redundant traversals or premature convergence to suboptimal paths. Given the hierarchical and multi-faceted nature of deep research, it is natural to model the process as a tree structure, like in Tree of Thoughts (Yao et al., 2023a).

Formally, we model the process as a directed tree $\mathcal{T} = (N^P \cup N^R, E)$ with disjoint node sets of *planning nodes* N^P and *research nodes* N^R.

A planning node n^P decomposes a query q^n into a finite set of subqueries:

$$n^P(q^n) \to \{q_1^n, \ldots, q_{b_n}^n\}, \quad q_j^n \in \mathcal{Q}. \tag{2}$$

Here, q^n can be either the initial query or a subquery generated by a previous planning node. Each $q_j^n \in n^P(q^n)$ instantiates a research node $n^R(q_j^n)$. The value $b_n = |n^P(q^n)|$ is the *breadth* chosen at that level. The tree root is therefore the planning node n_0^P that receives the initial query q.

A research node $n^R(q_j^n)$ executes retrieval and localized reasoning for its particular subquery q_j^n:

$$n^R(q_j^n) \to (C_{q_j^n}, F_{q_j^n}), \tag{3}$$

producing local contexts $C_{q_j^n} \subseteq C$ and findings $F_{q_j^n} \subseteq F$. Optionally, a research node may trigger recursion by spawning a single child planning node that further decomposes q_j^n, after which the alternating pattern continues. The *depth* d of the tree is defined by the number of research-node layers along the longest root-to-leaf path.

The final response $r_{\mathcal{T}}$ can then be synthesized as

$$r_{\mathcal{T}} = \sigma\left(q, \bigcup_{n_i \in N^R} C_i, \bigcup_{n_i \in N^R} F_i\right), \tag{4}$$

by aggregating each research node's local contexts and findings across the tree. The quality of the response can be subsequently measured by a utility function $U(r)$.

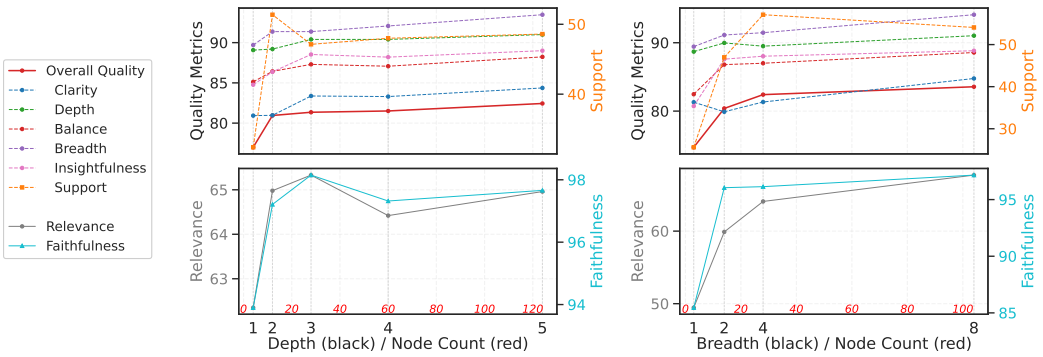
However, using fixed depths and breadths for the tree can lead to suboptimal performance: shallow trees might insufficiently explore the topic, while excessively deep or broad trees incur high costs with diminishing returns on quality. Therefore, the core challenge is to orchestrate the tree structure at runtime to maximize the response quality while adhering to a time budget t_{max}. Formally, we aim to solve:

$$\max_{\mathcal{T}} U(r_{\mathcal{T}}) \quad \text{s.t.} \quad t(\mathcal{T}) \leq t_{\text{max}}, \tag{5}$$

where $t(\mathcal{T})$ represents the total latency of the research process across all nodes and edges in the tree.

3.2 Motivating Experiments

Following the above formulation, we investigate how the tree structure can affect response quality in deep research tasks. We evaluated a tree-based deep research framework, GPT-Researcher, on a set of 100 complex queries randomly sampled from DeepResearchGym (Coelho et al., 2025). We varied the tree's maximum depth and breadth hyperparameters and measured performance across multiple metrics.



(a) Depth trade-off (b) Breadth trade-off

Figure 2: **Trade-offs between deep research tree structure and response quality.** Left figure (a) varies *depth* (breadth fixed at 4) and right figure (b) varies *breadth* (depth fixed at 3); in each, the *top* plot shows *Quality* metrics with sub-metric *Support* on the right y-axis, while the *bottom* plot shows *Relevance* (left) and *Faithfulness* (right). The red labels along the x-axis give total node counts as a proxy for computational cost. Early increases raise quality, but gains saturate as cost escalates.

Depth In Figure 2(a), increasing depth from 1 to 2 yields the largest gain, with overall quality score rising sharply from 77.00 to 80.95. Beyond depth 3, the curves flatten. Extra depth produces only marginal quality improvements while node number grows exponentially. Notably, *Relevance* and *Faithfulness* peak at depth 3 and then decline, as deeper searches bring in peripheral sources and redundant materials, diluting core evidence and complicating the write-up compression.

Breadth A similar pattern can be observed when varying the breadth. In Figure 2(b), widening the tree from breadth 1 to 2 delivers a substantial quality gain with a moderate increase in nodes. Quality continues to improve up to breadth 4, after which the gains taper off.

To summarize, initial increases in depth or breadth are valuable, but returns diminish as node counts escalate. This highlights that a one-size-fits-all approach is inefficient. Adaptive planning is essential to tailor depth and breadth to each query's complexity, optimizing the quality-cost tradeoff.

4 FLASHRESEARCH

FlashResearch consists of three core components: (1) an **Adaptive Research Planner**, (2) a **Real-Time Orchestration Layer**, and (3) a **Multi-Dimensional Parallelization Framework**. Together, these components dynamically expand and prune the research tree $\mathcal{T}$ in real time and execute sub-tasks concurrently.

4.1 ADAPTIVE RESEARCH PLANNING

To efficiently navigate vast information spaces, it is essential to adapt the breadth and depth of research to the scope, nature, and complexity of each query. For example, broad queries like *"What is the impact of climate change?"* can be decomposed into multiple subqueries that address distinct aspects.In contrast, narrower questions like *"What's the process for developing film in a darkroom?"* require less exploration but demand greater focus and precision.

Therefore, we propose an adaptive research planner that decomposes the query q^n into b_n sub-queries at each node $n^P \in N^P$, adjusting exploration breadth with a policy π_b contextualized on the accumulated research findings F:

$$n^P(q^n) = \pi_b(q^n, F) = (b_n, \{q_1^n, \ldots, q_{b_n}^n\}), \tag{6}$$

where b_n is the number of subqueries, $\{q_1^n, \ldots, q_{b_n}^n\}$ are the generated subqueries. A utility model guides the decision:

$$b_n = \underset{b \in [1, b_{\max}]}{\operatorname{argmax}} \ \mathbb{E}[U(b \mid q^n, F)], \tag{7}$$

Algorithm 1 Real-time Orchestration

function RESEARCHORCHESTRATOR(n_i^R, q^i, C_i, F_i, Φ_{min}, Ψ_{min})
  τ_i.should_terminate ← False ▷ Initialize termination flag for current subtree
  Async Execute research node n_i^R , updating C_i, F_i and parent node ▷ Interruptible
  Async Plan child queries
  for each child query q^j **do**
    Async Create n_j^R with C_j, F_j ▷ Speculative execution
    Async RESEARCHORCHESTRATOR(n_j^R, q^j, C_j, F_j, Φ_{min}, Ψ_{min}) ▷ Recursive monitor
  end for
  while not τ_i.should_terminate **do** ▷ Continuous monitor at current level
    Update C_i, F_i from node n_i^R and descendant nodes n_j^R
    Async Evaluate $(\delta_i, \phi_i, \psi_i) \leftarrow \pi_o(q^i, C_i, F_i)$
    if $\delta_i = 0$ and $\phi_i \geq \Phi_{\text{min}}$ and $\psi_i \geq \Psi_{\text{min}}$ **then**
      τ_i.should_terminate ← True
      Interrupt node n_i^R if ongoing ▷ Early termination
      for each descendant n_j^R **do**
        Terminate τ_j recursively ▷ Prune the descendant subtrees
      end for
    end if
    if n_i^R's task completed and all children completed/terminated **then**
      τ_i.should_terminate ← True
    end if
  end while
  return Aggregated results from C_i, F_i and children
end function

where U represents a utility function estimating the expected information gain and relevance of decomposing into b subqueries given the current context.

Once the breadth of exploration is set and the corresponding research is conducted, a policy π_d assesses whether to deepen the current research path based on the localized research findings F_i at each node $n_i^R \in N^R$:

$$\pi_d(q^i, F_i) = \mathbb{I}\{\mathbb{E}[U(F_{d+1} \mid q, F_i) - U(F_d)] > \tau\}, \tag{8}$$

where F_d represents the accumulated research findings at depth d, τ is a threshold for diminishing returns, and the output is a binary decision whether further exploration yields sufficient information gains to justify the continued exploitation of the current research path. In our work, π_b and π_d are instantiated with LLM agents to support adaptive and intelligent decision-making (see Appendix A.1 for details), though in principle such policies could also be realized through supervised training or reinforcement learning.

4.2 REAL-TIME ORCHESTRATION

To address the dynamic and iterative nature of research, where priorities can shift as new evidence emerges, breadth planning conducted *prior-research* and depth planning performed *post-research* can be limiting. Furthermore, depth planning can also delay the exploitation of promising research paths until decisions are finalized.

To mitigate these, we introduce a real-time orchestration layer that dynamically manages the research tree $\mathcal{T}$ based on *mid-research* signals, enabling speculative execution and resource reallocation. Each research node $n^R(q^i)$ is continuously monitored by the orchestration policy π_o based on the local query q^i, real-time context C_i, and accumulated findings F_i:

$$\pi_o(q_i, C_i, F_i) = (\delta_i, \phi_i, \psi_i) = \begin{cases} (0, \phi_i, \psi_i) & \text{if } \phi_i \geq \Phi_{\text{min}} \text{ and } \psi_i \geq \Psi_{\text{min}}, \\ (1, \phi_i, \psi_i) & \text{otherwise,} \end{cases} \tag{9}$$

where $\delta_i \in \{0, 1\}$ indicates task scheduling signals for continuation ($\delta_i = 1$) or termination ($\delta_i = 0$), based on whether the goal satisfaction score $\phi_i \in [0, 1]$ satisfies the goal satisfaction threshold Φ_{min},

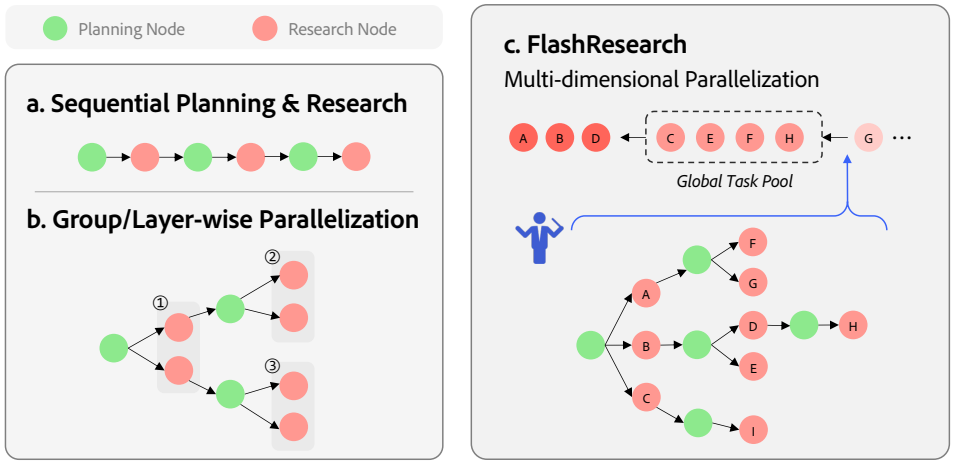


Figure 3: Sequential processing and group/layer parallelization introduce unnecessary latency by forcing nodes to wait for slow dependencies. FlashResearch supports multi-dimensional parallelization by submitting research nodes to a global task pool, where they are executed as soon as resources are available—so child nodes (e.g., D, E, F) can start immediately once their parents (A, B) finish, without being delayed by unrelated nodes like C.

and the quality score $\psi_i \in [0, 1]$ satisfies the quality threshold Ψ_{min}. The policy is also implemented via an LLM agent, detailed in Appendix A.2.

More importantly, this mechanism enables *speculative execution*: child nodes can be spawned and deepen the tree without awaiting the parent's planning decision. Child nodes' findings update the parent's C_i and F_i, even after the parent's research completes, enabling recursive and adaptive task management. As shown in Algorithm 1, upon evaluation at each hierarchy, low-yield nodes and their descendants are terminated early once the research goal is satisfied, pruning the subtree dynamically.

4.3 Multi-Dimensional Parallelization

To maximize efficiency in traversing the adaptive research tree $\mathcal{T}$, FlashResearch incorporates a multi-dimensional parallel execution engine that enables concurrent processing across multiple axes: breadth (parallel subqueries at the same level), depth (speculative deepening of paths), and across the real-time orchestrators at different recursive hierarchies in Algorithm 1.

The engine operates by submitting all research nodes n_i^R to a global asynchronous task pool as soon as they are planned and orchestrated. Each node is parameterized by its local query q^i, depth d_i, parent identifier p_i, and a unique task identifier t_i. Dependencies are enforced dynamically: a child node n_j^R (with $p_j = t_i$) becomes eligible for execution only once its parent n_i^R completes its initial research phase, but speculative spawning allows planning and partial execution to begin earlier under the real-time orchestrator's guidance.

FlashResearch's engine leverages non-blocking asynchronous calls, allowing tasks to progress independently. This approach mitigates bottlenecks inherent in sequential or coarse-grained parallel execution, where nodes must wait for unrelated dependencies to complete. For instance, as illustrated in Figure 3, child nodes (e.g., D, E, F) can initiate immediately upon their respective parents' (A, B) completion, without delays from slower siblings like C.

5 Experiments

5.1 Benchmarks and Evaluation Metrics

We evaluate our method on two recent deep research benchmarks.

DeepResearchGym (Coelho et al., 2025) is an open-source evaluation sandbox for deep research systems. The benchmark consists of the top 1,000 complex, high-engagement non-factoid queries from the Researchy Questions dataset (Rosset et al., 2025). We randomly sampled 100 for evaluation. It employs an LLM-as-a-judge protocol to assess generated reports along three dimensions:

- **Quality** Rates organization, clarity, and coherence of the synthesis. Prefers well-structured, readable reports that integrate evidence into concise, actionable takeaways.
- **Relevance** Judges whether the report directly answers the user's intent, covering the key subquestions and constraints in the prompt. Penalizes omissions and off-topic content.
- **Faithfulness** Assesses whether the cited evidence supports claims. Rewards correctly grounded statements and flags contradictions, unsupported claims, or hallucinations.

DeepResearch Bench (Du et al., 2025) comprises 100 PhD-level research tasks across 22 distinct fields. These tasks were designed by domain experts based on a statistical analysis of over 96,000 real-world user queries from web search–enabled LLM interactions. The benchmark includes 50 English and 50 Chinese tasks. We focus on the English subset in evaluation. The benchmark proposes two evaluation frameworks: RACE for report quality and FACT for citation trustworthiness.

- **RACE**:
  - **Comprehensiveness**: Evaluates thorough coverage of relevant aspects, including diverse perspectives and key subtopics, ensuring understanding without omissions.
  - **Depth**: Assesses level of detail, analysis, and insights beyond surface-level, including causes, impacts, and trends.
  - **Instruction following**: Checks adherence to query requirements, ensuring alignment with intent by following the topic and answering directly.
  - **Readability**: Assesses clarity through structure, language, and ease of understanding.
- **FACT**:
  - **Effective citation count**: Measures factual abundance by counting the number of unique, relevant citations that effectively support key statements in the report.
  - **Citation accuracy**: Evaluates citation trustworthiness by assessing the proportion of citations that effectively support the referenced claims.

5.2 Evaluation Setup

To ensure a fair comparison, we build FlashResearch on top of GPT-Researcher's agentic workflow (Elovic, 2023), and evaluate FlashResearch against the original GPT-Researcher, using FineWeb (Penedo et al., 2024) as the static web corpora to improve reproducibility. On DeepResearchGym, we fix maximum execution times at 2 and 10 minutes to reflect realistic usage scenarios:

- The 2-minute cutoff reflects human multitasking behavior, where information workers spend an average of $\sim$2–3 minutes on events or tools before task switching (González & Mark, 2004).
- The 10-minute threshold matches the average duration of a "working sphere" (González & Mark, 2004) and is supported by evidence from high-performance computing (Schlagkamp & Renker, 2015) and crowdsourcing tasks (Bernstein et al., 2011), indicating that a 10-minute window preserves task continuity without losing human-in-the-loop coordination.

5.3 Results

DeepResearchGym. We evaluated FlashResearch against both GPT-Researcher and an ablated variant, FlashResearch*, which omits adaptive research planning and real-time orchestration. The results in Table 1 show that FlashResearch consistently delivers superior throughput, processing substantially more research than the GPT-Researcher baseline (up to 4.11$\times$ in the 10-minute setup). This efficiency gain arises from adaptive planning and real-time orchestration, which enable speculative execution without compromising quality.

Beyond throughput, FlashResearch also improves overall response quality, with clear improvements in balance, breadth, and insight metrics. These gains highlight its ability to maintain a robust trade-off between expansive coverage and focused analysis, particularly under tight time constraints.

Table 1: Evaluation of deep research frameworks on DeepResearchGym under fixed time budgets. Scores are assessed by `gpt-4.1-mini-2025-04-14`, adhering to the benchmark's practice.

	Throughput	Quality							Relevance	Faithfulness
	# Nodes	**Overall**	Clarity	Depth	Balance	Breadth	Support	Insight	KPR + KPC	Cit. Recall
2 minutes										
GPT-Researcher	8.00	76.14	79.40	89.30	83.77	91.17	32.23	81.00	55.75	85.84
FlashResearch*	**21.14**	80.70	81.40	89.00	86.50	89.70	51.10	86.50	**68.00**	94.21
FlashResearch	19.42	**82.13**	**83.30**	**89.70**	**88.10**	**91.40**	**52.40**	**87.90**	62.41	**94.37**
10 minutes										
GPT-Researcher	23.94	81.19	81.77	**90.10**	86.77	91.40	49.03	88.07	64.56	95.51
FlashResearch*	68.00	85.25	**83.40**	89.70	87.40	94.10	**68.80**	88.10	**67.98**	**96.86**
FlashResearch	**98.43**	**85.40**	81.70	**90.10**	**88.10**	**95.40**	67.70	**89.40**	65.41	96.15

Table 2: Overall evaluation results on DeepResearch Bench under flexible time budgets, judged by `Gemini-2.5-flash` and `Gemini-2.5-pro`. Commercial deep research agents' results are directly from the DeepResearch Bench Leaderboard[1], which has no latency statistics reported.

Method	Throughput		RACE					FACT	
	# Nodes	**Latency**	**Overall**	**Comp.**	**Depth**	**Inst.**	**Read.**	**Cit. Acc.**	**Eff. Cit.**
Grok Deeper Search	-	-	38.22	36.08	30.89	46.59	42.17	73.08	8.58
Perplexity Research	-	-	40.46	39.10	35.65	46.11	43.08	82.63	31.20
OpenAI Deep Research	-	-	46.45	46.46	43.73	49.39	47.22	75.01	39.79
Gemini-2.5-Pro Deep Research	-	-	49.71	49.51	49.45	50.12	50.00	78.30	165.34
GPT-Researcher	23.12	554.41 s	41.15	38.58	37.55	46.03	45.62	65.58	9.40
FlashResearch*	27.88	**207.06** s	41.33	38.61	38.09	46.01	45.80	70.06	17.35
FlashResearch	**39.30**	367.88 s	**41.92**	**39.55**	**38.61**	**46.36**	**45.83**	58.25	22.94

Notably, the overall quality of FlashResearch with 2-minute execution even surpasses that of the GPT-Researcher baseline with 10 minutes, demonstrating a 5× speed-up while preserving quality. Overall, these results underscore FlashResearch 's capacity for dynamic adaptation, producing more comprehensive and higher-quality research outputs under constrained budgets.

DeepResearch Bench. To better assess our system's performance relative to other deep research agents, we also evaluate it under flexible time budgets on DeepResearch Bench. As shown in Table 2, FlashResearch achieves substantial efficiency gains, processing 39.3 nodes on average while reducing latency by 1.51× compared to the GPT-Researcher baseline.

In terms of quality, our proposed framework also consistently improves across all RACE submetrics. Compared to the ablated method, FlashResearch incurs higher latency but conducts more research at a comparable throughput, highlighting a favorable trade-off between efficiency and comprehensiveness. Importantly, FlashResearch also achieves a performance competitive with commercial systems like Grok Deeper Search and Perplexity Research, narrowing the gap with state-of-the-art proprietary agents. Beyond quantitative metrics, we also provide a detailed case analysis of FlashResearch's adaptability across diverse query conditions in Appendix B.

6 Conclusion

We present FlashResearch to enhance the efficiency of deep research tasks through adaptive planning, real-time orchestration, and multi-dimensional parallelization. By formulating deep research as a tree-structured process and dynamically allocating resources to promising paths, FlashResearch achieves significant improvements in research throughput and response quality, as demonstrated through extensive evaluations on two deep research benchmarks. Future work includes incorporating richer modalities beyond text and exploring tighter integration with human-in-the-loop monitoring and interruption to further improve transparency and usability.

[1] `https://huggingface.co/spaces/Ayanami0730/DeepResearch-Leaderboard`

Reproducibility Statement

Additional implementation details and experimental setups are included in the Appendix A. The complete source code and instructions for reproducing all experiments are available at the following anonymous repository: `https://anonymous.4open.science/r/FlashResearch/`.

Ethics Statement

This work does not involve human subjects, private data, or sensitive information. Experiments were conducted using publicly available datasets. While our framework aims to improve the efficiency and quality of deep research systems, we acknowledge the broader risks of misuse, including the potential amplification of biased or unreliable information. Responsible deployment requires careful selection of data sources, robust fact-checking, and adherence to ethical standards.

References

Salaheddin Alzubi, Creston Brooks, Purva Chiniya, Edoardo Contente, Chiara von Gerlach, Lucas Irwin, Yihan Jiang, Arda Kaz, Windsor Nguyen, Sewoong Oh, et al. Open deep search: Democratizing search with open-source reasoning agents. *arXiv preprint arXiv:2503.20201*, 2025.

Michael S Bernstein, Joel Brandt, Robert C Miller, and David R Karger. Crowds in two seconds: Enabling realtime crowd-powered interfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pp. 33–42, 2011.

Tianle Cai, Jiayi Li, Haotian Geng, Yuxin Ge, Shizhe Zhang, et al. Medusa: Simple llm inference acceleration framework with multiple decoding heads. *arXiv preprint arXiv:2401.10774*, 2024.

João Coelho, Jingjie Ning, Jingyuan He, Kangrui Mao, Abhijay Paladugu, Pranav Setlur, Jiahe Jin, Jamie Callan, João Magalhães, Bruno Martins, et al. Deepresearchgym: A free, transparent, and reproducible evaluation sandbox for deep research. *arXiv preprint arXiv:2505.19253*, 2025.

Yifu Ding, Wentao Jiang, Shunyu Liu, Yongcheng Jing, Jinyang Guo, Yingjie Wang, Jing Zhang, Zengmao Wang, Ziwei Liu, Bo Du, et al. Dynamic parallel tree search for efficient llm reasoning. *arXiv preprint arXiv:2502.16235*, 2025.

Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. Deepresearch bench: A comprehensive benchmark for deep research agents. *arXiv preprint arXiv:2506.11763*, 2025.

Assaf Elovic. GPT Researcher: LLM based autonomous agent that conducts deep local and web research on any topic and generates a long report with citations, July 2023. URL `https://github.com/assafelovic/gpt-researcher`.

Dane Gambrell. Ai for egovernance: Combining artificial intelligence and collective intelligence to develop evidence-based ai policy. In *2025 Eleventh International Conference on eDemocracy & eGovernment (ICEDEG)*, pp. 317–324. IEEE, 2025.

Victor M González and Gloria Mark. " constant, constant, multi-tasking craziness" managing multiple working spheres. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 113–120, 2004.

Michael Haman and Milan Školník. Fake no more: the redemption of chatgpt in literature reviews. *Accountability in Research*, pp. 1–3, 2025.

Shamsi T Iqbal and Eric Horvitz. Disruption and recovery of computing tasks: field study, analysis, and directions. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 677–686, 2007.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. Dspy: Compiling declarative language model calls into self-improving pipelines. *The Twelfth International Conference on Learning Representations*, 2024.

LangChain. Langgraph: Build resilient language agents as graphs, 2024. URL https://langchain-ai.github.io/langgraph/.

Langchain. Open deep research, 2025. URL https://github.com/langchain-ai/open_deep_research.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19274–19286. PMLR, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Gloria Mark, Daniela Gudith, and Ulrich Klocke. The cost of interrupted work: more speed and stress. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pp. 107–110, 2008.

Xiaohui Miao, Senzhang Zuo, Ang Li, Jiaqi Guo, Xiaoying Zhang, Yifan Xing, Xiaoliang Qian, Yanzhi Gao, Jingren Zhou, and Fan Zhou. Specinfer: Accelerating large language model serving with speculative inference. *arXiv preprint arXiv:2305.09781*, 2023.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Boye Niu, Yiliao Song, Kai Lian, Yifan Shen, Yu Yao, Kun Zhang, and Tongliang Liu. Flow: A modular approach to automated agentic workflow generation. *arXiv preprint arXiv:2501.07834*, 2025.

OpenAI. Swarm: Educational framework exploring ergonomic, lightweight multi-agent orchestration. https://github.com/openai/swarm, 2024.

Rui Pan, Yinwei Dai, Zhihao Zhang, Gabriele Oliaro, Zhihao Jia, and Ravi Netravali. Specreason: Fast and accurate inference-time compute via speculative reasoning. *arXiv preprint arXiv:2504.07891*, 2025.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, et al. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849, 2024.

Corbin Rosset, Ho-Lam Chung, Guanghui Qin, Ethan Chau, Zhuo Feng, Ahmed Awadallah, Jennifer Neville, and Nikhil Rao. Researchy questions: A dataset of multi-perspective, decompositional questions for deep research. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 3712–3722, 2025.

Stephan Schlagkamp and Johanna Renker. Acceptance of waiting times in high performance computing. In *International Conference on Human-Computer Interaction*, pp. 709–714. Springer, 2015.

Hao Wen, Yifan Su, Feifei Zhang, Yunxin Liu, Yunhao Liu, Ya-Qin Zhang, and Yuanchun Li. Parathinker: Native parallel thinking as a new paradigm to scale llm test-time compute. *arXiv preprint arXiv:2509.04475*, 2025.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 3(4), 2023.

Renjun Xu and Jingwen Peng. A comprehensive survey of deep research: Systems, methodologies, and applications. *arXiv preprint arXiv:2506.12594*, 2025.

Wang Yang, Xiang Yue, Vipin Chaudhary, and Xiaotian Han. Speculative thinking: Enhancing small-model reasoning with large model guidance at inference time. *arXiv preprint arXiv:2504.12329*, 2025.

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, 2023a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023b.

Guibin Zhang, Kaijie Chen, Guancheng Wan, Heng Chang, Hong Cheng, Kun Wang, Shuyue Hu, and Lei Bai. Evoflow: Evolving diverse agentic workflows on the fly. *arXiv preprint arXiv:2502.07373*, 2025.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, et al. Aflow: Automating agentic workflow generation. *arXiv preprint arXiv:2410.10762*, 2024.

Tong Zheng, Hongming Zhang, Wenhao Yu, Xiaoyang Wang, Xinyu Yang, Runpeng Dai, Rui Liu, Huiwen Bao, Chengsong Huang, Heng Huang, et al. Parallel-r1: Towards parallel thinking via reinforcement learning. *arXiv preprint arXiv:2509.07980*, 2025a.

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*, 2025b.

A Implementation Details

A.1 Adaptive Research Planning

The adaptive research planner dynamically decomposes queries into subqueries using LLM-based policies π_b and π_d. The following prompt is used by π_b to determine the optimal number of subqueries b_n, balancing exploration breadth based on the utility model in Equation (7):

Prompt 1

You are an expert researcher generating search queries. Your task is to determine the OPTIMAL number of clear, non-overlapping search queries.
EFFICIENCY IS CRITICAL: More subqueries do not necessarily lead to better research. Minimize waste and redundancy. Highly specific queries need fewer subqueries. Broad topics may need more.

SUBQUERY REQUIREMENTS:
- Do not exceed [max_breadth+flex_breadth] subqueries
- Keep queries clear and concise
- Make each subquery target a DISTINCT aspect
- Avoid near-duplicates and trivial variants
- Prefer fewer subqueries if coverage is maintained
- Ensure queries are relevant to the high-level research goal: [initial_query]
- Exclude overlap with existing learnings: [accumulated_learnings]

This prompt enables the generation of subqueries, adjusting the breadth b_n based on the query and accumulated research findings F, as described in Equation (6).

A.2 Real-Time Orchestration

The real-time orchestration layer monitors task execution with a policy π_o to assess goal satisfaction and quality. The following prompt implements π_o to compute goal satisfaction (ϕ_n) and quality (ψ_n) scores, enabling speculative execution and early termination as per Equation (9):

Prompt 2

You are an expert research quality evaluator. Determine if a research goal has been sufficiently satisfied based on current findings.

EVALUATION CRITERIA:
1. GOAL COVERAGE: Does the research adequately address the stated goal?
2. INFORMATION QUALITY: Are the findings comprehensive and reliable?
3. DEPTH SUFFICIENCY: Is there enough detail to answer the research question?
4. SOURCE DIVERSITY: Are findings from multiple credible sources?
5. COMPLETENESS: Are major aspects of the topic covered?

SATISFACTION SCORE:
- HIGH SATISFACTION (0.8-1.0): Goal fully satisfied, comprehensive coverage
- MEDIUM SATISFACTION (0.5-0.8): Goal mostly satisfied, minor gaps acceptable
- LOW SATISFACTION (0.3-0.5): Goal partially satisfied, significant gaps remain
- INSUFFICIENT (0.0-0.3): Goal not satisfied, major research needed

QUALITY SCORING:
- EXCELLENT (0.8-1.0): Comprehensive, well-sourced, detailed
- GOOD (0.5-0.8): Adequate coverage, some depth
- FAIR (0.3-0.5): Basic coverage, limited depth
- POOR (0.0-0.3): Insufficient information

Be conservative - only mark as satisfied if the research truly addresses the goal comprehensively.

This prompt facilitates the evaluation of task outputs against thresholds Φ_{min} and Ψ_{min}, pruning low-yield paths dynamically. In our experiments, both thresholds Φ_{min} and Ψ_{min} are set to 0.8.

A.3 Experimental Setups

For model configuration, we use `gpt-4.1-mini-2025-04-14` for the main research processing, and `o3-mini-2025-01-31` for major policy decisions, including adaptive research planning and real-time orchestration.

To ensure fair comparisons among deep research systems, we impose a maximum execution time for research trees. Once the time cut-off is reached, the research process terminates immediately, and the system generates a response based on the findings and context gathered up to that point. To allow all systems to fully utilize their time budgets, we set the maximum tree depth to 10 and the maximum breadth to 4 within the GPT-Researcher framework. To provide additional flexibility, the adaptive research planning module may expand the breadth up to 6 when necessary.

Additionally, to control the computational costs introduced by the real-time orchestration layer, we set an interval of 8 seconds between successive evaluations of goal satisfaction and research quality.

B Case Analysis

To illustrate how FlashResearch adapts its research process to different query conditions, we present the research trees from three cases in DeepResearch Bench. For a controlled comparison, we standardize the time cutoff to 2 minutes across all cases.

In Case 1 (Figure 4), the query concerns a broad topic: *investigating current non-alcoholic cocktails*. The tree expands widely across multiple layers of branching, encompassing diverse angles such as ingredient sourcing, sustainability, AI-driven methods, and integration with fine dining. This highlights FlashResearch 's ability to surface complementary perspectives when the query is exploratory and open-ended.

In contrast, Case 2 yields a more compact tree in Figure 5 for a narrow, domain-specific query on *cislunar situational awareness*. The decomposition focuses on specialized aspects, including hybrid sensor fusion, bias-correction methods, and federated filtering. With the research goal achievable through limited exploration, FlashResearch terminates at depth 2 to avoid redundant effort.

Finally, Case 3 depicts a common scenario where users explicitly specify particular focuses for a deeper investigation. As shown in Figure 6, FlashResearch tailors the tree to these requirements, deepening its analysis of *AI's disruptions across various industries*.

Collectively, these cases demonstrate FlashResearch 's adaptability: expanding broadly for open domains, conserving resources when goals can be met with focused research, and tailoring scope when users impose explicit constraints.

C Use of LLMs

We used LLMs as assistive tools for (i) polishing the manuscript; (ii) revising the phrasing of aforementioned LLM prompts, and (iii) refining code. LLMs did not contribute to research ideation, experimentation, or the formulation of claims.

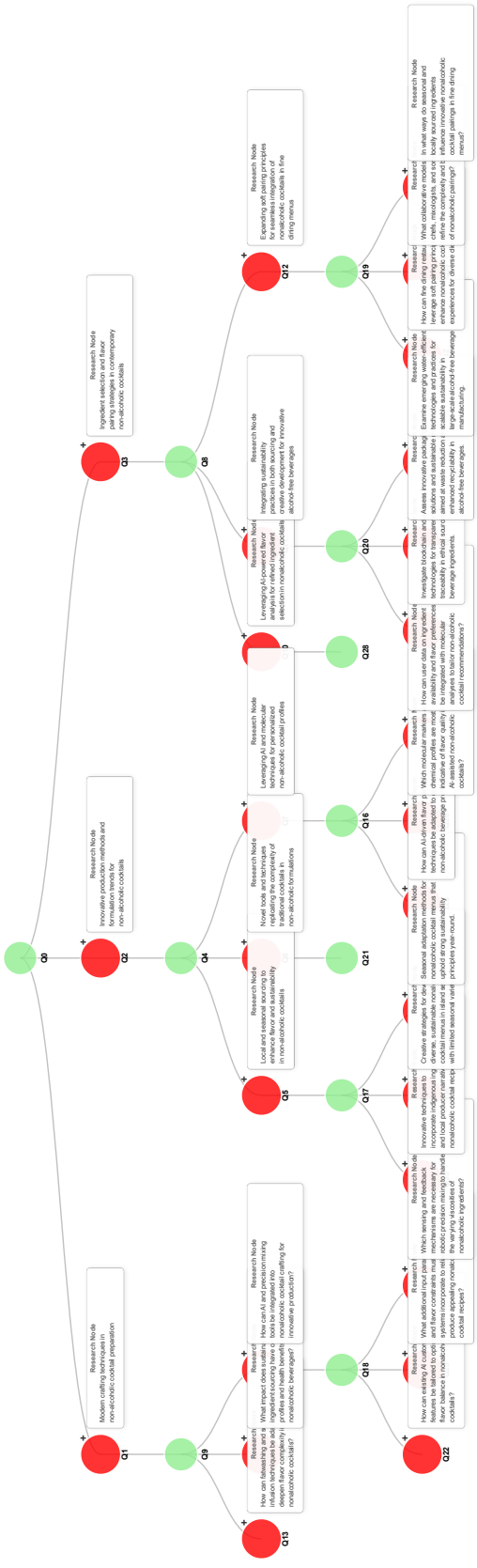


Figure 4: Research tree generated by FlashResearch for a broad-topic query: *"Research Topic: Crafting Techniques for Non-Alcoholic Cocktails. Objective: Investigate current non-alcoholic cocktails to discover innovative production methods and formulations."*

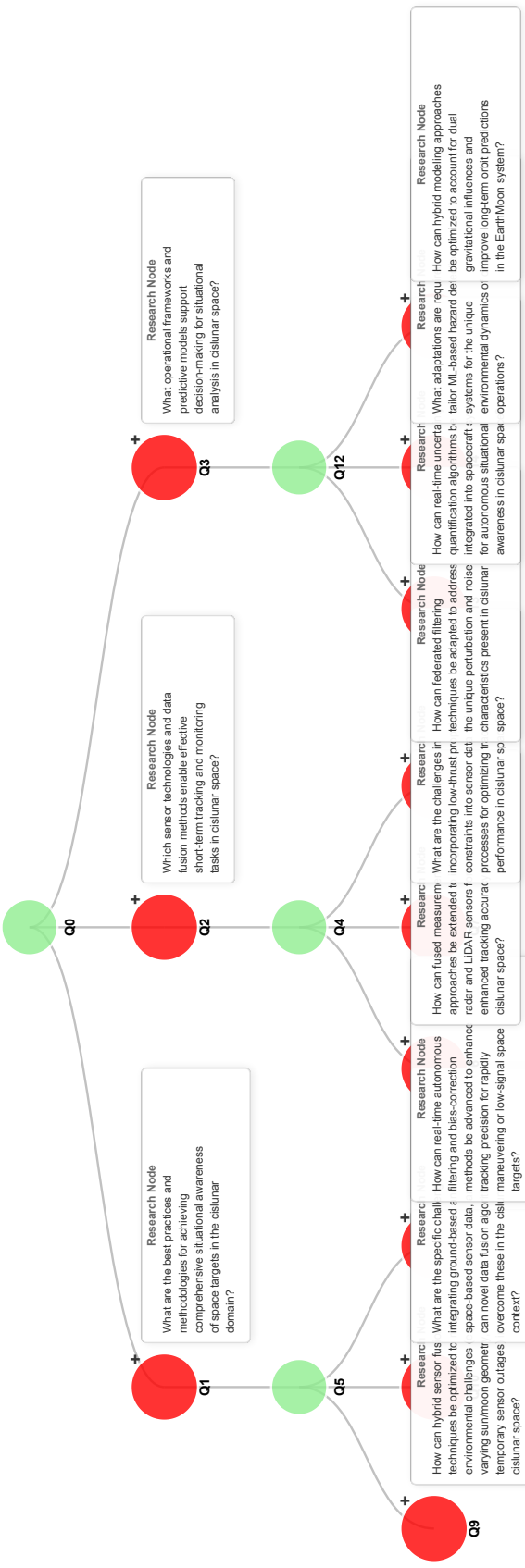


Figure 5: Research tree generated by FlashResearch for a narrow, domain-specific query: "*How to conduct comprehensive and accurate situational awareness of space targets in the cislunar space, and support the effectiveness of short-term cislunar space tracking and monitoring tasks?*"

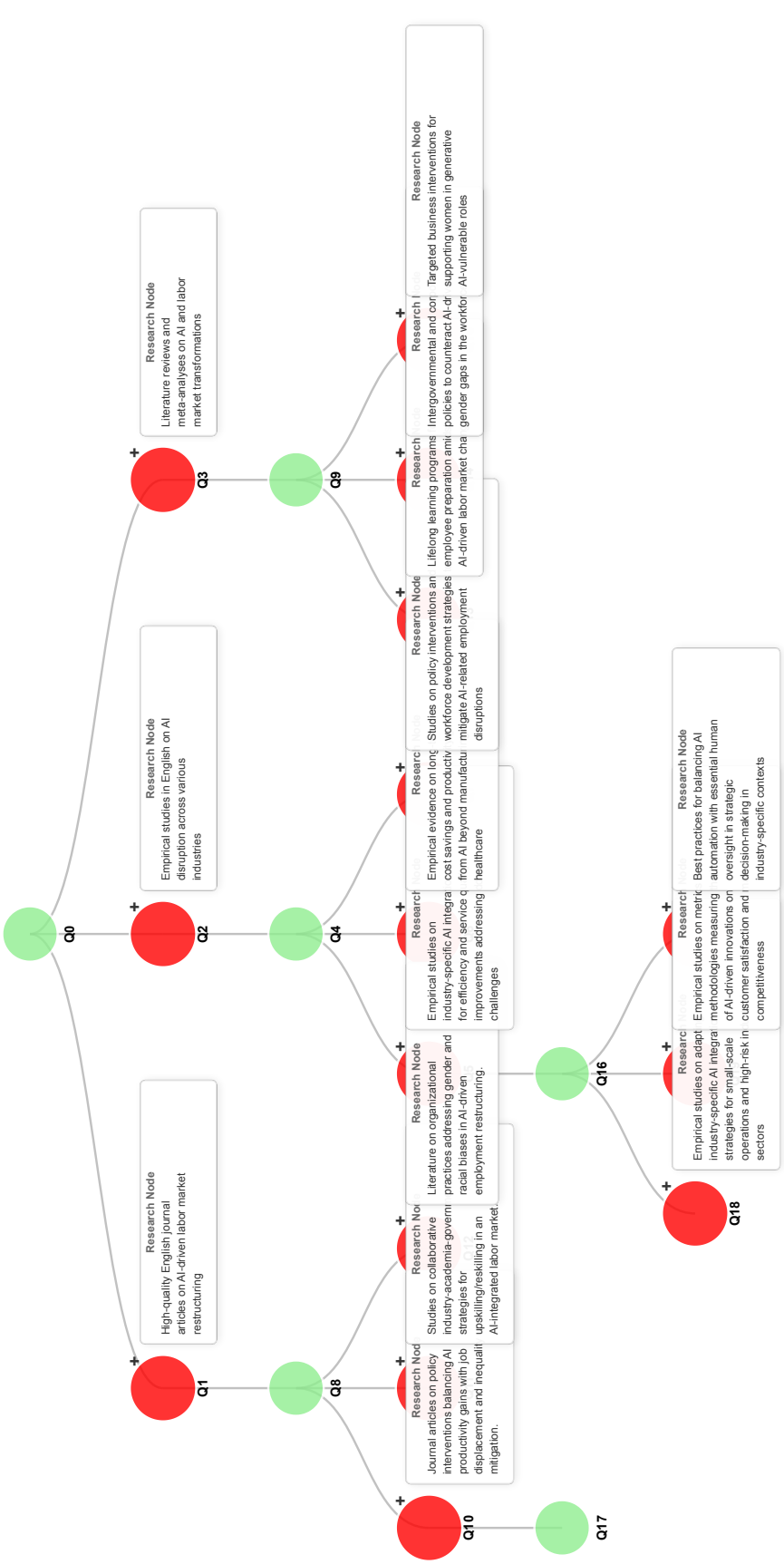


Figure 6: Research tree generated by FlashResearch for a user-focused query with explicit demands: *"Please write a literature review on the restructuring impact of Artificial Intelligence (AI) on the labor market. Focus on how AI, as a key driver of the Fourth Industrial Revolution, is causing significant disruptions and affecting various industries. Ensure the review only cites high-quality, English-language journal articles."*