MARCO-Law: Marginal-Aware Reinforcement Learning for Legal Tool Orchestration

Anonymous ACL submission

Abstract

Large-scale pretrained language models (LLMs) have achieved significant advances in natural language tasks, yet challenges persist in legal applications that demand high precision. Current approaches enhance model performance through long-chain reasoning and tool invocation, but often struggle with excessive resource consumption and suboptimal tool integration. To address these issues, this paper proposes a reinforcement learning-based framework for multi-tool collaborative invocation. The framework dynamically optimizes tool usage across multiple iterations, selecting tools and refining search terms based on marginal benefits, ensuring the execution of the most effective analysis strategy. Experimental results show that the proposed method improves both the accuracy of legal question answering and resource utilization, demonstrating the potential of multi-tool collaboration and adaptive strategy adjustment in the legal domain.

1 Introduction

Recent advancements in large-scale pretrained language models (LLMs) have led to significant breakthroughs in reasoning and natural language tasks. However, complex legal tasks, due to their specialized nature and high accuracy demands, still present substantial challenges. While existing methods enhance model performance through long-chain reasoning, self-reflection (Koa et al., 2024), and external knowledge integration (Jeong et al., 2024), they face key limitations. These include high resource consumption and cognitive load from lengthy reasoning processes, as well as ineffective integration of tool results, especially when transforming legal knowledge. This is particularly evident in the legal domain, where complex reasoning across statutes and judicial practices is required.

To address these challenges, enhancing LLM performance on legal tasks requires not only accurate reasoning but also a balance between efficiency and resource consumption (Wang et al., 2025). We propose **MARCO-Law** (Marginal-Aware Reinforcement Collaboration), a reinforcement learning-based framework that enables multi-tool collaboration and dynamic reasoning strategies. The framework integrates specialized tools and modules tailored to the legal domain, allowing it to flexibly handle a broad spectrum of legal tasks—from straightforward queries to complex, multi-step problems.

First, upon receiving a legal question, the system initially determines **whether external tools** should be invoked based on the task's complexity. This process involves: (1) modeling external legal resources (e.g., Caselaw, Google Search) as environmental tools (Jin et al., 2025) to support trajectory sampling; and (2) enabling the policy model to engage in multi-round interactions with these tools, dynamically refining its invocation strategies to enhance the quality and diversity of tool use while judiciously managing cost and frequency constraints. This framework supports both single-turn RL (e.g., GRPO(Shao et al., 2024)) and multi-turn RL (e.g., Archer(Zhou et al., 2024)), providing flexible applicability.

Second, in each turn, **marginal benefits**, defined as the accuracy improvement resulting from tool invocation compared to no invocation, are used to guide strategy learning. Based on this signal, we introduce a multi-round dynamic adjustment algorithm that decides whether to modify tool usage and retrieval parameters. When the marginal benefit is positive, it directs the reinforcement learning model to refine tool selection and retrieval terms, thereby enabling adaptive and effective strategy learning in each round.

Our contributions can be summarized as:

1. We address the challenge of balancing reasoning accuracy and resource efficiency in legal

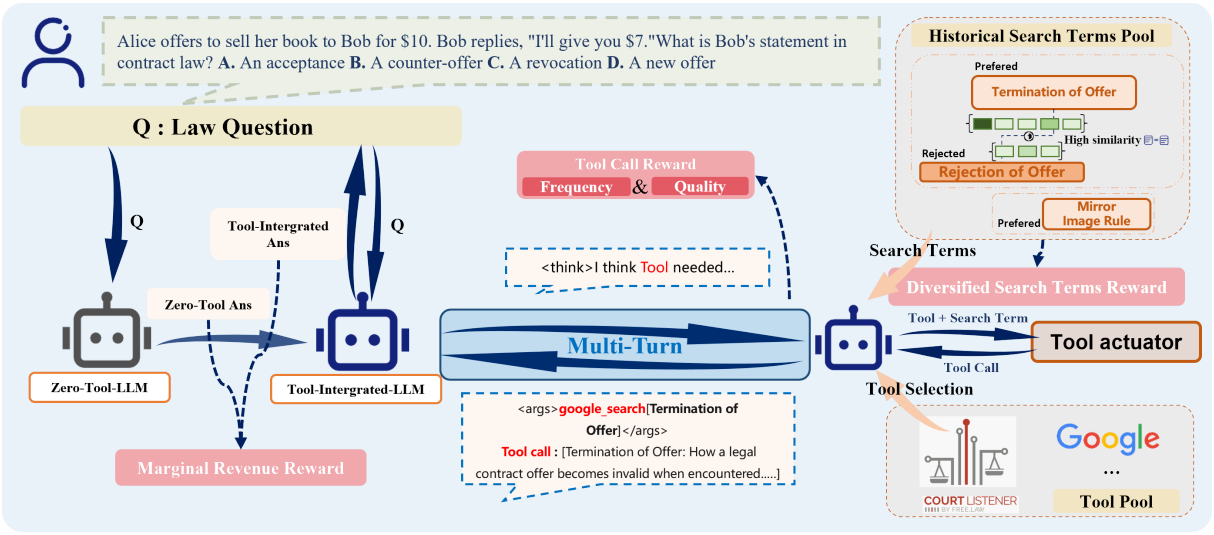

Figure 1: MARCO-Law Framework

tasks through effective multi-tool collaboration.

2. We introduce **MARCO-Law**, a reinforcement learning framework that dynamically optimizes tool invocation. MARCO-Law enhances both single-turn (via diversified tool/term selection) and multi-turn (via minimized usage) strategies by incorporating **Marginal Benefit Optimization**, which adapts tool use based on their round-specific effectiveness.

3. We experimentally validate MARCO-Law, showing significant improvements in legal QA accuracy and resource efficiency.

2 Literature Review

2.1 LLM Reasoning with Tools

LLMs have significantly advanced reasoning by integrating external tools(Shi et al., 2025). Early works demonstrated LLMs learning to use tools via APIs(Schick et al., 2023), with subsequent efforts fine-tuning models on extensive API collections for enhanced tool proficiency. To improve tool invocation efficiency, frameworks like SelfDC(Wang et al., 2024) leverage LLM self-awareness, especially in Retrieval-Augmented Generation (RAG)(Lewis et al., 2020). However, many tool integration methods still rely on complex prompting or extensive annotated data, likely limiting adaptability.

2.2 Reinforcement Learning for LLMs

Offline and Online RL. Reinforcement learning (RL) is widely applied to LLM training. Offline RL(Ghosh et al., 2022) learns cost-effectively from static datasets but faces distributional shift and reliance on high-quality reward annotations. Attempts to bridge offline and online methods, such as Archer(Zhou et al., 2024), still encounter challenges with advantage estimation from static data. Conversely, online RL methods like PPO(Schulman et al., 2017) and GRPO(Shao et al., 2024) offer greater adaptability via direct environmental interaction, but often at the cost of sample inefficiency, high computational demands, and training instability.

Single-turn and Multi-turn RL. Many LLM RL approaches, including PPO(Schulman et al., 2017), primarily optimize single-step preferences, excelling at response refinement but lacking the multi-step interactive learning crucial for complex, long-horizon reasoning. While multi-round RL holds promise for such tasks(Yu et al., 2024), its significant computational and resource overhead frequently limits practical large-scale deployment.

3 Method

3.1 Task Definition

Given a legal question q, which is represented in the form of multiple-choice questions. A set of candidate tools T=$\{T_1, T_2, \ldots, T_n\}$, the task is to learn a multi-round invocation policy $\Pi(o, r, a, o')$that generates tool-calling actions a_t at each step t. At each round, the state o_t comprises the question and

historical context. The policy $\Pi(o_t)$ outputs an action a_t specifying whether to invoke a tool, which tool to call, and with which search term.

Specifically, we optimize the strategy model by designing rewards from the perspective of marginal revenue reward. The marginal gain reward $r_t \in \{0, 1\}$ at step t is defined as a binary indicator of positive marginal benefit.

$$\Delta r_t = r_{\text{tool}} - r_{\text{non-tool}}, \quad r_t = \mathbf{1}_{\{\Delta r_t > 0\}} \quad (1)$$

where the marginal benefit Δr_t is computed as the difference between the accuracy with tool invocation and the accuracy without tool invocation. r_{tool} denotes the accuracy of the generated answer after invoking the tool, and $r_{\text{non-tool}}$ denotes the accuracy without invoking the tool.

3.2 MARCO-Law Framework

As shown in Figure 1, through multi-turn interactions with the physical world, the tool-integrated LLM aims to learn the optimal times for reasoning and receives marginal revenue rewards in each turn. Specifically, our objectives are understood at two levels: Overall Invocation Strategy and Per-Round Specific Invocation Strategy.

3.3 ArCHer-based MARCO

ArCHer with Tool Calls. ArCHer is a hierarchical reinforcement learning framework, well-suited for learning invocation strategies over extended interactions. It is particularly effective for optimizing the number of tool calls across multiple rounds of reasoning. In this study, we leverage this framework to guide the lower-level model effectively in each round, enabling the optimization of both tool selection and retrieval term adjustments at each turn. **The actor**: Policy function $\pi_\theta(a|s)$ with parameters θ.

$$\nabla_\theta J_1(\theta) = \mathbb{E}_{s\sim d^\pi}\Big[\sum_{a_1\in\mathcal{A}_1} \nabla_\theta \log \pi_\theta^1(a_1|s) \cdot \big(Q_\phi^1(s,a_1) - V_\phi^1(s)\big)\Big] \quad (2)$$

where $\mathcal{A}_1$: tool/retrieval args selection space; Q_ϕ^1: Critic evaluating tool choices.

Enhanced Action Policy

$$\nabla_\theta J_2(\theta) = \mathbb{E}_{s,m\sim d^\pi}\Big[\nabla_\theta \log \pi_\theta^2(a_2|s,m) \cdot \big(Q_\phi^2(s,m,a_2) - V_\phi^2(s,m)\big)\Big] \quad (3)$$

where $m = Tool(a_1, s)$ is tool-recall results.

The critic Value function $V_\phi(s)$ or $Q_\phi(s,a)$ with parameters ϕ. The critic minimizes the temporal difference (TD) error:

$$\mathcal{L}(\phi) = \mathbf{E}_{(s,a,\Delta r,s')}\left[\big(\Delta r + \gamma V_\phi(s') - V_\phi(s)\big)^2\right] \quad (4)$$

3.4 GRPO-based MARCO

GRPO for Tool-Augmented RL. To enhance policy optimization and reduce reliance on value function approximation, we adapt GRPO to integrate tool calls, which computes advantages based on the relative ranking of multiple sampled outputs. For an input q, a group of responses $\{y_i\}_{i=1}^G$ is sampled from a reference policy π_{ref}, incorporating tool calls.The policy π_θ is provided in Appendix A. Following search-r1(Jin et al., 2025), we mask the tool call's result section before loss calculation to prevent unintended learning dynamics from retrieved tokens during training.

Reward Design. Our reward signals guide policy training, extending beyond simple correctness. The R_{total} for query q and response y is:

$$R_{\text{total}}(q,y) = r_{\text{eff}} + r_{\text{err}} + r_{\text{gain}} + r_{\text{format}} \quad (5)$$

Components are defined as:

1. Tool Efficiency-Adjusted MCQ Reward (r_{eff}): This reward starts with the MCQ Accuracy score ($S_{\text{MCQ-ACC}}$ from Section 4.2, Eq. (9)) and is augmented to favor fewer tool calls (tc) for equivalent $S_{\text{MCQ-ACC}} > 0$.

$$r_{\text{eff}} = S_{\text{MCQ-ACC}} + k_1 \cdot f(\text{tc}, \text{tc}_{\text{group}}) \quad (6)$$

where $f(\text{tc}, \text{tc}_{\text{group}})$ is a normalized efficiency term (higher for lower tc within a group achieving the same $S_{\text{MCQ-ACC}}$).

2. Tool Error Penalty (r_{err}): A penalty discourages erroneous tool usage (e_{tool}):

$$r_{\text{err}} = -k_2 \cdot \frac{e_{\text{tool}}}{\text{tc} + \epsilon_0} \quad (7)$$

It is zero if no tools are called.

3. Marginal Gain Reward (r_{gain}): This rewards tool use only if it significantly improves $S_{\text{MCQ-ACC}}$ over a no-tool baseline ($S_{\text{MCQ-ACC,no-tool}}$):

$$r_{\text{gain}} = \begin{cases} +k_3, & \text{if } S_{\text{MCQ-ACC}} - S_{\text{MCQ-ACC, no-tool}} \geq \Delta_S \\ -k_3, & \text{if tc} > 0 \text{ and improvement} < \Delta_S \\ 0, & \text{if tc} = 0 \end{cases} \quad (8)$$

Table 1: The results of MARCO with different baselines in search.

Method	ACC-EM	ACC-LB	ACC-MCQ	P	R	Macro-F1	TC	TP
(base)	0.1780	0.6240	0.3327	0.7711	0.6258	0.6908	-	-
+SFT	0.1940	0.6200	0.3103	0.7665	0.6243	0.6879	-	-
+RAG	0.1770	0.6275	0.3463	0.7845	0.6139	0.6881	1	0.3463
+GRPO	0.2410	0.6745	0.3620	0.7684	0.7375	0.7519	-	-
+SFT+GRPO	0.2420	0.6695	0.3528	0.7629	0.7368	0.7492	-	-
+RAG+GRPO	0.1780	0.6202	0.2665	0.7434	0.6634	0.7010	1	0.2665
Ours	0.2610	0.6923	0.3850	0.7823	0.7506	0.7638	0.67	0.4890

with $k_3 \approx 0.1$ and $\Delta_S \approx 0.25$.

4. Format Reward (r_{format}): This is following the reward of deepseek-r1(Shao et al., 2024).

This multi-faceted reward promotes accurate, judicious, and efficient tool-augmented reasoning. Hyperparameters k_1, k_2, k_3, Δ_S are tuned.

The specific algorithm detailing how ARCO Law implements multiple rounds of tool calls within the GRPO framework is provided in Appendix C.

4 Experiments

4.1 Dataset and Baselines

Our experiments primarily utilize the **Legal Examination Question dataset**, featuring structured legal MCQs, and the **US-Caselaw-QA dataset**, which provides complex English legal queries with gold reasoning paths reformulated into MCQs; both are split 80/20 for training/testing. We use **Qwen2.5-3B-Instruct** as our **(base)** model and compare it against versions augmented with Supervised Fine-Tuning (**+SFT**), Retrieval-Augmented Generation (**+RAG**), and Group Relative Policy Optimization (**+GRPO**), as well as combinations thereof (**+SFT+GRPO**, **+RAG+GRPO**). These are benchmarked against our proposed MARCO approaches: **ArCHer-based MARCO** and **GRPO-based MARCO**.

4.2 Evaluation Metrics

We evaluate our model using several key metrics focusing on correctness in multi-choice questions (MCQ) and tool usage efficiency.

Our primary metric is MCQ Accuracy **(ACC-MCQ)**, which provides a fine-grained score for MCQs(multiple-choice questions). $S_{\text{ACC-MCQ}}(\mathcal{P}, \mathcal{G})$ is defined as:

$$S_{\text{MCQ-ACC}} = \begin{cases} 0, & \text{if } \mathcal{P} \setminus \mathcal{G} \neq \emptyset \\ 1.0, & \text{if } R = 1.0 \\ 0.75, & \text{if } 0.75 \leq R < 1.0 \\ 0.5, & \text{if } 0.5 \leq R < 0.75 \\ 0.25, & \text{if } 0.25 \leq R < 0.5 \\ 0.0, & \text{if } R < 0.25 \end{cases} \quad (9)$$

Table 2: Comparative results with ablation study on the US-Caselaw-QA dataset

Method	Rollout	Reward
Archer	-4.6875	-0.9868
MARCO w/o Margin RL	-3.9688	-0.9338
MARCO w/o Tool	1.0000	0.2000
MARCO w/ Tool & Margin	0.5651	0.4521

where $R = \frac{|\mathcal{P} \cap \mathcal{G}|}{|\mathcal{G}|}$ is the recall ratio, and the first condition (no false positives) must be met for any score greater than 0. The reported ACC-MCQ is the average over all samples.

For tool usage, following the OTC(Wang et al., 2025), we measure **Average Tool Calls (TC)**, the mean number of tool invocations per instance. For MCQs, we also define **Tool Productivity (TP)** in appendix B

Additionally, we report Exact Match Ratio **(EM)**, Label-based Accuracy **(Hamming Accuracy)**, **Micro F1**, and **Macro F1**.

4.3 Experimental Results

The results demonstrate the effectiveness of our multi-tool and marginal benefit-guided strategy. As shown in Table 1, our method achieves the best overall performance on the Legal Examination Question dataset, surpassing baselines in ACC-MCQ (0.3850), with improved precision and recall.

The ablation study in Table 2 further confirms the contribution of each component. The full MARCO model outperforms all variants, achieving the highest reward (0.4521), highlighting the importance of both tool invocation and marginal-guided learning in optimizing reasoning strategies.

5 Conclusion

Our findings validate that integrating dynamic, marginal-benefit-driven tool calls significantly enhances the model's ability to learn effective legal reasoning strategies, improving both accuracy and resource efficiency.

6 Limitations

Despite its advantages, our multi-tool invocation framework has limitations. It relies on domain-specific tools, which may limit its applicability across diverse industry systems. Additionally, the reinforcement learning setup can require considerable computational resources, posing challenges for large-scale deployment.

Acknowledgments

Additional elements were taken from the formatting instructions of the *International Joint Conference on Artificial Intelligence* and the *Conference on Computer Vision and Pattern Recognition*.

References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L1-regularized log-linear models. In *Proceedings of the 24th International Conference on Machine Learning*, pages 33–40.

Dibya Ghosh, Anurag Ajay, Pulkit Agrawal, and Sergey Levine. 2022. Offline rl policies should be trained to be adaptive. In *International Conference on Machine Learning*, pages 7513–7530. PMLR.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. 2024. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. *arXiv preprint arXiv:2403.14403*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Kelvin JL Koa, Yunshan Ma, Ritchie Ng, and Tat-Seng Chua. 2024. Learning to generate explainable stock predictions using self-reflective large language models. In *Proceedings of the ACM Web Conference 2024*, pages 4304–4315.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. *Computing Research Repository*, arXiv:1503.06733. Version 2.

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

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Zhengliang Shi, Shen Gao, Lingyong Yan, Yue Feng, Xiuyi Chen, Zhumin Chen, Dawei Yin, Suzan Verberne, and Zhaochun Ren. 2025. Tool learning in the wild: Empowering language models as automatic tool agents. In *Proceedings of the ACM on Web Conference 2025*, pages 2222–2237.

Hongru Wang, Cheng Qian, Wanjun Zhong, Xiusi Chen, Jiahao Qiu, Shijue Huang, Bowen Jin, Mengdi Wang, Kam-Fai Wong, and Heng Ji. 2025. Otc: Optimal tool calls via reinforcement learning. *arXiv preprint arXiv:2504.14870*.

Hongru Wang, Boyang Xue, Baohang Zhou, Tianhua Zhang, Cunxiang Wang, Huimin Wang, Guanhua Chen, and Kam-fai Wong. 2024. Self-dc: When to reason and when to act? self divide-and-conquer for compositional unknown questions. *arXiv preprint arXiv:2402.13514*.

Yuanqing Yu, Zhefan Wang, Weizhi Ma, Zhicheng Guo, Jingtao Zhan, Shuai Wang, Chuhan Wu, Zhiqiang Guo, and Min Zhang. 2024. Steptool: A step-grained reinforcement learning framework for tool learning in llms. *arXiv preprint arXiv:2410.07745*.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. 2024. Archer: Training language model agents via hierarchical multi-turn rl. *arXiv preprint arXiv:2402.19446*.

A Macro with GRPO policy model

The policy π_θ is optimized by maximizing:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{\substack{q\sim\mathcal{D}, \\ \{y_i\}_{i=1}^{G}\sim\tau_{\text{old}}(\cdot|q;\mathcal{E})}} \Bigg[\frac{1}{G}\sum_{i=1}^{G} \frac{1}{\sum_{t=1}^{|y_i|}\mathbb{I}(y_{i,t})} \cdot \cdot \sum_{t=1}^{|y_i|}\mathbb{I}(y_{i,t}) \min\left(p_t\hat{A}_{i,t}, \text{clip}(p_t, 1-\epsilon, 1+\epsilon)\hat{A}_{i,t}\right) \Bigg] - \beta\mathbb{D}_{KL}[\pi_\theta||\pi_{\text{ref}}] \tag{10}$$

Here, $p_t = \frac{\pi_\theta(y_{i,t}|x,y_{i,<t};\text{Tool})}{\pi_{\text{old}}(y_{i,t}|x,y_{i,<t};\text{Tool})}$ represents the importance sampling ratio at token t of response y_i, considering the context x, previous tokens $y_{i,<t}$, and any information obtained from tool calls (denoted by 'Tool'). $\hat{A}_{i,t}$ denotes the advantage at token t in response y_i, computed based on the relative ranking of rewards within the group. β controlling the KL-regularization strength. The clipping threshold ϵ ensures stable updates.

B Tool Productivity

$$\text{TP} = \frac{\sum_{i=1}^{N} S_{\text{MCQ-ACC},i}}{\sum_{i=1}^{N} \text{tc}_i + \delta} \tag{11}$$

where tc_i is its tool call count, N is the total samples, and δ is a small constant (e.g., 10^{-8}) to prevent division by zero if no tools are called.

C MARCO-Law in GRPO algorithm

The complete workflow is outlined in Algorithm 1

Algorithm 1 MARCO-Law in GRPO: Multi-Turn LLM Interaction with Tool Use and Composite Reward Calculation

Require: Initial user query x_{user}, LLM policy π_θ, Initial Tool Prompt $tool\_prompt_{init}$, Iteration Tool Prompt $tool\_prompt_{iter}$,zero Tool Prompt $prompt_{init}$, Tool executor $\mathcal{TE}$, Max LLM calls $MAX_{iteration}$, group size N_{group}, sreward parameters k_1, k_2, k_3.

Ensure: Final response y_{final}, A trained policy model.

Initialize first round respense with tool $y_0 \sim \pi_\theta(\cdot|\text{tool\_prompt}_{init}, x_{user})$.
Initialize zero tool respense $y_{zero\_tool} \sim \pi_\theta(\cdot|\text{prompt}_{init}, \text{x}_{user})$.
Initialize conversation history $H \leftarrow [\text{tool\_prompt}_{iter}, \text{x}_{user}, \text{" (Call \#1):"} + y_0]$
Initialize current LLM output to analyze $y_{current} \leftarrow y_0$
Initialize tool calls $c_{tool} \leftarrow 0$
Initialize tool errors $e_{tool} \leftarrow 0$
Initialize list of all LLM responses $Y_{list} \leftarrow [y_0]$
for $i \leftarrow 1$ **to** $MAX_{iteration}$ -1 **do**
    $continue\_interaction, call\_type, parsed\_output \leftarrow$ ParseLLMResponse$(y_{current})$
    if not $continue\_interaction$ **then** ▷ Direct answer found
        break
    $c_{tool} \leftarrow c_{tool} + 1$ ▷ Attempting a tool call
    if $call\_type =$ 'correct_call' **then**
        $q_{search} \leftarrow parsed\_output$["tool_query"]
        $q_{tool\_type} \leftarrow parsed\_output$["tool_type"]
        $d_{search} \leftarrow \mathcal{TE}(q_{search}, q_{tool\_type})$ ▷ Call Tool
        if $d_{search} \neq$ "no_result" **then**
            Update last part of H by inserting d_{search} where `"to_search"` was.
        else
            $e_{tool} \leftarrow e_{tool} + 1$
    else ▷ $call\_type =$ 'false_call'
        $e_{tool} \leftarrow e_{tool} + 1$
    if $c_{llm} < M_{LLM}$ **then**
        Construct prompt P_{llm} from H and system prompt.
        Generate new LLM response $y_{new} \sim \pi_\theta(\cdot|P_{llm})$
        Append " (Call " + c_{llm} + "):" + y_{new} to H
        $y_{current} \leftarrow y_{new}$
    else
        break ▷ Exceeded LLM call budget
$y_{final} \leftarrow$ ExtractFinalAnswerFromLastElementOf(H)
$S_{acc} \leftarrow$ CalculateMCQScore$(y_{final}$, gold_label$)$ ▷ Base accuracy score
▷ — Reward Calculation —
Reward 1 (Tool Efficiency): $R_1 \leftarrow$ CalculateToolEfficiencyReward$(S_{acc}, c_{tool}, N_{group}, k_1)$
▷ *Adjusts S_{acc} based on c_{tool} relative to others with same S_{acc} in a group.*
Reward 2 (Tool Error Penalty): $R_2 \leftarrow$ CalculateToolErrorPenalty$(c_{tool}, e_{tool}, k_2)$
▷ *Penalizes based on ratio of e_{tool} to c_{tool}. Max penalty $-k_2$.*
Reward 3 (Marginal Gain): $R_3 \leftarrow$ CalculateMarginalGainReward$(S_{acc}, y_{zero_tool}, c_{tool}, N_{group}, k_3)$
▷ *Rewards/penalizes based on S_{acc} vs S_{gold} if $c_{tool} > 0$. Reward $\pm k_3$.*
$R_{total} \leftarrow R_1 + R_2 + R_3 + R_{format}$ ▷ R_{total} is the total reward for a sample
Update parameters θ of policy π_θ by maximizing $\mathcal{J}_{GRPO_{with\_tool}}(\theta)$
return y_{final}, π_θ