DUALSPEC: ACCELERATING DEEP RESEARCH AGENTS VIA DUAL-PROCESS ACTION SPECULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model-based deep research agents have been increasingly popular for addressing long-horizon information-seeking tasks, but they often incur high end-to-end latency due to extensive reasoning and frequent tool use. Speculation frameworks aim to reduce latency by overlapping action execution with reasoning; however, existing approaches typically rely on uniform speculation strategies and strict action matching, which limits inference speedups and robustness. In this work, we revisit the speculate-verify paradigm for deep research agents through the lens of action heterogeneity. We show that *Search* and *Visit* actions exhibit fundamentally different reasoning and model capacity requirements: entropy-based analysis reveals that Search decisions have higher uncertainty and benefit significantly from explicit reasoning, whereas Visit decisions have lower entropy and depend primarily on model capacity. Motivated by this dual-process characteristic, we propose DualSpec, a heterogeneous speculation framework equipped with a lightweight, confidence-based semantic verifier. Experiments across multiple models and benchmarks demonstrate that DualSpec achieves up to 3.28× end-to-end speedup while maintaining accuracy comparable to fully reasoning agents.

1 INTRODUCTION

The increasing reasoning capabilities of large language models (LLMs) enable interaction with external tools and environments, driving the development of intelligent agents Yao et al. (2023); Shinn et al. (2023); Schick et al. (2023). Among these, deep research agents have emerged as a prominent application for addressing open-ended, long-horizon research tasks with high information-seeking and reasoning demands OpenAI (2025). By iteratively reasoning and invoking external tools such as search engines, these agents accumulate evidence and refine hypotheses, extending beyond static question answering to complex research problems.

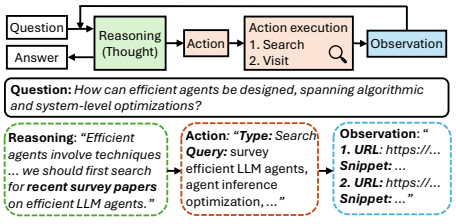

Figure 1: Deep research agent workflow following a Reason-Action-Observation loop.

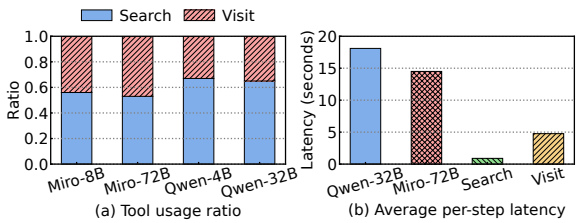

Figure 2: Deep research inference characteristics using different models and GAIA benchmark. "Miro" denotes "MiroThinker" while "Qwen" denotes "Qwen-3"

Despite their effectiveness, deep research agents often incur high inference latency. As shown in Figure 1, agents typically follow the ReAct paradigm Yao et al. (2023) with strict sequential dependencies: the model must complete a reasoning trace before emitting an action and then wait for the resulting observation before proceeding. Both reasoning and action execution can be time-consuming, particularly when using large models with long reasoning traces and external tools with variable response timesKim et al. (2023); Zhang et al. (2025a). This reasoning–action–observation cycle repeats until a final answer is produced, often taking minutes or longer per query.

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

A promising approach to reduce latency is the speculate–verify paradigm, where a lightweight model or strategy speculates the next action and executes it immediately, while the base model concurrently performs its reasoning Huang et al. (2025b); Guan et al. (2025). If the base model's action matches the speculative one, the speculative observation is directly accepted, saving execution time; otherwise, the base model executes the action as usual. Unlike speculative decoding Leviathan et al. (2023), this approach operates at the action level rather than the token level, enabling parallelism between reasoning and tool use. However, designing effective speculation and verification remains challenging: inaccurate speculation or conservative verification leads to frequent fallbacks and limited speedups, whereas overly permissive verification risks degrading agent performance.

In this work, we rethink speculate–verify for deep research agents through a principled analysis of action heterogeneity and verification trade-offs. Existing lightweight speculation methods generally adopt either (i) small models with explicit reasoning or (ii) large models that emit actions without reasoning. We observe that different action types exhibit distinct uncertainty profiles and thus require different speculation strategies. Deep research agents primarily use two actions: *Search*, which formulates a query to retrieve relevant webpages, and *Visit*, which selects and accesses a specific URL from a candidate set. Search involves high query uncertainty and benefits from reasoning, whereas Visit operates over a constrained action space and relies mainly on parametric knowledge.

We validate this distinction via end-to-end empirical evaluations, together with an entropy-based analysis of action decisions with and without reasoning. Across settings, Search actions exhibit much higher uncertainty than Visit actions; explicit reasoning helps reduce uncertainty for Search but provides marginal gains for Visit. This pattern aligns with the cognitive science distinction between *System 2* (deliberative) and *System 1* (intuitive) reasoning, with Search corresponding to the former and Visit to the latter. Guided by these insights, we show that matching speculation strategies to action characteristics – using a small reasoning model for Search and a large model without reasoning for Visit – significantly improves speculation accuracy.

Verification is also critical for achieving efficiency without sacrificing performance. Exact action matching is often overly restrictive, as semantically equivalent actions, especially queries, may differ at the token level. Moreover, action-based verification typically requires the base model to complete reasoning before verification, placing reasoning on the critical path and limiting latency reductions.

Based on these observations, we propose DualSpec, a heterogeneous action speculation framework for deep research agents that tailors speculation and verification to action-specific properties. DualSpec uses a small reasoning model to speculate actions, while allowing the base model to concurrently generate a Visit action by skipping reasoning. It dynamically selects an appropriate draft based on the reasoning state. For verification, DualSpec leverages the base model's internal confidence rather than explicit action matching, removing base-model reasoning from the critical path while preserving the agent performance.

We implement DualSpec and evaluate it on two representative reasoning models, MiroThinker Team et al. (2025) and Qwen3 Yang et al. (2025), using popular deep research benchmarks including GAIA-Text-103 Wu et al. (2025), XBench-DeepSearch Chen et al. (2025), and Seal-0 Pham et al. (2025). DualSpec achieves up to **3.28**$\times$ end-to-end latency speedup while maintaining performance comparable to the fully reasoning base model.

2 Background and Related Work

2.1 Deep Research Agents

Given an input question, deep research agents operate in a multi-step loop that alternates between reasoning to generate an action, executing the tool call, and incorporating the response into its context, until producing a final answer Zhang et al. (2025b); Huang et al. (2025a). Unless otherwise stated, micro-level analyses use GAIA; similar trends are observed across other benchmarks.

Most deep research agents rely on two core actions: Search and Visit. Search consists of a query used to retrieve candidate webpages with brief snippets, while Visit selects a URL and specifies an instruction for extracting relevant information. During execution, Search directly queries a search engine, whereas Visit accesses the webpage and typically invokes an LLM to summarize task-relevant content according to the instruction Nakano et al. (2021); Zhou et al. (2023). This design filters

irrelevant information and limits unnecessary context growth. Figure 2(a) shows that these actions occur at comparable frequencies, with Search used slightly more often due to query reformulation when results are unsatisfactory. Other tools (e.g., code execution) account for few steps.

Despite strong problem-solving performance, deep research agents often incur high latency due to their multi-step reasoning and tool-use workflows. Figure 2(b) reports the per-step time breakdown measured on the A100 GPU. Model reasoning dominates total latency, while tool execution introduces additional, though smaller, overheads. Accumulated over many iterations, these costs lead to long end-to-end response times, limiting usability and deployment.

2.2 Agent Optimization

Stronger backbone models and richer tool interactions improve agent performance on challenging tasks Shang et al. (2025), but often reduce time efficiency due to longer reasoning traces and more frequent tool use. Recent work observes that agent steps vary in difficulty Zhang et al. (2023); Saha et al. (2024), motivating approaches that delegate simpler steps to lightweight models while reserving complex reasoning for stronger models.

Speculate–verify paradigms have emerged as an effective strategy to reduce agent latency Ye et al. (2025); Guan et al. (2025); Hua et al. (2024); Wang et al. (2025). Dynamic Speculative Planning Guan et al. (2025) employs a small reasoning model to draft an action and obtain its result while a stronger model performs full reasoning; the speculative action is verified against the base model's action using criteria such as minimum edit distance, and its tool response is reused upon agreement. SPAgent Huang et al. (2025b) skips explicit reasoning in early stages and transitions to a speculate–verify phase later to maintain performance.

At a finer granularity, speculative decoding Leviathan et al. (2023) accelerates LLM inference at the token level by predicting future tokens with a smaller model and verifying them with a larger model. Combining it with agent-level speculation can further improve efficiency. SpecReason Pan et al. (2025) reduces reasoning overhead by dynamically offloading simpler reasoning steps to a smaller model; however, it is not designed for agent-based settings with iterative tool use.

2.3 Dual-Process Theory in LLMs

Dual-process theory Chaiken & Trope (1999) from cognitive science distinguishes between System 1, which is fast and intuitive, and System 2, which supports slower, deliberate reasoning. This framework has recently been applied to interpret and guide LLM reasoning. The System-1.x Planner Saha et al. (2024) decomposes tasks into simpler and more complex sub-steps, assigning System 1 strategies to the former and System 2 strategies to the latter to improve efficiency. However, it targets specific planning settings and requires extensive training. Applying dual-process principles to tool-using LLM agents, including deep research agents, remains largely unexplored.

3 Rethinking Speculate–Verify for Deep Research Agents

Existing speculate–verify frameworks typically apply a *uniform* speculation strategy across all actions, either by (i) reducing reasoning depth (e.g., skipping explicit reasoning) or (ii) reducing model capacity (e.g., using a smaller speculator). While effective in some settings, this overlooks a key property of deep research agents: *actions exhibit heterogeneous reasoning demands*. In this section, we provide empirical evidence that Search and Visit actions differ fundamentally in their sensitivity to reasoning depth and model capacity, motivating an action-aware speculation design. We further analyze verification trade-offs, highlighting the need to move beyond action matching.

3.1 Speculation Under Action Heterogeneity

We first examine the reasoning demands of different actions and then study how various speculation strategies affect the accuracy of drafting Search and Visit actions.

Observation 1: Search actions involve longer reasoning traces than Visit. Figure 3 shows the average reasoning length before emitting an action. Across all models, Search consistently requires

Table 1: End-to-end agent performance (pass@1 accuracy) on GAIA under action-level replacements. We selectively replace the generation strategy for Search or Visit using lightweight methods.

Configuration	MiroThinker	Qwen3
LLM with Reasoning	**63.11**	**29.13**
SLM with Reasoning for Search	**63.11**	**29.13**
LLM without Reasoning for Search	57.28	23.30
SLM without Reasoning for Search	55.34	22.33
SLM with Reasoning for Visit	56.31	22.33
LLM without Reasoning for Visit	**64.08**	**28.16**
SLM without Reasoning for Visit	54.37	24.27

1.65–2.95$\times$ more tokens than Visit, indicating that query formulation demands more deliberation than webpage selection. Thus, Search requires stronger reasoning.

Observation 2: Effective speculation strategies differ across actions. We measure the alignment of speculative actions with an Oracle agent (large model with full reasoning) using two representative strategies: a small language model (SLM) with explicit reasoning and a large language model (LLM) that skips reasoning. As shown in Figure 4(a), for Search actions, the SLM with reasoning consistently produces queries more aligned with the Oracle than the LLM without reasoning, measured via embedding-based cosine similarity Reimers & Gurevych (2019). This indicates that explicit reasoning is critical for query quality even under reduced model capacity. In contrast, Figure 4(b–c) shows that for Visit actions, the LLM without reasoning aligns more closely with the Oracle in both URL selection and extraction instruction, suggesting that deliberative reasoning is less essential for Visit, where pattern-based selection benefits more from model capacity.



Figure 3: Mean reasoning tokens before Search and Visit across models and benchmarks.

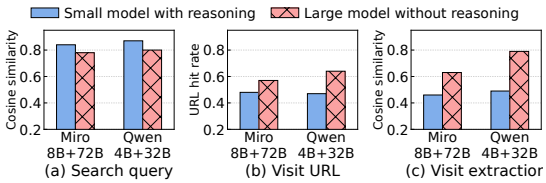

Figure 4: Oracle alignment: (a) query similarity, (b) URL hit rate, and (c) extraction similarity.

To assess end-to-end effects, we perform action-level interventions that selectively replace Search or Visit generation with lightweight methods while keeping the rest of the pipeline unchanged. We also include another variant of speculation combining reduced capacity and reduced reasoning (SLM without reasoning). Table 1 shows that assigning an SLM with reasoning to *Search* preserves overall accuracy, whereas using an LLM without reasoning degrades performance. Conversely, generating *Visit* with an LLM without reasoning yields the best results, while alternative choices lead to significant accuracy drops. Although speculative actions do not perfectly align with the Oracle (Figure 4), end-to-end accuracy remains high because of the tolerance of approximation in reasoning models, as long as appropriate inference pathways are chosen per action.

Key insight: Search as System 2, Visit as System 1. These results reveal a clear action-level dichotomy. Viewed through dual-process theory, Search exhibits System 2 behavior, requiring deliberative reasoning to translate underspecified research goals into effective queries. Visit aligns with System 1 behavior, where selection and extraction primarily rely on fast, pattern-based recognition encoded in model parameters. This distinction motivates action-aware speculation.

3.2 Verification Beyond Action Matching

Speculation alone is insufficient for reliable speedups; verification is essential to prevent error propagation. Most existing methods verify speculative actions via exact or approximate matching

with the base model output, but this has two limitations. First, action equivalence is hard to define. Exact matching is overly restrictive, as semantically equivalent Search queries may differ token-wise, causing unnecessary rejection, while approximate matching often requires additional modules (e.g., embedding models) and threshold tuning. Second, verification typically places the base model's full reasoning trace on the critical path, limiting latency gains. More aggressive designs Guan et al. (2025) allow multi-step speculation with parallel verification, but failures in the middle require rolling back all following speculative steps, wasting computation. Achieving better accuracy-efficiency trade-offs therefore requires verification strategies beyond simple action matching.

4 Theoretical Analysis

This section explains the empirical observations in Section 3. We analyze why different actions exhibit distinct sensitivities to explicit reasoning, leading to different optimal speculation strategies.

Our central claim is that Search and Visit actions differ in intrinsic decision uncertainty. We formalize this intuition via an entropy-based analysis of action policies and show that Search actions benefit significantly more from reasoning-induced uncertainty reduction than Visit actions, explaining the empirically observed System 2 versus System 1 dichotomy.

4.1 Preliminaries: Action Policies and Entropy

At each step, an agent observes a state s from the accumulated context and then samples an action a from a policy $\pi(\cdot \mid s)$. We denote the action spaces corresponding to Search and Visit by $\mathcal{A}_{\text{SEARCH}}$ and $\mathcal{A}_{\text{VISIT}}$, respectively.

A natural measure of decision uncertainty is the conditional entropy of the action policy:

$$H(\pi(\cdot \mid s)) \;=\; -\sum_{a \in \mathcal{A}} \pi(a \mid s) \log \pi(a \mid s). \tag{1}$$

Lower entropy indicates a more confident and concentrated decision, whereas higher entropy reflects ambiguity among many plausible actions.

In deep research agents, however, actions are expressed as open-ended language strings (e.g., search queries or extraction instructions), rendering exact computation of (1) intractable. We therefore adopt a token-level proxy based on the negative log-likelihood of the realized action. For an action token sequence $a = (t_1, \ldots, t_n)$, we define the mean token-level entropy proxy:

$$\bar{H}(a \mid s) \;=\; \frac{1}{n} \sum_{i=1}^{n} \left(-\log p(t_i \mid s, t_{<i})\right), \tag{2}$$

where smaller $\bar{H}(a \mid s)$ (higher mean token log probability) indicates lower decision uncertainty.

4.2 Intrinsic Entropy Gap Between Search and Visit

We first examine the baseline uncertainty of different actions when generated *without* explicit reasoning. Intuitively, Search actions map a broad and ambiguous intent to a concrete query, for which many formulations may be reasonable. In contrast, Visit operates on retrieved candidates and localized content, significantly constraining the decision space.

This intuition is reflected in the following inequality:

$$\mathbb{E}[\bar{H}(a \mid s) \mid a \in \mathcal{A}_{\text{SEARCH}}] \;>\; \mathbb{E}[\bar{H}(a \mid s) \mid a \in \mathcal{A}_{\text{VISIT}}]. \tag{3}$$

That is, under identical inference settings, Search actions exhibit higher average uncertainty than Visit actions. Figure 5 visualizes this gap. Each boxplot shows the distribution of mean token log probabilities for Search and Visit actions. When generated without reasoning, Search consistently exhibits lower log probabilities (higher $\bar{H}$) than Visit, suggesting less confident action distribution.

4.3 Why Reasoning Helps: Entropy Reduction via Intermediate Structure

We model explicit reasoning as the introduction of an intermediate latent variable z, corresponding to a reasoning trace that refines the decision context before the final action is generated. This transforms

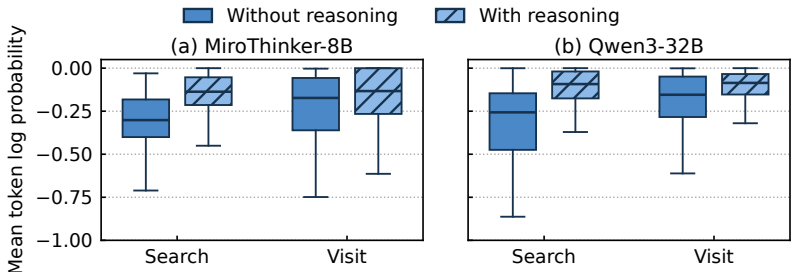

Figure 5: Distributions of mean action-token log probabilities for MiroThinker-8B and Qwen3-32B. Without reasoning (dark blue), Search has lower log probabilities and higher uncertainty than Visit. Reasoning (hatched light blue) increases log probabilities for both actions, with larger gains for Search, indicating a greater reduction in uncertainty.

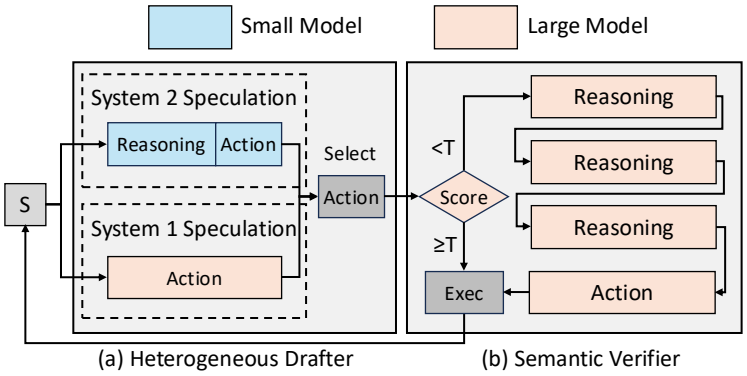

Figure 6: Overview of DualSpec.

the direct mapping $\pi(a \mid s)$ into a two-stage generation process:

$$\pi(a \mid s) \;=\; \sum_{z} \pi(z \mid s)\, \pi(a \mid s, z), \tag{4}$$

where the final action is conditioned not only on the original state s, but also on the intermediate reasoning state z. By a standard information-theoretic property Cover (1999), conditioning cannot increase entropy. Formally,

$$\mathbb{E}_{z\sim\pi(\cdot|s)}[H(\pi(\cdot \mid s, z))] \;\leq\; H(\pi(\cdot \mid s)). \tag{5}$$

Thus, a reasoning trace reduces action uncertainty and increases the realized action's likelihood.

As shown in Figure 5, incorporating reasoning consistently increases log probabilities, corresponding to a reduction in action-level uncertainty. This reduction is most pronounced when the target decision relies on non-local associations that are not directly specified in the immediate input Prystawski et al. (2023). Therefore, for Search, reasoning decomposes a global, underspecified mapping into a sequence of more localized sub-decisions, yielding a large reduction in uncertainty. In contrast, Visit actions already exhibit low baseline entropy due to strong grounding in retrieved content. Therefore, conditioning on a reasoning trace yields only a marginal additional reduction in uncertainty.

Taken together, this analysis explains why Search aligns more closely with *System 2* behavior, while Visit is closer to *System 1* behavior in deep research agents.

5 DualSpec Design

5.1 Overview

We propose DualSpec, a dual-process speculative framework for deep research agents. DualSpec allocates inference resources *heterogeneously* across actions for high speculation accuracy, while preserving end-to-end performance through lightweight, step-wise semantic verification.

As illustrated in Figure 6, DualSpec follows a draft–verify workflow. At each decision step, DualSpec generates two candidate actions in parallel: (i) a *System 2* draft produced by a small model with explicit reasoning, and (ii) a *System 1* draft via a large model skipping reasoning. The framework then selects a provisional draft based on the action type and the reasoning footprint from the small-model output. Finally, the selected draft is evaluated by a semantic verifier using a full-capacity base model. Drafts judged to be semantically consistent with the current reasoning trajectory are accepted and executed directly; otherwise, DualSpec falls back to full-capacity reasoning to regenerate the action.

5.2 Heterogeneous Draft

DualSpec implements heterogeneous drafting by producing two candidate actions at each step and adaptively selecting the one that best matches the inference demand of the current decision. Formally, given the current state s_t, we generate a System 2 draft (z_s, a_s) using SLM and a System 1 draft a_l using LLM. The key question is how to select the final drafted action while retaining reasoning information that is valuable for long-horizon planning.

Action-aware selection. We use the action type predicted by the small-model draft as the primary routing signal. If the small model generates a Search action, we retain a_s as the draft action, as Search typically benefits from explicit reasoning and can be reliably handled by a smaller model when paired with a reasoning trace. If the small model proposes a Visit action, we instead select the large-model draft a_l in most cases, since Visit actions rely more heavily on the large model's parametric capacity to make direct decisions on concrete inputs.

Preserving long-horizon reasoning. An important exception arises when the small-model draft produces a long reasoning trace before emitting its action. Empirically, such extended reasoning often contains global analysis or intermediate summaries that remain useful beyond the current step, regardless of whether the final action is Search or Visit. To preserve this information, when the length of the small-model reasoning exceeds a threshold τ_{think}, we choose the full draft (z_s, a_s) even if the action type is Visit. This mechanism ensures that high-level reasoning is not discarded when it may benefit subsequent decisions.

5.3 Semantic Verification

To maintain end-to-end accuracy, DualSpec performs lightweight semantic verification at every step. Instead of enforcing action-level matching, the verifier assesses whether the drafted reasoning and action are likely to make meaningful progress. This design is motivated by the observation that intermediate agent decisions are often tolerant to approximation as illustrated in table 1. Moreover, this approach speeds up verification by avoiding base-model reasoning delays.

Given the current state s_t and a draft consisting of an optional reasoning trace z_t and a candidate action a_t, we query the large model as a critic and ask it to answer `Yes` or `No`. The prompt instructs the critic to jointly assess (i) whether the reasoning is coherent (if present) and (ii) whether the proposed action is useful for making progress. The exact prompt template is provided in Appendix A.1.

Although the critic produces a discrete verdict, a *continuous* signal is necessary to trade off speed and reliability. We therefore convert the critic's output distribution into a real-valued confidence score.

Let $p_{\text{acc}}(s_t, z_t, a_t)$ and $p_{\text{rej}}(s_t, z_t, a_t)$ denote the critic's probabilities of answering `Yes` and `No`, respectively. We define the verification score as the log-probability margin:

$$\text{score}(s_t, z_t, a_t) = \log p_{\text{acc}}(s_t, z_t, a_t) - \log p_{\text{rej}}(s_t, z_t, a_t), \tag{6}$$

which gives the acceptance log-odds, a stable, monotonic measure of verifier confidence.

We accept the draft if its score exceeds a threshold τ:

$$\textsc{Accept}(z_t, a_t) \quad \text{if} \quad \text{score}(s_t, z_t, a_t) \geq \tau, \tag{7}$$

and otherwise trigger fallback. Fallback regenerates the step using the full-capacity model with explicit reasoning and continues execution with the regenerated action. This verification-and-fallback procedure follows a standard speculative pattern: propose a fast approximate step, validate it with a stronger critic, and only pay the cost of full reasoning when the draft is unlikely to be reliable.

Since the score scale depends on the critic model, we select τ offline on a held-out development set. We sweep candidate thresholds and choose a fixed τ that preserves end-to-end accuracy while maximizing the acceptance rate, and keep it fixed at runtime. This allows DualSpec to allocate expensive full-capacity reasoning only to the minority of uncertain steps, improving overall time-to-solution. Appendix A.2 provides further evidence of the verifier score's effectiveness.

6 EXPERIMENTS

6.1 EXPERIMENTAL SETUP

Models. We evaluate DualSpec under three dual-model configurations: MiroThinker-v1.0-72B + MiroThinker-v1.0-8B, MiroThinker-v1.0-72B + MiroThinker-v1.0-30B-A3B, and Qwen3-32B + Qwen3-4B Team et al. (2025); Yang et al. (2025). All models are deployed single-tenant with one NVIDIA A100 GPU per model and batch_size=4 for inference. The MiroThinker Models were quantized to 4-bit for inference, while the Qwen Models use native FP8 quantization, to avoid out-of-memory in GPUs.

Datasets. We evaluate on three deep-research benchmarks: GAIA-Text-103 Wu et al. (2025), XBench-DeepSearch Chen et al. (2025) and Seal-0 Pham et al. (2025).

Frameworks. We build our agent on the MiroMind deep-research framework. Tool invocation follows the MCP (Model Context Protocol) interface to standardize tool signatures and I/O, ensuring consistent argument formatting and result parsing across models and datasets. Within this setup, Search calls are executed via the Bing API, and Visit operations (page fetching and readable content extraction) are served by Jina, providing a fixed backend for web querying and page processing throughout all experiments. The models are serving under SGLangZheng et al. (2024).

Baselines. We compare with two speculative agent frameworks: DSP Guan et al. (2025) and SPAgent Huang et al. (2025b). DSP targets planning tasks and accepts a draft only if it matches the base action (minimum edit distance), while SPAgent is tailored to web search, skipping verification early and enforcing strict action matching later. In contrast to their uniform drafting and action-alignment verification, DualSpec uses heterogeneous drafting for System 2/System 1 actions and semantic verification that accepts trajectory-consistent drafts without exact action equivalence.

Verifier threshold. We tune the verifier threshold τ on a held-out split of GAIA, targeting an intervention rate of $\sim$20%, and reuse the same τ in our experiments.

6.2 MAIN RESULTS

Figure 7 reports end-to-end *latency* vs. *pass@1* across three model pairs and three deep-research benchmarks. Overall, DualSpec attains **1.33–3.28**$\times$ speedup over the base model, averaging $\sim$**2**$\times$, while maintaining comparable pass@1. Across datasets and model combinations, the points for DualSpec consistently move left (lower latency) with negligible accuracy degradation, indicating a better accuracy–latency operating point than uniform speculative baselines.

By analyzing each model pair, we observe 1.8$\times$ acceleration on MiroThinker-72B + 8B, 2.6$\times$ on MiroThinker-72B + 30B, and 1.5$\times$ on Qwen3-32B + 4B. The larger gain with the 30B configuration arises from its MoE design that each forward activates roughly *3B* parameters; at the same time, its stronger base capability reduces the number of base model interventions, further reducing end-to-end time. Consequently, using MiroThinker 30B-A3B as the draft model delivers the highest overall speedup among the evaluated pairs.

For agents accessed through token-metered APIs, token cost directly reflects the economic benefit of inference efficiency, complementing latency measurements that depend on hardware and service conditions. Under fixed reference pricing, DualSpec reduces estimated per-query token cost by **49.5–66.6%** relative to the base model by shifting generation to cheaper models, showing that its latency gains also translate into lower estimated API expenditure as illustrated in Appendix B.

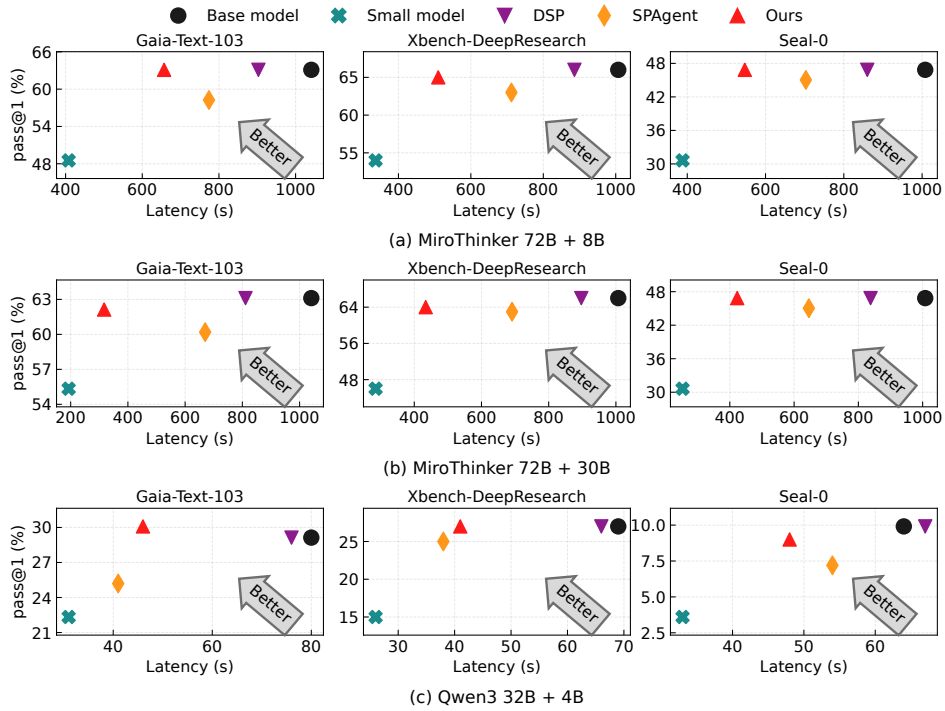

Figure 7: Accuracy (pass@1) and latency across model pairs and benchmarks. DualSpec achieves **1.33–3.28**× speedup (about 2× on average) over the base model at comparable accuracy, with a better accuracy–latency trade-off than DSP and SPAgent.

6.3 Speculation Ablation

Table 2: Pass@1 accuracy (%) / latency (s). Miro: 72B+8B; Qwen: 32B+4B. LLM-direct omits reasoning; SLM-CoT uses a small reasoning model.

Pair / dataset	Base	LLM-direct	SLM-CoT	DualSpec
Miro / GAIA	63.1 / 1041	59.2 / 575	56.3 / 651	63.1 / 605
Miro / XBench	66 / 1007	65 / 492	65 / 501	66 / 480
Qwen / GAIA	29.1 / 80	27.1 / 67	25.2 / 32	30.1 / 46
Qwen / XBench	27 / 69	25 / 46	26 / 49	27 / 41

Table 2 shows that heterogeneous drafting preserves base-model accuracy while lowering latency across both model families and datasets. Appendix C relates accuracy to intervention rate.

7 Conclusion

We introduce DualSpec, an efficient framework that accelerates deep research agents through heterogeneous action speculation. Our key insight is that actions exhibit different uncertainty levels: Search often requires deliberative reasoning, whereas Visit is typically more deterministic and can be executed without reasoning. Exploiting this asymmetry, DualSpec integrates action-specific draft policies with semantic verification to enable reliable speculative execution while removing large-model reasoning from the critical path. Experiments across multiple benchmarks show that DualSpec significantly reduces latency while maintaining strong task success rates, highlighting the importance of action-aware speculation for scalable agentic systems.

AI USE STATEMENT

Generative AI tools assisted with creating data visualizations, writing scripts for batch software runs, polishing the manuscript's language, and converting the LaTeX source to the ICLR 2027 format. The authors are responsible for the final content. The authors checked AI-assisted figures against the data, reviewed and tested the batch-run scripts, and reviewed all AI-polished prose.

REPRODUCIBILITY STATEMENT

Our implementation, experiment launchers, and setup instructions are available in the anonymous repository at `https://anonymous.4open.science/r/dualspec`. Section 6.1 describes the model pairs, quantization, hardware, benchmarks, tool backends, and verifier threshold selection. Appendix A.1 provides the verifier prompt, and Appendix B details the token-cost estimation procedure. The repository documents model endpoints, external service credentials, benchmark data formats, and evaluation commands. Reproduction requires separately obtained model checkpoints, benchmark data, and access to the search and webpage-reading services. Results may vary with live web content and service latency.

REFERENCES

Shelly Chaiken and Yaacov Trope. *Dual-process theories in social psychology*. Guilford Press, 1999.

Kaiyuan Chen, Yixin Ren, Yang Liu, Xiaobo Hu, Haotong Tian, Tianbao Xie, Fangfu Liu, Haoye Zhang, Hongzhang Liu, Yuan Gong, et al. xbench: Tracking agents productivity scaling with profession-aligned real-world evaluations. *arXiv preprint arXiv:2506.13651*, 2025.

Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

Yilin Guan, Qingfeng Lan, Sun Fei, Dujian Ding, Devang Acharya, Chi Wang, William Yang Wang, and Wenyue Hua. Dynamic speculative agent planning. *arXiv preprint arXiv:2509.01920*, 2025.

Wenyue Hua, Mengting Wan, Shashank Vadrevu, Ryan Nadel, Yongfeng Zhang, and Chi Wang. Interactive speculative planning: Enhance agent efficiency through co-design of system and user interface. *arXiv preprint arXiv:2410.00079*, 2024.

Yuxuan Huang, Yihang Chen, Haozheng Zhang, Kang Li, Huichi Zhou, Meng Fang, Linyi Yang, Xiaoguang Li, Lifeng Shang, Songcen Xu, et al. Deep research agents: A systematic examination and roadmap. *arXiv preprint arXiv:2506.18096*, 2025a.

Zixiao Huang, Wen Zeng, Tianyu Fu, Tengxuan Liu, Yizhou Sun, Ke Hong, Xinhao Yang, Chengchun Liu, Yan Li, Quanlu Zhang, et al. Reducing latency of llm search agent via speculation-based algorithm-system co-design. *arXiv preprint arXiv:2511.20048*, 2025b.

Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W Mahoney, Kurt Keutzer, and Amir Gholami. An llm compiler for parallel function calling. *arXiv preprint arXiv:2312.04511*, 2023.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pp. 19274–19286. PMLR, 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.

OpenAI. Introducing deep research, 2025. `https://openai.com/index/introducing-deep-research/`.

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.

Thinh Pham, Nguyen Nguyen, Pratibha Zunjare, Weiyuan Chen, Yu-Min Tseng, and Tu Vu. Sealqa: Raising the bar for reasoning in search-augmented language models. *arXiv preprint arXiv:2506.01062*, 2025.

Ben Prystawski, Michael Li, and Noah Goodman. Why think step by step? reasoning emerges from the locality of experience. *Advances in Neural Information Processing Systems*, 36:70926–70947, 2023.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 11 2019.

Swarnadeep Saha, Archiki Prasad, Justin Chih-Yao Chen, Peter Hase, Elias Stengel-Eskin, and Mohit Bansal. System-1. x: Learning to balance fast and slow planning with language models. *arXiv preprint arXiv:2407.14414*, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551, 2023.

Ning Shang, Yifei Liu, Yi Zhu, Li Lyna Zhang, Weijiang Xu, Xinyu Guan, Buze Zhang, Bingcheng Dong, Xudong Zhou, Bowen Zhang, et al. rstar2-agent: Agentic reasoning technical report. *arXiv preprint arXiv:2508.20722*, 2025.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652, 2023.

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

Zhihai Wang, Jie Wang, Jilai Pan, Xilin Xia, Huiling Zhen, Mingxuan Yuan, Jianye Hao, and Feng Wu. Accelerating large language model reasoning via speculative search. *arXiv preprint arXiv:2505.02865*, 2025.

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

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2023.

Naimeng Ye, Arnav Ahuja, Georgios Liargkovas, Yunan Lu, Kostis Kaffes, and Tianyi Peng. Speculative actions: A lossless framework for faster agentic systems. *arXiv preprint arXiv:2510.04371*, 2025.

Enhao Zhang, Erkang Zhu, Gagan Bansal, Adam Fourney, Hussein Mozannar, and Jack Gerrits. Optimizing sequential multi-step tasks with parallel llm agents. *arXiv preprint arXiv:2507.08944*, 2025a.

Jieyu Zhang, Ranjay Krishna, Ahmed H Awadallah, and Chi Wang. Ecoassistant: Using llm assistant more affordably and accurately. *arXiv preprint arXiv:2310.03046*, 2023.

Wentao Zhang, Ce Cui, Yilei Zhao, Yang Liu, and Bo An. Agentorchestra: A hierarchical multi-agent framework for general-purpose task solving. *arXiv preprint arXiv:2506.12508*, 2025b.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Livia Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Sglang: Efficient execution of structured language model programs. *Advances in neural information processing systems*, 37: 62557–62583, 2024.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.

A Verifier Prompt and Additional Analysis

A.1 Verifier Prompt Template

Given the current state s_t and a draft output consisting of an optional reasoning trace z_t and a candidate action a_t, we query the large model as a critic to output a binary judgment (`Yes` or `No`). The critic is instructed to jointly assess (i) whether the trajectory is making new progress toward the user goal and (ii) whether the proposed action is grounded and useful. When the draft pathway skips explicit reasoning, we set $z_t = \emptyset$. The exact prompt used for the critic is shown below.

> **[SYSTEM: TRAJECTORY AUDIT]**
> Review the recent steps (context). Is the agent making **NEW PROGRESS**?
> **REJECT ("No") if:**
>
> 1. **Stagnation**: Repeating similar queries or visiting same URLs (Looping).
> 2. **Ungrounded Answer**: The Final Answer is NOT supported by the *retrieved search results*.
> 3. **Lazy/Drift**: Queries are nested, vague, or irrelevant to User's Goal.
>
> Verdict: Is the trajectory HEALTHY and PROGRESSING?
> Answer only "Yes" or "No".

A.2 Verifier Score Distributions

To further validate that the verifier score provides a meaningful signal, we analyze the score distributions on complete trajectories. Specifically, we apply two full-size critics, MiroThinker-72B and Qwen3-32B, to evaluate step-level draft outputs produced by MiroThinker-8B and Qwen3-4B, respectively. We report two statistics per trajectory: the mean score across all steps (`Mean`) and the 25th percentile score (`p25`), which emphasizes low-confidence segments.

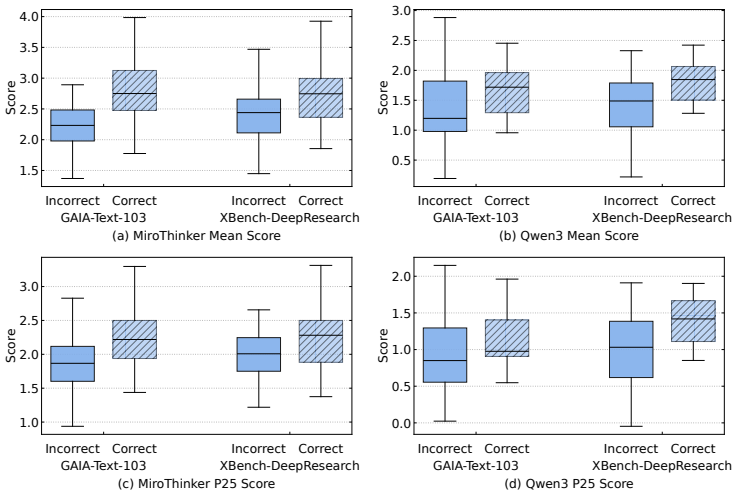

Figure 8: Verifier score distributions on GAIA and XBench-DeepResearch. We report trajectory-level aggregated verifier scores for (a) MiroThinker mean score, (b) Qwen mean score, (c) MiroThinker p25 score, and (d) Qwen p25 score. Correct trajectories consistently receive higher scores than incorrect ones, suggesting that the verifier provides a useful reliability signal.

Figure 8 shows the score distributions on GAIA and XBench-DeepResearch, grouped by whether the final answer is correct. Across both datasets and both model families, correct trajectories consistently exhibit higher verifier scores than incorrect ones under both `Mean` and `p25`. This indicates that the verifier score correlates with end-to-end task success, supporting its use as a lightweight reliability signal for controlling speculative execution.

B Token Usage and Estimated Cost

Token usage complements latency when evaluating agents accessed through metered APIs. We compare the base model, SPAgent, and DualSpec on GAIA using average token counts per query, including input, output, and verification tokens. Using Aliyun pricing as a reference,[1] we adopt fixed rates of 12, 8, 3, 2, and 1.2 RMB per million tokens for the 72B, 32B, 30B-A3B, 8B, and 4B models, respectively. We apply these parameter-scale rates uniformly to the reported token counts as a common cost proxy across model families. Given average token counts N_{LLM} and N_{SLM} and their respective rates p_{LLM} and p_{SLM}, the estimated cost per query is

$$C_{\mathrm{RMB}} = \frac{N_{\mathrm{LLM}} p_{\mathrm{LLM}} + N_{\mathrm{SLM}} p_{\mathrm{SLM}}}{10^6}. \tag{8}$$

This is a standardized estimate rather than an actual API bill: it does not separately model input/output pricing, caching discounts, or external tool fees.

Table 3: Average token usage and estimated cost per query on GAIA. Counts include all model tokens and are in thousands (k). The lowest estimated cost within each model pair is bold.

Model pair	Method	LLM (k)	SLM (k)	Cost (RMB)
MiroThinker 72B + 8B	Base model	10.0	0.0	0.12000
	SPAgent	3.7	10.3	0.06500
	DualSpec	2.0	15.0	**0.05400**
MiroThinker 72B + 30B-A3B	Base model	10.0	0.0	0.12000
	SPAgent	2.0	10.2	**0.05460**
	DualSpec	1.8	13.0	0.06060
Qwen3 32B + 4B	Base model	2.8	0.0	0.02240
	SPAgent	0.7	2.9	0.00908
	DualSpec	0.5	2.9	**0.00748**

Under this pricing schedule, DualSpec reduces estimated cost relative to the base model by 55.0%, 49.5%, and 66.6% for the 72B+8B, 72B+30B-A3B, and 32B+4B pairs, respectively. Although total token usage increases, shifting work from the more expensive LLM to the cheaper SLM lowers the estimated cost. Compared with SPAgent, DualSpec costs 16.9% less for 72B+8B and 17.6% less for 32B+4B. For 72B+30B-A3B, however, its greater SLM usage outweighs the LLM savings, yielding an estimated cost of 0.06060 RMB versus 0.05460 RMB for SPAgent (11.0% higher). Thus, cost savings depend on both token allocation and the relative prices of the paired models.

C Intervention Rate Ablation

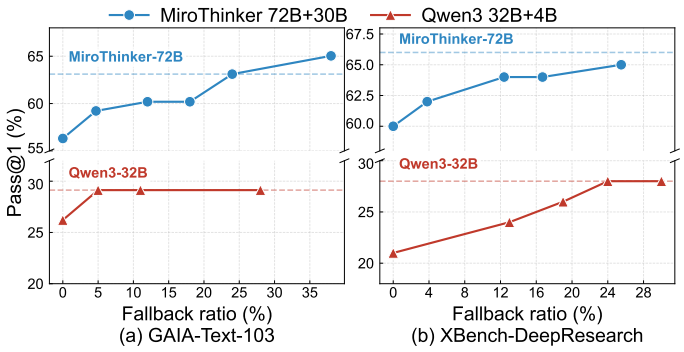

Figure 9: Accuracy (pass@1) versus reasoning intervention rate with a fixed drafting policy.

[1] https://help.aliyun.com/zh/model-studio/model-pricing. Published rates distinguish input and output tokens, deployment regions, and reasoning modes.

We study how accuracy changes with the frequency of the large model intervention. With a fixed speculator, we vary the verifier threshold, which indirectly controls the intervention rate. Accuracy increases as the intervention rate rises and then saturates. In practice, we observe that an intervention rate of about **20% to 30%** already reaches accuracy comparable to the base model, while retaining most of the latency benefit of heterogeneous speculation.

This trend is consistent across model families and datasets, with minor shifts in the saturation point. While tighter thresholds further improve accuracy, the returns diminish once the rate reaches the low-to-mid twenties. We therefore tune the threshold to target an intervention rate near 20%, recovering near-base accuracy without sacrificing the efficiency benefits of sparse large-model reasoning.