LEARNING OPTIMAL AGENTS ACROSS ANY BUDGET VIA BILEVEL OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Autonomous agents often achieve stronger task performance by consuming more execution resources, leading to increased deployment cost. To improve efficiency, existing budget-aware approaches train agents under particular budget conditions or expose them to multiple predefined budgets. However, in practice, available budgets vary across scenarios, motivating agents that can perform well across the entire budget range. In this work, we study how to train agents that perform reliably across varying budget constraints. We transform the objective of maximizing any-budget capability into minimizing the expected minimum sufficient budget required by individual tasks, which naturally yields a coupled bilevel optimization problem over task-specific budget adaptation and policy improvement. Based on this formulation, we propose **BABLE** (**B**ilevel **A**ny-**B**udget **L**earning for ag**E**nts), which solves the coupled problem through alternating optimization. BABLE combines a Task-Adaptive Budget Controller (TABC), which adapts task-specific budgets from successful executions sampled at the current stage, with Likelihood-Guided Conditional Advantage Optimization (LCAO), which strengthens policy learning from sparse successes under tight budget conditions. Experiments on four agentic benchmarks under both token level and tool-call level budgets show that BABLE consistently improves performance across the evaluated budget range compared with standard reinforcement learning and budget-aware baselines.

1 INTRODUCTION

How can an agent achieve its best possible performance across any budget constraints? Recent advances in reinforcement learning have substantially improved the reasoning, tool use, and interaction capabilities of autonomous agents (Guo et al., 2025; Yao et al., 2022; Schick et al., 2023), but stronger task performance often comes with higher execution cost (Wei et al., 2022; Wang et al., 2022; Snell et al., 2024). Recent work has therefore studied agent learning under explicit budget constraints, such as limits on generated tokens and tool invocations (Han et al., 2025; Wen et al., 2025; Liu et al., 2025). In practice, however, available budgets vary across applications and tasks. Optimizing for a single budget reduces to maximizing task success under that constraint, potentially specializing the policy to one regime while overlooking others. When an agent must perform well across the entire budget range, the objective shifts from optimizing a specific budget to maintaining strong performance across diverse budget constraints.

Existing approaches typically optimize different regions of the budget spectrum. Standard reinforcement learning can be viewed as training under loose budget conditions, where the primary objective is task success (Guo et al., 2025; Christiano et al., 2017). As the budget tightens, however, performance can degrade substantially because the learned policy may rely on long trajectories or costly interactions (Wei et al., 2022; Snell et al., 2024). In contrast, budget efficient methods are often optimized under restrictive budgets through prompting, budget signals (Wen et al., 2025), cost-sensitive rewards (Aggarwal & Welleck, 2025), or progressively tighter constraints (Lin et al., 2026; Liu et al., 2025; Yang et al., 2026; Wu et al., 2026). Such specialization can improve low budget performance but may fail to preserve or recover capability as the budget scales up. Anytime reasoning takes a step toward any budget optimization by training across varying budgets (Qi et al., 2026), but typically uses a predefined set of discrete budget conditions. This provides only coarse coverage of the budget space and does not adapt training budgets to the evolving needs of individual tasks. Consequently, substantial optimization potential may remain across the budget spectrum.

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

Research Question

How can we train an agent to achieve consistently strong performance under any budget constraint?

We introduce **BABLE**: **B**ilevel **A**ny **B**udget **L**earning for ag**E**nts. We first characterize any-budget capability by reliable task performance over the entire budget range, and equivalently transform the resulting objective into minimizing the expected minimum sufficient budget across tasks, yielding a bilevel optimization problem. The lower level problem seeks the minimum sufficient budget of each task under the current policy, while the upper level problem optimizes the policy to further reduce these task-specific budget requirements. The two levels are inherently coupled. Policy improvement changes the feasible budget boundary of each task, whereas tightening the budget changes the conditions under which the policy is optimized. This gives rise to two practical challenges. As illustrated in Fig. 1, *I) How can the task-specific budget boundary be estimated and adapted as policy capability evolves? II) How can the policy be effectively optimized when successful trajectories become increasingly sparse near this boundary?*

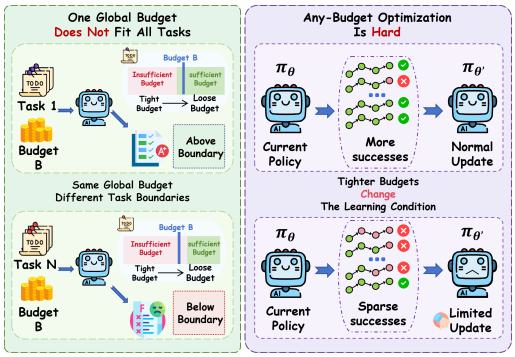


Figure 1: Problem Illustration. (a) ***The left part*** shows that global budgets fail for heterogeneous tasks. (b) ***The right part*** shows that tight budgets cause sparse successes and weak updates.

We solve the coupled problem through **alternating optimization**. **Task-Adaptive Budget Controller (TABC)** approximates the lower level update by adapting each task's training budget according to the costs of successful executions sampled at the current stage. Given these updated budgets, **Likelihood-Guided Conditional Advantage Optimization (LCAO)** realizes the upper level policy update through a budget conditioned log-success surrogate. As TABC adaptively adjusts task-specific budgets toward their sufficient budget boundaries, successes can become increasingly sparse, and LCAO strengthens learning under the resulting low success conditions. The improved policy in turn enables TABC to further tighten task-specific budgets, forming a naturally coupled optimization loop.

Positioning

Beyond proposing a budget aware reinforcement learning approach, this work rethinks any budget capability optimization. Rather than optimizing budget conditions separately, we characterize any budget capability by each task's minimum sufficient budget and formulate a coupled bilevel optimization problem, solved via alternating budget adaptation and policy optimization.

Our contributions are summarized as follows:

① **Problem Formulation.** We transform any budget agent learning into minimizing the expected minimum sufficient budget, and formulate the resulting objective as a bilevel optimization over task-specific budget adaptation and policy improvement.

② **Proposed Solution.** We solve the problem through alternating optimization. TABC adapts task-specific budgets toward sufficient budget boundaries, while LCAO strengthens policy optimization under sparse success conditions. Together, they couple budget adaptation with policy optimization.

③ **Experimental Validation.** We perform extensive experiments on four agentic benchmarks under both token level and tool-call level budget constraints. The results demonstrate that BABLE achieves superior performance across the entire budget range, while TABC and LCAO respectively improve budget scheduling and sparse success optimization.

2 Related Works

Budget-Aware Agent Learning. Existing work studies the trade-off between task performance and inference budget from different directions. Methods that primarily optimize task success may rely on longer trajectories or more tool use, making them less robust under tight budgets (Guo et al., 2025; Snell et al., 2024; Yuan et al., 2025). Budget aware methods instead improve efficiency through prompting (Han et al., 2025), strategy selection (Yang et al., 2026), budget signals (Liu et al., 2025),

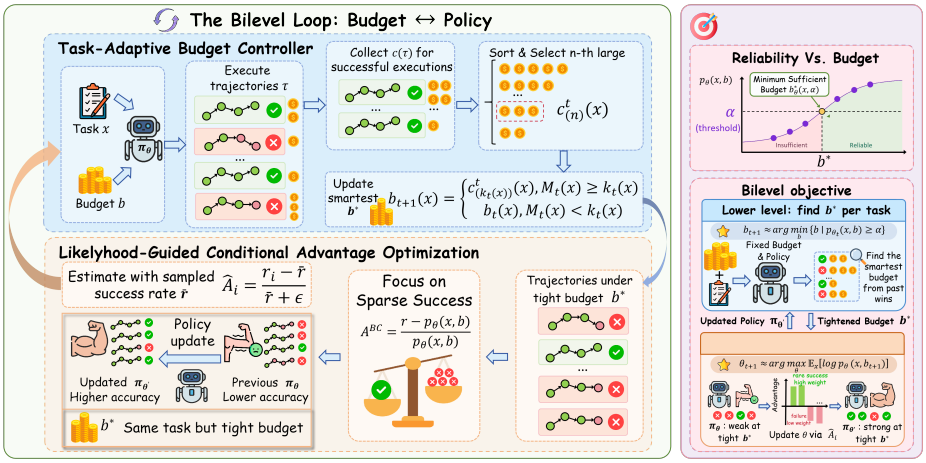


Figure 2: Overview of **BABLE**. Any budget agent learning is formulated as bilevel optimization between task-specific budget adaptation and policy improvement. TABC adapts budgets using successful executions, while LCAO strengthens learning from sparse successes under tight budgets. Their alternating updates jointly improve capability and reduce minimum sufficient budgets.

cost aware rewards (Aggarwal & Welleck, 2025; Yuan et al., 2025; Xiang et al., 2025; Ling et al., 2025), or tighter execution constraints (Wen et al., 2025). Anytime reasoning further trains policies across inference budgets, typically over a predefined set of conditions (Qi et al., 2026; Zhang et al., 2026; Li et al., 2026). However, these approaches generally specify budgets globally during training without modeling task-specific requirements. Our work instead optimizes over the entire budget range through each task's minimum sufficient budget.

Curriculum Learning and RL Optimization. Curriculum learning adapts training difficulty as model capability evolves, but existing methods often rely on predefined or globally shared schedules that expose samples to increasingly challenging conditions (Wen et al., 2025; Bengio et al., 2009; Kumar et al., 2010; Arora & Zanette, 2026). Such global progression may fail to capture the heterogeneous budget requirements and learning dynamics of individual tasks. In contrast, TABC maintains task-specific budgets and adapts them using successful executions sampled at the current stage. As tighter budgets induce sparse successes, policy optimization near the budget boundary becomes critical (Zhao et al., 2025; Ahmadian et al., 2024). GRPO constructs relative advantages from group-level reward statistics (Shao et al., 2024; Guo et al., 2025), while MaxRL reweights policy updates according to success probability through a maximum likelihood formulation (Tajwar et al., 2026; Dong et al., 2023; Li et al., 2023). Building on this perspective, LCAO applies likelihood-guided optimization to the sparse success task-budget conditions induced by TABC.

3 Methodology

BABLE formulates any budget agent learning as minimizing the expected minimum sufficient budget across tasks, yielding a bilevel optimization problem. As shown in Fig. 2, the lower level adapts task-specific budget boundaries under the current policy, while the upper level improves the policy under these budgets. We solve the coupled problem through alternating optimization: **Task-Adaptive Budget Controller (TABC)** updates budgets from successful executions, while **Likelihood-Guided Conditional Advantage Optimization (LCAO)** optimizes a budget conditioned log success objective that amplifies sparse successes under tight budgets. The training procedure is provided in Appendix A.

3.1 Problem Formulation

Given an agent task $x \sim \mathcal{D}$, a policy π_θ interacts with the environment and generates a trajectory $\tau = (s_0, a_1, s_1, \ldots, a_T, s_T)$, which receives a binary outcome reward $r(x, \tau) \in \{0, 1\}$ indicating whether the task is successfully completed. The success probability of task x under the policy is

$$p_\theta(x) = \mathbb{E}_{\tau \sim \pi_\theta(\cdot|x)} \left[r(x, \tau) \right]. \tag{1}$$

Conventional agent learning seeks to improve task performance by maximizing the expected success probability over the task distribution, $\mathbb{E}_{x\sim\mathcal{D}}[p_\theta(x)]$.

Under budget constrained execution, the budget b acts as an execution time truncation constraint. Given $b \in \mathcal{B}$, let $\tau^{(b)}$ denote the trajectory generated by $\pi_\theta(\cdot \mid x)$ and truncated when the cumulative execution cost reaches b. We define the corresponding success probability as

$$p_\theta(x, b) = \mathbb{E}\left[r(x, \tau^{(b)})\right]. \tag{2}$$

Here, b only determines when execution is truncated and does not alter the policy. For fixed b, the objective reduces to maximizing $\mathbb{E}_{x\sim\mathcal{D}}[p_\theta(x, b)]$. Our goal differs: rather than optimizing for a particular budget, we seek to improve capability across the entire budget range.

Any budget optimization. Our objective is to optimize policy capability across the entire budget range rather than at a fixed budget. We consider a continuous budget domain $\mathcal{B} = [0, B]$. We assume budget monotonicity, $p_\theta(x, b_1) \le p_\theta(x, b_2)$ for $b_1 \le b_2$, reflecting that a task solvable with less budget remains solvable with more. We discuss this assumption under early stopping execution in Appendix D. Given a target reliability level α, we define

$$s^\alpha_\theta(x, b) = \mathbb{I}\left[p_\theta(x, b) \ge \alpha\right], \qquad b \in [0, B], \tag{3}$$

which indicates whether task x is reliably solvable under budget b. We measure any budget capability by integrating task solvability over the full budget range:

$$\max_\theta J_{\mathrm{AB}}(\theta) = \max_\theta \mathbb{E}_{x\sim\mathcal{D}}\left[\int_0^B s^\alpha_\theta(x, b)\, db\right]. \tag{4}$$

This objective measures the area of reliable task solvability over the entire budget range. For each task, a larger value indicates that the policy can solve the task reliably under a broader range of budget conditions. Maximizing the expectation therefore encourages the policy to expand the reliably solvable region toward lower budgets across tasks.

Minimum sufficient budget. For each task x, we define its **minimum sufficient budget** as

$$b^*_\theta(x; \alpha) = \inf\left(\{b \in [0, B] : p_\theta(x, b) \ge \alpha\} \cup \{B\}\right). \tag{5}$$

The additional term $\{B\}$ handles tasks for which no budget in $[0, B]$ achieves the target reliability level α. In this case, the feasible set is empty and the minimum sufficient budget is assigned to the maximum budget B. From Eqs. (3) and (5), under budget monotonicity,

$$s^\alpha_\theta(x, b) = \mathbb{I}\left[b^*_\theta(x; \alpha) < b\right] \tag{6}$$

Substituting Eq. (6) into Eq. (4) gives

$$J_{\mathrm{AB}}(\theta) = \mathbb{E}_x\left[\int_0^B \mathbb{I}(b^*_\theta(x; \alpha) \le b)\, db\right] = B - \mathbb{E}_x\left[b^*_\theta(x; \alpha)\right]. \tag{7}$$

Since B is constant, maximizing any budget capability is equivalent to minimizing the expected minimum sufficient budget:

$$\max_\theta J_{\mathrm{AB}}(\theta) \quad \Longleftrightarrow \quad \min_\theta \mathbb{E}_{x\sim\mathcal{D}}\left[b^*_\theta(x; \alpha)\right]. \tag{8}$$

Minimum Sufficient Budget as the Optimization Target

The equivalence in Eq. (8) shows that any budget agent learning can be viewed as minimizing the budget needed for reliable task completion. The minimum sufficient budget thus provides a task-specific measure of capability: a stronger agent achieves the same reliability with fewer resources. Since this boundary depends on the current policy and evolves during training, the problem leads to a coupled bilevel optimization of budget adaptation and policy improvement.

3.2 BILEVEL OPTIMIZATION

The equivalence in Eq. (8) naturally leads to a bilevel formulation of any budget learning (Franceschi et al., 2018):

$$\min_{\theta} \mathbb{E}_{x\sim\mathcal{D}}\left[b^*_\theta(x;\alpha)\right], \qquad b^*_\theta(x;\alpha) = \inf\left(\{b\in[0,B] : p_\theta(x,b)\geq\alpha\}\cup\{B\}\right). \tag{9}$$

The lower level problem identifies the minimum sufficient budget of each task under the current policy, while the upper level problem optimizes the policy to further reduce these task-specific budget requirements. The two levels are coupled because policy improvement changes the feasible budget boundary, while budget tightening changes the conditions under which the policy is optimized.

Alternating optimization. Directly solving the bilevel problem is challenging because the task-specific budget boundary evolves with the policy. We therefore adopt a standard alternating optimization strategy to iteratively update budgets and policy parameters (Lorraine et al., 2020). Given the current policy θ_t, we approximate the lower level update as

$$b_{t+1}(x) \approx \inf\left(\{b\in[0,b_t(x)] : p_{\theta_t}(x,b)\geq\alpha\}\cup\{b_t(x)\}\right). \tag{10}$$

The term $\{b_t(x)\}$ handles the case where no budget in $[0, b_t(x)]$ achieves reliability level α, leaving the feasible set empty. In this case, the current budget is retained. Since the success probability and sufficient budget boundary are unavailable in practice, **Task-Adaptive Budget Controller (TABC)** approximates this update using successful executions sampled at the current stage.

Directly optimizing the outer objective is intractable because $b^*_\theta(x;\alpha)$ is implicitly defined by the reliability boundary. Increasing the success probability near this boundary expands the feasible budget region, enabling subsequent lower level updates to further reduce the minimum sufficient budget. We therefore use the tractable surrogate

$$\theta_{t+1} \approx \arg\max_{\theta} \mathbb{E}_{x\sim\mathcal{D}}\left[\log p_\theta(x, b_{t+1}(x))\right]. \tag{11}$$

This upper level update is realized by **Likelihood-Guided Conditional Advantage Optimization (LCAO)**, which strengthens learning signals from rare successes under tighter budget conditions. The improved policy then enables TABC to further tighten task-specific budgets in subsequent stages.

3.3 TASK-ADAPTIVE BUDGET CONTROLLER

The lower level optimization seeks the minimum sufficient budget of each task under the current policy. Since the success probability $p_{\theta_t}(x,b)$ and the corresponding budget boundary are unavailable in closed form, we approximate this update using the costs of successful executions sampled at the current stage. Specifically, the Task-Adaptive Budget Controller (TABC) maintains a separate training budget $b_t(x)$ for each task and adapts it as the policy evolves.

Task-specific budget adaptation. Let $\mathcal{R}_t(x)$ denote the trajectories sampled for task x at training under the current task-specific budget $b_t(x)$ stage t. We collect the realized costs of successful trajectories in the current stage as

$$\mathcal{C}_t(x) = \{c(\tau) \mid \tau\in\mathcal{R}_t(x),\ r(x,\tau)=1\}, \tag{12}$$

where $c(\tau)$ denotes the execution cost of trajectory τ. Let $N_t(x) = |\mathcal{R}_t(x)|$ denote the total number of trajectories sampled for task x at stage t, and let $M_t(x) = |\mathcal{C}_t(x)|$ denote the number of successful trajectories. We sort the successful costs in ascending order as $c^t_{(1)}(x) \leq c^t_{(2)}(x) \leq \cdots \leq c^t_{(M_t(x))}(x)$ and define $k_t(x) = \lceil \alpha N_t(x)\rceil$. TABC then updates the task-specific budget as

$$b_{t+1}(x) = c^t_{(k_t(x))}(x) \text{ if } M_t(x)\geq k_t(x), \text{ else } b_t(x). \tag{13}$$

The update has a direct reliability interpretation. When $M_t(x)\geq k_t(x)$, the selected cost is the smallest observed budget under which at least an α fraction of the current stage trajectories provide successful execution evidence. If fewer than $\lceil\alpha N_t(x)\rceil$ successful trajectories are observed, the current budget is retained, since the current stage rollouts provide insufficient evidence for further contraction. Thus, α directly controls the conservativeness of budget adaptation, with a larger α requiring stronger evidence before moving to a tighter budget. Because TABC updates each task independently, tasks can follow different budget trajectories according to their evolving policy capability. As successful executions become less costly, the assigned budget moves toward a tighter feasible region, providing the task-specific budget conditions for the upper level policy optimization.

3.4 LIKELIHOOD-GUIDED CONDITIONAL ADVANTAGE OPTIMIZATION

Given the task-specific budgets from TABC, the upper level problem optimizes the policy near each task's sufficient budget, where successful rollouts can become sparse. Under sparse successes, mean normalization assigns greater weight to rare successful rollouts than the standard deviation normalization used by standard GRPO.

Motivated by the likelihood perspective of MaxRL (Tajwar et al., 2026), we therefore couple TABC with a budget conditioned log success objective. Since the minimum sufficient budget is not directly differentiable with respect to the policy, we use

$$J_{\mathrm{LCAO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\left[\log p_\theta(x, b_t(x))\right]. \tag{14}$$

As TABC tightens $b_t(x)$ and reduces $p_\theta(x, b_t(x))$, the relation $d\log p = dp/p$ places stronger emphasis on improvements under these sparse success task-budget conditions. Together, they naturally couple budget adaptation with policy optimization: TABC adapts task-specific budgets toward their sufficient boundaries, while the likelihood objective strengthens learning under the resulting sparse success conditions.

Likelihood-guided conditional advantage. For a fixed task x under its current assigned budget b, using the score-function identity $\mathbb{E}_\tau[\nabla_\theta \log \pi_\theta(\tau \mid x, b)] = 0$, we obtain

$$\nabla_\theta \log p_\theta(x, b) = \mathbb{E}_{\tau\sim\pi_\theta(\cdot|x,b)}\left[\frac{r(x,\tau) - p_\theta(x,b)}{p_\theta(x,b)}\nabla_\theta \log \pi_\theta(\tau \mid x, b)\right]. \tag{15}$$

This identifies the population level conditional advantage as

$$A^{\mathrm{LCAO}}(x, b, \tau) = \frac{r(x,\tau) - p_\theta(x,b)}{p_\theta(x,b)}. \tag{16}$$

Sample based conditional advantage. In practice, $p_\theta(x, b)$ is unknown. For each task x under its current budget, we sample a group of G trajectories and let $K = \sum_{i=1}^{G} r_i$ and $\bar{r}_{x,b} = K/G$, where K is the number of successful trajectories and $\bar{r}_{x,b}$ is the empirical success rate. Replacing $p_\theta(x, b)$ with $\bar{r}_{x,b}$ yields, for $K > 0$,

$$\hat{g}(x, b) = \frac{1}{G}\sum_{i=1}^{G}\frac{r_i - \bar{r}_{x,b}}{\bar{r}_{x,b}}\nabla_\theta \log \pi_\theta(\tau_i \mid x, b) = \sum_{i=1}^{G}\left(\frac{r_i}{K} - \frac{1}{G}\right)\nabla_\theta \log \pi_\theta(\tau_i \mid x, b). \tag{17}$$

Accordingly, using $\bar{r}_{x,b} = K/G$, we define the stabilized sample based advantage as

$$\hat{A}_i^{\mathrm{LCAO}} = \frac{Gr_i - K}{K + G\epsilon}. \tag{18}$$

When $K = 0$, all trajectories fail and the advantage is automatically zero. For $K > 0$ and ignoring the small stabilization term, each successful trajectory receives a positive weight that increases as K decreases, whereas failed trajectories receive a negative weight. Thus, rare successes are amplified under low success task-budget conditions.

Integration with GRPO. We retain the clipped policy optimization structure of GRPO and replace its group standardized advantage with $\hat{A}_i^{\mathrm{LCAO}}$. For trajectory i at interaction step k, let $\omega_{i,k}(\theta) = \pi_\theta(a_{i,k} \mid s_{i,k-1}, x, b)/\pi_{\theta_{\mathrm{old}}}(a_{i,k} \mid s_{i,k-1}, x, b)$ denote the importance ratio at interaction step k. The resulting objective is

$$\mathcal{J}_{\text{LCAO-GRPO}}(\theta) = \mathbb{E}_{\tau_i,k}\left[\min\Big(\omega_{i,k}(\theta)\hat{A}_i^{\mathrm{LCAO}}, \mathrm{clip}(\omega_{i,k}(\theta), 1-\epsilon_c, 1+\epsilon_c)\,\hat{A}_i^{\mathrm{LCAO}}\Big)\right]. \tag{19}$$

LCAO therefore provides a tractable upper level policy update while retaining the clipped optimization structure of GRPO. Guided by the budget conditioned log success objective, it amplifies successful trajectories when the empirical success probability is low, improving the policy under the tight budgets supplied by TABC. Together, TABC and LCAO form the alternating optimization loop of BABLE: policy improvement enables TABC to adapt task-specific budgets toward their sufficient boundaries, while the adapted budgets provide updated conditions for policy optimization.

Table 1: Performance comparison across the evaluated budget range under different budget settings. **Any.** denotes the average accuracy across all evaluated budgets, **Max.** denotes the accuracy under the maximum budget, and **Low.** denotes the accuracy under a representative low budget. The best result in each column is highlighted in bold. Results are averaged over three evaluation runs. Standard deviations are reported in Appendix G. The best result in each column is highlighted in bold.

Methods	**MuSiQue**			**GAIA**			**WebWalkerQA**			**FRAMES**		
	Any.	**Max.**	**Low.**	**Any.**	**Max.**	**Low.**	**Any.**	**Max.**	**Low.**	**Any.**	**Max.**	**Low.**
budget: token												
Vanilla	35.12	42.90	23.76	21.72	28.03	13.58	6.77	9.08	4.10	24.37	32.76	16.38
GRPO	38.41	47.71	22.94	27.92	32.95	17.54	7.80	10.60	4.12	26.82	34.77	15.55
LCPO	37.00	41.43	31.24	21.91	26.07	**19.01**	8.25	9.67	**6.48**	25.34	30.68	**18.44**
BAMAS	37.74	46.99	24.96	26.42	33.42	16.37	7.64	9.63	4.75	26.24	34.19	14.74
BATS	39.96	48.87	29.97	26.25	33.93	17.82	8.01	10.51	5.09	26.74	35.20	16.07
BudgetThinker	40.20	47.19	29.00	20.81	32.10	16.08	7.64	9.75	5.10	26.38	34.36	15.44
AnytimeReasoner	39.69	48.18	31.56	22.90	28.87	18.03	8.51	11.25	5.61	25.32	33.74	17.58
BABLE	**41.59**	**48.97**	**31.90**	**28.83**	**36.75**	18.90	**9.66**	**12.40**	5.96	**28.46**	**36.15**	17.98
budget: tool-call												
Vanilla	33.58	42.90	8.45	21.58	28.03	13.19	6.93	9.08	1.61	25.32	32.76	7.68
GRPO	36.55	47.71	7.64	28.20	32.95	14.36	7.85	10.60	1.75	25.23	34.77	4.85
LCPO	36.38	41.43	**15.12**	24.64	26.07	**18.01**	8.28	9.67	**3.37**	25.44	30.68	4.85
BAMAS	36.55	46.99	10.47	28.79	33.42	16.10	7.80	9.63	2.68	25.69	34.19	7.19
BATS	34.54	48.87	3.35	27.14	33.93	12.19	8.12	10.51	2.16	25.49	35.20	6.17
BudgetThinker	35.33	47.19	8.43	26.53	32.10	13.26	7.56	9.75	2.05	25.19	34.36	6.32
AnytimeReasoner	29.77	48.18	5.23	18.10	28.87	7.96	7.91	11.25	2.84	18.26	33.74	4.02
BABLE	**37.57**	**48.97**	10.68	**31.12**	**36.75**	16.94	**9.82**	**12.40**	2.93	**26.85**	**36.15**	**8.34**

4 Experimental Evaluation

We evaluate our method's effectiveness, budget efficiency, and optimization behavior through three research questions: **RQ1:** How does our method perform compared with existing budget aware and reinforcement learning baselines across the entire budget range? **RQ2:** Can task-adaptive budget optimization reduce execution cost while preserving task performance? **RQ3:** Can likelihood-guided conditional advantage optimization sustain effective learning under sparse successes and tight budgets? Extended analyses are left in Appendices E–I.

Benchmarks. We evaluate on four multi-hop and agentic QA benchmarks, including MuSiQue (Trivedi et al., 2022), GAIA-text (Mialon et al., 2023), WebWalkerQA (Wu et al., 2025), and FRAMES (Krishna et al., 2025). All tasks are formulated as multi-turn search agent problems requiring iterative reasoning and tool use. See Appendix B for details.

Baselines and Setup. We use Qwen3-8B (Team et al., 2025) as the backbone model and evaluate two execution budgets: token-level budgets measured by generated tokens and tool-call budgets measured by tool invocations. Additional results on LLaMA-3.1-8B are provided in Appendix E. We compare BABLE with seven representative baselines from two groups. **(1) General baselines** include **Vanilla**, which directly evaluates the pretrained agent without training, **GRPO** (Guo et al., 2025), which optimizes task performance without explicit budget control, and **LCPO** (Aggarwal & Welleck, 2025), which introduces length penalty to encourage efficient generation. **(2) Budget-aware baselines** include **BAMAS** (Yang et al., 2026), which adapts agent execution strategies to the available budget, **BATS** (Liu et al., 2025), which exposes remaining budget information to guide planning and tool use, **BudgetThinker** (Wen et al., 2025), which learns budget aware behavior through budget signals and curriculum reinforcement learning, and **AnytimeReasoner** (Qi et al., 2026), which trains a single policy across varying budget conditions. We adapt all baselines to the same multi-turn search agent setting for fair comparison. Implementation details are provided in Appendix C.

Evaluation Metrics. We report accuracy (Acc.) under token level and tool-call level budgets. We evaluate three metrics: **Anytime Acc.**, which measures the average accuracy across the evaluated

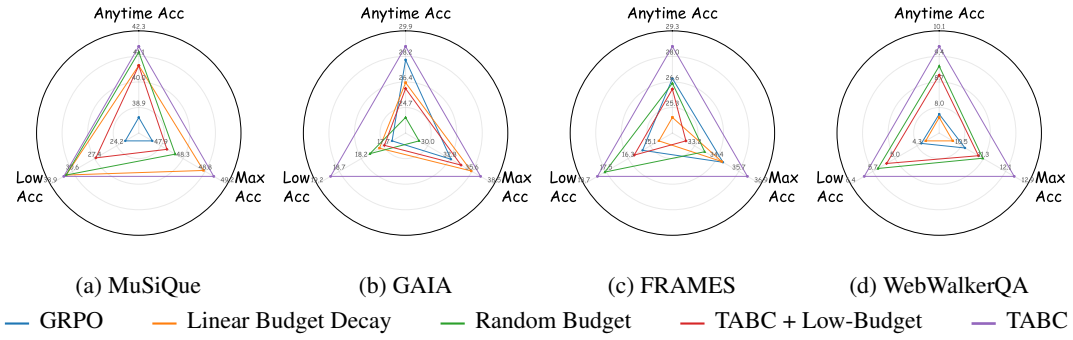


Figure 3: Comparison of different budget optimization strategies across four benchmarks. Each radar chart reports Anytime Accuracy, Max budget Accuracy, and Low budget Accuracy.

budget range, **Low budget Acc.**, which measures accuracy under a limited resource setting, and **Max budget Acc.**, which measures accuracy under the largest available budget. Detailed budget settings and evaluation protocols are provided in Appendix C.

4.1 RQ1: Overall Performance

To address **RQ1**, we compare BABLE with standard and budget aware baselines under token level and tool-call level budget constraints. Table 1 reports performance under anytime, maximum budget, and low budget settings across four benchmarks. We summarize the main observations as follows. **Obs. ① BABLE achieves consistently strong performance across budget conditions.** Under token-level budgets, BABLE achieves the highest Anytime Acc. on all four benchmarks, outperforming the strongest baseline by 0.91 to 1.64 points. The same pattern holds under tool-call budgets, with gains of 1.02 to 2.33 points over the best competing method. This consistency across two budget representations shows that BABLE improves performance across the evaluated budget range rather than under a specific notion of execution cost. **Obs. ② BABLE preserves performance when sufficient resources are available.** BABLE also achieves the highest Max Acc. on all four benchmarks under both budget representations, with gains of up to 2.82 points over the strongest baseline. Thus, improving performance across varying budgets does not sacrifice capability when sufficient resources are available. This suggests that the gains under constrained budgets do not come at the expense of full budget performance. **Obs. ③ Optimizing for a specific budget regime does not consistently improve the overall budget-performance profile.** Several baselines perform strongly under particular budget conditions but do not maintain the same advantage elsewhere. For example, LCPO obtains the best low budget accuracy in several settings, yet these gains do not consistently extend to anytime or maximum budget evaluations. In contrast, BABLE achieves the strongest anytime and maximum budget performance while remaining competitive under low budgets. This suggests that effective any budget learning should balance performance across the full budget range rather than specialize in a single regime.

> **Insight 1:** BABLE provides a consistently strong budget performance profile across budget representations and resource conditions, improving anytime performance while preserving full budget capability and remaining competitive under constrained budgets.

4.2 RQ2: Budget Scheduling

To address **RQ2**, we investigate how training budget schedules affect policy performance across budget conditions. We fix the policy optimization objective as LCAO and vary the budget scheduling strategy. We compare TABC with a fixed decreasing schedule, random budget sampling, and a low budget training stage after TABC. We also include unconstrained GRPO as a reinforcement learning baseline. This comparison isolates the effect of budget scheduling. **Obs. ① Preset budget schedules exhibit inconsistent behavior across tasks.** Fixed budget decay and random sampling can improve performance on some benchmarks while degrading it on others. This suggests that predefined budget schedules may not adapt well to heterogeneous task requirements. **Obs. ② TABC provides the most consistent performance across budget regimes.** TABC achieves the best result across all 12

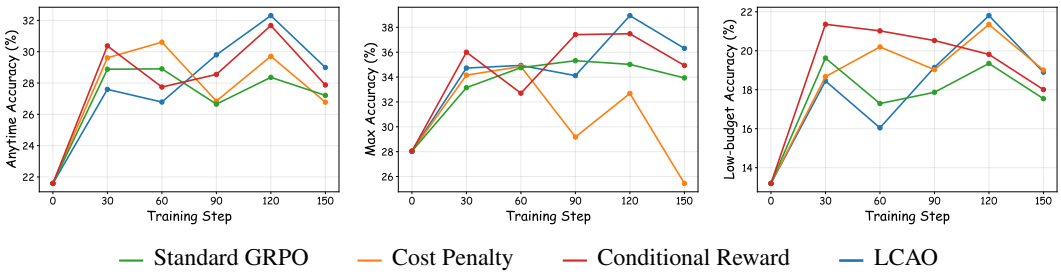


Figure 4: Training dynamics of learning signals on GAIA with Qwen3-8B under budget constraints. We compare Standard GRPO, Cost Penalty, Conditional Reward, and LCAO across training steps. The subfigures report Anytime Accuracy, Max budget Accuracy, and Low budget Accuracy, respectively.

benchmark metric comparisons, covering anytime, maximum budget, and low budget performance. This shows that task-adaptive scheduling provides a more consistent budget performance profile than preset or random budgets. **Obs. ③ Further specialization to a low budget regime does not improve the trade-off.** Adding a low budget training stage after TABC consistently degrades performance, including low budget accuracy across all four benchmarks. This indicates that further specialization to a tight budget regime can disrupt the broader budget-performance profile learned by TABC.

> **Insight 2:** Effective budget aware training depends on how training budgets are adapted. Compared with preset or randomly sampled schedules, TABC maintains more consistent performance across budget regimes, while further specialization to a low budget can be harmful.

4.3 RQ3: Learning Optimization

To address **RQ3**, we investigate how different learning signals affect optimization under sparse successes and tight budget constraints. We compare Standard GRPO, Cost Penalty, Conditional Reward, and LCAO. Conditional Reward and LCAO both use TABC for budget scheduling. The methods differ in how budget information is incorporated into the learning signal, while their training dynamics are evaluated using anytime, maximum budget, and low budget performance. **Obs. ① Budget aware learning signals outperform unconstrained rewards.** Both Cost Penalty and Conditional Reward improve over Standard GRPO, showing that incorporating budget information can better guide policy optimization. However, their training dynamics remain less stable under tighter budget conditions. **Obs. ② Sparse successes limit the effectiveness of conditional rewards.** Conditional Reward improves budget alignment by focusing on executions that satisfy the budget constraint, but its performance becomes unstable as successful trajectories become sparse. This suggests that binary budget-aware feedback alone may provide an insufficient learning signal in low success regimes. **Obs. ③ LCAO provides more stable optimization across budget regimes.** LCAO maintains stronger performance across anytime, maximum budget, and low budget evaluations throughout training. These results indicate that likelihood-guided advantage weighting more effectively exploits sparse successful trajectories under constrained budgets.

> **Insight 3:** Effective budget aware learning requires not only incorporating budget information, but also extracting learning signals from sparse successes. LCAO strengthens policy optimization in such low success regimes and provides more stable learning across budget conditions.

5 Conclusions

In this work, we study any budget agent learning and transform its objective into minimizing the minimum sufficient budget required by individual tasks, which naturally leads to a bilevel optimization formulation. We propose BABLE, an alternating optimization framework that solves this problem through task-adaptive budget scheduling with TABC and sparse success policy optimization with LCAO. Experiments across four agent benchmarks under both token level and tool-call level budget constraints demonstrate that BABLE improves budget efficiency while maintaining strong performance across the entire budget range.

AI USE STATEMENT

Generative AI tools were used to improve grammar, spelling, and word choice during manuscript editing. The authors reviewed all AI-assisted edits and take full responsibility for the final content of this work.

REFERENCES

Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*, 2025.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Daman Arora and Andrea Zanette. Training language models to reason efficiently. *Advances in Neural Information Processing Systems*, 38:60770–60808, 2026.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*, 2023.

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *International conference on machine learning*, pp. 1568–1577. PMLR, 2018.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24842–24855, 2025.

Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqui. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4745–4759, 2025.

M Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. *Advances in neural information processing systems*, 23, 2010.

Junyan Li, Wenshuo Zhao, Yang Zhang, and Chuang Gan. Steering llm thinking with budget guidance. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 37437–37451, 2026.

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models. *arXiv preprint arXiv:2310.10505*, 2023.

Yuxiang Lin, Zihan Wang, Mengyang Liu, Yuxuan Shan, Longju Bai, Junyao Zhang, Xing Jin, Boshan Chen, Jinyan Su, Xingyao Wang, et al. Bagen: Are llm agents budget-aware? *arXiv preprint arXiv:2606.00198*, 2026.

Zehui Ling, Deshu Chen, Hongwei Zhang, Yifeng Jiao, Xin Guo, and Yuan Cheng. Fast on the easy, deep on the hard: Efficient reasoning via powered length penalty. *arXiv preprint arXiv:2506.10446*, 2025.

Tengxiao Liu, Zifeng Wang, Jin Miao, I Hsu, Jun Yan, Jiefeng Chen, Rujun Han, Fangyuan Xu, Yanfei Chen, Ke Jiang, et al. Budget-aware tool-use enables effective agent scaling. *arXiv preprint arXiv:2511.17006*, 2025.

Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *International conference on artificial intelligence and statistics*, pp. 1540–1552. PMLR, 2020.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*, 2023.

Penghui Qi, Zichen Liu, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Optimizing anytime reasoning via budget relative policy optimization. *Advances in Neural Information Processing Systems*, 38:23429–23451, 2026.

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Fahim Tajwar, Guanning Zeng, Yueer Zhou, Yuda Song, Daman Arora, Yiding Jiang, Jeff Schneider, Ruslan Salakhutdinov, Haiwen Feng, and Andrea Zanette. Maximum likelihood reinforcement learning. *arXiv preprint arXiv:2602.02710*, 2026.

Qwen Team et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 6(7):13, 2025.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Hao Wen, Xinrui Wu, Yi Sun, Feifei Zhang, Liye Chen, Jie Wang, Yunxin Liu, Yunhao Liu, Ya-Qin Zhang, and Yuanchun Li. Budgetthinker: Empowering budget-aware llm reasoning with control tokens. *arXiv preprint arXiv:2508.17196*, 2025.

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

Yong Wu, YanZhao Zheng, TianZe Xu, ZhenTao Zhang, YuanQiang Yu, JiHuai Zhu, Chao Ma, BinBin Lin, BaoHua Dong, HangCheng Zhu, et al. Contextbudget: Budget-aware context management for long-horizon search agents. *arXiv preprint arXiv:2604.01664*, 2026.

Violet Xiang, Chase Blagden, Rafael Rafailov, Nathan Lile, Sang Truong, Chelsea Finn, and Nick Haber. Just enough thinking: Efficient reasoning with adaptive length penalties reinforcement learning. *arXiv preprint arXiv:2506.05256*, 2025.

Liming Yang, Junyu Luo, Xuanzhe Liu, Yiling Lou, and Zhenpeng Chen. Bamas: Structuring budget-aware multi-agent systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 29802–29810, 2026.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Danlong Yuan, Tian Xie, Shaohan Huang, Zhuocheng Gong, Huishuai Zhang, Chong Luo, Furu Wei, and Dongyan Zhao. Efficient rl training for reasoning models via length-aware optimization. 2025.

Xuanming Zhang, Shwan Ashrafi, Aziza Mirsaidova, Amir H Rezaeian, Miguel Ballesteros, Lydia Chilton, Zhou Yu, and Dan Roth. Budget-aware anytime reasoning with llm-synthesized preference data. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 8587–8599, 2026.

Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, et al. Absolute zero: Reinforced self-play reasoning with zero data, 2025. *URL https://arxiv. org/abs/2505.03335*, 27, 2025.

Appendix

A **Algorithm Workflow** 14

B **Information of Training and Test Dataset** 14

C **Implementation Details** 15

D **Discussion of the Budget Monotonicity** 15

E **Generalization to LLaMA-3.1-8B** 16

F **Analysis of Budget Recovery** 17

G **Statistical Uncertainty and Robustness** 18

H **Analysis of Budget Adaptation and Advantage Optimization** 18

I **Analysis of Budget-Performance Trade-off and Sufficient Budgets** 19

J **System Prompts** 25

A Algorithm Workflow

Algorithm 1 BABLE: Bilevel Any-Budget Learning for Agents

Input: Task distribution $\mathcal{D}$; initial policy π_{θ_1}; maximum budget B; reliability level α; group size G; stabilization constant ϵ; clipping parameter ϵ_c.
Output: Optimized policy π_θ and task-specific budgets $\{b(x)\}$.

Initialize $b_1(x) \leftarrow B$ for each task x
for $t \leftarrow 1$ **to** T **do**
  Initialize $\mathcal{J}_{\mathrm{BABLE}}(\theta) \leftarrow 0$
  foreach *task* x *sampled at stage* t **do**
    // Sample once under the maximum budget
    Sample $\mathcal{R}_t(x) = \{\tau_i\}_{i=1}^{G}$ by executing $\pi_{\theta_t}(\cdot \mid x)$ with truncation budget $b_t(x)$
    // Lower level: Task-Adaptive Budget Controller
    $\mathcal{C}_t(x) \leftarrow \{c(\tau_i) \mid \tau_i \in \mathcal{R}_t(x),\, r(x, \tau_i) = 1\}$
    $M_t(x) \leftarrow |\mathcal{C}_t(x)|, \quad k_t(x) \leftarrow \lceil \alpha G \rceil$
    Sort $c^t_{(1)}(x) \le \cdots \le c^t_{(M_t(x))}(x)$
    if $M_t(x) \ge k_t(x)$ **then**
      $b_{t+1}(x) \leftarrow c^t_{(k_t(x))}(x)$
    else
      $b_{t+1}(x) \leftarrow b_t(x)$
    // Upper level: Likelihood-Guided Conditional Advantage Optimization
    for $i \leftarrow 1$ **to** G **do**
      $\tau_i^{(b_{t+1})} \leftarrow \mathrm{Truncate}(\tau_i, b_{t+1}(x))$
      $r_i \leftarrow r\left(x, \tau_i^{(b_{t+1})}\right)$
    $K \leftarrow \sum_{i=1}^{G} r_i$
    for $i \leftarrow 1$ **to** G **do**
      $\hat{A}_i^{\mathrm{LCAO}} \leftarrow \dfrac{G r_i - K}{K + G\epsilon}$
      foreach *interaction step* k *in* $\tau_i^{(b_{t+1})}$ **do**
        $h_{i,k} \leftarrow (s_{i,k-1}, x)$
        $\omega_{i,k}(\theta) \leftarrow \dfrac{\pi_\theta(a_{i,k} \mid h_{i,k})}{\pi_{\theta_t}(a_{i,k} \mid h_{i,k})}, \quad \bar{\omega}_{i,k}(\theta) \leftarrow \mathrm{clip}\left(\omega_{i,k}(\theta), 1-\epsilon_c, 1+\epsilon_c\right)$
        $L_{i,k}(\theta) \leftarrow \min\left(\omega_{i,k}(\theta)\hat{A}_i^{\mathrm{LCAO}}, \bar{\omega}_{i,k}(\theta)\hat{A}_i^{\mathrm{LCAO}}\right)$
        $\mathcal{J}_{\mathrm{BABLE}}(\theta) \leftarrow \mathcal{J}_{\mathrm{BABLE}}(\theta) + L_{i,k}(\theta)$
  // Policy update
  $\theta_{t+1} \leftarrow \theta_t + \eta \nabla_\theta \mathcal{J}_{\mathrm{BABLE}}(\theta)\big|_{\theta=\theta_t}$
return $\pi_{\theta_{T+1}}$ *and* $\{b_{T+1}(x)\}$

B Information of Training and Test Dataset

Benchmarks. We consider four benchmarks covering multi-hop reasoning, tool assisted interaction, and structured decision making:

- **MuSiQue** Trivedi et al. (2022): a multi-hop question answering benchmark constructed by composing connected single hop questions, designed to require genuine multi step reasoning.
- **GAIA-text** Mialon et al. (2023): evaluates general AI assistants on real world questions requiring reasoning, web browsing, and tool use proficiency.
- **WebWalkerQA** Wu et al. (2025): benchmarks multi-step web traversal and evidence aggregation through structured navigation.
- **FRAMES** Krishna et al. (2025): a unified benchmark for evaluating factuality, retrieval, and reasoning in end-to-end retrieval augmented generation.

Training and Test Splits. We train our models only on 5,975 MuSiQue instances and evaluate them on a held-out MuSiQue test set. For GAIA-text, WebWalkerQA, and FRAMES, we use no

training data and directly evaluate the trained models on their released test sets following the official evaluation protocols. The strong performance across these unseen benchmarks demonstrates the cross-benchmark generalization of our method.

Data Release. To ensure reproducibility and facilitate future research, we will release all training and test splits used in our experiments, together with preprocessing scripts and evaluation protocols, upon publication.

C Implementation Details

Training Setup. Our optimization framework leverages pretrained weights from the *Qwen-3-8B* and *LLaMA-3.1-8B* instruction-tuned models. We employ the AdamW optimizer with a fixed learning rate of 10^{-5} and a global batch size of 256. To enhance trajectory diversity, we generate 8 rollouts per input instance. Training is conducted for 150 gradient steps on 8×A100 GPUs with data parallelism. Hyperparameters remain consistent across benchmark evaluations unless otherwise noted. We set the reliability level to $\alpha = 0.25$.

Tool Execution. Environment interactions are mediated through a service interface built on FastMCP. The agent directly invokes external tools through this interface. For web exploration, the agent accesses a Baidu based search service, providing a pipeline of *baidu_search* and webpage scraping for realtime information retrieval.

Evaluation Metrics. We report accuracy (Acc.) under token-level and tool-call budgets. For token-level budgets, **Anytime Acc.** is averaged over thirteen budget levels from 2K to 8K tokens with 500-token intervals, while **Low budget Acc.** and **Max budget Acc.** are measured at 3K and 10K tokens, respectively. For tool-call budgets, **Anytime Acc.** is averaged over five call limits from 1 to 5 calls, with **Low budget Acc.** and **Max budget Acc.** corresponding to 1 call and the unconstrained setting, respectively.

Our theoretical formulation and empirical evaluation characterize complementary aspects of any-budget capability. The reliability threshold α defines when a task is sufficiently solvable, and Eq. (8) relates the resulting solvability objective to the expected minimum sufficient budget under budget monotonicity. Directly estimating this quantity during large-scale evaluation is costly, since it requires repeated executions of each task across multiple budgets to estimate task-level success probabilities. We therefore use Anytime Accuracy as the primary practical metric, which can be estimated from binary execution outcomes across the evaluated budgets without precisely estimating each task's success probability. To further connect the evaluation with our formulation, we additionally estimate the empirical minimum sufficient budget on GAIA through repeated executions at $\alpha = 0.25$. BABLE achieves lower sufficient budgets under both token and tool-call constraints across different backbones. Details are provided in Appendix I.

D Discussion of the Budget Monotonicity

Our any-budget formulation adopts the budget monotonicity assumption

$$p_\theta(x, b_1) \le p_\theta(x, b_2), \qquad b_1 \le b_2,$$

which states that increasing the available execution budget should not reduce the probability of successfully completing a task. Here, we provide additional intuition for this assumption under an early-stopping execution mechanism.

We consider the execution setting in which the budget acts as a hard upper bound on the amount of computation or interaction available to the agent. Importantly, the budget is not treated as an additional policy input. Under different budget settings, the agent follows the same policy π_θ and generates the same trajectory prefix until the execution terminates or reaches the budget limit. A larger budget only extends the maximum allowed execution length without changing the policy behavior or modifying previously generated actions. Therefore, an execution under a smaller budget can be viewed as a prefix of the execution under a larger budget.

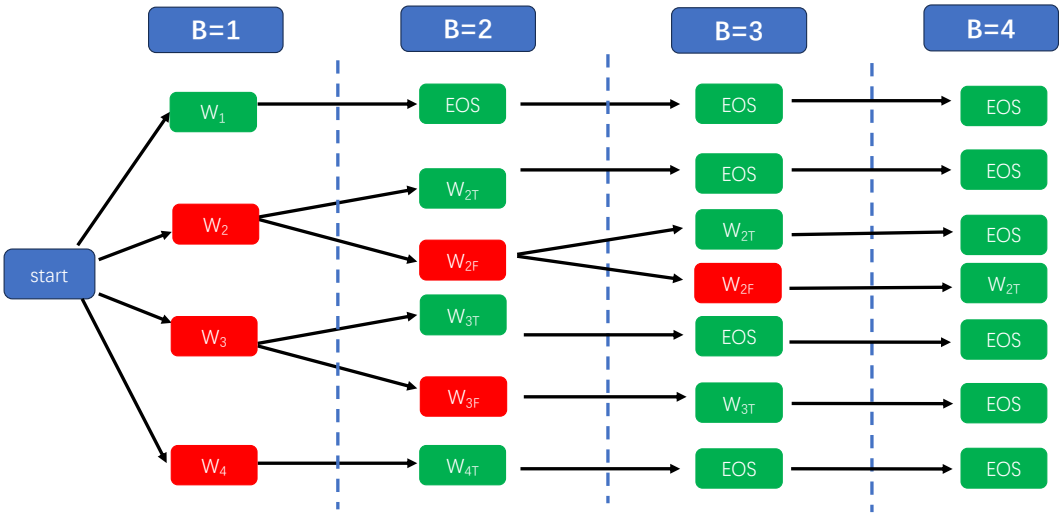


Figure 5: Illustration of budget monotonicity under early stopping. Green nodes denote successful executions, while red nodes denote executions that have not yet succeeded. Once an execution succeeds, it terminates with `EOS` and remains successful under larger budgets. In contrast, an unsuccessful execution may either remain unsuccessful or become successful when additional budget is available. Thus, increasing the execution budget preserves previous successes while providing additional opportunities for success.

As illustrated in Figure 5, an execution that has already successfully completed the task terminates through early stopping. Once a successful state is reached, additional budget does not alter its outcome, and the subsequent execution is represented by an `EOS` state. In contrast, an execution that has not yet succeeded may either remain unsuccessful or become successful when additional computation or interaction is available.

From this perspective, increasing the budget preserves executions that have already succeeded while providing additional opportunities for previously unsuccessful executions to succeed. Let $\mathcal{C}_b(x)$ denote the event that task x has been successfully completed within budget b. Under the early stopping execution mechanism illustrated in Figure 5, the successful events naturally satisfy

$$\mathcal{C}_{b_1}(x) \subseteq \mathcal{C}_{b_2}(x), \qquad b_1 < b_2.$$

This nested structure provides the intuition behind the budget monotonicity assumption

$$p_\theta(x, b_1) \leq p_\theta(x, b_2), \qquad b_1 < b_2.$$

This interpretation applies to the setting where budget controls only the maximum execution length while the underlying policy remains unchanged. Increasing the budget therefore extends the same execution process, and early stopping preserves trajectories that have already completed the task. This corresponds to the role of the budget in our any-budget formulation, where it specifies the maximum amount of computation or tool interaction available during execution.

E Generalization to LLaMA-3.1-8B

To evaluate the generalization capability of BABLE beyond the primary backbone, we further conduct experiments with *LLaMA-3.1-8B-Instruct* as the base model. All training configurations and optimization procedures are kept consistent with those used for the Qwen3-8B experiments, except for the backbone initialization. We evaluate each method under both token-level and tool-call level budget constraints across four agent benchmarks, including MuSiQue, GAIA, WebWalkerQA, and FRAMES.

Table 2 reports the performance comparison. Following the main experiments, **Any.** denotes the average accuracy over the evaluated budget range, while **Max.** denotes the performance without

Table 2: Performance comparison on *LLaMA-3.1-8B* across the evaluated budget range under token-level and tool-call level constraints. **Any.** denotes the average accuracy across all evaluated budgets, **Max.** denotes the accuracy without budget constraints, and **Low.** denotes the accuracy under a 3K token budget or a 2-call budget, respectively. For token-level budgets, **Any.** is averaged from 2K to 8K tokens with an interval of 500 tokens. The best result in each column is highlighted in bold.

Methods	**MuSiQue**			**GAIA**			**WebWalkerQA**			**FRAMES**		
	Any.	**Max.**	**Low.**	**Any.**	**Max.**	**Low.**	**Any.**	**Max.**	**Low.**	**Any.**	**Max.**	**Low.**
backbone: LLaMA-3.1-8B, budget: token												
Vanilla	18.11	29.24	9.01	8.76	13.34	5.31	4.81	7.09	2.68	11.96	19.54	5.45
GRPO	24.54	37.56	13.30	12.08	21.03	4.78	5.38	8.35	2.61	14.31	23.82	4.97
LCPO	0.00	0.00	0.00	0.00	0.00	0.00	0.22	0.22	0.22	0.00	0.00	0.00
BAMAS	20.88	30.28	13.04	7.68	13.44	3.68	4.67	6.26	3.58	9.13	15.39	4.43
BATS	24.27	37.55	12.89	11.43	20.54	4.04	4.84	7.45	2.68	12.68	21.73	5.41
BudgetThinker	28.31	39.41	20.21	11.01	16.95	**7.94**	4.99	6.65	3.54	14.71	23.55	8.46
AnytimeReasoner	28.10	35.38	**21.35**	8.19	11.47	5.92	5.07	6.59	**3.64**	16.20	20.67	**11.74**
BABLE	**31.40**	**43.81**	20.91	**13.11**	**21.98**	7.39	**5.80**	**8.39**	3.28	**17.98**	**29.28**	9.26
backbone: LLaMA-3.1-8B, budget: tool-call												
Vanilla	13.61	29.24	1.02	6.23	13.34	0.00	3.91	7.09	0.90	8.42	19.54	0.83
GRPO	20.94	37.56	2.03	10.54	21.03	0.00	4.97	8.35	0.79	12.33	23.82	0.95
LCPO	0.00	0.00	0.00	0.00	0.00	0.00	0.22	0.22	0.22	0.00	0.00	0.00
BAMAS	16.11	30.28	2.04	5.81	13.44	1.26	3.90	6.26	1.12	7.32	15.39	0.83
BATS	21.16	37.55	3.39	10.02	20.54	1.27	4.24	7.45	0.67	11.27	21.73	1.78
BudgetThinker	26.66	39.41	3.47	11.10	16.95	3.20	4.74	6.65	**2.16**	14.39	23.55	1.68
AnytimeReasoner	24.69	35.38	**5.14**	6.90	11.47	1.89	4.52	6.59	1.41	13.85	20.67	**3.58**
BABLE	**27.19**	**43.81**	3.84	**11.22**	**21.98**	**3.25**	**5.14**	**8.39**	1.48	**15.27**	**29.28**	2.28

budget constraints. For token-level budgets, the **Any.** metric is computed over budgets from 2K to 8K tokens with an interval of 500 tokens, while for tool-call budgets it is computed over the corresponding evaluated call limits. The **Low.** metric represents performance under a strict resource constraint, using a 3K-token budget for token-level evaluation and a 2-call budget for tool-call evaluation.

Across both budget settings, BABLE consistently achieves competitive performance under varying resource constraints. In particular, BABLE obtains the highest average accuracy on most benchmarks, demonstrating that the proposed budget-aware learning framework remains effective when transferred to a different language model backbone. Moreover, the improvements under low resource settings indicate that BABLE can better preserve task performance when the available computation budget is limited.

LCPO exhibits a degenerate reward hacking behavior on *LLaMA-3.1-8B-Instruct*: under the cost penalty, the policy collapses to very short refusal like responses that minimize execution cost without solving the task. This results in near zero accuracy on most benchmarks and illustrates a potential failure mode of direct cost penalized optimization.

F Analysis of Budget Recovery

We further examine whether allowing task-specific budgets to increase after contraction improves optimization. We consider a recovery-enabled variant, denoted as SLACK. Unlike TABC, SLACK samples trajectories under the maximum budget B. Let $k = \lceil \alpha G \rceil$ and $c_t^{(1)}(x) \le \cdots \le c_t^{(M_t(x))}(x)$ denote the sorted costs of successful executions. The budget is updated as

$$b_{t+1}(x) = \begin{cases} c_t^{(k)}(x), & M_t(x) \ge k, \\ b_t(x), & M_t(x) < k. \end{cases} \tag{20}$$

Since executions are sampled under B, $c_t^{(k)}(x)$ may exceed the current budget $b_t(x)$, allowing the budget to increase and recover from a potentially over-contracted state.

Table 3: Comparison between the recovery enabled SLACK variant and BABLE under token budgets.

Method	MuSiQue			GAIA			WebWalkerQA		
	Any.	Max.	Low.	Any.	Max.	Low.	Any.	Max.	Low.
SLACK	40.23	47.19	28.49	25.15	35.28	17.31	9.05	11.39	5.43
BABLE	**41.59**	**48.97**	**31.90**	**28.83**	**36.75**	**18.90**	**9.66**	**12.40**	**5.96**

Table 4: Performance comparison in terms of average accuracy across all evaluated budgets. Results are reported as mean ± standard deviation over three random seeds.

Methods	MuSiQue	GAIA	WebWalkerQA	FRAMES
budget: token				
Vanilla	35.12 ± 0.46	21.72 ± 0.20	6.77 ± 0.35	24.37 ± 0.31
GRPO	38.41 ± 0.52	27.92 ± 0.49	7.80 ± 0.42	26.82 ± 0.23
LCPO	37.00 ± 0.43	21.91 ± 0.20	8.25 ± 0.31	25.34 ± 0.48
BAMAS	37.74 ± 0.20	26.42 ± 0.30	7.64 ± 0.47	26.24 ± 0.51
BATS	39.96 ± 0.31	26.25 ± 0.53	8.01 ± 0.57	26.74 ± 0.18
BudgetThinker	40.20 ± 0.56	20.81 ± 0.50	7.64 ± 0.38	26.38 ± 0.27
AnytimeReasoner	39.69 ± 0.55	22.90 ± 0.38	8.51 ± 0.24	25.32 ± 0.24
BABLE	**41.59 ± 0.49**	**28.83 ± 0.54**	**9.66 ± 0.56**	**28.46 ± 0.52**
budget: tool-call				
Vanilla	33.58 ± 0.50	21.58 ± 0.46	6.93 ± 0.41	25.32 ± 0.51
GRPO	36.55 ± 0.48	28.20 ± 0.55	7.85 ± 0.50	25.23 ± 0.53
LCPO	36.38 ± 0.50	24.64 ± 0.21	8.28 ± 0.32	25.44 ± 0.35
BAMAS	36.55 ± 0.23	28.79 ± 0.32	7.80 ± 0.24	25.69 ± 0.35
BATS	34.54 ± 0.56	27.14 ± 0.40	8.12 ± 0.40	25.49 ± 0.31
BudgetThinker	35.33 ± 0.34	26.53 ± 0.54	7.56 ± 0.57	25.19 ± 0.55
AnytimeReasoner	29.77 ± 0.28	18.10 ± 0.52	7.91 ± 0.28	18.26 ± 0.41
BABLE	**37.57 ± 0.57**	**31.12 ± 0.56**	**9.82 ± 0.51**	**26.85 ± 0.59**

As shown in Table 3, allowing budget recovery does not improve the budget-performance profile. BABLE achieves higher Anytime, Max budget, and Low budget accuracy on all three benchmarks. In particular, BABLE improves Anytime Accuracy by 1.36 points on MuSiQue, 3.68 points on GAIA, and 0.61 points on WebWalkerQA. These results suggest that immediately relaxing the budget in response to rollout variation can undo previously established training constraints. In contrast, the monotonic update in TABC preserves tighter budget targets and provides a more stable curriculum for policy optimization.

G STATISTICAL UNCERTAINTY AND ROBUSTNESS

Statistical robustness. Table 4 reports average accuracy over three random seeds. BABLE achieves the highest mean performance across all four benchmarks under both token and tool-call budgets, while exhibiting standard deviations comparable to those of the baselines. These results indicate that performance gains are consistent across random seeds rather than arising from a single run.

H ANALYSIS OF BUDGET ADAPTATION AND ADVANTAGE OPTIMIZATION

Analysis of Budget Adaptation and Advantage Optimization To examine the contributions of budget adaptation and advantage optimization, we compare four training variants: (1) fixed budget with GRPO (FixedGRPO), (2) fixed budget with LCAO (FixedLCAO), (3) TABC with GRPO (TABC-GRPO), and (4) TABC with LCAO, corresponding to the full BABLE framework. These variants form a two factor comparison of the budget scheduling strategy and the policy update rule. All four variants use Qwen3-8B and are evaluated under the same token-level and tool-call budget settings.

MaxRL develops likelihood-guided policy optimization based on task conditioned success probabilities. BABLE extends this perspective to task-specific execution constraints by considering the budget-conditioned success probability $p_\theta(x, b_t(x))$, where $b_t(x)$ specifies the token or tool-call budget available to an execution. This execution budget is distinct from the number of rollouts used to estimate a likelihood-guided policy update.

In BABLE, the execution budget is a task-specific quantity that evolves with the policy. TABC adapts $b_t(x)$ using successful executions sampled from the current policy, and LCAO improves the policy through the corresponding budget conditioned log-success surrogate, $\mathbb{E}_{x\sim\mathcal{D}}[\log p_\theta(x, b_t(x))]$. Subsequent policy updates change the success evidence available for further budget adaptation. BABLE thus couples likelihood-guided policy learning with adaptive control of the execution conditions under which learning takes place.

FixedLCAO serves as a control for this likelihood-guided update under a fixed training budget. Comparing FixedLCAO with BABLE examines the additional benefit of task-adaptive budget scheduling while retaining the same update rule. This analysis therefore focuses on the empirical contribution of TABC and its combination with likelihood-guided policy optimization.

Table 5 shows that introducing TABC improves Anytime Accuracy across all four datasets and both budget types, for both GRPO and LCAO updates. In particular, relative to FixedLCAO, BABLE improves Anytime Accuracy by 0.85–3.63 percentage points under token budgets and 2.92–4.99 percentage points under tool-call budgets. These comparisons support the benefit of adapting training budgets at the task level for average performance over the evaluated budget range.

The effect of replacing GRPO with LCAO depends on the budget schedule and evaluation metric. Under fixed budgets, LCAO improves Anytime Accuracy in six of the eight dataset budget combinations, with FRAMES being the exception under both budget types. Specifically, FRAMES Anytime Accuracy decreases from 26.38 to 24.83 under token budgets and from 23.26 to 21.86 under tool-call budgets. Changes in Max budget and Low budget Accuracy are also mixed under fixed budgets. These results indicate that mean reward normalization alone does not yield uniform improvements across datasets and operating points.

With TABC, however, replacing GRPO with LCAO improves Anytime Accuracy in all eight comparisons and matches or improves the reported Max budget and Low budget accuracies. The contrast between the fixed budget and adaptive budget results suggests that the effectiveness of the likelihood-guided update depends in part on the budget conditions encountered during training.

BABLE achieves the highest reported Anytime and Max budget accuracies in all eight dataset budget combinations, together with the highest or joint highest Low budget Accuracy. The joint optimum at the low token budget on WebWalkerQA is shared with TABC-GRPO. Taken together, these results provide empirical support for combining task-adaptive budget scheduling with likelihood-guided policy updates in the evaluated setting, while also revealing the limitations of using either component in isolation.

I Analysis of Budget-Performance Trade-off and Sufficient Budgets

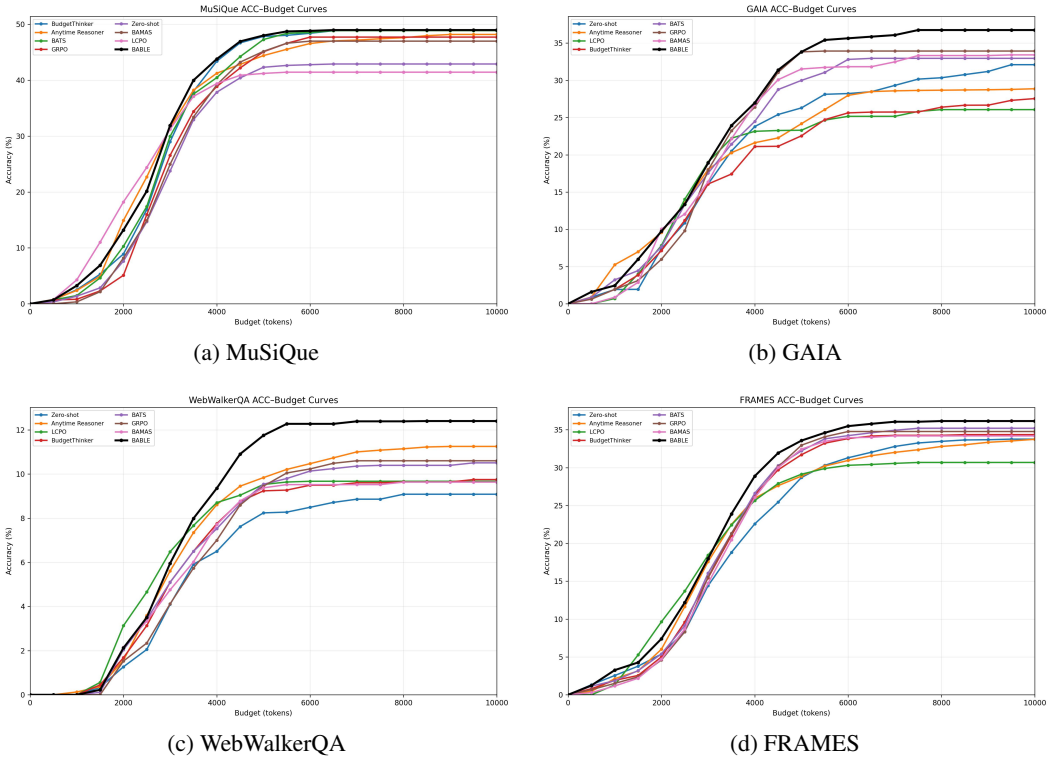
We provide additional analysis to characterize the relationship between accuracy and budget, as well as the empirical computation cost required under different resource constraints. Experiments are conducted with both Qwen3-8B and LLaMA-3.1-8B backbones on MuSiQue, GAIA, WebWalkerQA, and FRAMES. We consider two types of execution budgets: token budgets and tool call budgets.

For each dataset, the accuracy budget curve measures the achievable accuracy under different available budgets. A higher curve indicates that a method achieves better performance under the same resource constraint. In addition, we analyze the empirical completion budget to quantify the computation required for successful task completion.

Accuracy Budget Curves Figures 6 and 7 present the accuracy budget curves of Qwen3-8B under token budgets and tool call budgets, respectively. The corresponding results on LLaMA-3.1-8B are shown in Figures 8 and 9.

Table 5: Component analysis of budget adaptation and advantage optimization. FixedGRPO and FixedLCAO use a fixed training budget with GRPO and LCAO updates, respectively. TABC-GRPO uses task-adaptive budgets with GRPO, while BABLE combines TABC with LCAO. Any., Max., and Low. denote Anytime, Max budget, and Low budget accuracy (%), respectively. Bold entries indicate the highest reported value for each dataset and metric within each budget setting, including ties.

Methods	**MuSiQue**			**GAIA**			**WebWalkerQA**			**FRAMES**		
	Any.	**Max.**	**Low.**	**Any.**	**Max.**	**Low.**	**Any.**	**Max.**	**Low.**	**Any.**	**Max.**	**Low.**
backbone: Qwen3-8B, budget: token												
FixedGRPO	40.63	48.54	29.47	20.36	27.92	15.61	7.91	9.31	5.56	25.89	34.81	15.02
FixedLCAO	40.74	48.76	31.56	26.37	35.55	17.82	7.71	10.15	3.98	24.83	34.70	14.67
TABC-GRPO	40.36	48.64	30.56	26.60	35.22	18.14	9.36	11.98	**5.96**	27.04	35.61	15.01
BABLE	**41.59**	**48.97**	**31.90**	**28.83**	**36.75**	**18.90**	**9.66**	**12.40**	**5.96**	**28.46**	**36.15**	**17.98**
backbone: Qwen3-8B, budget: tool-call												
FixedGRPO	32.74	48.54	8.31	22.93	27.92	14.29	7.21	9.31	2.48	22.82	34.81	6.79
FixedLCAO	34.00	48.76	8.14	26.66	35.55	11.78	6.90	10.15	1.52	21.86	34.70	4.96
TABC-GRPO	33.93	48.64	7.48	27.03	35.22	15.75	8.44	11.98	2.61	23.96	35.61	5.94
BABLE	**37.57**	**48.97**	**10.68**	**31.12**	**36.75**	**16.94**	**9.82**	**12.40**	**2.93**	**26.85**	**36.15**	**8.34**



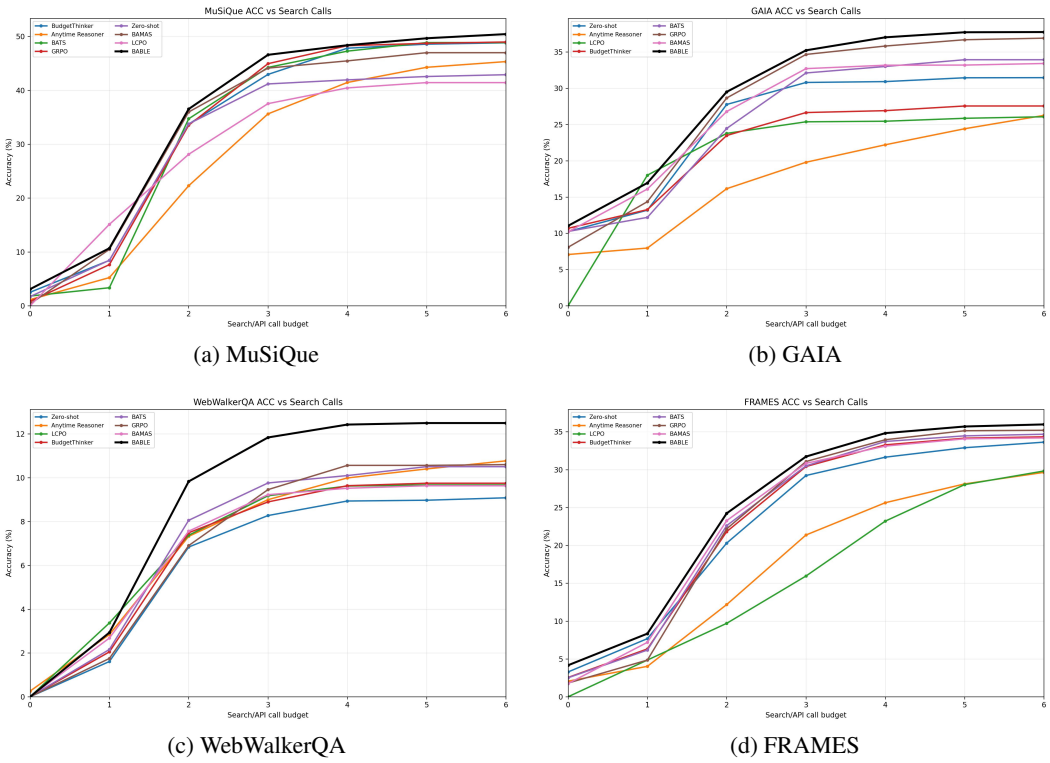
(a) MuSiQue (b) GAIA

(c) WebWalkerQA (d) FRAMES

Figure 6: Accuracy under different token budgets on Qwen3-8B. Higher curves indicate better performance under the same token constraint.

Across different backbones and budget settings, BABLE maintains strong performance over a broad range of operating budgets, demonstrating its ability to allocate computation according to task difficulty rather than relying on a fixed resource allocation.

Empirical Minimum Sufficient Budget Analysis To directly evaluate the minimum sufficient budget in our formulation, we empirically estimate $b^*(x;\alpha)$ at $\alpha = 0.25$. For each task x_i and method



(a) MuSiQue (b) GAIA

(c) WebWalkerQA (d) FRAMES

Figure 7: Accuracy under different tool-call budgets on Qwen3-8B. Higher curves indicate better performance under the same tool-interaction constraint.

m, we perform $G = 8$ independent executions under the maximum budget B. Let $r_{m,i,j} \in \{0, 1\}$ denote the outcome of the j-th execution and $c_{m,i,j}$ its execution cost.

We sort the costs of successful executions as

$$c_{m,i}^{(1)} \leq \cdots \leq c_{m,i}^{(K_{m,i})}, \qquad K_{m,i} = \sum_{j=1}^{G} r_{m,i,j}. \tag{21}$$

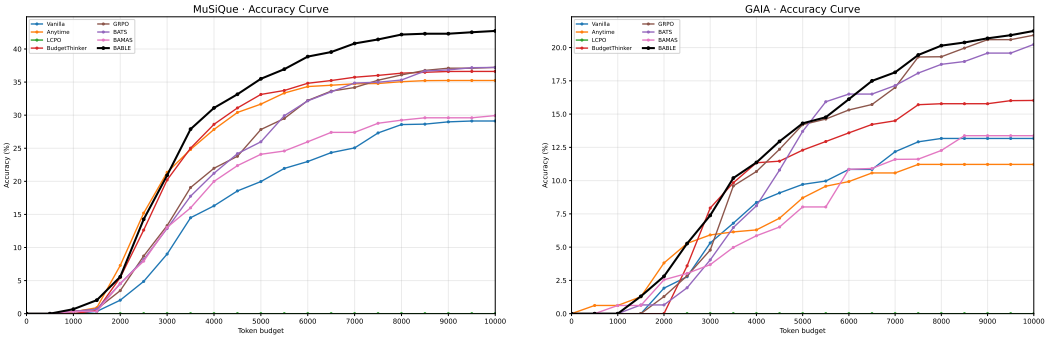
Following the finite-sample criterion used by TABC, we set $k = \lceil \alpha G \rceil = 2$ and estimate

$$\hat{b}_m^*(x_i; \alpha) = \begin{cases} c_{m,i}^{(k)}, & K_{m,i} \geq k, \\ B, & K_{m,i} < k. \end{cases} \tag{22}$$

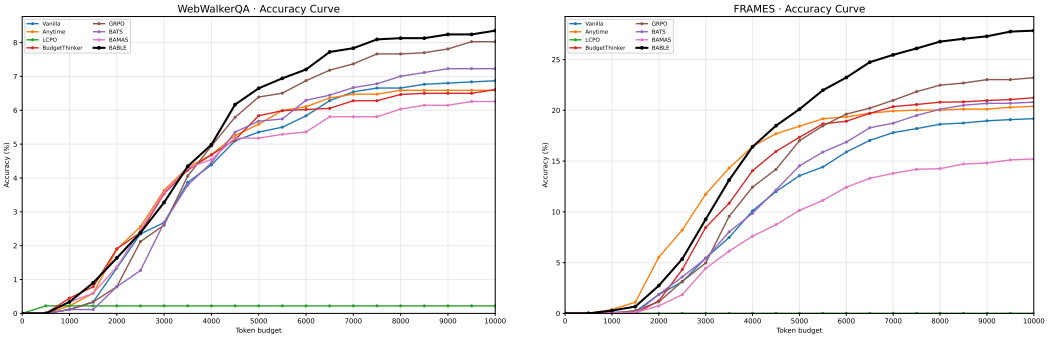
Thus, the estimate corresponds to the second smallest successful execution cost, while tasks with insufficient successful executions are assigned the maximum budget.

To account for differences in task success coverage, we further normalize the empirical sufficient budgets by Max Accuracy. For each evaluation setting, all methods use the same reference accuracy, defined as the highest Max Accuracy among the compared methods. This adjustment penalizes methods with lower success coverage and prevents a method from appearing more budget-efficient simply because it succeeds on fewer tasks. We then average the normalized estimates over tasks. Lower values indicate that a policy reaches the target reliability with less execution budget while maintaining comparable task coverage.

Figures 10 and 11 report the results on GAIA for Qwen3-8B and LLaMA-3.1-8B under token and tool-call constraints, respectively. BABLE achieves the lowest empirical minimum sufficient budget under both budget types and both backbones.

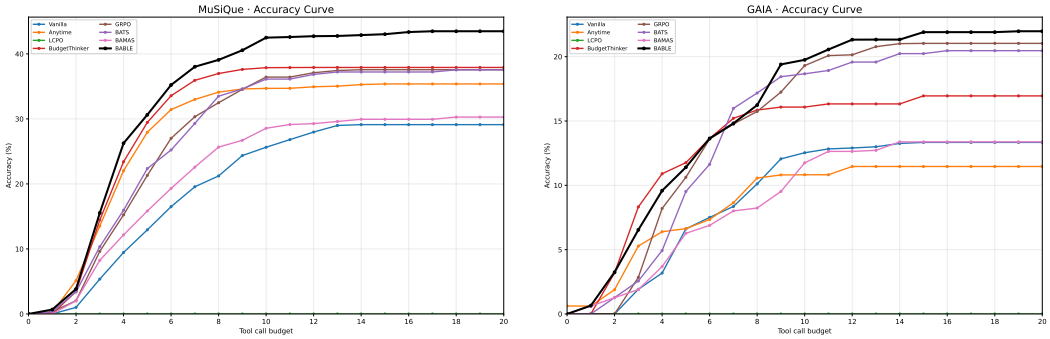


(a) MuSiQue (b) GAIA

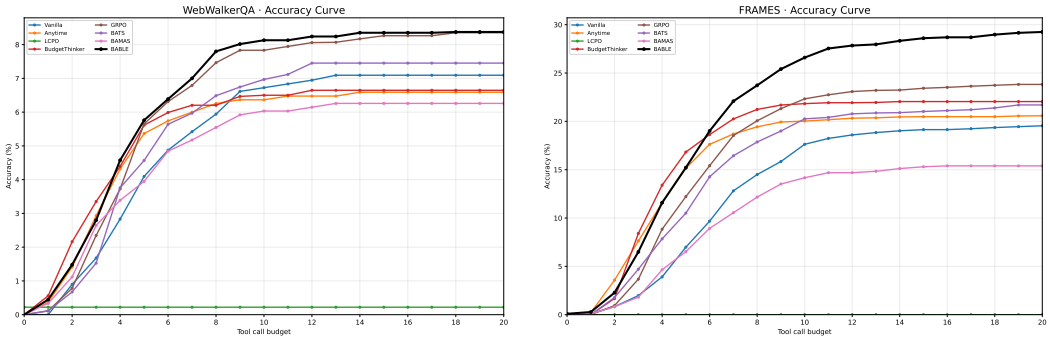


(c) WebWalkerQA (d) FRAMES

Figure 8: Accuracy under different token budgets on LLaMA-3.1-8B. Higher curves indicate better performance under the same token constraint.



(a) MuSiQue (b) GAIA



(c) WebWalkerQA (d) FRAMES

Figure 9: Accuracy under different tool-call budgets on LLaMA-3.1-8B. Higher curves indicate better performance under the same tool-interaction constraint.

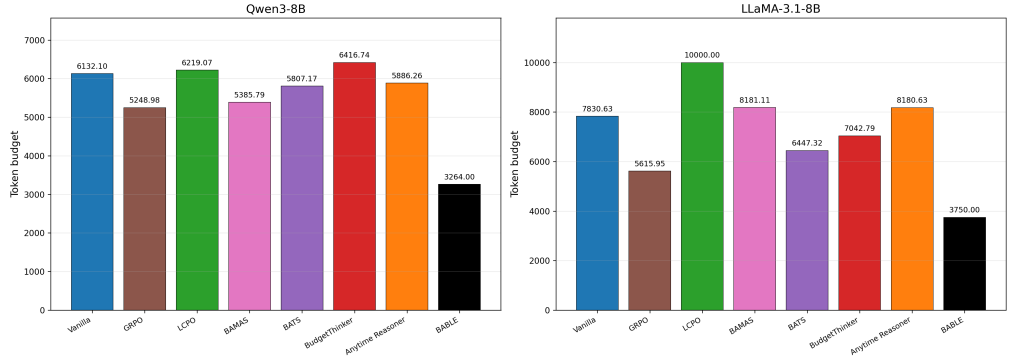


Figure 10: Empirical minimum sufficient budget on GAIA at $\alpha = 0.25$ under token constraints.

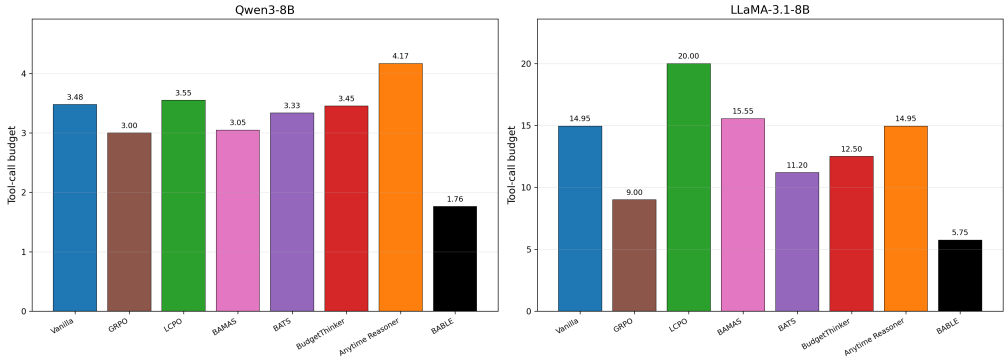


Figure 11: Empirical minimum sufficient budget on GAIA at $\alpha = 0.25$ under tool-call constraints.

J SYSTEM PROMPTS

System Prompt

In this environment you have access to a set of tools you can use to answer the user's question.
You only have access to the tools provided below. You can only use one tool per message, and will receive the result of that tool in the user's next response. You use tools step-by-step to accomplish a given task, with each tool-use informed by the result of the previous tool-use. Today is: 2025-07-16.

Tool-Use Formatting Instructions
Tool-use is formatted using XML-style tags. The tool-use is enclosed in <use_mcp_tool> and </use_mcp_tool>, and each parameter is similarly enclosed within its own set of tags.
The Model Context Protocol (MCP) connects to servers that provide additional tools and resources to extend your capabilities. You can use the server's tools via the use_mcp_tool interface.

Description
Request to use a tool provided by an MCP server. Each MCP server can provide multiple tools with different capabilities. Tools have defined input schemas that specify required and optional parameters.

Parameters

- server_name: (required) The name of the MCP server providing the tool.
- tool_name: (required) The name of the tool to execute.
- arguments: (required) A JSON object containing the tool's input parameters, following the tool's input schema.

Usage

```
<use_mcp_tool>
  <server_name>server_name_here</server_name>
  <tool_name>tool_name_here</tool_name>
  <arguments>
  {
    "param1": "value1",
    "param2": "value2 \"escaped string\""
  }
  </arguments>
</use_mcp_tool>
```

Important Notes

- Tool-use must be placed at the end of your response, at the top level, and not nested within other tags.
- Always adhere to this format for tool use to ensure proper parsing and execution.
- String and scalar parameters should be specified as is, while lists and objects should use JSON format.
- The output is not expected to be valid XML and is parsed with regular expressions.

Available Functions
Server name: search_and_scrape

Tool name: search
This tool searches the web for relevant sources given a query and returns a list of results (e.g., title, snippet, and URL). Use it to locate candidate sources before scraping.

Input JSON Schema (search)

```
{
  "properties": {
    "query": {
      "title": "Query",
      "type": "string"
    },
    "recency_days": {
      "title": "RecencyDays",
      "type": "integer"
    },
    "domains": {
      "title": "Domains",
      "type": "array",
      "items": { "type": "string" }
    }
  },
  "required": ["query"],
  "title": "searchArguments",
  "type": "object"
}
```

Tool name: scrape
This tool fetches and extracts the content from a given URL.
Use it to retrieve the full text needed for answering and for quoting/verifying claims.

Input JSON Schema (scrape)

```
{
  "properties": {
    "url": {
      "title": "URL",
      "type": "string"
    }
  },
  "required": ["url"],
  "title": "scrapeArguments",
  "type": "object"
}
```

General Objective
You accomplish a given task iteratively, breaking it down into clear steps and working through them methodically.

Task Strategy

1. Analyze the user's request and set clear, achievable sub-goals.
2. Start with a concise, numbered, step-by-step plan before taking any action.
3. Work through these sub-goals sequentially and adjust the plan as needed.
4. Use tools strategically to accomplish each sub-goal.
5. Revise earlier steps if new information emerges.

Tool-Use Guidelines

1. Each step must involve a single tool call, unless the task is already solved.
2. Before each tool call:
   - Summarize what is known.
   - Identify what is missing.

- Choose the most relevant tool.
- Verify all required parameters.

3. All tool queries must include full context.
4. Avoid vague queries. Each call should retrieve actionable information.
5. Extract and summarize partial information if a tool result is incomplete.

Tool-Use Communication Rules

1. Do not include tool results in your response.
2. Do not present the final answer until the entire task is complete.
3. Do not mention tool names.
4. Do not engage in unnecessary back-and-forth.
5. Do not use non-existent tools.
6. Respond in the same language as the user's message.
7. If the task does not require tool use, answer directly.

Agent Specific Objective

You are an agent that performs the task of searching and browsing the web for specific information and generating the desired answer. Your task is to retrieve reliable, factual, and verifiable information that fills in knowledge gaps.
Do not infer, speculate, summarize broadly, or attempt to fill in missing parts yourself. Only return factual content.
Critically assess the reliability of all information:

- If the credibility of a source is uncertain, clearly flag it.
- Do not treat information as trustworthy just because it appears.
- If you find conflicting or ambiguous information, include all relevant findings and flag the inconsistency.

Be cautious and transparent in your output:

- Always return all related information, even if incomplete, and flag any uncertainty.
- Never assume or guess. If an exact answer cannot be found, say so clearly.
- Prefer quoting or excerpting original source text rather than interpreting or rewriting it.
- Provide the URL of each source when available.
- If more context is needed, return a clarification request and do not proceed with tool use.