# AgentSwing: Adaptive Parallel Context Management Routing for Long-Horizon Web Agents

Anonymous ACL submission


## Abstract

As large language models (LLMs) evolve into autonomous agents for long-horizon information-seeking, managing finite context capacity has become a critical bottleneck. Existing context management methods typically commit to a single fixed strategy throughout the entire trajectory. Such static designs may work well in some states, but they cannot adapt as the usefulness and reliability of the accumulated context evolve during long-horizon search. To formalize this challenge, we introduce a probabilistic framework that characterizes long-horizon success through two complementary dimensions: search efficiency and terminal precision. Building on this perspective, we propose **AgentSwing**, a state-aware adaptive parallel context management routing framework. At each trigger point, AgentSwing expands multiple context-managed branches in parallel and uses lookahead routing to select the most promising continuation. Experiments across diverse benchmarks and agent backbones show that AgentSwing consistently outperforms strong static context management methods, often matching or exceeding their performance with substantially fewer interaction turns while also improving the ultimate performance ceiling of long-horizon web agents. Beyond the empirical gains, the proposed probabilistic framework provides a principled lens for analyzing and designing future context management strategies for long-horizon agents.


## 1 Introduction

As large language models (LLMs) evolve from single-turn question answering assistants into autonomous agents capable of web browsing and sequential tool use, long-horizon information-seeking has emerged as a critical testbed of their real-world capabilities (Wu et al., 2025b,a; Team, 2025a; Fang et al., 2025; Li et al., 2025c; Tao et al., 2025; Li et al., 2025b). In such tasks, solving a problem often requires tens or even hundreds of steps of searching, visiting, verifying, and backtracking before the agent can locate the key evidence and produce a final answer.

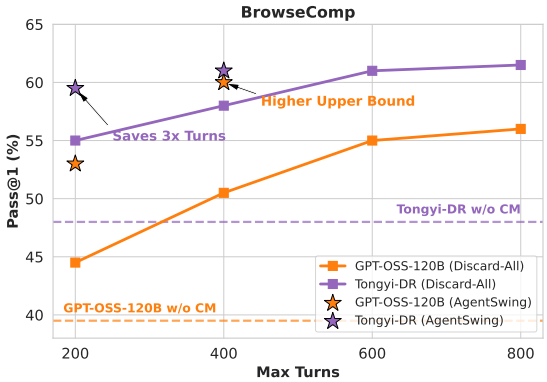


Figure 1: Performance on BrowseComp under different interaction budgets. Dashed lines denote the baselines without context management.

A central bottleneck in deep information-seeking is the tension between finite context capacity and the need for long-horizon exploration (Wei et al., 2025; Phan et al., 2025; Wong et al., 2025). Under a fixed context budget, an agent may exhaust its available workspace before completing a sufficiently informative search trajectory. As a result, **context management** has become a key mechanism shaping the performance ceiling of long-horizon agents (Anthropic, 2025a; Liu et al., 2025a). Recent frontier systems have shown that aggressive context management, such as *Discard-All*, can substantially improve long-horizon performance by enabling agents to discard the accumulated context to sustain more interaction turns (Liu et al., 2025a; Team et al., 2026; Zeng et al., 2026a). Most existing context management approaches rely on a single fixed strategy that is repeatedly applied throughout the entire trajectory. This design is inherently limited in long-horizon search, where the quality of the accumulated context evolves over time. Some trajectory states contain useful inter-

mediate structures that should be retained, while others are dominated by noise, drift, or unproductive search history and therefore call for more aggressive intervention.

To make this limitation explicit, we introduce the first probabilistic perspective for deep information-seeking agents that characterizes success through two complementary dimensions: **search efficiency** and **terminal precision**. Search efficiency measures whether an agent can reach a stopping point before exhausting available resources, while terminal precision measures whether the final answer is correct conditioned on reaching such a stopping point. This view reveals that commonly reported metrics such as Pass@1 or accuracy are not monolithic indicators in long-horizon settings. Instead, end-to-end success depends jointly on whether the agent can arrive at a terminal state with final answer and whether it can answer correctly once there.

Building on this perspective, we propose **AgentSwing**, an adaptive parallel context management routing framework for long-horizon web agents. Instead of committing to a single context management operation at every trigger point, AgentSwing expands multiple context-managed branches from the current trajectory state and uses a lookahead routing mechanism to select the most promising continuation. In this way, AgentSwing leverages the complementary strengths of heterogeneous context management strategies and moves beyond the efficiency-precision trade-off of static context management methods. Experiments on several challenging long-horizon benchmarks with diverse open-source backbones, including GPT-OSS-120B (OpenAI, 2025b), DeepSeek-v3.2 (Liu et al., 2025a), and Tongyi-DR-30B-A3B (Team, 2025b), show that AgentSwing consistently outperforms strong static methods. Under constrained interaction budgets, it reaches or exceeds the performance of static strategies that require substantially more interaction turns, while also achieving a higher ultimate performance ceiling (see Figure 1). It pushes DeepSeek-v3.2 to 71.3 on BrowseComp-ZH and 44.4 on HLE, surpassing several proprietary foundation models, and establishes leading performance for Tongyi-DR-30B-A3B among information-seeking agents of comparable scale. Our core contributions are as follows:

- We introduce the first probabilistic framework for long-horizon web agents that characterizes context management through two complementary dimensions, search efficiency $\eta$ and terminal precision $\rho$, providing a unified lens for understanding the behavior of different strategies.
- We propose AgentSwing, a state-aware adaptive context management framework that dynamically switches among candidate strategies according to the quality of the current trajectory and continuations, thereby balancing search efficiency and terminal reliability and improving overall long-horizon agent performance.
- Extensive experiments across multiple long-horizon benchmarks and model backbones demonstrate the effectiveness and generalization of AgentSwing, and provide a fine-grained analysis of how different context management strategies behave and why adaptive routing works.

## 2 A Complementary Probabilistic View of Long-Horizon Web Agents

We begin with a probabilistic characterization of long-horizon web agents under resource-constrained execution. In deep information-seeking, end-to-end success cannot be understood solely by final answer accuracy. Before producing a correct answer, the agent must first navigate a long interaction trajectory, accumulate sufficient evidence, and reach a stopping point before exhausting its available resources, such as context budget and maximum interaction turns. Accordingly, failures arise from two distinct sources: the agent may fail to reach a stopping point within the allowed resources, or it may terminate but produce an incorrect answer.

### 2.1 Two Perspectives on Success: Search Efficiency and Terminal Precision

We assume tasks $\tau$ are independently sampled from an underlying task distribution $\mathcal{T}$. For a task $\tau$, consider an agent executed under a test-time strategy $\pi$, where $\pi$ specifies the execution protocol, including context management, stopping rules, and resource constraints. Let $S^\pi$ denote the event that the agent reaches a stopping point and emits a final answer under strategy $\pi$, and let $C^\pi$ denote the event that this answer is correct. We define two task-level quantities:

$$\eta_\tau^\pi := P(S^\pi \mid \tau), \quad \rho_\tau^\pi := P(C^\pi \mid S^\pi, \tau). \tag{1}$$

Here, $\eta_\tau^\pi$ is the agent's *search efficiency*, i.e., the probability of reaching a stopping point before the

protocol terminates, and $\rho_\tau^\pi$ is its *terminal precision*, i.e., the probability that the answer is correct conditioned on reaching such a stopping point.

Task-level success then follows from the chain rule:

$$P(\text{Success}^\pi \mid \tau) = P(S^\pi \cap C^\pi \mid \tau) = \eta_\tau^\pi \rho_\tau^\pi. \quad (2)$$

Thus, success requires both reaching a terminal state and answering correctly once there. At the population level, we define

$$\eta^\pi := P(S^\pi) = \mathbb{E}_{\tau\sim\mathcal{T}}[\eta_\tau^\pi], \quad (3)$$

$$\rho^\pi := P(C^\pi \mid S^\pi) = \frac{P(C^\pi \cap S^\pi)}{P(S^\pi)} = \frac{\mathbb{E}_{\tau\sim\mathcal{T}}[\eta_\tau^\pi \rho_\tau^\pi]}{\mathbb{E}_{\tau\sim\mathcal{T}}[\eta_\tau^\pi]}. \quad (4)$$

Accordingly, the population-level success probability can be written as

$$Pass@1^\pi = P(\text{Success}^\pi) = P(S^\pi \cap C^\pi) = \eta^\pi \rho^\pi. \quad (5)$$

This decomposition shows that commonly used end-to-end metrics such as Pass@1 or accuracy should not be treated as monolithic indicators in long-horizon settings. Instead, they jointly reflect search efficiency and terminal precision.

In practice, suppose a benchmark contains $M$ tasks. For a fixed strategy $\pi$, let $N_{\text{finish}}^\pi$ denote the number of tasks on which the agent reaches a stopping point and emits a final answer, and let $N_{\text{correct}}^\pi$ denote the number of tasks on which the final answer is correct. Following Team et al. (2026); Zeng et al. (2026a), tasks that exhaust the allowed resources before producing a final answer are directly counted as failed. We estimate

$$\eta^\pi \approx \frac{N_{\text{finish}}^\pi}{M}, \qquad \rho^\pi \approx \frac{N_{\text{correct}}^\pi}{N_{\text{finish}}^\pi}, \quad (6)$$

with the corresponding empirical end-to-end success rate

$$\text{Pass@1}^\pi = \eta^\pi \rho^\pi \approx \frac{N_{\text{correct}}^\pi}{M}. \quad (7)$$

Since different strategies may finish on different task subsets, we additionally report *aligned terminal precision* for cross-strategy comparison. Let $N_{\text{aligned-finish}}$ be the number of tasks that finish under all compared strategies or settings, and let $N_{\text{aligned-correct}}^\pi$ be the number of these tasks answered correctly by strategy $\pi$. We compute

$$\rho_{\text{align}}^\pi \approx \frac{N_{\text{aligned-correct}}^\pi}{N_{\text{aligned-finish}}}. \quad (8)$$

By reporting terminal precision on the shared finished subset, this metric enables a fairer comparison across strategies or settings.

## 2.2 Discard-All vs. Baseline

We use *Discard-All* as a concrete case study to instantiate the framework above and explain why context management can outperform the standard *w/o context management* baseline.

Let $\pi = \text{std}$ denote the baseline without context management. Under this protocol, the agent continuously appends its interaction history and follows a single uninterrupted search trajectory. It therefore either reaches a stopping point and produces a final answer, or exhausts the maximum context length and is counted as failed. In contrast, *Discard-All* ($\pi = \text{DA}$) introduces a context-management trigger. Once the accumulated context exceeds a predefined threshold, the agent discards the full trajectory history and continues from the original user prompt only. As a result, the same task execution under *Discard-All* may contain multiple reset-based attempts. If the maximum turn budget is exhausted before a final answer is produced, the task is counted as failed.

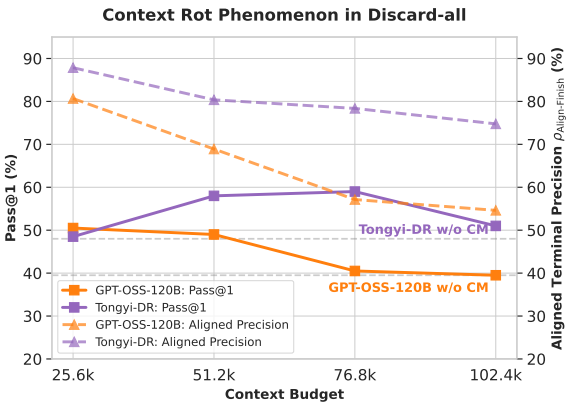


Figure 2: Performance on BrowseComp under *Discard-All* with different context budgets.

We next study how the trigger threshold affects the performance of *Discard-All*, while also using it to understand the difference between *Discard-All* and the baseline. We vary the trigger ratio while fixing the maximum interaction turns to 400, so that the primary changing factor is the effective context budget per attempt. Figure 2 shows that, for both Tongyi-DR-30B-A3B and GPT-OSS-120B, aligned terminal precision consistently decreases as the context budget increases. This indicates that larger working contexts lead to more severe context rot at termination (Hsieh et al., 2024; Modarressi et al., 2025; Hong et al., 2025; Fang et al., 2026). Since the baseline corresponds to the largest context regime, it is also the least favorable for terminal precision. At the same time, an appropriate con-

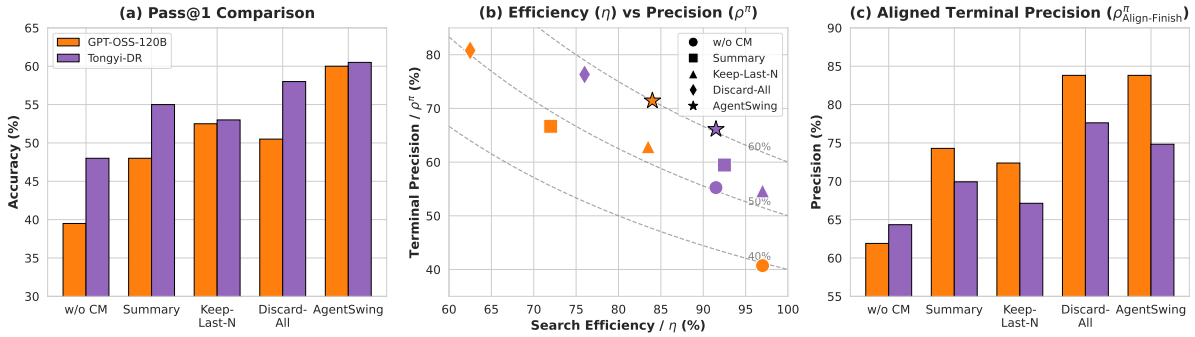


Figure 3: Comparison of context management strategies through the efficiency-precision lens. (a) Pass@1 on BrowseComp. (b) Search efficiency $\eta$ versus terminal precision $\rho$. (c) Aligned terminal precision, where *Align-Finish* refers to the common finished cases shared by different strategies within the same model.

text budget allows *Discard-All* to outperform the baseline in overall performance.

This phenomenon can be further interpreted through our efficiency-precision framework. In Figure 3b, the standard baseline typically has the lowest terminal precision, consistent with the trend in Figure 2, but also the highest search efficiency. In other words, it reaches stopping points on more tasks, yet the resulting terminal states are less reliable. By contrast, *Discard-All* usually has lower search efficiency $\eta$, because each reset-based attempt operates under a smaller effective context budget and is less likely to finish on its own. However, this efficiency loss can be alleviated by increasing the number of reset opportunities $N$. For a task $\tau$, let $S_i$ denote the event that the agent reaches a stopping point during the $i$-th reset-based attempt, and suppose at most $N$ such attempts are allowed. Then

$$\eta_\tau^{\mathrm{DA}} = P\Big(\bigcup_{i=1}^{N} S_i \,\Big|\, \tau\Big), \tag{9}$$

which, under a conditional independence approximation across reset-based segments, becomes

$$\eta_\tau^{\mathrm{DA}} = 1 - \prod_{i=1}^{N}\big(1-\eta_{\tau,i}^{\mathrm{DA}}\big) \approx 1 - \big(1-\eta_{\tau,\mathrm{single}}^{\mathrm{DA}}\big)^N. \tag{10}$$

Although each individual attempt is less likely to finish than the baseline, increasing $N$ provides more chances to reach a stopping point. Combined with the higher precision of smaller contexts, this allows *Discard-All* to outperform the baseline.

### 2.3 Static Context Management Strategies in the Efficiency-Precision Plane

The same perspective extends naturally beyond *Discard-All* to other context management strategies. Figure 3 compares *Summary*, *Discard-All*, *Keep-Last-N*, and *AgentSwing* under maximum interaction budget of 400 turns. As shown in Figure 3a, all context management strategies outperform the baseline in Pass@1, but through different efficiency-precision trade-offs.

Figure 3b shows that static strategies occupy different operating points in the efficiency-precision plane, forming an empirical trade-off frontier. As discussed above, the standard baseline is high-efficiency but low-precision, whereas *Discard-All* lies near the opposite end. *Summary* and *Keep-Last-N* fall between these extremes, improving search efficiency over *Discard-All* but not matching its terminal precision; see also Figure 3c. In contrast, *AgentSwing* moves to a more favorable region of the plane by adaptively routing multiple strategies, leading to the strongest overall performance.

## 3 AgentSwing

AgentSwing consists of two components: (1) *Parallel Context Management* and (2) *Lookahead Routing*, as illustrated in Figure 4. We consider the standard deep information-seeking setting, where an agent starts from a user prompt $q$ and interacts with the environment through repeated (`<thinking>`, `<tool call>`, `<tool response>`) turns. When the current context length exceeds a predefined fraction $r$ of the model's maximum context length, the framework activates context management over the accumulated trajectory.

**(1) Parallel Context Management.** At each trigger point, AgentSwing applies multiple candidate context management strategies to the same raw context in parallel, producing a set of alternative managed contexts. In this work, we consider three representative strategies:

- **Keep-Last-N:** Preserves only the latest $N$ interaction turns, i.e., the last $N$ (`<thinking>`,

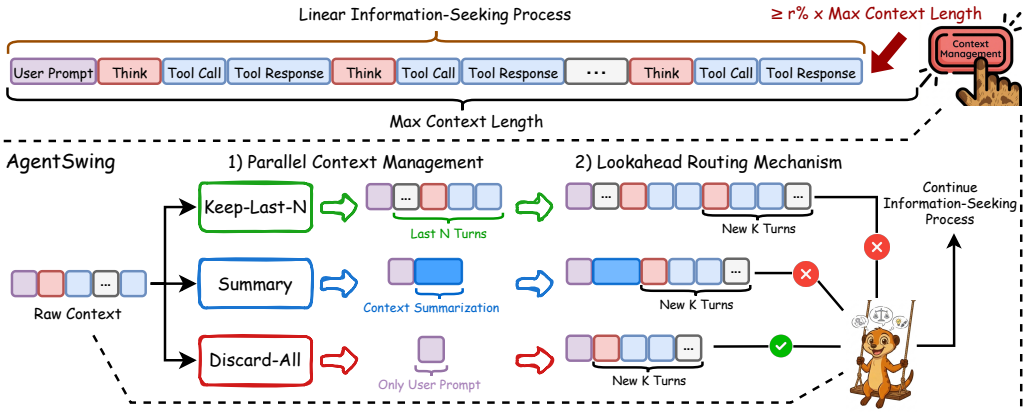


Figure 4: Overview of AgentSwing. AgentSwing triggers context management once the accumulated context exceeds a predefined threshold, executes multiple candidate strategies in parallel, extends each branch for $K$ new turns, and dynamically routes to the most promising continuation.

<tool call>, <tool response>) tuples, and discards earlier history (Liu et al., 2025a; Zeng et al., 2026a).

- **Summary:** Compresses the accumulated trajectory into a summarized text and keeps the context in the form of the original user prompt together with the summary, i.e., $(q, \mathrm{Sum})$ (Liu et al., 2025a; Anthropic, 2025a).
- **Discard-All:** Discards the entire accumulated interaction history and keeps only the original user prompt $q$ (Liu et al., 2025a; Team et al., 2026; Zeng et al., 2026a).

Applying these strategies in parallel can further yield multiple candidate continuations from the same trajectory state, each corresponding to a different way of managing the accumulated context.

**(2) Lookahead Routing Mechanism.** After parallel context management, AgentSwing does not immediately select a branch. Instead, it performs short-horizon lookahead for each managed context. Concretely, each branch continues interacting with the environment for $K$ additional turns. After the lookahead phase, AgentSwing presents the candidate continuations together with the original raw context to the agent model, which then selects the most promising branch for subsequent exploration. The full routing prompt is provided in Appendix E. The remaining branches are discarded, and the selected continuation becomes the new main trajectory. This design allows branch selection to depend not only on the managed context itself, but also on its short-term downstream behavior under real environment feedback. In this way, AgentSwing differs from static context management methods, which repeatedly apply a single fixed strategy throughout the entire search process.

# 4 Experiments

## 4.1 Setup

**Benchmarks.** We evaluate AgentSwing on three challenging deep information-seeking benchmarks: BrowseComp (Wei et al., 2025), BrowseComp-ZH (Zhou et al., 2025), and Humanity's Last Exam (HLE) (Phan et al., 2025). These benchmarks jointly assess deep search and reasoning ability. For efficient evaluation, we use sampled subsets for the larger benchmarks: 200 randomly selected tasks from BrowseComp and 500 text-only tasks from HLE, following prior work (Li et al., 2025d; Nguyen et al., 2025). For BrowseComp-ZH, we use the full set of 289 tasks.

**Tools.** We adopt the standard tool configuration used by deep information-seeking agents (Wu et al., 2025a; Li et al., 2025b), with Search and Visit as the core tools. For HLE, following Chen et al. (2026), we further include Google Scholar and a Python Interpreter. More details are in Appendix A.

**Agent Models.** We use three open-source models with diverse parameter scales and strong tool-use capability for deep information-seeking tasks: GPT-OSS-120B (OpenAI, 2025b), DeepSeek-v3.2 (Liu et al., 2025a), and Tongyi-DeepResearch-30B-A3B (Tongyi-DR-30B-A3B) (Team, 2025b). All models are invoked under their official function-calling protocol. Unless otherwise specified, we use the same agent model for both stages in AgentSwing.

**Baselines.** In addition to the standard baseline without context management (*w/o CM*), we compare AgentSwing with several representative static context management strategies introduced in Section 3, including *Discard-All*, *Keep-Last-N* ($N = 5$), and *Summary*. The summarization step is always per-

| Model / Framework | Context Management | BrowseComp | BrowseComp-ZH | HLE |
|---|---|---|---|---|
| ***Foundation Models with Tools*** | | | | |
| Claude-4.5-Opus (Anthropic, 2025b) | w/o CM | 37.0‡ | 62.4 | 43.4‡ |
| Gemini-3.0-Pro (DeepMind, 2025) | w/o CM | 37.8‡ | 66.8 | 45.8‡ |
| GPT-5.1 High (OpenAI, 2025c) | w/o CM | 50.8‡ | – | 42.7‡ |
| OpenAI-o3 (OpenAI, 2025d) | w/o CM | 49.7‡ | 58.1 | – |
| ***Deep Information-Seeking Agents*** | | | | |
| OpenAI DeepResearch (OpenAI, 2025a) | - | 51.5‡ | 42.9 | 26.6‡ |
| ASearcher-Web-32B (Gao et al., 2025) | - | 5.2‡ | 15.6 | 12.5‡ |
| DeepMiner-32B-RL (Tang et al., 2025) | - | 33.5‡ | 40.1 | – |
| IterResearch-30B-A3B (Chen et al., 2026) | - | 37.3‡ | 45.2 | 28.8‡ |
| AgentFold-30B-A3B (Ye et al., 2026) | - | 36.2‡ | 47.3 | – |
| AgentFounder-30B-A3B (Su et al., 2026) | - | 39.9‡ | 43.3 | 31.5‡ |
| MiroThinker-v1.5-30B-A3B (Team, 2026) | - | 56.1‡ | 66.8 | 31.0‡ |
| ***Open-Source Models with Context Management*** | | | | |
| GPT-OSS-120B | Baseline (w/o CM) | 39.5 | 28.4 | 33.2 |
| | Discard-All | 50.5 | 31.5 | 34.2 |
| | Keep-Last-N | 52.5 | 33.6 | 34.1 |
| | Summary | 48.0 | 30.8 | 34.4 |
| | **AgentSwing (Ours)** | **60.0** | **38.0** | **35.1** |
| DeepSeek-v3.2 | Baseline (w/o CM) | 51.4‡ / 43.5 | 65.0‡ / 61.6 | 40.8‡ / 40.2 |
| | Discard-All | 58.0 | 70.2 | 42.0 |
| | Keep-Last-N | 52.0 | 69.9 | 39.6 |
| | Summary | 48.5 | 69.2 | 43.5 |
| | **AgentSwing (Ours)** | **62.5** | **71.3** | **44.4** |
| Tongyi-DR-30B-A3B | Baseline (w/o CM) | 43.4‡ / 48.0 | 46.7‡ / 47.1 | 32.9‡ / 31.7 |
| | Discard-All | 58.0 | 53.9 | 32.7 |
| | Keep-Last-N | 53.0 | 50.1 | 32.2 |
| | Summary | 55.0 | 49.1 | 32.0 |
| | **AgentSwing (Ours)** | **60.5** | **56.7** | **33.1** |

Table 1: Overall performance on long-horizon agentic benchmarks. Scores marked with ‡ represent full-benchmark results, whereas unmarked scores correspond to our benchmark settings.

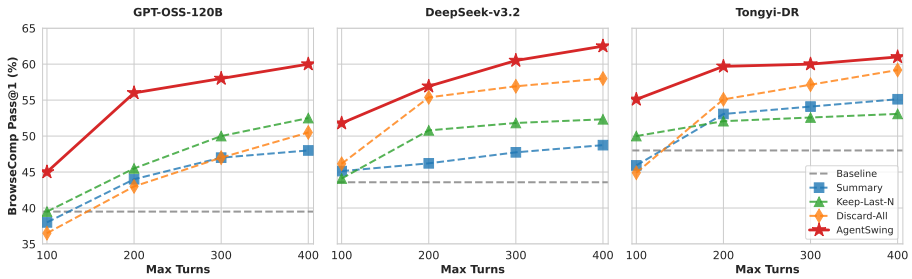


Figure 5: Performance of different context management strategies on BrowseComp over maximum interaction turns.

formed by GPT-OSS-120B as a fixed external module. More details about the evaluation metrics and parameters are in Appendix B.

### 4.2 Overall Performance

Table 1 shows that AgentSwing consistently achieves strong performance across all benchmarks and agent backbones, outperforming both the standard baseline and representative context management strategies. Notably, AgentSwing pushes DeepSeek-v3.2 to 71.3 on BrowseComp-ZH and 44.4 on HLE, surpassing several proprietary foundation models. It also establishes leading performance for Tongyi-DR-30B-A3B among deep information-seeking agents of comparable scale. These results show that adaptive context management is a strong and general test-time scaling mechanism for long-horizon web agents.

### 4.3 Analysis and Ablation

We next provide a fine-grained analysis of AgentSwing. We examine how different context management strategies scale with interaction budget, compare their behavior on aligned harder cases, ablate the lookahead routing mechanism, and present case studies. Further analyses of token efficiency, strategy transitions, and strategy combinations are deferred to Appendices C to F.

**Analysis of Context Management Strategies.** Figure 5 shows how different strategies scale with

| Model | Strategy | $N_{align}$ | $N_{finish}$ | $N_{correct}$ | $\eta$ (%) | $\rho$ (%) | Pass@1 (%) | $\overline{N_{turn}}$ |
|---|---|---|---|---|---|---|---|---|
| GPT-OSS-120B | Discard-All | 122 | 51 | 35 | 41.8 | **68.6** | 28.7 | 297.2 |
| | Summary | | 68 | 35 | 55.7 | 51.5 | 28.7 | 248.0 |
| | Keep-Last-N | | 91 | 43 | **74.6** | 47.3 | 35.2 | 205.4 |
| | AgentSwing | | 90 | 51 | 73.8 | 56.7 | **41.8** | **190.3** |
| DeepSeek-v3.2 | Discard-All | 73 | 40 | 24 | 54.8 | **60.0** | 32.9 | 268.3 |
| | Summary | | 72 | 22 | **98.6** | 30.6 | 30.1 | **132.2** |
| | Keep-Last-N | | 53 | 23 | 72.6 | 43.4 | 31.5 | 183.5 |
| | AgentSwing | | 68 | 26 | 93.2 | 38.2 | **35.6** | 151.9 |
| Tongyi-DR-30B-A3B | Discard-All | 45 | 11 | 9 | 24.4 | **81.8** | 20.0 | 340.8 |
| | Summary | | 35 | 9 | 77.8 | 25.7 | 20.0 | 215.7 |
| | Keep-Last-N | | 42 | 9 | **93.3** | 21.4 | 20.0 | **153.0** |
| | AgentSwing | | 34 | 14 | 75.6 | 41.2 | **31.1** | 203.6 |

Table 2: Performance comparison of different context management methods on aligned cases that trigger context management under all evaluated strategies ($\rho_{\text{Align-CM}}$).

| Model | Strategy | CM Triggered/Case | Tokens/Case | Tokens/CM |
|---|---|---|---|---|
| GPT-OSS -120B | Summary | 15.3 | 3988K | 261K |
| | Keep-Last-N | 10.2 | 3308K | 324K |
| | Discard-All | 17.8 | 4729K | 266K |
| | **AgentSwing** | 13.6 | 3450K | **254K** |
| DeepSeek -v3.2 | Summary | 4.3 | 3629K | 844K |
| | Keep-Last-N | 5.0 | 5571K | 1114K |
| | Discard-All | 8.1 | 7286K | 900K |
| | **AgentSwing** | 6.2 | 4826K | **778K** |
| Tongyi-DR -30B-A3B | Summary | 2.9 | 9354K | 3225K |
| | Keep-Last-N | 2.0 | 6227K | 3114K |
| | Discard-All | 8.5 | 10213K | 1202K |
| | **AgentSwing** | 4.9 | 6342K | **1294K** |

Table 3: Token-level cost on aligned CM-triggered cases. Tokens/Case = average tokens per case. Tokens/CM = average tokens per case / average CM triggers.

the maximum interaction budget on BrowseComp. Under small budgets, context management provides limited gains since the baseline retains strong single-attempt search efficiency. As the budget grows, all context management strategies surpass the baseline, consistent with the analysis in Section 2.3. AgentSwing outperforms the baseline even under limited budgets and maintains a consistent advantage across the full scaling curve.

To further isolate strategy behavior on harder cases, Table 2 reports results on the cases where context management is triggered under all strategies. *Keep-Last-N* and *Summary* achieve stronger search efficiency $\eta$, while *Discard-All* achieves the strongest terminal precision $\rho$. AgentSwing combines the strengths of both regimes, leading to the highest Pass@1 across all three models while maintaining turn counts efficiently.

Table 3 reports the token-level cost on these aligned CM-triggered cases. AgentSwing achieves moderate total tokens per case and the lowest average tokens per CM trigger across all three models, confirming that its gains are not obtained by paying a substantially higher compute cost.

**Ablation of the Lookahead Routing Mechanism.** Table 4 compares AgentSwing with two variants: *random* selection and *w/o Lookahead* (routing without rollout). Both consistently underperform AgentSwing, showing that the gains come from using short-horizon lookahead to evaluate downstream consequences before routing.

| Routing Mechanism | GPT-OSS-120B | Tongyi-DR-30B-A3B |
|---|---|---|
| random | 51.0 | 56.5 |
| w/o Lookahead | 50.0 | 57.0 |
| Lookahead ($k = 1$) | 52.5 | 58.0 |
| Lookahead ($k = 3$) | **60.0** | **60.5** |
| Lookahead ($k = 5$) | 55.0 | 59.0 |

Table 4: Ablation on lookahead strategy. We compare random selection, disabling lookahead, and different lookahead depths $k$.

We further vary the lookahead depth $k$, i.e., the number of newly generated turns per branch before routing. The results show that moderate lookahead is most effective. In particular, $k = 3$ generally provides the strongest performance across models. Compared with $k = 1$, it exposes richer future trajectory information, while larger lookahead such as $k = 5$ does not always improve performance further, since it may risk exceeding maximum length constraints of agent models.

## 4.4 Case Study

Figure 6 shows a case from DeepSeek-v3.2. When context management is triggered at Turn 23, the current history contains both substantial distractions arising from incorrect hypotheses ("Nipsey Hussle", "Lil Durk", and "Hit-Boy") and a newly surfaced local clue ("$tupid Young"). This mixed state makes static context management brittle. *Discard-All* resets to broad exploration; *Summary* preserves the incorrect "Lil Durk" hypothesis; *Keep-Last-N*

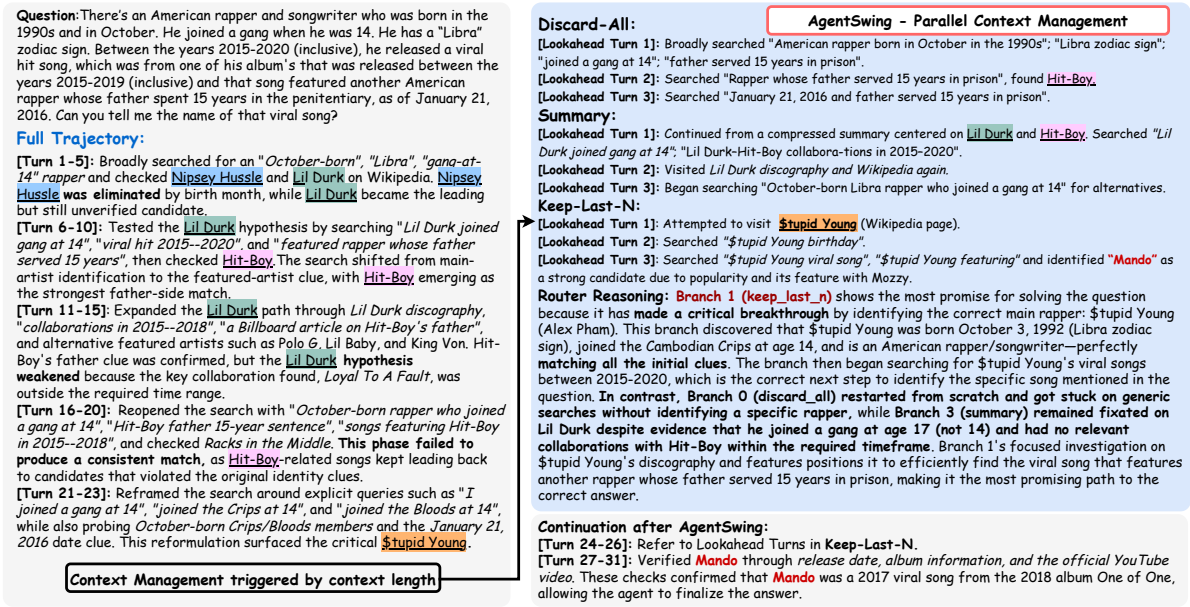


Figure 6: Case Study. AgentSwing correctly routes to *Keep-Last-N* branch and reaches the final answer afterward.

preserves the recent clue, enabling identification of "Mando" within lookahead turns. The router correctly selects *Keep-Last-N* and the agent reaches the final answer shortly afterward. Appendix G provides a detailed turn-by-turn summary together with a complementary GPT-OSS-120B case.

## 5 Related Work

**Long-horizon web agents.** LLM-based web agents have rapidly evolved from single-turn assistants into autonomous systems capable of web browsing, tool use, and long-horizon information seeking (Wu et al., 2025b,a; Li et al., 2025c; Fang et al., 2025; Liu et al., 2025b). Recent efforts from both academia and industry have demonstrated strong potential on deep information-seeking tasks, while also highlighting the importance of test-time scaling and long-horizon interaction design (Chai et al., 2025; Huang et al., 2025; Li et al., 2025a; Zeng et al., 2026b). However, most existing agents still rely on ReAct-style trajectories (Yao et al., 2023), making them increasingly vulnerable to context saturation, drift, and error accumulation as the search horizon grows (Fang et al., 2026).

**Context management for LLM agents.** Context management, or context engineering, aims to provide LLM-based agents with a more effective working context (Anthropic, 2025a; Qiao et al., 2025). Within long-horizon agents, prior methods mainly rely on static intra-task context curation, including reset-based policies such as *Discard-All*, recent-turn retention such as *Keep-Last-N* (Liu et al., 2025a; Team et al., 2026; Zeng et al., 2026a), and context compaction strategies closely related to *Summary* (Anthropic, 2025a; Liu et al., 2025a). Recent work such as AgentFold (Ye et al., 2026), MemAgent (Yu et al., 2025), and IterResearch (Chen et al., 2026) explores more sophisticated context operations, studying questions around designing specific memory or compression mechanisms, which are related to *Summary* and are complementary to our approach.

## 6 Conclusion

In this work, we introduce the first probabilistic framework that decomposes the end-to-end success of long-horizon deep information-seeking agents into two complementary dimensions, search efficiency and terminal precision, providing a unified view of how context management strategies affect long-horizon performance. Building on this perspective, we propose AgentSwing, an adaptive framework that moves beyond a single static context management strategy by expanding multiple parallel context management branches and dynamically selecting among them through a lookahead routing mechanism. Experiments across multiple benchmarks and agent backbones demonstrate that AgentSwing is both effective and generalizable, consistently improving long-horizon agent performance over static context management baselines.

## Limitations

Our work focuses on test-time context management as an external control mechanism for long-horizon agents. The proposed perspective helps clarify the efficiency-precision trade-off and leads to strong empirical gains. A more fundamental direction is to translate these principles into model-level competence, for example, by training agents that are intrinsically more efficient under smaller context budgets or more reliable under long-horizon noisy trajectories. In addition, the current routing mechanism is still performed by the agent model itself. Although this design is simple and effective, it may not be optimal. A stronger dedicated router, verifier, or trajectory evaluator with better foresight may further improve branch selection quality and therefore unlock additional gains for adaptive context management.

## References


Anthropic. 2025a. Effective context engineering for ai agents.

Anthropic. 2025b. Introducing claude opus 4.5.

Jingyi Chai, Shuo Tang, Rui Ye, Yuwen Du, Xinyu Zhu, Mengcheng Zhou, Yanfeng Wang, Yuzhi Zhang, Linfeng Zhang, Siheng Chen, et al. 2025. Scimaster: Towards general-purpose scientific ai agents, part i. x-master as foundation: Can we lead on humanity's last exam? *arXiv preprint arXiv:2507.05241*.

Guoxin Chen, Zile Qiao, Xuanzhong Chen, Donglei Yu, Haotian Xu, Xin Zhao, Ruihua Song, Wenbiao Yin, Huifeng Yin, Liwen Zhang, Kuan Li, Minpeng Liao, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2026. Iterresearch: Rethinking long-horizon agents via markovian state reconstruction. In *The Fourteenth International Conference on Learning Representations*.

Google DeepMind. 2025. A new era of intelligence with gemini 3.

Runnan Fang, Shihao Cai, Baixuan Li, Jialong Wu, Guangyu Li, Wenbiao Yin, Xinyu Wang, Xiaobin Wang, Liangcai Su, Zhen Zhang, et al. 2025. Towards general agentic intelligence via environment scaling. *arXiv preprint arXiv:2509.13311*.

Shicheng Fang, Yuxin Wang, XiaoRan Liu, Jiahao Lu, Chuanyuan Tan, Xinchi Chen, Yining Zheng Huang, Xipeng Qiu, et al. 2026. Agentlongbench: A controllable long benchmark for long-contexts agents via environment rollouts. *arXiv preprint arXiv:2601.20730*.

Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. 2025. Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous rl. *arXiv preprint arXiv:2508.07976*.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. 2024. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*.

Kelly Hong, Anton Troynikov, and Jeff Huber. 2025. Context rot: How increasing input tokens impacts llm performance. Technical report, Chroma.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. 2024. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*.

Yuchen Huang, Sijia Li, Minghao Liu, Wei Liu, Shijue Huang, Zhiyuan Fan, Hou Pong Chan, and Yi R Fung. 2025. Environment scaling for interactive agentic experience collection: A survey. *arXiv preprint arXiv:2511.09586*.

Baixuan Li, Dingchu Zhang, Jialong Wu, Wenbiao Yin, Zhengwei Tao, Yida Zhao, Liwen Zhang, Haiyang Shen, Runnan Fang, Pengjun Xie, et al. 2025a. Parallelmuse: Agentic parallel thinking for deep information seeking. *arXiv preprint arXiv:2510.24698*.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Rui Ye, Yida Zhao, Liwen Zhang, Litu Ou, Dingchu Zhang, Xixi Wu, Jialong Wu, Xinyu Wang, Zile Qiao, Zhen Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025b. Websailor-v2: Bridging the chasm to proprietary agents via synthetic data and scalable reinforcement learning. *Preprint*, arXiv:2509.13305.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, Weizhou Shen, Junkai Zhang, Dingchu Zhang, Xixi Wu, Yong Jiang, Ming Yan, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025c. Websailor: Navigating super-human reasoning for web agent. *Preprint*, arXiv:2507.02592.

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

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. 2025a. Deepseek-v3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Junteng Liu, Yunji Li, Chi Zhang, Jingyang Li, Aili Chen, Ke Ji, Weiyu Cheng, Zijia Wu, Chengyu Du, Qidi Xu, et al. 2025b. Webexplorer: Explore and evolve for training long-horizon web agents. *arXiv preprint arXiv:2509.06501*.

Ali Modarressi, Hanieh Deilamsalehy, Franck Dernoncourt, Trung Bui, Ryan A. Rossi, Seunghyun Yoon, and Hinrich Schuetze. 2025. Nolima: Long-context evaluation beyond literal matching. In *Forty-second International Conference on Machine Learning*.

Xuan-Phi Nguyen, Shrey Pandit, Revanth Gangi Reddy, Austin Xu, Silvio Savarese, Caiming Xiong, and Shafiq Joty. 2025. Sfr-deepresearch: Towards effective reinforcement learning for autonomously reasoning single agents. *arXiv preprint arXiv:2509.06283*.

OpenAI. 2025a. Deep research system card.

OpenAI. 2025b. gpt-oss-120b & gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

OpenAI. 2025c. Gpt-5.1: A smarter, more conversational chatgpt.

OpenAI. 2025d. Introducing openai o3 and o4-mini.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, et al. 2025. Humanity's last exam. *arXiv preprint arXiv:2501.14249*.

Zile Qiao, Shen Huang, Jialong Wu, Kuan Li, Wenbiao Yin, Xinyu Wang, Liwen Zhang, Baixuan Li, Zhengwei Tao, Weizhou Shen, Xixi Wu, Yong Jiang, Pengjun Xie, Fei Huang, Jun Zhang, and Jingren Zhou. 2025. WebResearcher: Unleashing unbounded reasoning capability in long-horizon agents. *Preprint*, arXiv:2509.13309.

Liangcai Su, Zhen Zhang, Guangyu Li, Zhuo Chen, Chenxi Wang, Maojia Song, Xinyu Wang, Kuan Li, Jialong Wu, Xuanzhong Chen, Zile Qiao, Zhongwang Zhang, Huifeng Yin, Shihao Cai, Runnan Fang, Zhengwei Tao, Wenbiao Yin, Rui Ye, Yong Jiang, Ningyu Zhang, Pengjun Xie, Fei Huang, Kai Ye, Kewei Tu, Chenxiong Qian, and Jingren Zhou. 2026. Scaling agents via continual pre-training. In *The Fourteenth International Conference on Learning Representations*.

Qiaoyu Tang, Hao Xiang, Le Yu, Bowen Yu, Yaojie Lu, Xianpei Han, Le Sun, WenJuan Zhang, Pengbo Wang, Shixuan Liu, et al. 2025. Beyond turn limits: Training deep search agents with dynamic context window. *arXiv preprint arXiv:2510.08276*.

Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025. WebShaper: Agentically data synthesizing via information-seeking formalization. *Preprint*, arXiv:2507.15061.

Kimi Team. 2025a. Kimi researcher tech report.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Y Charles, HS Che, Cheng Chen, Guanduo Chen, et al. 2026. Kimi k2. 5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*.

MiroMind Team. 2026. Introducing mirothinker 1.5: 30b parameters that outperform 1t models.

Tongyi DeepResearch Team. 2025b. Tongyi deepresearch: A new era of open-source ai researchers. https://github.com/Alibaba-NLP/DeepResearch.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. 2025. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*.

Ryan Wong, Jiawei Wang, Junjie Zhao, Li Chen, Yan Gao, Long Zhang, Xuan Zhou, Zuo Wang, Kai Xiang, Ge Zhang, et al. 2025. Widesearch: Benchmarking agentic broad info-seeking. *arXiv preprint arXiv:2508.07999*.

Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Gang Fu, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025a. Webdancer: Towards autonomous information seeking agency. *Preprint*, arXiv:2505.22648.

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, and Fei Huang. 2025b. Webwalker: Benchmarking llms in web traversal. *Preprint*, arXiv:2501.07572.

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

Rui Ye, Zhongwang Zhang, Kuan Li, Huifeng Yin, Zhengwei Tao, Yida Zhao, Liangcai Su, Liwen Zhang, Zile Qiao, Xinyu Wang, Yong Jiang, Pengjun Xie, Fei Huang, Siheng Chen, and Jingren Zhou. 2026. Agentfold: Long-horizon web agents with proactive context folding. In *The Fourteenth International Conference on Learning Representations*.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, et al. 2025. Memagent: Reshaping long-context llm with multi-conv rl-based memory agent. *arXiv preprint arXiv:2507.02259*.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chengxing Xie, Cunxiang Wang, et al. 2026a. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*.

Weihao Zeng, Keqing He, Chuqiao Kuang, Xiaoguang Li, and Junxian He. 2026b. Pushing test-time scaling limits of deep search with asymmetric verification. In *The Fourteenth International Conference on Learning Representations*.

Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, et al. 2025. Browsecomp-zh: Benchmarking web browsing ability of large language models in chinese. *arXiv preprint arXiv:2504.19314*.

## A Tools

- **Search:** Performs batched Google queries and returns the top-10 ranked results for each query.
- **Visit:** Fetches webpages from URLs and extracts information relevant to the specified goal.
- **Google Scholar:** Returns top-10 academic search results with snippets, citations, and scholarly metadata.
- **Python Interpreter:** Executes arbitrary Python code in a secure sandbox for computational tasks and data analysis. We use Code Sandbox[1] to ensure secure and isolated execution.

## B Evaluation Metrics and Hyper-parameters

All evaluations are conducted under the LLM-as-a-Judge protocol (Gu et al., 2024), using the official evaluation prompts and judging models released by each benchmark. For all agent models, we set the maximum context length to 128k tokens. Unless otherwise specified, we set the maximum interaction budget to 400 turns for all context management strategies. To ensure fair comparison and reproducibility, model-specific hyper-parameters follow the officially recommended or empirically optimal settings of each agent backbone. For all experiments involving context management, we set the context budget as a fixed ratio $r$ of the 128k maximum context length. Specifically, we use $r = 0.2$ for GPT-OSS-120B and $r = 0.4$ for both Tongyi-DR-30B-A3B and DeepSeek-v3.2. The rationale behind these settings is discussed in Section 2.2.

## C Gains from Parallel Context Management Combinations

Figure 7 compares different candidate context management combinations within AgentSwing. Although some single strategies, especially *Discard-All*, already perform strongly, combining multiple candidate strategies generally yields further gains. This is particularly clear for combinations such as *Discard-All* + *Summary*, which outperform either constituent strategy used alone. These results suggest that different context management operations provide complementary advantages, and that AgentSwing benefits from routing over a richer set of candidate continuations than any single static policy can offer. More broadly, they indicate that AgentSwing's effectiveness depends not only on the routing mechanism itself, but also on the diversity and complementarity of the candidate strategy set. This also suggests that exploring richer or more specialized candidate strategies is a promising direction for further improving performance.

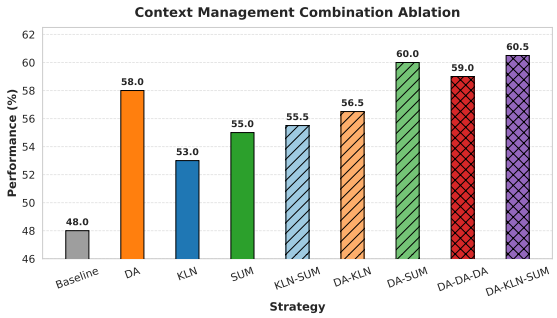


Figure 7: Performance of Tongyi-DR-30B-A3B on BrowseComp under AgentSwing with different context management combinations.

## D Comparison of Token Efficiency

Figure 8 compares token efficiency on the aligned cases used in Table 2. Each point denotes one finished task, plotted by its total interaction turns and cumulative token count at termination. Although *AgentSwing* introduces additional token usage due to lookahead routing, the overhead remains modest in practice. One reason is that efficiency-oriented strategies such as *Keep-Last-N* often incur higher cumulative token usage at similar turn counts, since they retain more trajectory history in the context. By contrast, *Discard-All* tends to accumulate fewer tokens, but usually requires more turns to finish. Taken together, these results show that AgentSwing does not achieve its gains by paying a substantially larger overall cost.

## E Routing Prompt

The following is the full routing prompt used in AgentSwing. At each context management trigger, the router receives the original question, the full conversation history before context management, and each branch's managed context plus $K$-step lookahead results.

```
You are a Context Management Router
evaluator. Your task is to evaluate
multiple branches that have been executed
with different context-management
strategies and k-turn lookahead, then
select the branch that shows the most
promise for solving the question.

## Original Question
```

[1] https://github.com/bytedance/SandboxFusion

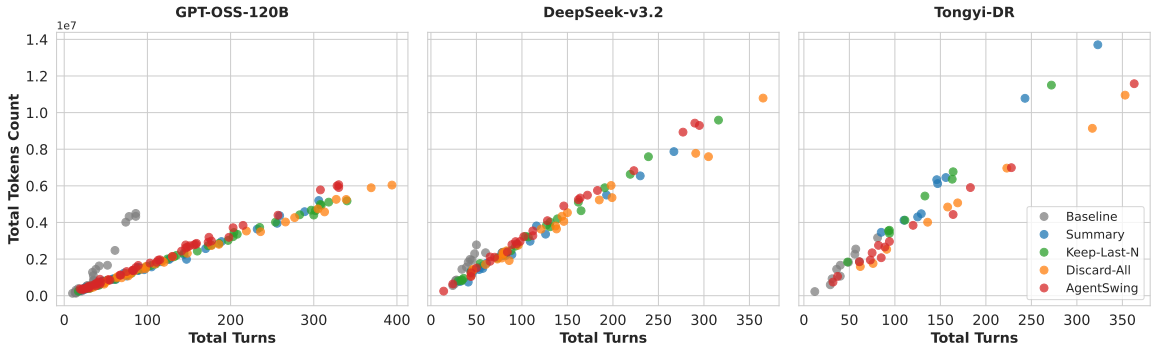


Figure 8: Token efficiency on the aligned cases used in Table 2. Each point corresponds to one finished task and is plotted by its total interaction turns and cumulative token count at termination.

```
{question}

## Messages Before Context Management
The following is the complete conversation
history before context management:
{messages_before_cm}

## Branch Results
All branches start from the same
conversation history before context
management. Each branch independently:
1. Applies one context-management
   operation, resulting in a managed
   context.
2. Executes k-step lookahead based on
   its own managed context.
3. Cannot see the lookahead results
   of other branches.

For each branch, we provide:
- The resulting message count after
  context management.
- The k-step lookahead execution results,
  containing only the new messages
  generated during lookahead.

### Branch {branch_index}: {strategy_name}
Resulting message count after context
management: {num_messages_after_cm}
Lookahead K Steps
({num_lookahead_messages} new messages):
{lookahead_messages}

[Repeat for all branches.]

## Evaluation Task
Based on the original question, the
conversation history before context
management, and each branch's
context-managed result plus lookahead
progress, evaluate which branch shows the
most promise for solving the question
if execution continues.

Consider:
- Which branch demonstrates better
  progress toward answering the question?
- Which branch shows more relevant and
  useful reasoning or tool usage?
- Which branch is more likely to lead
  to a correct final answer?

## Output Format
Respond with a JSON object in the
following format:

{
  "best_branch_index": <integer>,
  "reasoning": "<string>"
}

Now evaluate and select the best branch:
```

To verify that the router does not rely on strategy-name priors, we run a blind-routing ablation where strategy names are removed from branch headers. In the blind-routing variant, explicit strategy names (e.g., "Branch 1: keep_last_n") are replaced with neutral labels (e.g., "Branch 1"). On 50 randomly sampled GPT-OSS-120B cases repeated twice, non-blind routing obtains 59.0 while blind routing obtains 60.0, confirming that routing is driven by observed lookahead progress.

## F Analysis of Strategy Transitions under AgentSwing

Figure 9 shows the empirical strategy-transition probabilities under AgentSwing. The transition matrices are clearly non-uniform, indicating that routing behavior is not random. Instead, the preferred transitions depend on the underlying backbone. DeepSeek-v3.2 and Tongyi-DR tend to favor *Summary*, whereas GPT-OSS-120B more often transitions to *Discard-All*.

## G Case Study

Tables 5 and 6 provide a more detailed turn-by-turn summary of the main trajectory in the Mando case from Figure 6, while Table 7 reports the corresponding lookahead continuations under different context management strategies.

In addition, Tables 8, 9, and 10 present a complementary case from GPT-OSS-120B in which

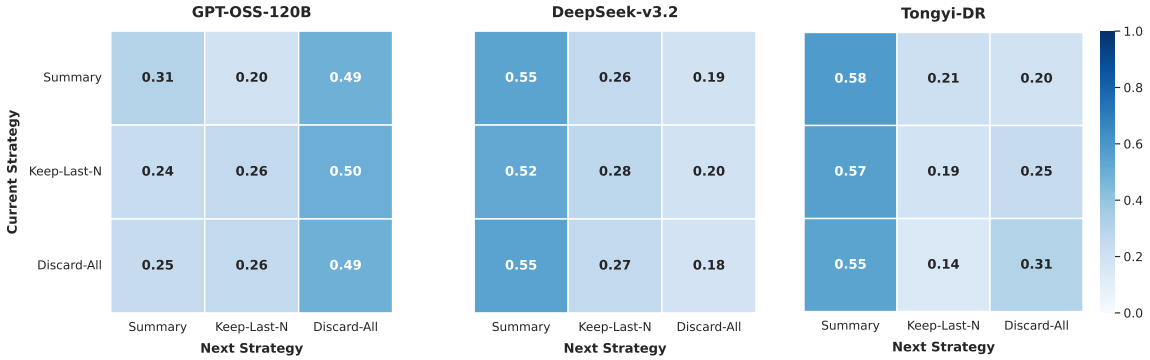


Figure 9: Statistical analysis of strategy transitions under AgentSwing. Each entry gives the probability of routing to the next strategy conditioned on the current one.

*Discard-All* is selected as the context management strategy. This case demonstrates that long-horizon web agents may become trapped in trajectories dominated by substantial noise, repetition, and failed local exploration. In such situations, aggressively resetting the accumulated context through *Discard-All* allows the agent to recover a cleaner search state, refocus on the most relevant clues, and ultimately improve the chance of reaching the correct answer.

| Turn ID | Tool Call | Content | Key State |
|---|---|---|---|
| Turn 1 | search | American rapper born in October in the 1990s; Libra zodiac sign; joined a gang at 14. | Started with broad candidate retrieval based on birth month, zodiac sign, and gang involvement; surfaced candidates such as Lil Durk. |
| Turn 2 | search | Rapper joined gang at age 14; American rapper gang member at 14. | Narrowed the search to rappers who explicitly joined a gang at age 14; exposed promising clues including $tupid Young and Nipsey Hussle. |
| Turn 3 | visit | Nipsey Hussle (Wikipedia) | Verified that Nipsey Hussle was born in August and therefore did not satisfy the October/Libra clue. |
| Turn 4 | search | October-born rapper who joined a gang at 14; Libra rapper with gang affiliation. | Searched again for October-born gang-affiliated rappers; Lil Durk became a stronger candidate. |
| Turn 5 | visit | Lil Durk (Wikipedia) | Confirmed that Lil Durk was born on October 19, 1992 (Libra), but his gang-entry age remained unsupported and appeared inconsistent with the clue. |
| Turn 6 | search | Lil Durk joined gang at 14; Durk Banks gang age. | Tried to verify whether Lil Durk joined a gang at age 14; no direct evidence supported this claim. |
| Turn 7 | search | Lil Durk viral hit (2015–2020); featured rapper whose father served 15 years. | Shifted from identity matching to song matching; attempted to find a Lil Durk viral song and a featured rapper with the father-in-prison clue, but results were inconclusive. |
| Turn 8 | search | Rapper whose father served 15 years in prison. | Focused on the featured-artist clue; Hit-Boy emerged as the strongest match. |
| Turn 9 | visit | Hit-Boy (Wikipedia) | Checked Hit-Boy's profile; found ambiguity because one source suggested 9 years, creating tension with the 15-year clue. |
| Turn 10 | search | Lil Durk collaborations in 2015–2018. | Collected candidate Lil Durk collaborations and featured artists for downstream filtering. |
| Turn 11 | visit | Lil Durk discography (Wikipedia) | Examined Lil Durk's discography in detail to match songs, years, and featured artists. |
| Turn 12 | search | Rapper whose father was in prison for 15 years. | Revisited the father clue; Hit-Boy remained the dominant candidate. |
| Turn 13 | visit | Hit-Boy's father article (Billboard) | Confirmed through Billboard that Hit-Boy's father had indeed served 15 years, resolving the earlier inconsistency. |
| Turn 14 | search | Lil Durk and Hit-Boy collaboration. | Looked for Lil Durk–Hit-Boy collaborations; found "Loyal To A Fault," but it was released in 2021 and thus fell outside the required window. |
| Turn 15 | search | Polo G father prison; Lil Baby father prison; King Von father prison. | Explored alternative featured rappers connected to Lil Durk's songs, but none fit the father clue clearly. |
| Turn 16 | search | October-born rapper who joined a gang at 14. | Reopened the identity search because the Lil Durk hypothesis was weakening; results remained noisy. |

Table 5: Main-branch trajectory for the "Mando" case (Part I).

| Turn ID | Tool Call | Content | Key State |
|---|---|---|---|
| Turn 17 | search | Hit-Boy father 15-year sentence; Big Hit prison timeline. | Further validated the 15-year imprisonment timeline around Hit-Boy's father, strengthening the father-side clue. |
| Turn 18 | search | Songs featuring Hit-Boy in 2015–2018. | Looked for songs where Hit-Boy was explicitly credited as a featured artist; results were still fragmented. |
| Turn 19 | visit | Racks in the Middle (Wikipedia) | Checked "Racks in the Middle" and confirmed that although Hit-Boy was involved, Nipsey Hussle did not fit the October/1990s clue. |
| Turn 20 | search | Viral songs featuring Hit-Boy in 2017–2019. | Continued searching for a viral song featuring Hit-Boy; no decisive match was found yet. |
| Turn 21 | search | October-born Crips/Bloods member in the 1990s. | Explored West Coast gang-affiliated rappers born in October; results were mostly generic gang references. |
| Turn 22 | search | January 21, 2016 and father served 15 years in prison. | Investigated the date anchor around January 21, 2016, but this direction produced little value. |
| Turn 23 | search | I joined a gang at 14; joined the Crips/Bloods at 14. | Reframed the search around explicit self-reports of joining a gang at 14; this surfaced the critical **$tupid Young** clue. |
| CM | – | Context management triggered. | The router selected the **Keep-Last-N** branch so that it can continue with $tupid Young clue. |
| Turn 24 | visit | $tupid Young (Wikipedia page lookup) | Attempted to open a Wikipedia page for **$tupid Young**, but the page was unavailable. |
| Turn 25 | search | $tupid Young birthday; Stupid Young birth date. | Verified the rapper's identity: $tupid Young was born on October 3, 1992, matching both the 1990s and Libra clues. |
| Turn 26 | search | $tupid Young viral song; $tupid Young hit song; $tupid Young featuring. | Searched his discography and identified "Mando" as a strong candidate due to popularity and its feature with Mozzy. |
| Turn 27 | search | Mozzy father prison 15 years; Mozzy dad penitentiary. | Verified the featured-artist clue; found a Vice article stating that Mozzy's father had spent 15 years in prison. |
| Turn 28 | visit | Mando (Wikipedia page lookup) | Attempted to visit a dedicated page for "Mando," but no useful Wikipedia entry was available. |
| Turn 29 | search | Mando by $tupid Young and Mozzy; release date; album information. | Confirmed that "Mando" was released on November 17, 2017 and linked to the 2018 album *One of One*. |
| Turn 30 | visit | Mando official YouTube video | Used the YouTube page to confirm the song's viral status via its massive view count (68M+). |
| Turn 31 | answer | – | Integrated all evidence and finalized the answer: **"Mando"**. |

Table 6: Main-branch trajectory for the "Mando" case (Part II).

| Branch | Turn ID | Tool Call | Content | Key State |
|---|---|---|---|---|
| Discard-All | LA Turn 1 | search | American rapper born in October in the 1990s; Libra zodiac sign; joined a gang at 14; father served 15 years in prison. | Restarted from scratch with broad retrieval. It surfaced generic October-born candidates such as Lil Durk, but made no decisive progress on the real target. |
| | LA Turn 2 | search | Rapper whose father served 15 years in prison. | Focused on the father clue and rediscovered Hit-Boy, but still lacked a correct main-rapper hypothesis. |
| | LA Turn 3 | search | January 21, 2016 and father served 15 years in prison. | Pursued the date-anchored clue without traction. This branch remained broad and under-focused. |
| Keep-Last-N | LA Turn 1 | visit | $tupid Young (Wikipedia page lookup) | Attempted to open a Wikipedia page for $tupid Young, but the page was unavailable. |
| | LA Turn 2 | search | $tupid Young birthday; Stupid Young birth date. | Verified the rapper's identity: $tupid Young was born on October 3, 1992, matching both the 1990s and Libra clues. |
| | LA Turn 3 | search | $tupid Young viral song; $tupid Young hit song; $tupid Young featuring. | Searched his discography and identified "Mando" as a strong candidate due to popularity and its feature with Mozzy. |
| Summary | LA Turn 1 | search | Lil Durk joined gang at 14; Lil Durk–Hit-Boy collaborations in 2015–2020. | Continued from a compressed summary centered on Lil Durk and Hit-Boy. This preserved structure but also inherited a misleading focus. |
| | LA Turn 2 | visit | Lil Durk (Wikipedia); Lil Durk discography (Wikipedia). | Verified that Lil Durk joined the Black Disciples at age 17 rather than 14, and found no Lil Durk–Hit-Boy collaboration within 2015–2020. |
| | LA Turn 3 | search | October-born Libra rapper who joined a gang at 14. | Only after falsifying the Lil Durk path did this branch begin searching for alternative rappers; within the lookahead horizon, it did not reach the $tupid Young breakthrough. |

Table 7: Lookahead branches triggered by context management in the "Mando" case.

| Turn ID | Tool Call | Content | Key State |
|---|---|---|---|
| Turn 1 | search | Performer who stapled paper to his forehead; sideshow stapling act. | Started with direct retrieval on the stapling clue, but results were dominated by noisy modern pages, social media posts, and irrelevant literal uses of "stapled paper." |
| Turn 2 | search | Paper-to-forehead sideshow performer who ate something live. | Tried to combine the stapling clue with the "ate live creatures" clue; surfaced sideshow-related entities such as Jim Rose Circus, but no stable performer identity. |
| Turn 3 | search | Strongwoman associated with "beef, game, and plenty of vegetables." | Shifted to a secondary clue in the question, but the retrieved results were largely noisy and did not yet identify the relevant strongwoman. |
| Turn 4 | search | Bethel, Connecticut; Manhattan museum; Feejee Mermaid. | Used the Bethel / museum / Feejee Mermaid clue to infer the publication domain; this pointed toward P. T. Barnum and the historical oddities / sideshow space. |
| Turn 5 | search | Bethel, Connecticut; sideshow; Manhattan museum; Feejee Mermaid. | Repeated the supporting-entity search, but the results were still not sufficiently specific to identify the source publication. |
| Turn 6 | search | Strongwoman who threw a heckler across a tent. | Switched to another distinctive supporting clue in order to identify the common source through a secondary figure. |
| Turn 7 | search | Strongwoman threw a heckler. | A simplified version of the query surfaced references to Minerva, helping move the search toward historical strongwoman material. |
| Turn 8 | visit | Victorian strongwomen article (iNews). | Visited the article and confirmed that the strongwoman was Josephine Schauer Blatt (Minerva), establishing that the question belongs to the historical sideshow / freak-show domain. |
| Turn 9 | search | Circus performer who staples paper to his forehead. | Returned to the stapling clue after confirming the domain; the results now included more circus / sideshow-related pages, but still no exact match. |
| Turn 10 | visit | Jelly Boy the Clown article (East Bay Times). | Found a modern performer who allowed money to be stapled to his face, but this did not match the clue about eating live creatures. |
| Turn 11 | search | Stapling performer who also eats live creatures. | Tried to jointly resolve the two key attributes, but the results still lacked a decisive source text. |
| Turn 12 | search | Paper on forehead; eating live creatures. | Continued direct clue search, but the retrieval remained noisy and failed to identify the exact publication or performer. |
| Turn 13 | search | Exact phrase: "stapled paper to his forehead." | Achieved the first major breakthrough: search results surfaced the *PDF* **The Victorian Sideshow**, with a snippet containing the critical phrase "has willingly stapled paper to his forehead ... eaten a mouthful ..." |
| Turn 14 | visit | The Victorian Sideshow PDF (direct access attempt). | Tried to open the PDF directly, but the tool returned no extractable content. This established the central bottleneck of the case. |
| Turn 15 | search | The Victorian Sideshow PDF. | Searched for alternative paths to the same PDF, but the results still pointed back to the same inaccessible source. |
| Turn 16 | visit | The Victorian Sideshow PDF (second access attempt). | Repeated the PDF visit attempt, but the extraction failure persisted. |
| Turn 17 | search | Paper to his forehead; sideshow. | Looked for alternative source surfaces after the failed PDF access; results still pointed mainly to the same PDF and its mirrors. |
| Turn 18 | visit | Scribd mirror of *Sideshow*. | Attempted to recover the content through Scribd, but the page was effectively inaccessible. |

Table 8: Main-branch trajectory for the "live-crickets" case (Part I).

| Turn ID | Tool Call | Content | Key State |
|---|---|---|---|
| Turn 19 | search | Stapled paper; forehead; sideshow; eaten. | Combined the snippet clues again, but the results still revolved around the unresolved PDF source. |
| Turn 20 | search | Full snippet phrase including “eaten a mouthful.” | Queried the visible snippet directly; this helped confirm the source phrase, but still did not reveal the missing object after “eaten a mouthful of ...” |
| Turn 21 | search | Performer name from the stapling clue. | Tried to infer the performer identity directly from the snippet description, but the retrieval remained inconclusive. |
| Turn 22 | search | Exact phrase: “has willingly stapled paper to his forehead.” | Repeated exact-phrase retrieval to localize the passage more precisely, but still without extractable full text. |
| Turn 23 | search | The Victorian Sideshow PDF. | Re-confirmed that **The Victorian Sideshow** was the likely shared source behind the unusual individuals in the question. |
| Turn 24 | visit | The Victorian Sideshow PDF (targeted extraction attempt). | Made a more targeted attempt to extract the paragraph about the stapling performer and the live-creature clue, but the visit tool still failed. |
| CM | – | Context management triggered. | Context management was triggered because the trajectory had become long, noisy, and partially repetitive. The router evaluated three branches and selected **Discard-All**. |
| Turn 25 | search | Stapled paper to forehead performer. | After the reset, restarted with a cleaner search plan; quickly re-entered the correct search space without carrying over the accumulated local noise. |
| Turn 26 | search | Feejee Mermaid; Minerva; Jo-Jo; supporting clue bundle. | Used multiple supporting entities together to verify that the publication family was correct and that the search was grounded in the historical sideshow domain. |
| Turn 27 | visit | Jo-Jo the Dog-Faced Boy article. | Confirmed another supporting figure from the same source family, increasing confidence that the publication hypothesis was correct. |
| Turn 28 | search | Exact stapling-performer phrasing. | Returned to the core unresolved clue after re-confirming the correct publication **The Victorian Sideshow**. |
| Turn 29 | visit | The Victorian Sideshow PDF. | Direct extraction still failed, confirming that the bottleneck was tool-access related rather than search-related. |
| Turn 30 | visit | Alternative text-extraction endpoint for the PDF. | Achieved the decisive breakthrough by using an alternative access path that successfully returned the source text, revealing that the performer had “eaten a mouthful of live crickets.” |
| Turn 31 | answer | – | Integrated all evidence and produced the final answer: the person who stapled paper to his forehead ate a mouthful of **live crickets**. |

Table 9: Main-branch trajectory for the "live-crickets" case (Part II).

| Branch | Turn ID | Tool Call | Content | Key State |
|---|---|---|---|---|
| Discard-All | LA Turn 1 | search | Stapled paper to forehead performer. | Restarted from scratch and quickly re-entered the correct search space around the stapling-performer clue, without inheriting the noisy local loop around failed PDF extraction. |
| | LA Turn 2 | search | Feejee Mermaid; Minerva; Jo-Jo; supporting clue bundle. | Used multiple supporting entities together to verify that the publication family was correct and that the search was grounded in the historical sideshow domain. |
| | LA Turn 3 | visit | Jo-Jo the Dog-Faced Boy article. | Also revisited supporting clues from the question, indicating that the branch was reconstructing the source-publication hypothesis through multiple entities rather than overfitting to one failed access path. |
| Keep-Last-N | LA Turn 1 | visit | Scribd mirror of *Sideshow*. | Preserved the most recent local context, which was already dominated by failed source-extraction attempts; immediately re-entered the same bottleneck. |
| | LA Turn 2 | search | The Victorian Sideshow PDF. | Continued searching for alternative access points to the same PDF, but remained trapped in the same unresolved extraction problem. |
| | LA Turn 3 | visit | The Victorian Sideshow PDF. | Attempted direct PDF access again and failed, showing that preserving the most recent context mainly preserved the local dead end rather than useful progress. |
| Summary | LA Turn 1 | search | Exact stapling phrase; sideshow; live-creature clue. | Used the summary-preserved hypothesis that The Victorian Sideshow was likely the correct source, and re-centered search on the key unresolved phrase. |
| | LA Turn 2 | search | Repeated phrase-centered retrieval. | Continued operating at the correct abstraction level, but still remained dependent on search-result snippets and inaccessible source pages. |
| | LA Turn 3 | search | Repeated snippet-oriented search behavior. | Maintained a cleaner high-level focus than Keep-Last-N, but did not produce a concrete recovery step that would break the extraction bottleneck. |

Table 10: Lookahead branches triggered by context management in the "live-crickets" case.