

000 001 002 003 004 005 COST-EFFECTIVE AGENT TEST-TIME SCALING 006 VIA BUDGET-AWARE THINKING 007 008 009

010 **Anonymous authors**
011 Paper under double-blind review
012
013
014
015
016
017
018
019
020
021
022
023
024
025
026
027
028
029

030 ABSTRACT 031

032 Scaling test-time computation improves performance across different tasks on
033 large language models (LLMs), which has also been extended to tool-augmented
034 agents. For these agents, scaling involves not only “thinking” in tokens but also
035 “acting” via tool calls. The number of tool calls directly bounds the agent’s in-
036 teraction with the external environment. However, we find that simply granting
037 agents a larger tool-call budget fails to improve performance, as they lack “budget
038 awareness” and quickly hit a performance ceiling. To address this, we study how
039 to scale such agents effectively under explicit tool-call budgets, focusing on web
040 search agents. We first introduce the **Budget Tracker**, a lightweight plug-in that
041 provides the agent with continuous budget awareness, enabling simple yet effec-
042 tive scaling. We further develop **BATS** (Budget Aware Test-time Scaling), an ad-
043 vanced framework that leverages this awareness to dynamically adapt its planning
044 and verification strategy, deciding whether to “dig deeper” on a promising lead or
045 “pivot” to new paths based on remaining resources. To analyze cost-performance
046 scaling in a controlled manner, we formalize a unified cost metric that jointly ac-
047 counts for token and tool consumption. We provide the first systematic study on
048 budget-constrained agents, showing that budget-aware methods produce more fa-
049 vorable scaling curves and push the cost-performance Pareto frontier. Our work
050 offers empirical insights toward a more transparent and principled understanding
051 of scaling in tool-augmented agents.
052

053 1 INTRODUCTION

054 Scaling test-time compute in large language models (LLMs) helps improve the performance across
055 a wide range of tasks including reasoning, coding (Wu et al., 2025b; Snell et al., 2024; Muennighoff
056 et al., 2025; Chen et al., 2025b). Mainstream scaling strategies such as sequential (Madaan et al.,
057 2023) and parallel scaling (Wang et al., 2023; Brown et al., 2024) enable models to utilize more
058 effort, elicit deeper reflection, and refine their outputs, often leading to substantial gains in answer
059 quality (Zhang et al., 2025a). These successes motivate recent efforts to extend test-time scaling
060 to tool-augmented agents (Zhu et al., 2025b; Wang et al., 2025a), where LLMs are equipped with
061 various tools to interact with the external environment such as search engines or APIs.
062

063 Test-time scaling for tool-augmented agents expands both thinking (tokens) and acting (tool calls).
064 While textual reasoning tasks are usually scaled by thinking with more tokens, solely increasing
065 the internal thinking effort in agents proves to be suboptimal and does not always translate to per-
066 formance scaling (Shen et al., 2025). Unlike controlling the token count in textual reasoning (Han
067 et al., 2025b; Pu et al., 2025), in agent tasks such as web browsing, the number of tool calls directly
068 determines the depth and breadth of exploration (Anthropic, 2025c; Team et al., 2025b), defining
069 the effective boundary of external information access.
070

071 However, simply granting agents more test-time tool-call resources does not guarantee better per-
072 formance. Our analysis reveals a critical bottleneck: *standard agents lack inherent budget awareness*.
073 Without explicit signals, they often perform shallow searches (Lu et al., 2025) and fail to utilize ad-
074 dditional resources, even when available. As our empirical findings later demonstrate, standard agents
075 quickly hit a “performance ceiling”, saturating their capabilities regardless of how much extra bud-
076 get is allocated. The challenge is not simply spending more, but spending wisely: the marginal
077 benefit per tool call is uncertain, so every call must be spent strategically.
078

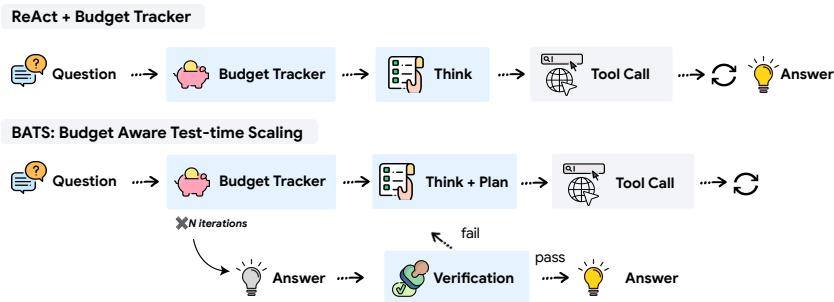


Figure 1: Budget Tracker is a lightweight plug-in that can be applied to both a standard ReAct agent (top) and more advanced orchestration frameworks like BATS (bottom). In this figure, blue boxes highlight modules that adapt to the budget.

This unique complexity brings up a critical research question: How can tool-augmented agents scale *effectively* by making the best use of a given resource budget? We study this question in a budget-constrained setting, grounding our analysis in search agents equipped with search and browse tools, which are widely used in practice (Google, 2025; OpenAI, 2025) and inherently require extensive tool calls to collect external information. To ensure a fair and transparent comparison, we formalize a unified cost metric that jointly accounts for the economic costs of both internal token consumption and external tool interactions. This metric enables us to trace a true cost-performance scaling trend.

We first develop an intuitive, light-weight approach: **Budget Tracker** (Figure 1 (top)), which is a plug-and-play module compatible with most ReAct-based agents (Yao et al., 2023). Budget Tracker provides the agent with a continuous signal of resource availability, proving to be a simple yet effective method for enabling budget-aware tool use. Through extensive experiments, we show that this simple plug-in improves performance across various budget constraints. Combining this explicit budget awareness with traditional test-time scaling strategies consistently enables more effective scaling behaviors while continuously pushing the cost-performance Pareto frontier.

While simple awareness is effective, it still operates within the agent’s predefined logic. To further break the scaling ceiling and fully internalize resource constraints, we introduce **BATS** (Budget Aware Test-time Scaling) (Figure 1 (bottom)), a framework designed to maximize agent performance under any given budget. At its core, BATS maintains a continuous signal of remaining resources, and it uses this information to dynamically adapt its behavior. A planning module adjusts stepwise effort to match the current budget, while a verification module decides whether to “dig deeper” into a promising lead or “pivot” to alternative paths based on resource availability.

Our empirical study systematically evaluates the relationship between overall resource cost and task performance by comparing different scaling frameworks under varying budgets. The results show that BATS produces more favorable scaling curves: it achieves higher performance while using fewer tool calls and incurring lower overall cost than competing methods. These findings demonstrate the potential of budget-aware design for creating effective and efficient tool-augmented agents, highlighting the importance of explicitly accounting for cost in agent test-time scaling.

We summarize our contributions as follows:

- We provide the first systematic study of budget-constrained tool-use agents by formalizing agent test-time scaling with explicit tool-call budgets and introducing a unified cost metric.
- We introduce **Budget Tracker**, a light-weight, plug-in module compatible with any agent orchestration framework that enables effective budget-aware tool use.
- We develop **BATS**, a budget-aware framework that dynamically adapts planning and verification strategies based on real-time resource tracking, flexibly switching between deepening a lead and branching to alternatives.
- We conduct systematic experiments under varying budgets and unified costs with search agents, demonstrating that BATS is more cost-effective and yields more favorable scaling curves and better cost–performance trade-offs.

108 **2 PROBLEM FORMULATION**109 **2.1 AGENT TEST-TIME SCALING**

110 We formulate agent test-time scaling as how an agent’s performance scales with its budget for ex-
 111 ternal tool-call interaction, refining the broader concept of test-time interaction scaling as discussed
 112 in Shen et al. (2025). To make agent test-time scaling cost-effective, an ideal agent should be able to
 113 achieve its best possible performance under an arbitrary budget constraint on the scaling curve. To
 114 this end, our target is to design an effective and efficient agent framework, π , that maximizes answer
 115 accuracy while adhering to a strict tool-call budget.

116 Assume the agent is equipped with a set of K tools as $\mathcal{T} = \{t_1, \dots, t_K\}$. For a given question
 117 $x \in \mathcal{X}$, the agent works under a budget $\mathbf{b} = (b_1, \dots, b_K)$, where b_i is the maximum number of
 118 invocations of tool $t_i \in \mathcal{T}$. Let $\hat{y}_\pi(x)$ denote the agent’s predicted answer for question x with
 119 ground truth $y(x)$ and let $c_i(x; \pi)$ be the realized number of calls to tool t_i on x . We formulate the
 120 agent test-time scaling problem as a budget-constrained optimization objective:

$$\begin{aligned} 123 \quad \max_{\pi} \quad & \text{Acc}_{\mathbf{b}}(\pi) = \mathbb{E}_x [\mathbf{1}\{\hat{y}_\pi(x) = y(x)\}] \\ 124 \quad \text{s.t.} \quad & c_i(x; \pi) \leq b_i \quad \text{for all } i = 1, \dots, K, \text{ and every } x \in \mathcal{X} \end{aligned} \quad (1)$$

125 Here the objective is the expected accuracy over all questions. The constraint ensures that for any
 126 given question, the number of realized calls for each tool never exceeds its allocated budget. By
 127 evaluating the agent performance at various budget levels, we can trace the performance-cost curve,
 128 which characterizes the agent’s test-time scaling behavior, showing how effectively it leverages bud-
 129 get resources to its problem solving capabilities.

130 **Budget vs. Cost.** We distinguish between the preset budget constraint, which specifies the maxi-
 131 mum number of tool calls available to the agent, and the realized cost, which reflects the resources
 132 actually consumed during execution. While the budget imposes a hard upper limit, the final cost
 133 depends on the agent’s strategy. To facilitate a more consistent and comprehensive comparison, we
 134 introduce in Section 2.2 a unified post-hoc cost metric for analyzing agents’ test-time scaling.

135 **Choice of Budget.** Among possible constraints, we prioritize a tool-call budget over a token-based
 136 budget for its *relevance*, *consistency*, and *practicability*. A tool-call limit offers a more relevant and
 137 direct constraint on an agent’s ability to acquire external knowledge than the tokens used for internal
 138 reasoning. This choice is also consistent with and justified by established practices (Shen et al.,
 139 2025; Team et al., 2025b). Furthermore, it offers greater practicability, as it is often non-trivial to
 140 pre-determine an appropriate token budget for complex, multi-step agentic tasks. While the budget
 141 is defined by tool calls, we still incorporate token usage into our unified cost metric (Section 2.2) to
 142 ensure a more fair and transparent comparison.

143 **2.2 PROBLEM INSTANTIATION WITH SEARCH AGENT**

144 In our work, we instantiate the test-time scaling problem with search agent, a setting selected for its
 145 broad applicability and the presence of established benchmarks.

146 A search agent is an LLM that answers an information-seeking question x by retrieving external
 147 evidence and reasoning over it. The agent follows an iterative ReAct-style loop (Yao et al., 2023),
 148 alternating between internal thinking and external actions. The agent has access to two primary tools
 149 for interacting with the world:

150 **(1) Search.** This tool helps perform a standard search engine query. Given a text query, it returns a
 151 list of search results, each including a title, a brief snippet, and a URL.

152 **(2) Browse.** Given a specific URL, this tool scrapes the full content of the corresponding webpage,
 153 providing detailed information that is often unavailable in a search snippet.

154 **Unified Cost Metric.** We model the agent’s total cost as the sum of consumed resources along two
 155 dimensions: tokens and tool calls. To create a unified metric, we map both to their corresponding
 156 economic costs.

157 **(1) Token cost.** This represents the agent’s internal cognitive effort, including its reasoning, plan-
 158 ning, and parametric knowledge processing. Token costs are calculated based on the pricing of the
 159 model provider, distinguishing among input, output, and cache hit tokens. In multi-round iterative

162 frameworks, the output of iteration i becomes part of the input for iteration $i + 1$. Any overlap is a
 163 cache hit, thereby lowering the token consumption cost.

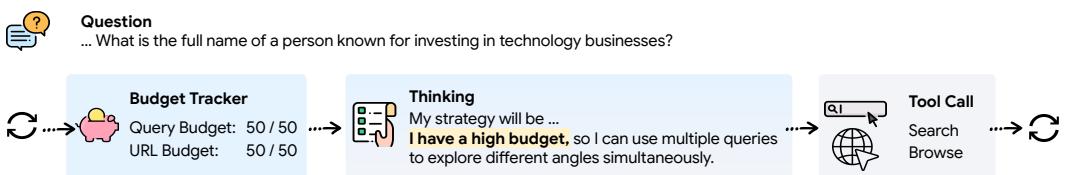
164 **(2) Tool call cost.** This represents the agent’s active interaction with the external environment
 165 through information-seeking actions. Each invocation of an external service (e.g., a search query
 166 or a browsing request) incurs a cost determined by the pricing of the corresponding API provider.

167 The total unified cost, $C_{unified}(x; \pi)$, for solving a given question x under policy π is the sum of the
 168 token cost and the tool call cost. Let $c_i(x; \pi)$ be the number of actual invocations to tool t_i and P_i
 169 be the economic cost per invocation of t_i . The unified cost metric is then defined as:
 170

$$171 \quad C_{unified}(x; \pi) = \underbrace{c_{token}(x; \pi)}_{\text{Token Cost}} + \underbrace{\sum_{i=1}^K c_i(x; \pi) \cdot P_i}_{\text{Total Tool Cost}} \quad (2)$$

175 Here, $c_{token}(x; \pi)$ represents the total cost incurred from token consumption during the agent’s reasoning
 176 process for question x . This formulation allows us to uniformly analyze the actual incurred
 177 costs for each execution. These two dimensions are inherently coupled: additional tool calls generally
 178 increase token consumption, as the agent must process and reason over the retrieved external
 179 information. Measuring this unified cost under varying budget constraints, b , enables a more comprehensive
 180 and consistent analysis across different policies.

182 3 BUDGET AWARENESS



191 Figure 2: At each interaction round, the agent is provided with its current and remaining budget
 192 through the budget tracker before generating the next thinking step and the tool call actions.

194 3.1 BUDGET TRACKER

196 Each tool invocation introduces non-negligible costs in context and computation. We hypothesize
 197 that providing explicit budget signals enables the model to internalize resource constraints and adapt
 198 its strategy without requiring additional training. We introduce **Budget Tracker**, a lightweight,
 199 plug-and-play module that surfaces real-time budget states inside the agent’s reasoning loop. In
 200 the beginning iteration, the tracker provides a brief policy guideline describing the budget regimes
 201 and corresponding tool-use recommendations (see Appendix F). At each subsequent iteration, this
 202 tracker appends a budget status block showing the remaining and used budgets for each available
 203 tool (see Appendix A). Despite its simplicity, Budget Tracker operates purely at the prompt level and
 204 makes the agent explicitly aware of its resource consumption and remaining budget, enabling it to
 205 condition subsequent reasoning steps on the updated resource state.

207 3.2 SCALING WITH BUDGET AWARENESS

208 To better understand how agent behaviors scale under different budget constraints, we study two
 209 mainstream test-time scaling paradigms: *sequential scaling* and *parallel scaling*. In *sequential scaling*,
 210 we adopt the budget-forcing strategy (Muennighoff et al., 2025) by appending the following
 211 message when the agent proposes an answer: “Wait, you still have remaining tool budget, use more
 212 search and browse tools to explore different information sources before concluding.” This encour-
 213 ages the model to reevaluate its reasoning and make better use of its available tool calls. In *parallel*
 214 *scaling*, we fix the per-run budget and conduct multiple independent runs in parallel. We report
 215 Majority Vote, Best-of-N and Pass@N results, following common test-time scaling practice (Wei
 et al., 2025). More details can be found in Appendix E.2.

216 3.3 RESULTS AND ANALYSIS
217

218 We evaluate Budget Tracker on three information-seeking QA datasets requiring external search:
 219 BrowzComp (Wei et al., 2025), BrowzComp-ZH (Zhou et al., 2025) and HLE-Search (Phan et al.,
 220 2025; Han et al., 2025a). Dataset details and full experimental setup are provided in Section B.
 221 Building on the ReAct framework, we incorporate the Budget Tracker immediately after each tool
 222 response to inform the agent of its remaining budget. Each tool is assigned a budget of 100, and the
 223 reasoning process stops when any budget is exhausted or the agent reaches a final answer.

224 Table 1: Effect of Budget Tracker on tool-use behavior and performance. Adding the tracker con-
 225 sistently improves answer accuracy across models and datasets. Results are averaged across 3 runs.
 226

Model	Method	BrowzComp	BrowzComp-ZH	HLE-Search
Gemini-2.5-Pro	ReAct + Budget Tracker	12.6 14.6	31.5 32.9	20.5 21.8
Gemini-2.5-Flash	ReAct + Budget Tracker	9.7 10.7	26.5 28.7	14.7 17.3
Claude-Sonnet-4	ReAct + Budget Tracker	12.2 14.0	29.1 31.1	20.5 23.0

236 **Budget Tracker enables higher performance under the same budget.** Table 1 compares ReAct
 237 with and without the Budget Tracker under identical budget limits. Across different models, adding
 238 the tracker consistently improves accuracy on all the datasets. This demonstrates that making budget
 239 signals explicit encourages more strategic and positive agent tool-use behavior.

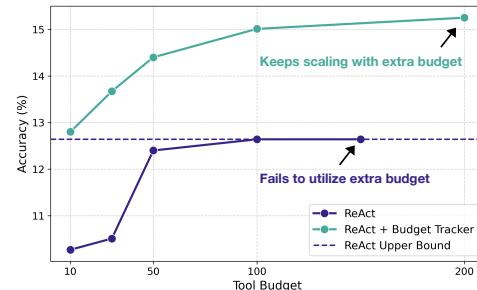
240 **Budget Tracker achieves lower resource us-
 241 age and cost under similar accuracy.** In Ta-
 242 ble 2, the consumed resources report the av-
 243 erage number of search and browse tool calls,
 244 as well as the unified cost combining tool and
 245 token consumption per question. Increasing
 246 the tool-use budget in ReAct improves per-
 247 formance but also raises both tool call frequency
 248 and total cost. In contrast, Budget Tracker
 249 achieves comparable accuracy with less budget
 250 (10 vs. 100), while using 40.4% fewer search
 251 calls, 21.4% fewer browse calls, and reduc-
 252 ing overall cost by 31.3%. This demonstrates
 253 that explicit budget awareness enables more ef-
 254 ficient tool use without sacrificing accuracy.

255 **Budget Tracker facilitates continued scaling
 256 through effective resource utilization by sim-
 257 plly adding budget awareness.** Figure 3 illus-

258 trates Gemini-2.5-Pro’s performance on BrowzComp as tool budget increases. Lacking budget
 259 awareness, the standard ReAct baseline saturates at a budget of 100. Beyond this point, it fails to
 260 utilize any additionally allocated budget, even though the context window is not filled up. This ceil-
 261 ing occurs because the agent’s reasoning process terminates prematurely: it either believes it has
 262 found a sufficient answer or concludes it is stuck and gives up, unaware of the unused resources.
 263 In contrast, Budget Tracker explicitly informs the model of its remaining resources, enabling it to
 264 leverage larger budgets and continue scaling its performance.

265 3.4 ANALYSIS ON TEST-TIME SCALING STRATEGIES
266

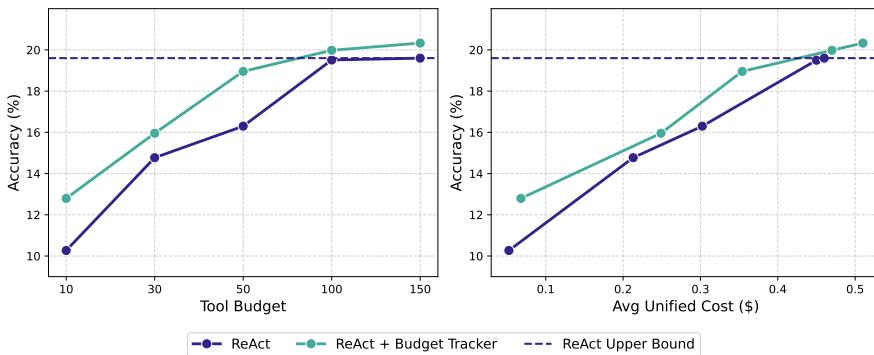
267 Across different test-time scaling strategies, **Budget Tracker consistently pushes the Pareto fron-
 268 tier and demonstrates robust effectiveness.** Sequential scaling encourages the agent to issue more
 269 tool calls by prompting it to continue reasoning and exploring additional information sources. The
 prompting stops once the agent refuses to invoke any tools for five iterations. As shown in Figure 4,



268 Figure 3: ReAct saturates and fails to utilize
 269 additional tool budget, reaching a performance ceiling.
 In contrast, ReAct + Budget Tracker continues
 to scale effectively with larger budgets, achieving
 consistent accuracy improvements.

270
271 Table 2: Performance and resource consumption comparison under different budgets using Gemini-
272 2.5-Pro. With $10\times$ less budget (10 vs. 100), Budget Tracker achieves comparable accuracy while
273 requiring fewer tool calls and lower total cost, highlighting the benefits of explicit budget awareness.

274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323	Methods	Budget	Acc (%)	Consumed resources		
				# search	# browse	Unified cost (€)
ReAct	ReAct	10	10.3	7.87	0.60	5.2
		30	10.5	13.77	1.33	8.0
		100	12.6	14.24	1.36	9.9
ReAct + Budget Tracker		10	12.8	8.48	1.09	6.8



294 Figure 4: Comparison of Budget Tracker and ReAct in sequential scaling using Gemini-2.5-Pro.
295 With explicit budget awareness, Budget Tracker consistently improves upon ReAct at equal budgets.
296 ReAct plateaus early as it cannot utilize extra resources, while Budget Tracker adapts its spending
297 to gain further improvements and extend the cost–performance frontier.

300 while this strategy enables ReAct to further improve accuracy, it eventually encounters a performance ceiling, where the agent confidently concludes that no further tool calls are necessary. In contrast, Budget Tracker achieves a consistently better scaling curve and sustains performance gains beyond ReAct’s plateau. Furthermore, it advances the Pareto frontier of the cost–performance curve, demonstrating not only more effective performance scaling but also more efficient tool utilization.

306 For parallel scaling, we set a fixed budget of 50 per run for Gemini-2.5-Pro, as empirical observations
307 indicate that only 0.8% of the data in BrowseComp requires exceeding this limit (see details in
308 Appendix E.1). In Figure 5, we use identical sample sizes for both settings and report Majority
309 Vote and Best-of-N results (Pass@N results are provided in Appendix E.2). On the right, the figure
310 illustrates the cost–performance scaling, where Budget Tracker consistently yields a superior curve.

4 BATS: BUDGET-AWARE AGENT TEST-TIME SCALING

314 Given the demonstrated effectiveness of budget awareness, we now explore how it enhances agent
315 orchestration and influences key behaviors such as planning and self-verification. In this section,
316 we propose **BATS**, a Budget-Aware Test-time Scaling framework for tool-augmented agents under
317 explicit budget. As shown in Figure 6, given an information-seeking question and a tool call budget,
318 BATS begins with internal reasoning to formulate a structured action plan and decide which tools
319 to invoke. Tool calls are triggered through specific tokens, and the returned tool responses are
320 appended to the reasoning sequence, expanding the context with new evidence. When the agent
321 proposes a candidate answer, the verification module checks its validity and decides whether to
322 continue the current sequence or initiate a new attempt with the remaining budget. The iterative
323 process terminates once any budgeted resource is exhausted. Finally, an LLM-as-a-judge selects the
best answer across all verified answers. The full prompts of BATS are provided in Appendix F.

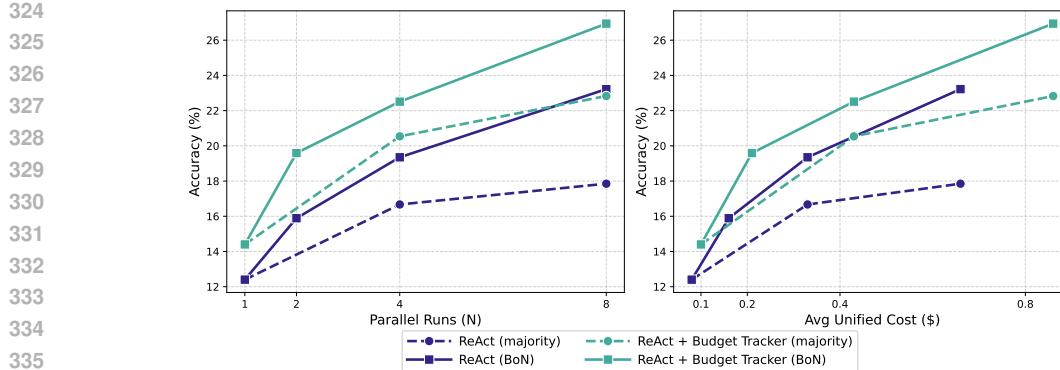


Figure 5: Comparison of Budget Tracker and ReAct in parallel scaling using Gemini-2.5-Pro. The left subfigure shows accuracy scaling with increasing parallel runs, while the right subfigure illustrates the corresponding cost–performance trend.

The central design principle of BATS is *budget awareness*. Throughout the execution, the Budget Tracker continuously updates both resource usage and remaining budget at every iteration. This persistent awareness shapes the agent’s planning, tool-use strategy, and verification behavior, enabling adaptive and efficient use of constrained resources. We next describe how each module in BATS incorporates budget awareness.

4.1 BUDGET-AWARE PLANNING

Planning in BATS incorporates both *constraint decomposition* and *structured dynamic planning*. Selecting an appropriate starting point is critical: a well-chosen entry narrows the search space and conserves budget, while a poor choice can quickly exhaust the budget (see example in Appendix G.1). To address this, we prompt the agent to first perform **constraint decomposition** and to categorize the clues implied in the question into two types: (1) *exploration*, which expand the candidate space, and (2) *verification*, which validate specific properties. While a verification clue can sometimes provide a direct shortcut to the answer, relying on it prematurely is risky, as it may consume resources without guaranteeing progress.

The agent is further instructed to generate and maintain an explicit plan throughout execution. This **tree-structured plan** acts as a dynamic checklist, recording step status, resource usage, and allocation, while guiding future actions. It is never overwritten: completed, failed, or partial steps remain recorded to prevent redundant tool calls and to maintain a full execution trace. We provide the full prompt of planning module in Appendix F.2. As shown in the planning block from Figure 6, a single step in the plan represents a subtask that requires multiple tool calls to complete, for instance, getting a candidate list. After every iteration, new information may create additional branches, resolve pending steps, or invalidate unproductive paths. The planning module adjusts exploration breadth and verification depth based on the current remaining budget.

This integration of constraint decomposition, persistent structured planning, and continuous budget-conditioned updating allows BATS to maintain a controlled and interpretable search process while efficiently allocating available tool calls across exploration and verification subtasks.

4.2 BUDGET-AWARE SELF-VERIFICATION

Once the agent proposes an answer, the self-verification module re-evaluates the reasoning trajectory and corresponding resource usage. The full verifier prompt is provided in Appendix F.3. This process begins with a constraint-by-constraint backward check. Using the previously extracted exploration and verification clues, the module assesses each constraint and determines whether it has been satisfied, contradicted, or remains unverifiable. This detailed check grounds the proposed answer directly against the question’s requirements and clues.

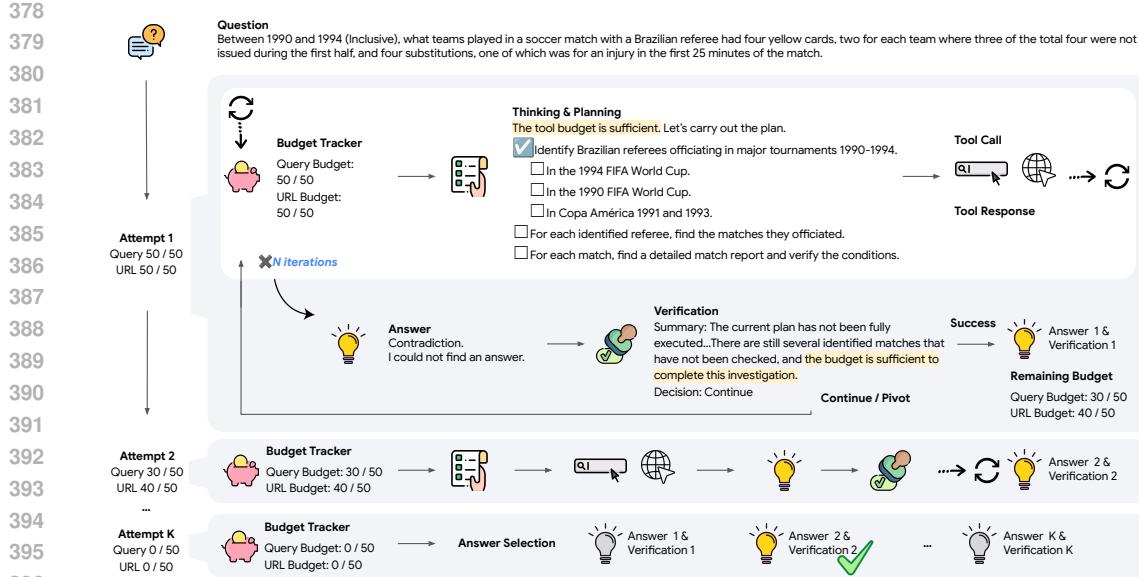


Figure 6: Overview of the BATS framework. Given a question and per-tool budgets, the agent begins with budget-aware thinking and planning, structured as a checklist. The agent keeps iterating by reasoning over new information and updated budgets. When an answer is proposed, BATS performs verification and decides to either continue, pivot, or initiate a new attempt with the remaining budget. BATS terminates when any of the budgets are exhausted.

Based on this analysis, the module then makes a verification decision based on the above assessment and budget status. If all constraints are satisfied, the answer is marked as a *SUCCESS*. If several constraints remain unverifiable but the trajectory appears promising, provided the budget is sufficient for deeper exploration, the outcome is to *CONTINUE* exploration. In contrast, if contradictions are identified or the remaining budget cannot support further investigation towards this lead, the agent is expected to terminate expensive or low-yield directions early and *PIVOT* toward a different direction to avoid wasting tool call resources while resources are still sufficient to pursue alternatives. When the decision is to *CONTINUE* or *PIVOT*, the module also generates a concise summary that replaces the raw trajectory in context. This includes key reasoning steps, intermediate findings, failure causes, and suggestions for optimization to avoid redundant exploration. By compressing the reasoning trajectory into a compact and informative summary, the verifier reduces context length while ensuring that subsequent attempts remain grounded in previously acquired information.

Together, structured constraint checking, explicit decision rules, and budget-aware trajectory summarization allow BATS to terminate useless trajectories early, continue promising ones efficiently, and maintain reliable progress toward the correct answer within strict budget constraints.

5 EXPERIMENTS AND RESULTS

We use three challenging information-seeking benchmarks: BrowseComp (Wei et al., 2025), BrowseComp-ZH (Zhou et al., 2025) and HLE-Search (Han et al., 2025a). More experiment details can be found in Appendix B.

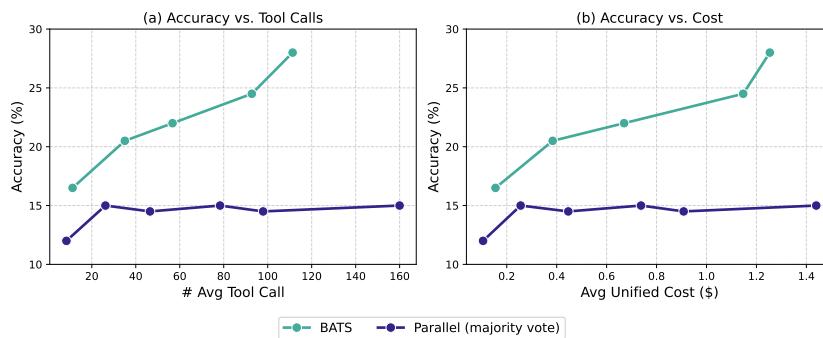
5.1 MAIN RESULTS

Table 3 shows the performance comparison across different web search agents. Under the strict budget constraints of 100 tool uses for BrowseComp, BATS consistently achieves better results than baselines, obtaining 24.6% on BrowseComp, 46.0% on BrowseComp-ZH and 27.0% on HLE-Search using Gemini-2.5-Pro. Crucially, while many comparing agents rely on extensive task-specific training to boost performance, BATS is entirely training-free. This result highlights the ef-

432
 433 Table 3: Performance comparison across web search agents. We denote results from our own ex-
 434 periments with *; other baseline scores are cited from their respective publications. The “Training”
 435 column specifies whether the model has been specifically trained on agentic web search tasks. For
 436 our budget-constrained setting, each agent is provided a budget of 100 tool uses per tool.

437	Method	Training	BrowseComp	BrowseComp-ZH	HLE-Search
438	<i>Model Only</i>				
439	GPT-4o	✗	0.6	6.2	-
440	Claude-3.7-Sonnet	✗	2.3	11.8	-
441	Gemini-2.5-Flash*	✗	2.7	23.9	2.8
442	Gemini-2.5-Pro*	✗	6.3	27.8	8.6
443	OpenAI o1	✗	9.9	29.1	-
444	<i>Training-based Agents</i>				
445	ASearcher	✓	5.2	15.6	-
446	WebSailor	✓	12.0	30.1	-
447	DeepDive	✓	14.8	25.6	-
448	WebExplorer	✓	15.7	32.0	-
449	OpenAI Deep Research	✓	51.5	42.9	29.1
450	<i>Budget-constrained</i>				
451	Claude-Sonnet-4	ReAct	✗	12.2	29.1
452	Claude-Sonnet-4	BATS (Ours)	✗	19.1	41.5
453	Gemini-2.5-Flash	ReAct	✗	9.7	26.5
454	Gemini-2.5-Flash	BATS (Ours)	✗	14.3	34.3
455	Gemini-2.5-Pro	ReAct	✗	12.6	31.5
456	Gemini-2.5-Pro	BATS (Ours)	✗	24.6	46.0
457					

458 effectiveness of our budget-aware design, which maximizes the efficiency of every tool call to achieve
 459 superior results in resource-constrained settings without the need for additional fine-tuning.
 460



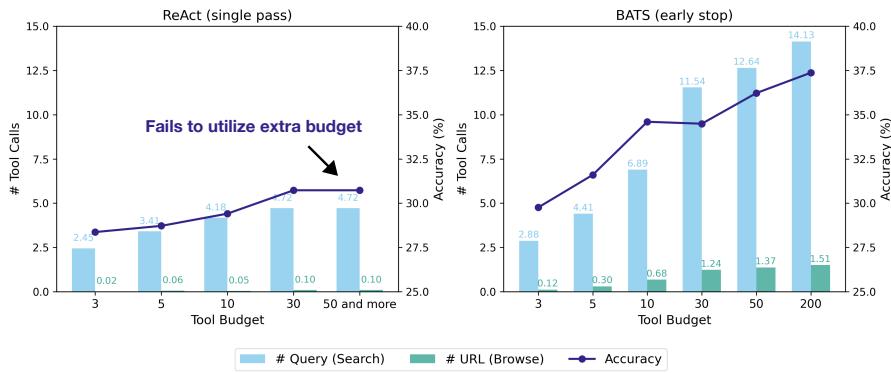
473 Figure 7: Scaling behaviors along (a) total number of tool calls and (b) average unified cost, eval-
 474 uated on a 200-example subset of BrowseComp using Gemini-2.5-Pro.
 475

476 **BATS achieves higher performance under the same budget constraint.** To better understand
 477 the scaling behavior, we vary the tool-call budget and evaluate performance on a random subset of
 478 200 examples from BrowseComp for a manageable analysis. Figure 7 (left) shows the accuracy
 479 against the average number of tool calls, including both search and browse. Across all budget levels,
 480 BATS consistently outperforms the parallel majority-vote baseline, demonstrating that budget-aware
 481 adaptation leads to more effective use of limited tool calls.

482 **BATS achieves higher performance when accounting for unified costs.** Beyond tool-call counts,
 483 we measure performance under a unified cost metric that incorporates both token usage and tool-call
 484 expenses. As shown in Figure 7 (right), BATS achieves more favorable scaling curves, delivering
 485 higher accuracy at comparable or lower costs. This indicates that BATS not only improves effec-
 486 tiveness under budget constraints but also yields better cost–performance trade-offs.

486 5.2 EARLY STOPPING
487

488 We analyze how effectively agents can solve questions under budget constraints when allowed to
489 stop early, without exhausting all tool calls. This setting reflects realistic scenarios where efficient,
490 confident answers are preferable to prolonged exploration. To isolate this behavior, we focus on the
491 agent’s *first attempt* only. For BATS, this is the first answer that successfully passes self-verification.
492 For baselines that lack verification, we use the output from a single generation pass as their first
493 attempt. If any tool budget is exhausted before this first attempt completes, the agent will stop
494 and output a final answer based on its progress so far. This evaluation captures both efficiency and
495 robustness, testing whether agents can solve questions quickly without using their full budgets.
496



507
508 Figure 8: Performance and resource utilization under early stopping on BrowseComp-ZH using
509 Gemini-2.5-Pro. The bars represent the average number of Search and Browse tool calls (left axis),
510 while the line plots indicate accuracy (right axis). While ReAct plateaus and fails to utilize additional
511 budget, BATS effectively scales up tool usage to achieve higher accuracy as the budget increases.
512
513

514 **Budget awareness enables BATS to scale effectively with increased resources.** Figure 8 presents
515 the early stopping performance on BrowseComp-ZH using Gemini-2.5-Pro. The x-axis denotes the
516 predefined tool call budget for both search and browse tool calls. Bars show the average number
517 of tool calls actually used, while the line plot reports the resulting accuracy. The ReAct baseline
518 (Figure 8 left) exhibits poor budget utilization: it consistently underuses the browse tool (fewer than
519 0.1 calls on average) and reaches a performance plateau early, with accuracy capped at 30.7% for all
520 budgets of 30 and above. This behavior highlights its lack of budget awareness and its inability to
521 benefit from additional resources. In contrast, BATS (Figure 8 right) shows strong budget-adaptive
522 behavior. As the budget increases, it strategically raises its use of both search and browse tools,
523 leading to steady gains in accuracy, rising from 29.8% (budget=3) to 37.4% (budget=200). No-
524 tably, with a budget of just 5, BATS already surpasses the baseline’s best achievable accuracy. This
525 demonstrates BATS’s ability to make more strategic and cost-effective decisions, delivering higher
526 accuracy even with the same or fewer resources. We provide ablation studies in Appendix C. More
527 analysis can be found in Appendix E.4.
528
529

530 6 CONCLUSION
531

532 In this work, we present the first systematic study of budget-constrained tool-use agents and their
533 test-time scaling behaviors. We identify that standard agents often hit a performance ceiling due to
534 a lack of resource awareness. To overcome this, we introduce the **Budget Tracker**, a lightweight
535 plug-in that provides budget awareness, and **BATS**, a comprehensive framework that dynamically
536 adapts planning and verification based on real-time resource status. Extensive experiments across
537 multiple models and information-seeking benchmarks show that budget awareness enables stronger
538 agent scaling and consistently pushes the cost-performance Pareto frontier. By formalizing tool-call
539 budgets as a critical scaling dimension, our work offers a principled foundation for building efficient
and adaptive agents.

540 THE USE OF LARGE LANGUAGE MODELS (LLMs)
541542 We acknowledge the use of LLMs (ChatGPT and Gemini) exclusively for editing the text to correct
543 grammatical errors and improve clarity and flow. All core scientific content and research ideas were
544 authored solely by the human authors.
545546 REPRODUCIBILITY STATEMENT
547548 We conduct evaluations exclusively on publicly available benchmarks and query LLMs through
549 public providers. All experimental settings, including temperature, context length, tool calling,
550 configuration parameters, and prompts, are detailed in Appendix B and F.
551552 ETHICS STATEMENT
553554 The authors confirm adherence to the ICLR Code of Ethics. This work aims to reduce the computa-
555 tional and economic costs of LLM agents, contributing positively to accessibility and sustainability.
556 At the same time, we recognize that web search agents may inherit web-based biases or propagate
557 misinformation. Addressing these risks requires our continued attention and responsible research
558 practices within the research community.
559560 REFERENCES
561562 Anthropic. Claude 3.7 sonnet system card. <https://assets.anthropic.com/m/785e231869ea8b3b/original/clause-3-7-sonnet-system-card.pdf>,
563 2025a.564 Anthropic. Claude 4 system card: Claude opus 4 & claude sonnet 4. <https://www-cdn.anthropic.com/6d8a8055020700718b0c49369f60816ba2a7c285.pdf>, 2025b.565 Anthropic. Effective context engineering for ai agents, September 2025c. URL <https://www.anthropic.com/engineering/effective-context-engineering-for-ai-agents>.
566567 Bradley C. A. Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *CoRR*, abs/2407.21787, 2024. doi: 10.48550/ARXIV.2407.21787. URL <https://doi.org/10.48550/arXiv.2407.21787>.
568569 Jianhao Chen, Zishuo Xun, Bocheng Zhou, Han Qi, Qiaosheng Zhang, Yang Chen, Wei Hu, Yuzhong Qu, Wanli Ouyang, and Shuyue Hu. Do we truly need so many samples? multi-llm repeated sampling efficiently scales test-time compute. *CoRR*, abs/2504.00762, 2025a. doi: 10.48550/ARXIV.2504.00762. URL <https://doi.org/10.48550/arXiv.2504.00762>.
570571 Jiefeng Chen, Jie Ren, Xinyun Chen, Chengrun Yang, Ruoxi Sun, and Sercan Ö. Arik. SETS: leveraging self-verification and self-correction for improved test-time scaling. *CoRR*, abs/2501.19306, 2025b. doi: 10.48550/ARXIV.2501.19306. URL <https://doi.org/10.48550/arXiv.2501.19306>.
572573 Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, Sahel Sharifmoghaddam, Yanxi Li, Haoran Hong, Xinyu Shi, Xuye Liu, Nandan Thakur, Crystina Zhang, Luyu Gao, Wenhui Chen, and Jimmy Lin. Browsecomp-plus: A more fair and transparent evaluation benchmark of deep-research agent. *CoRR*, abs/2508.06600, 2025c. doi: 10.48550/ARXIV.2508.06600. URL <https://doi.org/10.48550/arXiv.2508.06600>.
574575 Gheorghe Comanici, Eric Bieber, Mike Schaeckermann, Ice Pasupat, Noveen Sachdeva, Inderjit S. Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marrs, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szekely, Nan-Jiang Jiang, Krishna Haridasan, Ahmed Omran, Nikunj Saunshi, Dara Bahri, Gaurav
576

594 Mishra, Eric Chu, Toby Boyd, Brad Hekman, Aaron Parisi, Chaoyi Zhang, Kornraphop Kaw-
 595 intiranon, Tania Bedrax-Weiss, Oliver Wang, Ya Xu, Ollie Purkiss, Uri Mendlovic, Ilai Deu-
 596 tel, Nam Nguyen, Adam Langley, Flip Korn, Lucia Rossazza, Alexandre Ramé, Sagar Wagh-
 597 mare, Helen Miller, Nathan Byrd, Ashrith Sheshan, Raia Hadsell Sangnie Bhardwaj, Paweł
 598 Janus, Tero Rissa, Dan Horgan, Sharon Silver, Ayzaan Wahid, Sergey Brin, Yves Raimond,
 599 Klemen Kloboves, Cindy Wang, Nitesh Bharadwaj Gundavarapu, Ilia Shumailov, Bo Wang,
 600 Mantas Pajarskas, Joe Heyward, Martin Nikoltchev, Maciej Kula, Hao Zhou, Zachary Garrett,
 601 Sushant Kafle, Sercan Arik, Ankita Goel, Mingyao Yang, Jiho Park, Koji Kojima, Parsa Mah-
 602 moudieh, Koray Kavukcuoglu, Grace Chen, Doug Fritz, Anton Bulyenov, Sudeshna Roy, Dim-
 603 itris Paparas, Hadar Shemtov, Bo-Juen Chen, Robin Strudel, David Reitter, Aurko Roy, An-
 604 drey Vlasov, Changwan Ryu, Chas Leichner, Haichuan Yang, Zelda Mariet, Denis Vnukov,
 605 Tim Sohn, Amy Stuart, Wei Liang, Minmin Chen, Praynaa Rawlani, Christy Koh, JD Co-
 606 Reyes, Guangda Lai, Praseem Banzal, Dimitrios Vytiniotis, Jieru Mei, and Mu Cai. Gemini
 607 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next genera-
 608 tion agentic capabilities. *CoRR*, abs/2507.06261, 2025. doi: 10.48550/ARXIV.2507.06261. URL
 609 <https://doi.org/10.48550/arXiv.2507.06261>.

610 Mehul Damani, Idan Shenfeld, Andi Peng, Andreea Bobu, and Jacob Andreas. Learning how hard to
 611 think: Input-adaptive allocation of LM computation. In *The Thirteenth International Conference*
 612 *on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.
 613 URL <https://openreview.net/forum?id=6qUUGw9bAZ>.

614 Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu.
 615 Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous RL.
 616 *CoRR*, abs/2508.07976, 2025. doi: 10.48550/ARXIV.2508.07976. URL <https://doi.org/10.48550/arXiv.2508.07976>.

617 Google. Gemini deep research — your personal research assistant. <https://gemini.google/overview/deep-research/>, 2025.

618 Rujun Han, Yanfei Chen, Zoey CuiZhu, Lesly Miculicich, Guan Sun, Yuanjun Bi, Weiming Wen,
 619 Hui Wan, Chunfeng Wen, Solène Maître, George Lee, Vishy Tirumalashetty, Emily Xue, Zizhao
 620 Zhang, Salem Haykal, Burak Gokturk, Tomas Pfister, and Chen-Yu Lee. Deep researcher with
 621 test-time diffusion. *CoRR*, abs/2507.16075, 2025a. doi: 10.48550/ARXIV.2507.16075. URL
 622 <https://doi.org/10.48550/arXiv.2507.16075>.

623 Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-
 624 budget-aware LLM reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mo-
 625 hammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics, ACL*
 626 2025, Vienna, Austria, July 27 - August 1, 2025, pp. 24842–24855. Association for Compu-
 627 tational Linguistics, 2025b. URL <https://aclanthology.org/2025.findings-acl.1274/>.

628 Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Os-
 629 trow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Madry, Alex Baker-Whitcomb,
 630 Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino,
 631 Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Ka-
 632 mali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoonchian, Ananya
 633 Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codis-
 634 poti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, An-
 635 gela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano,
 636 Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben
 637 Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby
 638 Spero, Bogo Giertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian
 639 Guaraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll L. Wainwright,
 640 Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Con-
 641 ger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont,
 642 Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey,
 643 Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei,
 644 Constantin Koumouzelis, and Dane Sherburn. Gpt-4o system card. *CoRR*, abs/2410.21276, 2024.

648 doi: 10.48550/ARXIV.2410.21276. URL <https://doi.org/10.48550/arXiv.2410.21276>.

649

650

651 Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec
 652 Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard
 653 Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett,
 654 Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrei
 655 Mishchenko, Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghor-
 656 bani, Ben Rossen, Benjamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao
 657 Hao, Bowen Baker, Brandon Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lu-
 658 garesi, Cary Bassin, Cary Hudson, Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen,
 659 Chong Zhang, Chris Koch, Chris Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan
 660 Roberts, Daniel Kappler, Daniel Levy, Daniel Selsam, David Dohan, David Farhi, David Mely,
 661 David Robinson, Dimitris Tsipras, Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Ed-
 662 mund Wong, Elizabeth Proehl, Enoch Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan
 663 Mays, Fan Wang, Felipe Petroski Such, Filippo Raspa, Florencia Leoni, Foivos Tsimpourlas,
 664 Francis Song, Fred von Lohmann, Freddie Sulit, Geoff Salmon, Giambattista Parascandolo,
 665 Gildas Chabot, Grace Zhao, Greg Brockman, Guillaume Leclerc, Hadi Salman, Haiming Bao,
 666 Hao Sheng, Hart Andrin, Hessam Bagherinezhad, Hongyu Ren, Hunter Lightman, Hyung Won
 667 Chung, Ian Kivlichan, Ian O'Connell, Ian Osband, Ignasi Clavera Gilaberte, and Ilge Akkaya.
 668 Openai o1 system card. *CoRR*, abs/2412.16720, 2024. doi: 10.48550/ARXIV.2412.16720. URL
 669 <https://doi.org/10.48550/arXiv.2412.16720>.

670

671

672 Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. Search-
 673 r1: Training llms to reason and leverage search engines with reinforcement learning. *CoRR*,
 674 abs/2503.09516, 2025. doi: 10.48550/ARXIV.2503.09516. URL <https://doi.org/10.48550/arXiv.2503.09516>.

675

676 Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Hufeng Yin,
 677 Jialong Wu, Jingren Zhou, Kuan Li, Liangcai Su, Litu Ou, Liwen Zhang, Pengjun Xie, Rui
 678 Ye, Wenbiao Yin, Xinmiao Yu, Xinyu Wang, Xixi Wu, Xuanzhong Chen, Yida Zhao, Zhen
 679 Zhang, Zhengwei Tao, Zhongwang Zhang, Zile Qiao, Chenxi Wang, Donglei Yu, Gang Fu,
 680 Haiyang Shen, Jiayin Yang, Jun Lin, Junkai Zhang, Kui Zeng, Li Yang, Hailong Yin, Maojia
 681 Song, Ming Yan, Peng Xia, Qian Xiao, Rui Min, Ruixue Ding, Runnan Fang, Shaowei Chen,
 682 Shen Huang, Shihang Wang, Shihao Cai, Weizhou Shen, Xiaobin Wang, Xin Guan, Xinyu
 683 Geng, Yingcheng Shi, Yuning Wu, Zhuo Chen, Zijian Li, and Yong Jiang. Tongyi deepre-
 684 search technical report. *CoRR*, abs/2510.24701, 2025a. doi: 10.48550/ARXIV.2510.24701. URL
 685 <https://doi.org/10.48550/arXiv.2510.24701>.

686

687 Kuan Li, Zhongwang Zhang, Hufeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baix-
 688 uan Li, Zhengwei Tao, Xinyu Wang, Weizhou Shen, Junkai Zhang, Dingchu Zhang, Xixi Wu,
 689 Yong Jiang, Ming Yan, Pengjun Xie, Fei Huang, and Jingren Zhou. Websailor: Navigating super-
 690 human reasoning for web agent. *CoRR*, abs/2507.02592, 2025b. doi: 10.48550/ARXIV.2507.
 691 02592. URL <https://doi.org/10.48550/arXiv.2507.02592>.

692

693 Weizhen Li, Jianbo Lin, Zhuosong Jiang, Jingyi Cao, Xinpeng Liu, Jiayu Zhang, Zhenqiang Huang,
 694 Qianben Chen, Weichen Sun, Qiexiang Wang, Hongxuan Lu, Tianrui Qin, Chenghao Zhu, Yi Yao,
 695 Shuying Fan, Xiaowan Li, Tiannan Wang, Pai Liu, King Zhu, He Zhu, Dingfeng Shi, Piaohong
 696 Wang, Yeyi Guan, Xiangru Tang, Minghao Liu, Yuchen Eleanor Jiang, Jian Yang, Jiaheng Liu,
 697 Ge Zhang, and Wangchunshu Zhou. Chain-of-agents: End-to-end agent foundation models via
 698 multi-agent distillation and agentic RL. *CoRR*, abs/2508.13167, 2025c. doi: 10.48550/ARXIV.
 699 2508.13167. URL <https://doi.org/10.48550/arXiv.2508.13167>.

700

701 Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang,
 702 and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. *CoRR*,
 703 abs/2501.05366, 2025d. doi: 10.48550/ARXIV.2501.05366. URL <https://doi.org/10.48550/arXiv.2501.05366>.

704

705 Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Xinglin Wang, Bin Sun, Heda Wang, and
 706 Kan Li. Escape sky-high cost: Early-stopping self-consistency for multi-step reasoning. In

702 *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net, 2024. URL <https://openreview.net/forum?id=ndR8Ytrzhh>.*

703

704

705 Junteng Liu, Yunji Li, Chi Zhang, Jingyang Li, Aili Chen, Ke Ji, Weiyu Cheng, Zijia Wu,
706 Chengyu Du, Qidi Xu, Jiayuan Song, Zhengmao Zhu, Wenhui Chen, Pengyu Zhao, and Junxian He. Webexplorer: Explore and evolve for training long-horizon web agents. 2025a. URL
707 <https://api.semanticscholar.org/CorpusID:281204359>.

708

709

710 Licheng Liu, Zihan Wang, Linjie Li, Chenwei Xu, Yiping Lu, Han Liu, Avirup Sil, and Manling
711 Li. A simple "try again" can elicit multi-turn LLM reasoning. *CoRR*, abs/2507.14295, 2025b.
712 doi: 10.48550/ARXIV.2507.14295. URL <https://doi.org/10.48550/arXiv.2507.14295>.

713

714 Rui Lu, Zhenyu Hou, Zihan Wang, Hanchen Zhang, Xiao Liu, Yujiang Li, Shi Feng, Jie Tang, and
715 Yuxiao Dong. Deepdive: Advancing deep search agents with knowledge graphs and multi-turn
716 rl, 2025. URL <https://arxiv.org/abs/2509.10446>.

717

718 Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegr
719 effe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bod
720 hisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and
721 Peter Clark. Self-refine: Iterative refinement with self-feedback. In Alice Oh, Tris
722 stan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Ad
723 vances in Neural Information Processing Systems 36: Annual Conference on Neural Infor
724 mation Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16,
725 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

726

727 Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi,
728 Luke Zettlemoyer, Percy Liang, Emmanuel J. Candès, and Tatsunori Hashimoto. s1: Simple test
729 time scaling. *CoRR*, abs/2501.19393, 2025. doi: 10.48550/ARXIV.2501.19393. URL <https://doi.org/10.48550/arXiv.2501.19393>.

730

731 Sania Nayab, Giulio Rossolini, Giorgio C. Buttazzo, Nicolamaria Manes, and Fabrizio Giacomelli.
732 Concise thoughts: Impact of output length on LLM reasoning and cost. *CoRR*, abs/2407.19825,
733 2024. doi: 10.48550/ARXIV.2407.19825. URL <https://doi.org/10.48550/arXiv.2407.19825>.

734

735 OpenAI. Introducing deep research. <https://openai.com/index/introducing-deep-research/>, 2025.

736

737

738 Xianghe Pang, Shuo Tang, Rui Ye, Yuwen Du, Yixin Du, and Siheng Chen. Browsemaster: Towards
739 scalable web browsing via tool-augmented programmatic agent pair. *CoRR*, abs/2508.09129,
740 2025. doi: 10.48550/ARXIV.2508.09129. URL <https://doi.org/10.48550/arXiv.2508.09129>.

741

742 Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Sean Shi, Michael
743 Choi, Anish Agrawal, Arnav Chopra, Adam Khoja, Ryan Kim, Jason Hausenloy, Oliver Zhang,
744 Mantas Mazeika, Daron Anderson, Tung Nguyen, Mobeen Mahmood, Fiona Feng, Steven Y.
745 Feng, Haoran Zhao, Michael Yu, Varun Gangal, Chelsea Zou, Zihan Wang, Jessica P. Wang,
746 Pawan Kumar, Oleksandr Pokutnyi, Robert Gerbicz, Serguei Popov, John-Clark Levin, Mstyslav
747 Kazakov, Johannes Schmitt, Geoff Galgon, Alvaro Sanchez, Yongki Lee, Will Yeadon, Scott
748 Sauers, Marc Roth, Chidozie Agu, Søren Riis, Fabian Giska, Saitheja Utpala, Zachary Giboney,
749 Gashaw M. Goshu, Joan of Arc Xavier, Sarah-Jane Crowson, Mohinder Maheshbhai Naiya, Noah
750 Burns, Lennart Finke, Zerui Cheng, Hyunwoo Park, Francesco Fournier-Facio, John Wydallis,
751 Mark Nandor, Ankit Singh, Tim Gehrunger, Jiaqi Cai, Ben McCarty, Darling Duclosel, Jung
752 bae Nam, Jennifer Zampese, Ryan G. Hoerr, Aras Bacho, Gautier Abou Loume, Abdallah Galal,
753 Hangrui Cao, Alexis C. Garretson, Damien Sileo, Qiuyu Ren, Doru Cojoc, Pavel Arkhipov, Us
754 man Qazi, Lianghui Li, Sumeet Motwani, Christian Schröder de Witt, Edwin Taylor, Johannes
755 Veith, Eric Singer, Taylor D. Hartman, Paolo Rissone, Jaehyeok Jin, Jack Wei Lun Shi, Chris G.
Willcocks, Joshua Robinson, Aleksandar Mikov, Ameya Prabhu, Longke Tang, Xavier Alapont,
Justine Leon Uro, Kevin Zhou, Emily de Oliveira Santos, Andrey Pupasov Maksimov, Edward

756 Vendrow, Kengo Zenitani, Julien Guillo, Yuqi Li, Joshua Vendrow, Vladyslav Kuchkin, and
 757 Ng Ze-An. Humanity's last exam. *CoRR*, abs/2501.14249, 2025. doi: 10.48550/ARXIV.2501.
 758 14249. URL <https://doi.org/10.48550/arXiv.2501.14249>.

759

760 Xiao Pu, Michael Saxon, Wenyue Hua, and William Yang Wang. THOUGHTTERMINA-
 761 TOR: benchmarking, calibrating, and mitigating overthinking in reasoning models. *CoRR*,
 762 abs/2504.13367, 2025. doi: 10.48550/ARXIV.2504.13367. URL <https://doi.org/10.48550/arXiv.2504.13367>.

763

764 Zile Qiao, Guoxin Chen, Xuanzhong Chen, Donglei Yu, Wenbiao Yin, Xinyu Wang, Zhen Zhang,
 765 Baixuan Li, Hufeng Yin, Kuan Li, Rui Min, Minpeng Liao, Yong Jiang, Pengjun Xie, Fei Huang,
 766 and Jingren Zhou. Webresearcher: Unleashing unbounded reasoning capability in long-horizon
 767 agents. 2025. URL <https://api.semanticscholar.org/CorpusID:281325175>.

768

769 Tianrui Qin, Qianben Chen, Sinuo Wang, He Xing, King Zhu, He Zhu, Dingfeng Shi, Xinxin Liu,
 770 Ge Zhang, Jiaheng Liu, Yuchen Eleanor Jiang, Xitong Gao, and Wangchunshu Zhou. Flash-
 771 searcher: Fast and effective web agents via dag-based parallel execution. *CoRR*, abs/2509.25301,
 772 2025. doi: 10.48550/ARXIV.2509.25301. URL <https://doi.org/10.48550/arXiv.2509.25301>.

773

774 Junhong Shen, Hao Bai, Lunjun Zhang, Yifei Zhou, Amirth Setlur, Shengbang Tong, Diego Caples,
 775 Nan Jiang, Tong Zhang, Ameet Talwalkar, and Aviral Kumar. Thinking vs. doing: Agents that
 776 reason by scaling test-time interaction. *CoRR*, abs/2506.07976, 2025. doi: 10.48550/ARXIV.
 777 2506.07976. URL <https://doi.org/10.48550/arXiv.2506.07976>.

778

779 Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute opti-
 780 mally can be more effective than scaling model parameters. *CoRR*, abs/2408.03314, 2024. doi: 10.
 781 48550/ARXIV.2408.03314. URL <https://doi.org/10.48550/arXiv.2408.03314>.

782

783 Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen
 784 Zhang, Xinyu Wang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. Webshaper: Agen-
 785 tically data synthesizing via information-seeking formalization. *CoRR*, abs/2507.15061, 2025.
 786 doi: 10.48550/ARXIV.2507.15061. URL <https://doi.org/10.48550/arXiv.2507.15061>.

787

788 Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen,
 789 Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv*
 790 preprint [arXiv:2507.20534](https://arxiv.org/abs/2507.20534), 2025a.

791

792 MiroMind Team, Song Bai, Lidong Bing, Carson Chen, Guanzheng Chen, Yuntao Chen, Zhe Chen,
 793 Ziyi Chen, Jifeng Dai, Xuan Dong, Wenhan Dou, Yue Deng, Yunjie Fu, Junqi Ge, Chenxia Han,
 794 Tammy Huang, Zhenhang Huang, Jerry Jiao, Shilei Jiang, Tianyu Jiao, Xiaoqi Jian, Lei Lei,
 795 Ruilin Li, Ryan Luo, Tiantong Li, Xiang Lin, Ziyuan Liu, Zhiqi Li, Jie Ni, Qiang Ren, Pax Sun,
 796 Shiqian Su, Chenxin Tao, Bin Wang, Hellen Wang, Haonan Wang, James Wang, Jin Wang, Jojo
 797 Wang, Letian Wang, Shizun Wang, Weizhi Wang, Zixuan Wang, Jinfan Xu, Sen Xing, Chenyu
 798 Yang, Hai Ye, Jiaheng Yu, Yue Yu, Muyan Zhong, Tianchen Zhao, Xizhou Zhu, Yanpeng Zhou,
 799 Yifan Zhang, and Zhi Zhu. Mirothinker: Pushing the performance boundaries of open-source
 800 research agents via model, context, and interactive scaling, 2025b. URL <https://arxiv.org/abs/2511.11793>.

801

802 UncleCode. Crawl4ai: Open-source llm friendly web crawler scraper. <https://github.com/unclecode/crawl4ai>, 2024.

803

804 Guangya Wan, Yuqi Wu, Jie Chen, and Sheng Li. Reasoning aware self-consistency: Leveraging
 805 reasoning paths for efficient LLM sampling. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.),
 806 *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association
 807 for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long
 808 Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pp. 3613–3635. Association for
 809 Computational Linguistics, 2025. doi: 10.18653/V1/2025.NAACL-LONG.184. URL <https://doi.org/10.18653/v1/2025.naacl-long.184>.

810 Fali Wang, Hui Liu, Zhenwei Dai, Jingying Zeng, Zhiwei Zhang, Zongyu Wu, Chen Luo, Zheng
 811 Li, Xianfeng Tang, Qi He, and Suhang Wang. Agentts: Large language model agent for test-
 812 time compute-optimal scaling strategy in complex tasks. *CoRR*, abs/2508.00890, 2025a. doi: 10.
 813 48550/ARXIV.2508.00890. URL <https://doi.org/10.48550/arXiv.2508.00890>.

814 Ningning Wang, Xavier Hu, Pai Liu, He Zhu, Yue Hou, Heyuan Huang, Shengyu Zhang, Jian Yang,
 815 Jiaheng Liu, Ge Zhang, Changwang Zhang, Jun Wang, Yuchen Eleanor Jiang, and Wangchunshu
 816 Zhou. Efficient agents: Building effective agents while reducing cost. *CoRR*, abs/2508.02694,
 817 2025b. doi: 10.48550/ARXIV.2508.02694. URL <https://doi.org/10.48550/arXiv.2508.02694>.

818 Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha
 819 Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language
 820 models. In *The Eleventh International Conference on Learning Representations, ICLR 2023,
 821 Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL <https://openreview.net/forum?id=1PL1NIMMrw>.

822 Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won
 823 Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecmp: A simple yet
 824 challenging benchmark for browsing agents. *CoRR*, abs/2504.12516, 2025. doi: 10.48550/
 825 ARXIV.2504.12516. URL <https://doi.org/10.48550/arXiv.2504.12516>.

826 Sean Welleck, Amanda Bertsch, Matthew Finlayson, Hailey Schoelkopf, Alex Xie, Graham Neu-
 827 big, Ilia Kulikov, and Zaïd Harchaoui. From decoding to meta-generation: Inference-time al-
 828 gorithms for large language models. *Trans. Mach. Learn. Res.*, 2024, 2024. URL <https://openreview.net/forum?id=eskQMCIbMS>.

829 Ryan Wong, Jiawei Wang, Junjie Zhao, Li Chen, Yan Gao, Long Zhang, Xuan Zhou, Zuo Wang,
 830 Kai Xiang, Ge Zhang, Wenhao Huang, Yang Wang, and Ke Wang. Widesearch: Benchmarking
 831 agentic broad info-seeking. *CoRR*, abs/2508.07999, 2025. doi: 10.48550/ARXIV.2508.07999.
 832 URL <https://doi.org/10.48550/arXiv.2508.07999>.

833 Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang,
 834 Zekun Xi, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. Webdancer: Towards au-
 835 tonomous information seeking agency. *CoRR*, abs/2505.22648, 2025a. doi: 10.48550/ARXIV.
 836 2505.22648. URL <https://doi.org/10.48550/arXiv.2505.22648>.

837 Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws:
 838 An empirical analysis of compute-optimal inference for LLM problem-solving. In *The Thirteenth
 839 International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*.
 840 OpenReview.net, 2025b. URL <https://openreview.net/forum?id=VNckp7JEHn>.

841 Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan
 842 Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International
 843 Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenRe-
 844 view.net, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

845 Rui Ye, Zhongwang Zhang, Kuan Li, Hufeng Yin, Zhengwei Tao, Yida Zhao, Liangcai Su, Li-
 846 wen Zhang, Zile Qiao, Xinyu Wang, Pengjun Xie, Fei Huang, Siheng Chen, Jingren Zhou, and
 847 Yong Jiang. Agentfold: Long-horizon web agents with proactive context management. *CoRR*,
 848 abs/2510.24699, 2025. doi: 10.48550/ARXIV.2510.24699. URL <https://doi.org/10.48550/arXiv.2510.24699>.

849 Howard Yen, Ashwin Paranjape, Mengzhou Xia, Thejas Venkatesh, Jack Hessel, Danqi Chen, and
 850 Yuhao Zhang. Lost in the maze: Overcoming context limitations in long-horizon agentic search.
 851 *CoRR*, abs/2510.18939, 2025. doi: 10.48550/ARXIV.2510.18939. URL <https://doi.org/10.48550/arXiv.2510.18939>.

852 Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Zhihan Guo, Yufei Wang, Irwin
 853 King, Xue Liu, and Chen Ma. What, how, where, and how well? A survey on test-time scaling in
 854 large language models. *CoRR*, abs/2503.24235, 2025a. doi: 10.48550/ARXIV.2503.24235. URL
 855 <https://doi.org/10.48550/arXiv.2503.24235>.

864 Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew C. Yao. Cumulative reasoning with large
 865 language models. *Trans. Mach. Learn. Res.*, 2025, 2025b. URL <https://openreview.net/forum?id=grW15p4eq2>.

866

867 Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling
 868 Jin, Chenxuan Xie, Meng Cao, Yuxin Gu, Sixin Hong, Jing Ren, Jian Chen, Chao Liu, and
 869 Yining Hua. Browsecomp-zh: Benchmarking web browsing ability of large language models
 870 in chinese. *CoRR*, abs/2504.19314, 2025. doi: 10.48550/ARXIV.2504.19314. URL <https://doi.org/10.48550/arXiv.2504.19314>.

871

872

873 He Zhu, Tianrui Qin, King Zhu, Heyuan Huang, Yeyi Guan, Jinxiang Xia, Yi Yao, Hanhao Li,
 874 Ningning Wang, Pai Liu, Tianhao Peng, Xin Gui, Xiaowan Li, Yuhui Liu, Yuchen Eleanor Jiang,
 875 Jun Wang, Changwang Zhang, Xiangru Tang, Ge Zhang, Jian Yang, Minghao Liu, Xitong Gao,
 876 Jiaheng Liu, and Wangchunshu Zhou. Oagents: An empirical study of building effective agents.
 877 *CoRR*, abs/2506.15741, 2025a. doi: 10.48550/ARXIV.2506.15741. URL <https://doi.org/10.48550/arXiv.2506.15741>.

878

879 King Zhu, Hanhao Li, Siwei Wu, Tianshun Xing, Dehua Ma, Xiangru Tang, Minghao Liu,
 880 Jian Yang, Jiaheng Liu, Yuchen Eleanor Jiang, Changwang Zhang, Chenghua Lin, Jun Wang,
 881 Ge Zhang, and Wangchunshu Zhou. Scaling test-time compute for LLM agents. *CoRR*,
 882 abs/2506.12928, 2025b. doi: 10.48550/ARXIV.2506.12928. URL <https://doi.org/10.48550/arXiv.2506.12928>.

883

884

885 A BUDGET TRACKER

886

887

888 Budget Tracker

889

```
<budget>
890 Tool1 Budget Used: ##, Tool1 Budget Remaining: ##
891 Tool2 Budget Used: ##, Tool2 Budget Remaining: ##
892 Make the best use of the available resources.
893 </budget>
```

894

895 B EXPERIMENT DETAILS

896

897 B.1 SETUP

898

899 **Datasets.** To evaluate web search agents, we use three challenging information-seeking benchmarks: BrowsecComp (Wei et al., 2025), a dataset of 1,266 difficult web-browsing questions requiring persistent retrieval; BrowsecComp-ZH (Zhou et al., 2025), a 289-question Chinese benchmark designed to test agents’ performance in region-specific web environments; and HLE-Search (Han et al., 2025a), a curated 200-question subset of Human’s Last Exam (Phan et al., 2025) focusing on queries that explicitly require search rather than pure reasoning.

900

901 **Baselines.** We compare BATS against a range of models and agentic frameworks, including both
 902 general-purpose base models (Hurst et al., 2024; Jaech et al., 2024; Comanici et al., 2025; Anthropic,
 903 2025a;b) and those specifically fine-tuned for agentic search tasks (Li et al., 2025b; Liu et al., 2025a;
 904 Gao et al., 2025; Lu et al., 2025). To evaluate the final answer accuracy, we use Gemini-2.5-Flash
 905 as the judge model and adopt the evaluation prompt from Phan et al. (2025). Baseline results for
 906 HLE-Search are scraped from Han et al. (2025a).

907

908 For scaling methods, we evaluate sequential and parallel scaling approaches applied to the Re-
 909 Act (Yao et al., 2023) baseline. For **sequential scaling**, to encourage the agent to use more tools
 910 during its iterative execution, we follow the approach from textual reasoning (Muennighoff et al.,
 911 2025) and append the instruction that encourages it to rethink and use more tools whenever the agent
 912 provides an answer. This process is repeated until the agent’s tool budget is exhausted, after which
 913 it is prompted to produce the final answer. For **parallel scaling**, we use temperature sampling to
 914 generate diverse reasoning paths. To aggregate the results, by default, we use Gemini-2.5-Flash
 915

918 to select the most common answer via a majority vote (Wang et al., 2023) (see Appendix E.2 for
 919 details). For experiments in BATS, to enforce the budget constraint for each question, we continue
 920 sampling new sequences until the budget is fully consumed. Thus the number of sampled sequences
 921 may differ across questions.
 922

923 B.2 IMPLEMENTATION DETAILS

925 We use Gemini-2.5-Flash, Gemini-2.5-Pro (Comanici et al., 2025) and Claude-Sonnet-4 (Anthropic,
 926 2025b) as the backbone models in our framework. By default, we disable the thinking mode by
 927 setting the thinking budget as 0 for Gemini-2.5-Flash and 1024 for Gemini-2.5-Pro models. The
 928 maximum number of new tokens for generation was set to 65,536 for Gemini models and 64,000
 929 for Claude. We use a temperature of 0.7 during agent execution to encourage exploration, and use
 930 a deterministic temperature of 0.0 for final answer selection and answer evaluation. We use the
 931 Google Custom Search JSON API¹ for search tools, Jina.ai² and Crawl4AI (UncleCode, 2024) for
 932 web browsing.
 933

934 To keep the context length manageable, we employ several simple context management strategies.
 935 For each browse tool call, the fetched webpage is truncated to 150,000 characters before being sent
 936 to the model. At every iteration, we discard tool outputs from previous steps and retain only the
 937 most recent tool response, preventing the context from growing with accumulated tool results. In
 938 BATS’s verification module, we further control context size by periodically replacing the historical
 939 trajectory with summary. Since the agent determines when to activate the verification module, we
 940 perform a check during each invocation: if more than K iterations have passed since the last update
 941 (with $K = 10$ in our experiments), we replace the older reasoning trace with a concise summary
 942 derived from the verification outputs.
 943

944 B.3 RESOURCE COST

945 The cost of tool calls is determined by the pricing of the providers. To standardize billing, we’ve
 946 established a unified rate of \$0.001 per invocation for both search API calls and web browsing
 947 actions. Because different URLs produce varying amounts of text and tokens, this per-call rate is
 948 derived as an average computed from post hoc statistics over all our experiments. The consumption
 949 of tokens is billed separately, adhering to the official pricing models of the API provider pricing³.
 950

951 C ABLATIONS

952 Table 4: Ablation results of BATS on three benchmarks with Gemini-2.5-Pro. The agent is provided
 953 a budget of 100 tool uses per tool. Removing different modules leads to various performance drops
 954 across datasets.
 955

956 Method	957 BrowseComp	958 BrowseComp-ZH	959 HLE-Search
960 BATS	18.7	39.1	23.0
961 w/o Planning	17.0	34.6	20.0
962 w/o Verification	15.4	37.7	22.0
963 w/o Planning & Verification	14.6	32.9	21.5

964 In this section, we present an ablation analysis of the key modules within BATS using Gemini-2.5-
 965 Pro. The full comparison is shown in Table 4. Across all settings and datasets, the agent is allocated
 966 a budget of 100 tool uses per tool. To provide a clearer view of the orchestration framework, we
 967 focus on BATS under the early-stopping mechanism, where generation terminates once an answer
 968 is verified as successful.
 969

970 Results indicate that removing the planning module leads to a moderate performance decrease. How-
 971 ever, removing the verification module causes a more significant drop on BrowseComp (from 18.7%
 972 to 14.6%).

¹<https://developers.google.com/custom-search/v1/overview>

²<https://jina.ai/>

³<https://cloud.google.com/vertex-ai/generative-ai/pricing>

972 to 15.4%). This suggests that the verification module is crucial for enabling the agent to navigate the
 973 solution space and accurately assess its current progress. Removing both modules results in lower
 974 performance (14.6% on BrowseComp), while BrowseComp-ZH and HLE-Search exhibit some vari-
 975 ance, likely due to the smaller dataset size. In addition, questions in HLE-Search are typically
 976 shorter and contain fewer details that can be verified, which limits the contribution of the verifica-
 977 tion module. Overall, the trend confirms that both modules contribute positively to the orchestration
 978 framework’s effectiveness.

980 D RELATED WORK

982 D.1 TEST-TIME SCALING

984 Test-time scaling (TTS) (Snell et al., 2024; Zhang et al., 2025a) strategies typically fall into two
 985 categories. The first is sequential scaling, where a model iteratively refines its output based on self
 986 feedback or reflection (Madaan et al., 2023; Zhang et al., 2025b; Muennighoff et al., 2025; Liu
 987 et al., 2025b). The second category is parallel scaling, where multiple reasoning paths are sam-
 988 pled and an aggregation strategy is used to determine the final answer (Brown et al., 2024; Wang
 989 et al., 2023). Further, hybrid scaling attempts to combine their complementary benefits (Chen et al.,
 990 2025a,b; Wan et al., 2025; Li et al., 2024). While prior work focuses on text-only reasoning, we
 991 extend TTS to tool-augmented agents, where scaling accounts for both tokens and tool calls under
 992 budget constraints. As these methods push performance by increasing computation, a complemen-
 993 tary line of work examines how to constrain the effort. Typical constraints are defined over tokens,
 994 sampled sequences, or FLOPs (Nayab et al., 2024; Welleck et al., 2024; Damani et al., 2025; Pu
 995 et al., 2025). Specifically, AgentTTS (Wang et al., 2025a) optimizes LLM size and sampling num-
 996 bers under a unified FLOPs budget, while SLIM (Yen et al., 2025) uses periodic summarization to
 997 manage context growth in long-horizon agents. In contrast, we formalize and constrain the tool-call
 998 budget, shifting the focus from token-related limits in text reasoning to budget-constrained scaling
 999 of tool-augmented agents.

1000 D.2 WEB SEARCH AGENTS

1002 Web search agents use search and browse tools to solve complex, multi-hop queries (Chen et al.,
 1003 2025c; Wong et al., 2025; Team et al., 2025a; Han et al., 2025a; Li et al., 2025a). One research
 1004 direction builds training data and applies various training methods to specialize the models (Jin
 1005 et al., 2025; Wu et al., 2025a; Li et al., 2025b; Liu et al., 2025a; Tao et al., 2025; Ye et al., 2025;
 1006 Li et al., 2025c). MiroThinker (Team et al., 2025b) shows scaling interactive tool use is an effective
 1007 dimension, but it often yields redundant or inefficient tool calls. Another explores inference-time
 1008 strategies (Li et al., 2025d; Zhu et al., 2025a; Qiao et al., 2025; Qin et al., 2025), such as incorpo-
 1009 rating programmatic execution to perform multiple tool call actions (Pang et al., 2025), finding an
 1010 optimal, statically efficient configuration (Wang et al., 2025b), or exploring various design choices
 1011 when scaling test-time compute (Zhu et al., 2025b). Instead, our work focuses on dynamic, cost-
 1012 effective performance, providing the first analysis of agent scaling behavior under explicit budget
 1013 constraints.

1014 E ADDITIONAL ANALYSIS

1015 E.1 TOOL USAGE

1018 In Table 5, we examine how the ReAct baseline utilizes tools under different budget constraints on
 1019 the BrowseComp dataset. As expected, increasing the allowed tool budget enables the agent to inter-
 1020 act more with the environment, leading to improved accuracy. The Over-Budget% column shows
 1021 the proportion of examples that exhaust the allocated tool budget. For Gemini-2.5-Pro, only 0.8% of
 1022 the data requires more than 50 tool calls, whereas Gemini-2.5-Flash shows a higher reliance on tool
 1023 usage, with 2.6% of the queries exceeding a budget of 100. Notably, Flash models issue roughly
 1024 twice as many search queries as Pro models. This suggests that the stronger parametric knowl-
 1025 edge of Gemini-2.5-Pro allows it to navigate the search space more efficiently, reducing reliance on
 externally gathered evidence and enabling it to solve queries with fewer tool calls.

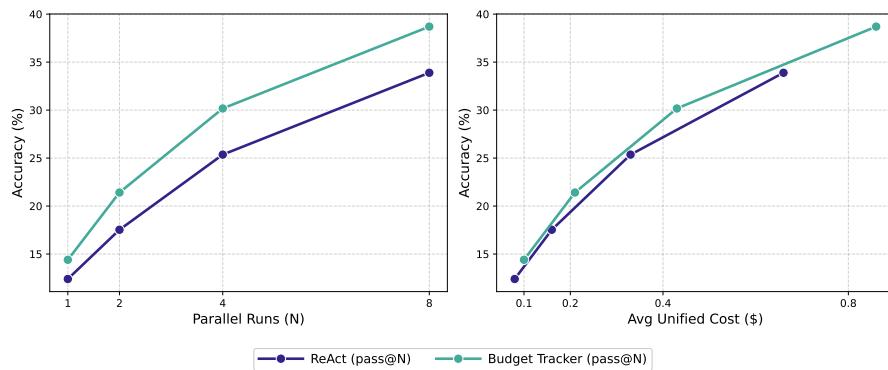
1026 Table 5: Resource usage statistics of the ReAct baseline using Gemini-2.5-Pro and Gemini-2.5-
 1027 Flash on the BrowseComp dataset. Over-Budget % denotes the percentage of data that reaches the
 1028 given tool budget limit.

1029

1030 1031 1032 1033 1034 1035 1036 1037 1038	1039 1040 1041 1042 1043 1044 1045 1046 1047 1048	1049 1050 1051 1052 1053 1054 1055 1056 1057 1058	Model	Budget	Acc %	Query		URL	
			Gemini-2.5-Pro	30	10.5	Avg. #	Over-Budget %	Avg. #	Over-Budget %
				50	12.4	13.77	9.24	1.33	0.00
				100	12.6	13.95	0.79	1.31	0.00
			Gemini-2.5-Flash	30	9.0	14.24	0.00	1.36	0.00
				50	9.7	22.77	42.81	1.04	0.00
				100	10.0	28.88	14.22	1.24	0.00
						29.93	2.6	1.36	0.00

E.2 DETAILS OF PARALLEL SCALING

For Majority Vote, we aggregate the final answers across different samples and use a judge model (Gemini-2.5-Flash) to identify the consensus answer. For Best-of-N, we provide all response trajectories to the judge model and select the most promising one. For Pass@N, the score is calculated as 1 if any of the sampled responses contain the correct answer. The prompts used for these evaluations are provided in Appendix F.4. In Figure 9, we additionally report the Pass@N results for ReAct and Budget Tracker under parallel scaling settings. Budget Tracker consistently achieves higher overall accuracy across varying budgets and cost levels.



1062 Figure 9: Pass@N results of Budget Tracker and ReAct in parallel scaling. The left subfigure shows
 1063 accuracy scaling with increasing parallel runs, while the right subfigure illustrates the corresponding
 1064 cost–performance trend.

E.3 RESOURCE USAGE

1068 Table 6 reports the detailed breakdown of tool calls and token consumption in BATS using Gemini-
 1069 2.5-Pro on BrowseComp. We compare our approach against the variant BATS-response, which
 1070 keeps the full tool responses from previous rounds. In contrast, BATS removes these responses to
 1071 optimize context length. In the table, token usage is reported as a triplet: (Input / Output / Cache).
 1072 The results demonstrate that removing historical tool responses does not compromise performance
 1073 (maintaining similar accuracy) but significantly reduces computational overhead. This efficiency is
 1074 evidenced by the substantial decrease in cache tokens and the reduction in overall cost.

E.4 PLANNING MODULE

1078 To evaluate the impact of the planning module, we augment the ReAct baseline with our proposed
 1079 design, which integrates constraint analysis and a dynamically structured checklist plan. The tool-
 call budget is capped at 200 for both search queries and browse URLs. As shown in Table 7, the

1080 Table 6: Resource usage comparison. Token counts are reported as Input / Output / Cache (in
 1081 10,000). BATS achieves comparable accuracy to the full-context baseline (BATS-response) while
 1082 significantly reducing cache token usage and cost.

	Acc (%)	# search	# browse	Total tokens	Veri. tokens	Cost (\$)
BATS	24.6	87.3	13.6	32.1 / 6.9 / 39.3	1.1 / 1.2 / 7.6	1.1
BATS-response	24.3	84.4	14.6	33.6 / 6.5 / 91.8	1.0 / 1.3 / 17.1	1.2

1088 addition of planning module alone improves the agent’s ability to organize exploration and utilize
 1089 tool usage more effectively, resulting in a performance gain of 1.8%.

1091 Table 7: Effect of the planning module. With the same budget, the planning module encourages
 1092 better and yields higher average performance.

Method	Acc %	Avg. # Query	Avg. # URL
ReAct	11.0	7.75	0.35
ReAct + planning	12.8	13.81	0.82

1099 E.5 EARLY STOPPING

1101 **BATS pushes the cost-performance Pareto**
 1102 **frontier even further by leveraging early**
 1103 **stopping.** To provide a direct and transpar-
 1104 ent comparison of cost-efficiency, Figure 10
 1105 shows accuracy against the actual unified cost.
 1106 BATS demonstrates a much steeper perfor-
 1107 mance curve, indicating that it achieves higher
 1108 accuracy for a lower cost compared to the par-
 1109 allel majority vote baseline. BATS reaches over
 1110 37% accuracy for approximately \$0.23, while
 1111 the parallel baseline requires more than double
 1112 that cost (over \$0.50) to achieve a comparable
 1113 result. This efficiency benefits from its budget-
 1114 aware verification module, which enables early
 1115 termination upon finding a satisfactory answer
 1116 and minimizes unnecessary spending.

1117 F PROMPTS

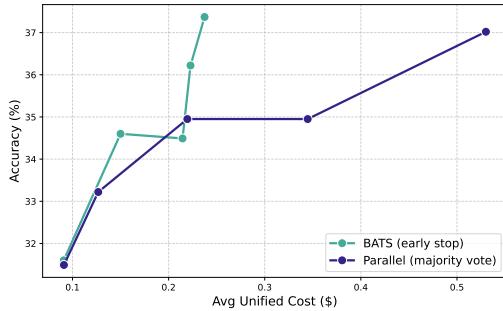
1119 Our prompt for web search agents is developed based on Jin et al. (2025) and Li et al. (2025b).

1121 F.1 REACT+ BUDGET TRACKER

1124 ReAct + Budget Tracker

```

1126 You are an AI reasoner with Google Search and Browsing tools. Solve the question by
1127 iterating: think, tool_code, tool_response, answer.
1128 ## Tools
1129 You have access to 2 tools: search and browse.
1130 {
1131   "name": "search",
1132   "description": "Performs batched web searches: supply an array 'query'; the tool
1133   retrieves the top 10 results for each query in one call.",
1134   "parameters": {
1135     "type": "object",
1136     "properties": {
1137   }
1138 }
```



1120 Figure 10: Average unified cost analysis on
 1121 BrowseComp-ZH using Gemini-2.5-Pro.

```

1134
1135     "query": {
1136         "type": "array",
1137         "items": {
1138             "type": "string"
1139         },
1140         "description": "Array of query strings. Include multiple complementary search
1141         queries in a single call."
1142     },
1143 },
1144     "name": "browse",
1145     "description": "Visit webpage(s) and return the summary of the content.",
1146     "parameters": {
1147         "type": "object",
1148         "properties": {
1149             "url": {
1150                 "type": "array",
1151                 "items": {"type": "string"},
1152                 "description": "The URL(s) of the webpage(s) to visit. Can be a single
1153                 URL or an array of URLs."
1154             },
1155             "goal": {
1156                 "type": "string",
1157                 "description": "The specific information goal for browsing webpage(s)."
1158             }
1159         },
1160         "required": [
1161             "url",
1162             "goal"
1163         ]
1164     }
1165 }
1166
1167 You should start with one or more cycles of (thinking about which tool to use ->
1168     performing tool code -> waiting for tool response), and end with (thinking about
1169     the answer -> answer of the question). The thinking processes, tool codes, tool
1170     responses, and answer are enclosed within their tags. There could be multiple
1171     thinking processes, tool codes, tool call parameters and tool response parameters.
1172
1173 ## Budget
1174
1175 You have two independent budgets:
1176 - Query Budget (for search)
1177 - URL Budget (for browse)
1178
1179 Each string in 'query' or 'url' consumes 1 unit respectively.
1180
1181 After each <tool_response>, a <budget> tag shows remaining units.
1182 You must ADAPT your strategy dynamically to the current budget state.
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
```

```

1188
1189     </budget>
1190
1191     Repeat <think><tool_code> until you have the final answer.
1192     <answer> Final solution only. </answer>
1193
1194     ## About answers
1195
1196     * Only write the final answer inside <answer> and </answer>.
1197     * If you cannot find the answer, write <answer>None</answer>.
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241

```

Figure 11: Prompts used in ReAct + Budget Tracker.

F.2 PLANNING MODULE

Budget-aware Planning Module in BATS

```

1201
1202
1203     ## About questions
1204
1205     Questions contain two types of constraints: exploration and verification.
1206     * Exploration: Broad, core requirements (e.g., birthday, profession). Use these for
1207         initial searches to surface candidates. You may combine 1-2 to form stronger
1208         queries.
1209     * Verification: Narrow, specific details. Apply these only after you have candidates,
1210         to confirm or filter them. Never begin with verification constraints.
1211     Start with exploration queries, then use verification to validate the results.
1212
1213     ## About planning
1214
1215     Maintain a tree-structured checklist of actionable steps (each may require several tool
1216         calls).
1217     - Mark each step with its status: [ ] pending, [x] done, [!] failed, [~] partial.
1218     - Use numbered branches (1.1, 1.2) to represent alternative paths or candidate leads.
1219     - Log resource usage after execution: (Query=#, URL=#).
1220     - Keep all executed steps, never delete them, retain history to avoid repeats.
1221     - Update dynamically as you reason and gather info, adding or revising steps as needed.
1222     - Always consider current and remaining budget when updating the plan.
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241

```

Figure 12: Prompts used in planning module of BATS.

F.3 SELF-VERIFICATION MODULE

Budget-aware Self-verification Module in BATS

```

1222
1223
1224     You are an AI Strategic Verifier. Your primary goal is to evaluate a proposed answer,
1225         assess the viability of the current problem-solving plan, and decide the best
1226         course of action: declare success, continue with the current plan, or pivot to a
1227         new one.
1228
1229     ### Given Inputs
1230
1231     * Question: The original user question. An answer is believed to exist.
1232     * Trajectory: The sequence of reasoning steps and tool calls taken so far in the
1233         current attempt.
1234     * Current Answer: The final answer produced by the current attempt.
1235     * Budget Status: Information on current tool call budget utilization and remaining
1236         budget, including search queries and browsing urls.
1237
1238     ### Your Task: A 3-Step Process
1239
1240     You must proceed in the following order:
1241
1242     #### Step 1: Conduct Verification Analysis
1243
1244     First, perform a strict verification of the `Current Answer`.
1245     * Go through each constraint from the original Question one by one.
1246     * For each constraint, compare it against the `Current Answer` and the `Trajectory`.
1247     * State your finding for each constraint: satisfied, contradicted, or unverifiable.
1248
1249     #### Step 2: Make a Strategic Decision
1250
1251     Based on your verification and the budget, make one of three decisions.
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
```

```

1242
1243 1. SUCCESS: If the verification in Step 1 passed (all constraints are satisfied). The
1244 task is complete.
1245 2. CONTINUE: If the verification failed because few constraints are unverifiable, but
1246 the overall plan is still sound and salvageable. This is the choice if both of
1247 these conditions are true:
1248 * Promising Path: The 'Trajectory' is generally sound, and the failure was due to a
1249 correctable error.
1250 * Sufficient Budget: There is enough 'Remaining Budget' to attempt a correction on
1251 this path.
1252 3. PIVOT: If the verification failed, signal to abandon the current plan and switch to
1253 another one. You should pivot if any of these conditions are true:
1254 * Dead End: The 'Trajectory' reveals a fundamental flaw in the current plan's logic
1255 that cannot be easily fixed.
1256 * Failed Tool Calls: The Trajectory shows repeated, unsuccessful attempts to find
1257 certain info.
1258 * Insufficient Budget: The 'Remaining Budget' is too low to make another meaningful
1259 attempt or correction within the current plan.

1260 ##### Step 3: Summarize for the Next Step
1261
1262 This is the most critical step for guiding future actions.
1263 You need to first provide a trajectory summary: Summarize the agent's reasoning
1264 trajectory into a concise narrative. Explain its initial goal, the logical steps
1265 taken, key findings and the final conclusion, emphasizing how key findings or
1266 contradictions caused the agent to change its strategy.

1267 Then, provide additional details tailored to your decision in Step 2.
1268
1269 * If the decision is SUCCESS:
1270   * No further detail needed.
1271
1272 * If the decision is CONTINUE / PIVOT:
1273   * Failure Analysis: Diagnose the root cause of the failure. Identify the critical
1274     flaw (e.g., poor query design, flawed logic, misinterpreted evidence) and name the
1275     general failure pattern to prevent its recurrence.
1276   * Useful information: Any useful intermediate findings or results from the current
1277     'Trajectory' that could be valuable inputs for the next attempt. This prevents
1278     redundant work.
1279   * Strategic Recommendations: Provide actionable advice for the agent's next attempt
1280     . Suggest strategic pivots, new angles of investigation, or different ways to
1281     combine the problem's constraints. Explicitly state if it should backtrack to and
1282     resume from a specific step in the previous plan to avoid re-doing work.

1283 ##### Output Requirement**
1284
1285 Your final output must be a single JSON object with the following structure. Do not add
1286 any text before or after this JSON block.
1287
1288 ````json
1289 {
1290   "verification": "Verification analysis",
1291   "decision": "SUCCESS | CONTINUE | PIVOT",
1292   "justification": "A concise explanation for your strategic decision. Why is it a
1293     success, a dead end, or a correctable error?",
1294   "trajectory_summary": "The informative trajectory summary.",
1295   "details": "A JSON object containing the additional details required by Step 3. For a
1296     SUCCESS decision, this can be an empty object {}."
1297 }
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371
3372
3373
3374
3375
3376
3377
3378
3379
3380
3381
3382
3383
3384
338
```

```

1296
1297     First, provide a brief justification explaining why the chosen answer is the most
1298     accurate and specific choice. Then, on a new line, output the letter of the best
1299     option inside a box.
1300
1301     **Example:**  

1302     Justification: Answer B is the most specific correct location...  

1303
1304
1305
1306     You are an expert evaluator. Your task is to select the answer that best represents the
1307     **majority vote** among the provided candidates. The question has a deterministic
1308     answer, and the goal is to identify which option most responses converge on. You'
1309     ll be provided with several answers and their corresponding trajectories/
1310     verifications.
1311
1312     **Instructions:**  

1313     1. **Identify the Core Question:** Determine the exact piece of information the
1314     question is asking for (e.g., a person, a location, a date).
1315     2. **Tally the Votes:** Review all candidate answers and count how many times each
1316     distinct answer (or near-equivalent variant) appears. Treat semantically
1317     equivalent responses as votes for the same candidate.
1318     3. **Select the Majority:** Choose the answer that has the highest number of votes. If
1319     there is a tie, pick the option that is the most specific and consistent with the
1320     question. Never choose "None" or refuse to make a choice.
1321
1322     **Output Format:**  

1323     First, provide a brief justification explaining why the chosen answer was selected (e.g
1324     .., "Answer C has the majority of votes across candidates"). Then, on a new line,
1325     output the letter of the best option inside a box.
1326
1327     **Example:**  

1328     Justification: Answer B received the majority of votes and aligns most consistently
1329     with the question.  

1330     Answer: \boxed{B}

```

Figure 14: Prompts used in answer selection.

G CASE STUDY

This section presents outputs from different agents. To avoid contaminating the internet with the cases shown, we have removed essential details and display only snippets of the agent trajectories.

G.1 ADAPTIVE THINKING AND PLANNING

We demonstrate how Budget Tracker enables the agent to dynamically adapt its behavior based on budget awareness, choosing different strategies under high (Figure 15) or low (Figure 16) budget. These cases are collected from Gemini-2.5-Pro trajectories on BrowseComp. Without budget awareness, ReAct uses a brute-force planning strategy to exhaust direct searches. This causes it to quickly consume all available resources, failing to answer the question correctly.

G.2 ADAPTIVE VERIFICATION

In Figure 17, we demonstrate how self-verification module makes budget aware decision based on the trajectory and the resource status. It decides to pivot from the current pass because the remaining budget still allows for further investigation. These cases are collected from Gemini-2.5-Pro trajectories on BrowseComp.

H LIMITATIONS

More resource constraints. While our study presents the first empirical analysis of tool-call budgets, a more realistic and challenging scenario involves managing multiple resource constraints jointly. Examples include token limits, inference latency, and tool-call budgets. Understanding and controlling agent behavior under multi-dimensional constraints is important for deploying scalable systems in real environments.

1350

1351

1352

High Budget

1353

1354

**Question**

What is the full name of a person known for investing in technology businesses? [Other constraints]...

1355

1356

ReAct [Wrong answer]

1357

The most unique and searchable clue is the blog post titled "Mxx" about a trip to Rxx. **This is a very specific combination of terms. I will start by searching for this.**

1358

1359

ReAct + Budget Tracker [Correct answer]

1360

1361

My strategy will be to start with the most specific clue: the blog post. I will combine it with other keywords to narrow down the results. **I have a high budget, so I can use multiple queries to explore different angles simultaneously.**

1362

Figure 15: Adaption to high budget constraints. Being aware of the high budget, the agent chooses to expand its search queries starting from a specific keyword, whereas ReAct immediately applies narrow conditions and fails.

1363

1364

1365

1366

1367

1368

1369

1370

Low Budget

1371

1372

**Question**

In what year did the event occur that led to the loss of lives [Other constraints]...

1373

1374

ReAct [Wrong answer]

1375

1376

This requires a multi-pronged search. I will look for famous monuments in each city and then check their details... At this point, I **have exhausted the direct search** for a monument that fits all criteria.

1377

1378

ReAct + Budget Tracker [Correct answer]

1379

This is too many queries. I will try to be more **efficient**. [performs search] **I have a low query budget, so this must be precise.**

1380

1381

Figure 16: Adaption to low budget constraints. Under a low budget, ReAct fails due to an exhaustive search strategy. In contrast, budget tracker helps the agent adapt by prioritizing query efficiency and precision, successfully solving the task.

1385

1386

1387

1388

1389

1390

**Question**

There is a person ... What date was ...

1391

1392

Verification Decision: PIVOT

1393

1394

Justification:The agent concluded it had exhausted all logical paths, but this is incorrect. ... The core problem remains unsolved, and **since significant budget remains, a focused pivot back to the original, most logical strategy is required.**

1395

1396

Recommendation:The next attempt must **pivot back to the original**, most logical strategy and execute it with more rigor: xxx This is the key that unlocks the puzzle. Do not deviate to other strategies until this one is exhausted.

1397

1398

1399

1400

1401

Figure 17: In BATS, the verification process dynamically adapts to budget constraints. It successfully identifies the cause of failure and provides strategic recommendations based on the current trajectory. Following this advice allows the agents to better utilize the budget and arrive at the correct answer in the next attempt.

1402

1403

1404
1405 **Resource allocation.** Although we observe that budget awareness enables more effective scaling,
1406 we do not explore how an agent should allocate its available resources. Our empirical analysis
1407 suggests that models often underestimate their actual resource consumption, which can lead to sub-
1408 optimal performance. Developing accurate resource estimation and principled budget allocation
1409 strategies is a promising direction for future work.

1410 **More intelligent context management.** Although we adopt several simple context control tech-
1411 niques such as removing tool responses and summarizing intermediate trajectories, more advanced
1412 context engineering remains largely unexplored. Designing more effective memory formats and
1413 identifying the right balance between context length and performance are important open challenges
1414 for building robust and efficient agents.

1415 **Prompt Sensitivity.** While we ensure reproducibility by providing the prompts used in our experi-
1416 ments, we acknowledge that agent performance is inherently sensitive to prompt variations. Exhaus-
1417 tively enumerating prompt combinations to fully evaluate robustness is computationally infeasible
1418 and remains a fundamental challenge in LLM research.

1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457