MARTI: A FRAMEWORK FOR MULTI-AGENT LLM SYSTEMS REINFORCED TRAINING AND INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

We present MARTI (Multi-Agent Reinforced Training and Inference), an open-source framework designed to facilitate scalable and efficient learning of multi-agent LLM systems. MARTI supports centralized multi-agent interactions and distributed policy training, with the added capability of multi-turn asynchronous rollouts to enhance training efficiency. The framework includes dynamic workflows for multi-agent interactions, which integrate both rule-based verifiable rewards and LLM-based generative rewards. We validate the effectiveness of MARTI through comprehensive experiments on diverse mathematical tasks, demonstrating that multi-agent LLM-based systems outperform single-agent systems within the same inference budget after convergence. Our contributions lay the foundation for exploring scalable collaborations within LLM-based multi-agent systems and advancing the capabilities of large reasoning models.

1 INTRODUCTION

Large Reasoning Models (LRMs), such as DeepSeek-R1 (Guo et al., 2025) and OpenAI o1/o3 (El-Kishky et al., 2025), highlight the significant role Reinforcement Learning (RL) plays in enhancing the reasoning capabilities of Large Language Models (LLMs) for solving complex problems. Notably, LRMs can explore and generate extended chains of thought using only rule-based outcome rewards. This RL paradigm has also demonstrated considerable progress in other domains, including visual reasoning (Liu et al., 2025d; Zhou et al., 2025; Team et al., 2025) and agentic reasoning (Wang et al., 2025c; Jin et al., 2025) tasks. These studies indicate the effectiveness of scaling up test-time inference computations using RL. However, further performance improvements through post-training RL typically demand substantial computational resources. Additionally, recent research suggests that RL primarily activates intrinsic capabilities and reflective patterns established during pre-training (Gandhi et al., 2025; Yue et al., 2025a; Shah et al., 2025). Consequently, the initial model's passk performance sets an upper bound for RL-based enhancements (Yue et al., 2025a), which means the base model determines the reasoning limit. Therefore, the most viable approach for significantly boosting policy model performance remains within the scaling laws (Kaplan et al., 2020; Brown et al., 2020), either by training models on larger datasets or increasing the model's parameter size. Regarding the reinforcement learning stage, effectively leveraging the potential of exploration and environmental interaction remains a critical challenge (Silver & Sutton, 2025).

Meanwhile, LLM-based Multi-Agent Systems (MAS) (Han et al., 2024; Guo et al., 2024) scale inference computation by expanding the number of agents, each adaptively responding to specific tasks. Numerous open-source frameworks for LLM-based MAS are currently available, including AutoGen (Wu et al., 2023a), CAMEL (Li et al., 2023), and MetaGPT (Hong et al., 2024). However, these frameworks predominantly rely on LLM inference. This reliance makes their efficacy highly dependent on the instruction-following capabilities of the LLMs, a factor that, as recent studies (Pan et al., 2025) indicate, can readily contribute to operational failures. Concurrently, several RL frameworks (e.g., OpenRLHF (Hu et al., 2024b), veRL (Sheng et al., 2025), TRL (von Werra et al., 2020)), designed to train LLMs, can enhance LLM reasoning abilities but do not support LLM-based MAS. This observation prompts an essential question: *can we leverage RL to improve LLM-based MAS and thereby achieve superior reasoning performance?* Addressing this question requires mitigating the gap between inference and RL training in LLM-based MAS, which in turn necessitates a unified framework integrating multi-agent reinforcement learning with inference capabilities.

Table 1: Comparison between Multi-Agent and RL Framework.

Framework	MAS Inference	Single RL	MAS RL
CAMEL (Li et al., 2023)	✓	✗	✗
AutoGen (Wu et al., 2023b)	✓	✗	✗
Meta-GPT (Hong et al., 2023)	✓	✗	✗
GPTSwarm (Zhuge et al., 2024)	✓	✗	✗
TRL (von Werra et al., 2020)	✗	✓	✗
OpenRLHF (Hu et al., 2024a)	✗	✓	✗
Verl (Sheng et al., 2024)	✗	✓	✗
AReaL (Fu et al., 2025)	✗	✓	✗
MARTI (Our)	✓	✓	✓

In this work, we propose the **M**ulti-**A**gent **R**einforced **T**raining and **I**nference (**MARTI**) framework for LLM-based multi-agent systems. MARTI is built upon the OpenRLHF framework (Hu et al., 2024b), which enables scalable and high-performance RL for LLMs. For multi-agent inference, we integrate asynchronous workflows to facilitate dynamic interactions. MARTI employs a centralized interaction design for its built-in workflows (e.g., Multi-Agent Debate, Chain-of-Agents, and Mixture-of-Agents) and customizable workflows, while utilizing distributed policy training for individual agents. During inference, MARTI supports rule-based rewards as used in DeepSeek-R1 (Guo et al., 2025), along with generative reward models (Liu et al., 2025c). Prior to transferring rollout experiences to distributed agent policies, MARTI incorporates several reward shaping techniques (Park et al., 2025; Motwani et al., 2024) and credit assignment strategies to allocate rewards effectively.

Our preliminary experiments demonstrate that MARTI enhances multi-agent workflow performance, achieving a higher upper bound than single-agent RL training under the same inference budget. For instance, our multi-agent debate workflow based on DeepScaleR-1.5B-Preview (Luo et al., 2025) attains a score of 65.0 on the AIME benchmark, surpassing the single-agent baseline (53.5), which relies on large reasoning models with test-time RL. However, challenges remain in multi-agent RL, including the need for improved reward models for multi-agent systems (Pan et al., 2025) and real-world applicability (Li et al., 2025; Zheng et al., 2025). We will continue optimizing MARTI to advance MAS training for high-value applications.

Our contributions can be summarized as follows:

- We propose and open-source the Multi-Agent Reinforced Training Infrastructure (MARTI), a framework that facilitates centralized multi-agent interactions and distributed policy training, enabling scalable multi-agent learning. MARTI also supports multi-turn asynchronous rollouts during training to enhance the efficiency of multi-agent learning.
- We implement dynamic workflows for multi-agent interactions that support both rule-based verifiable rewards and LLM-based generative rewards.
- We conduct comprehensive experiments on various mathematical tasks, which demonstrate that multi-agent LLM-based systems can achieve superior performance than single agent under the same inference budget after convergence.

2 MARTI: Multi-Agent Reinforced Training and Inference

2.1 Framework Design

We designed the MARTI framework based on the principle of centralized multi-agent interaction with distributed policy training, where all agent interactions and reward allocation occur centrally, while policy training is distributed across individual agents. As illustrated in Figure 1, MARTI consists of three core modules for rollout generation and policy training: Multi-Agent World, Centralized Reward Models, and Agent Policy Trainer. The relationships and detailed descriptions of each module are provided in the following section.

Multi-Agent World. This module serves as the environment for all multi-agent interactions and experience generation. Its core functions are to execute prompt-driven rollouts according to specified interaction workflows, manage the credit assignment mechanism for resulting trajectories, and

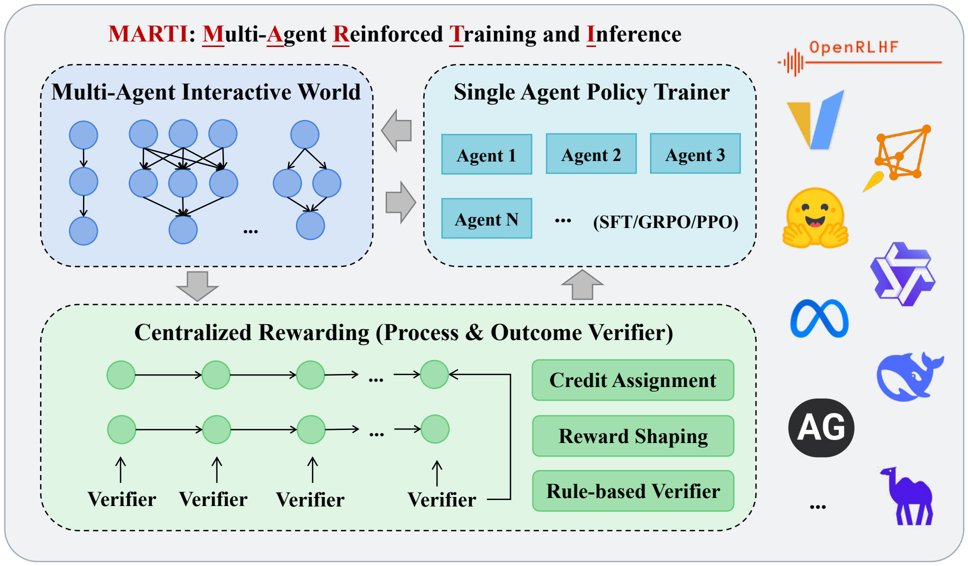


Figure 1: Overview and motivation behind of MARTI.

perform the necessary data format conversion for downstream RL training of individual agents. To ensure maximal experimental flexibility, the system supports the dynamic injection of custom workflows, which define the multi-agent interaction logic and provide access to vLLM engines [1]. Trajectories collected in this environment are subsequently processed using Centralized Reward Models. A key architectural feature is the workflow's support for asynchronous generation, which significantly enhances data throughput. The abstract workflow interface is defined in Code 1, and a complete code example is provided in Appendix B.3 for reference.

Centralized Reward Models. Following world interactions, this module collects trajectories and performs credit assignment and reward shaping. Initial global rewards are computed using either rule-based strategies or generative reward models, which are then decomposed into agent-level rewards for subsequent agent training. Section 2.2.1 introduces rule-based rewards (e.g., DeepSeek-R1) and influence-aware reward shaping for MAS. For open-domain applications, Section 2.2.2 presents generative reward models that extend to LLM-as-judge approaches (Zheng et al., 2023; Gu et al., 2024) for multi-agent reward allocation across roles and collaborations.

Agent Policy Trainer. After trajectory collection and reward allocation, MARTI distributes agent-specific trajectories and rewards to individual policy trainers. Here, backbone LLMs undergo supervised fine-tuning or reinforcement learning. Section 2.2.3 discusses policy training strategies. Leveraging distributed training capabilities, we implement various RL algorithms from OpenRLHF, including REINFORCE++ (Hu, 2025), GRPO (Shao et al., 2024), and PPO (Schulman et al., 2017), while maintaining extensibility for novel algorithms such as PRIME (Cui et al., 2025). We additionally integrate supervised fine-tuning during on-policy rollout training to enhance stability and accelerate convergence. These dynamic training strategies warrant further investigation regarding on-policy and off-policy combinations (Yan et al., 2025; Tang et al., 2025).

2.2 Algorithms Implementation

In this section, we present the implementation details of reward allocation and policy training for multi-agent training in MARTI. For reward allocation, we first discuss rule-based reward shaping (Section 2.2.1), followed by generative reward models for open-domain applications (Section 2.2.2), and finally policy training strategies (Section 2.2.3).

[1] https://github.com/vllm-project/vllm

2.2.1 RULE-BASED REWARD SHAPING

For mathematical problems with verifiable solutions, we employ rule-based reward models such as DeepSeek-R1 (Guo et al., 2025). This approach is particularly effective for mixture-of-agents (Wang et al., 2025a) and multi-agent debate (Du et al., 2024) scenarios, where each agent's output can be directly evaluated against the ground truth solution, enabling precise reward assignment based on predefined scoring rules. To improve temporal consistency and leverage historical information in multi-turn interactions, we introduce an inference-aware reward shaping strategy from MAPoRL (Park et al., 2025). This method integrates past performance estimates with current rewards. Specifically, the approach combines an immediate correctness reward from a task verifier with a dynamic adjustment derived from the agent's historical performance. This historical performance is calculated as the average reward across previous interactions.

We implement two variants: (1) a Quality Mode, which encourages consistency by aligning current performance with historical correctness, and (2) a Margin Mode, which directly rewards agents for surpassing their historical average performance. Additionally, two historical evaluation scopes are provided: one considers only the most recent interaction, offering immediate but potentially variable feedback, while the other averages across all past interactions for more stable and reliable estimates. These modular and flexible strategies effectively reduce overfitting to single-turn outcomes, enhancing long-term collaboration effectiveness in multi-turn scenarios.

Let $R_t^i \in [0, 1]$ denote the immediate correctness reward assigned by a task verifier for agent i at turn t, and let $Q_t^i \in [0, 1]$ represent the historical performance estimate of the agent, computed over a set of previous interactions:

$$Q_t^i = \frac{1}{|\mathcal{H}_t^i|} \sum_{k \in \mathcal{H}_t^i} R_k^i, \tag{1}$$

where $\mathcal{H}_t^i \subset \{1, \ldots, t-1\}$ denotes the historical evaluation scope (e.g., most recent round or all previous rounds). We define the dynamic shaping term Δ_t^i under two modes:

$$\text{Margin Mode:} \quad \Delta_t^i = R_t^i - Q_t^i, \tag{2}$$

$$\text{Quality Mode:} \quad \Delta_t^i = Q_t^i \cdot R_t^i - (1 - Q_t^i)(1 - R_t^i). \tag{3}$$

The final shaped reward $\tilde{R}_t^i$ is then given by: $\tilde{R}_t^i = R_t^i + \alpha \cdot \Delta_t^i$ where $\alpha \in \mathbb{R}_{\geq 0}$ is a tunable hyperparameter controlling the influence of historical consistency.

2.2.2 GENERATIVE REWARD MODELS

Recent advances have demonstrated that LLMs can effectively evaluate response quality, enabling their use as generative reward models (GenRMs) to enhance policy model reasoning capabilities (Zhang et al., 2025e; Mahan et al., 2024; Zhao et al., 2025). Building on these developments, we implement GenRMs in MARTI for both verifiable and open-domain problems. Our framework supports GenRMs through either local vLLM engines or OpenAI-compatible APIs, with a defined GenRM that assigns scalar rewards to given problem-trajectory pairs.

Furthermore, we investigate specialized GenRMs for multi-agent systems (MAS) that explicitly address common failure modes identified in prior work (Cemri et al., 2025; Zhang et al., 2025f). These models show particular promise for improving collaborative behaviors in MAS. We continue to optimize this functionality, with further discussion reserved for future work.

2.2.3 POLICY MODEL TRAINING

Upon obtaining rollout experiences comprising individual trajectories and corresponding rewards for each agent, we initiate distributed training of agent policy models. The training leverages adapted implementations from OpenRLHF, supporting various reinforcement learning algorithms including REINFORCE++ (Hu, 2025), GRPO (Shao et al., 2024), and PPO (Schulman et al., 2017). Notably, all agent policies are trained using identical RL algorithms to maintain consistency.

Furthermore, we augment the training process by incorporating additional imitation learning strategies during on-policy rollouts. These include supervised fine-tuning (SFT) and direct preference optimization (DPO) (Rafailov et al., 2023), extending beyond OpenRLHF's native capabilities. This integration enables dynamic selection of training strategies tailored to specific application requirements, such as stable training and faster convergence.

3 Experiments

3.1 Experimental Setup

Datasets. We utilize competition-level mathematical datasets for our experiments, including AIME24 (AI-MO, 2024a), AMC (AI-MO, 2024b), and MATH-500 (Lightman et al., 2024). All datasets are adapted from the publicly released DeepScaleR project materials [2].

Models. For non-reasoning models, we use `Qwen2.5-3B` and `Qwen2.5-3B-Instruct` (Yang et al., 2024). For reasoning models, we utilize `Qwen3-1.7B` (Team, 2025) and `DeepScaleR-1.5B-Preview` (Luo et al., 2025) for experiments. We also incorporate results for `Qwen2.5-7B/14B-Instruct`, `DeepSeek-R1-Qwen-7/14B`, and `OpenAI-o1-mini`.

Inference Details. For multi-agent workflow inference, we use a temperature of 0.6 and top_p of 0.95 for all models. The max generation token is set to 8192 for non-reasoning models and 16384 for reasoning models. For reasoning models with outputs like "<think> reasoning </think><answer>final answer</answer>", agents exclusively exchange final answers without their intermediate thinking processes with each other.

Training Details. We employ the MARTI framework to train both base and reasoning models, specifically `Qwen2.5-3B` and `DeepScaleR-1.5B-Preview`. For `Qwen2.5-3B`, we implement DeepSeek-R1 zero-like reinforcement learning training using Level 3-5 samples from the MATH dataset (Hendrycks et al., 2021) like previous works (Zeng et al., 2025; Liu et al., 2025b). The `DeepScaleR-1.5B-Preview` model, which exhibits strong inherent reasoning capabilities but presents training challenges, undergoes test-time reinforcement learning (TTRL) (Zuo et al., 2025) adaptation on AIME benchmark data. We maintain the same maximum generation tokens and temperature settings as used during inference, while extending the maximum prompt token length to 8192. For multi-agent reinforcement learning, we employ a cluster configuration consisting of 3 nodes, each equipped with 8 A800 80GB GPUs, allocating one full node per agent.

Evaluation Metrics. We evaluate model performance using accuracy scores computed for all datasets with open-source scripts from `Qwen2.5-Math`[3]. Additionally, we measure `Pass@1` by averaging scores across multiple responses and compute `Maj@N` (where $N = 4$ or 6) under the same inference budget using multi-agent reinforcement learning (RL). For conciseness, we abbreviate multi-agents debate, mixture-of-agents, and chain-of-agents as `MAD`, `MoA`, and `CoA`, respectively.

3.2 Main Results

We present comparative results for both non-reasoning and reasoning models across different training and inference configurations in Figure 2 (instruction models) and Figure 3 (reasoning models), followed by a multi-perspective analysis:

Failures of Multi-agent Workflows. Our experimental results demonstrate that both non-reasoning and reasoning models achieve superior performance through majority voting compared to multi-agent workflows under equivalent computational budgets. This observation aligns with existing literature documenting failures in LLM-based multi-agent systems (Cemri et al., 2025; Zhang et al., 2025f;c), which identifies two key limitations: (1) inability to adhere to role specifications, and (2) failure to effectively utilize inter-agent interaction information. We attribute these shortcomings to the predominant single-agent training paradigm of current LLMs, which inherently lacks exposure to multi-agent dynamics. These findings motivate our proposed MARTI framework, which implements MARL to develop advanced reasoning capabilities through structured agent interactions.

MARTI Enhances Base Models Using Zero-like RL. We further investigate training base models with zero-like RL using MARTI. For the `Qwen2.5-3B` model, we compare standard RL training

[2] https://github.com/agentica-project/deepscaler
[3] https://github.com/QwenLM/Qwen2.5-Math

Figure 2: Average scores of Qwen2.5-3B base and instruct models under different budget and settings.

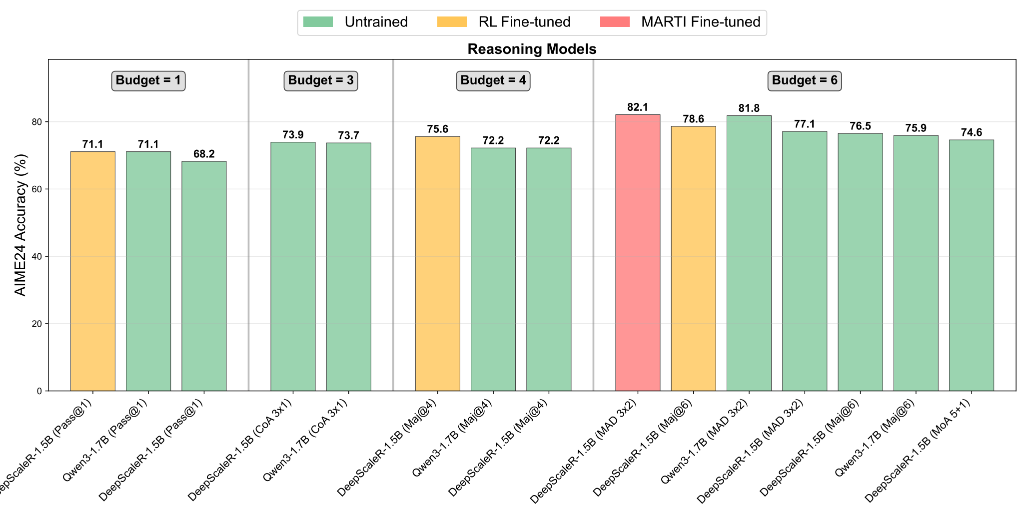


Figure 3: Average scores of reasoning models under different budget and settings.

with MARTI-based training and observe that multi-agent systems trained with MARTI achieve a higher performance upper bound than single-agent systems. Notably, multi-agent debate yields the best results under the same computational budget. Consistent with prior research, our results demonstrate that base models enhanced with reinforcement learning achieve comparable performance to instructed models. This finding further supports the established conclusion that RL primarily enhances a model's intrinsic pre-trained capabilities rather than imparting new knowledge. These findings suggest the need to explore novel reinforcement learning approaches that enhance individual model capabilities, such as through multi-agent interaction paradigms.

MARTI Enhances Large Reasoning Models. To explore the upper bound of multi-agent training, we apply test-time reinforcement learning (TTRL) to large reasoning models (`DeepScaleR-1.5B-Preview`). Our results demonstrate TTRL's effectiveness for LRMs, particularly on complex tasks. Notably, Multi-Agent Debates (MAD) achieve a score of 66.7 on AIME, significantly outperforming other same-cost configurations, including OpenAI-o1-mini.

Multi-Agent RL Achieves a Higher Performance Upper Bound Than Single-Agent Systems. After analyzing the performance of both base models and large reasoning models trained with MARTI, we find that multi-agent RL consistently achieves a higher performance upper bound than single-agent systems. This demonstrates that, under the same inference budget, reinforced multi-agent models attain superior benchmark scores compared to their single-agent counterparts. Furthermore, these results suggest that reinforced multi-agent training can enhance advanced reasoning capabilities, presenting a promising direction for future research in reasoning optimization.

Table 2: Results for Llama-3.2-3B-Instruct across various workflows and training configurations. Under an equivalent inference budget, MARTI consistently outperforms both single-agent reinforcement learning and majority-vote baselines.

Llama-3.2-3B-Instruct	AIME	AMC	MATH500	Avg
Single Agent (Pass@1)	3.3	12.4	32.2	16.0
+ RL	11.7	25.6	48.9	28.7
Single Agent (Maj@4)	6.6	18.1	36.6	20.4
+ RL	11.7	27.7	50.6	30.0
MAD 2×2	3.3	16.9	38.4	19.5
+ RL (MARTI)	13.3	29.5	53.6	32.1
MoA 3×1	6.6	16.9	37.2	20.2
+ RL (MARTI)	11.7	28.7	52.6	31.0

Table 3: Comparison of REINFORCE++ (RF++) and GRPO on Qwen2.5-3B. Both algorithms produce strong performance gains; GRPO achieves marginally better results on most evaluated metrics.

Qwen2.5-3B	AIME	AMC	MATH500	Avg
Single-Agent + RF++	10.0	36.1	66.7	37.6
Single-Agent + GRPO	13.3	34.6	66.0	37.9
MAD 2×2 + RF++	16.7	49.4	70.8	45.6
MAD 2×2 + GRPO	16.7	50.0	71.2	46.0

Table 4: Ablation study on reward shaping for Qwen2.5-3B. Removing reward shaping results in substantial performance degradation for both MAD and MoA architectures.

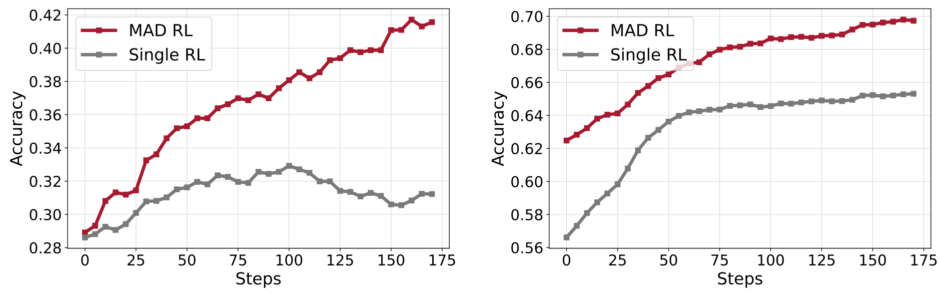
Qwen2.5-3B	AIME	AMC	MATH500	Avg
MAD 2×2 w/ reward shaping	16.7	49.4	70.8	45.6
MAD 2×2 w/o reward shaping	6.6	36.6	66.7	36.6
MoA 3×1 w/ reward shaping	13.3	47.0	69.0	43.1
MoA 3×1 w/o reward shaping	10.0	38.9	65.4	38.1

3.3 Ablation Studies

Different Model Families To examine whether MARTI generalizes beyond Qwen-based models, we apply both single-agent and multi-agent RL to the Llama-3.2-3B-Instruct backbone. Table 2 summarizes the results. Single-agent RL already brings a large improvement over the supervised Pass@1 baseline (from 16.0 to 28.7 on average). On top of this, applying MARTI to multi-agent workflows yields further gains: for example, MAD 2×2 with RL reaches an average score of 32.1, outperforming both single-agent RL (28.7) and majority-vote RL (30.0). MoA 3×1 with RL achieves a similar average score of 31.0. These trends are consistent across all three benchmarks, indicating that the benefits of MARTI are not tied to a specific model architecture.

Different Algorithms We further compare two policy-gradient estimators, REINFORCE++ (RF++) and GRPO, on Qwen2.5-3B. As shown in Table 3, both algorithms improve substantially over the supervised single-agent baseline, and GRPO provides a slight but consistent edge. For instance, MAD 2×2 with GRPO achieves an average score of 46.0, compared to 45.6 with RF++. These results indicate that MARTI is robust to the specific choice of policy-gradient algorithm, and that the main gains arise from the multi-agent RL formulation itself.

Reward Shaping. MARTI employs a delta-style reward shaping mechanism that compares the current turn of an agent with its own historical performance, thereby rewarding relative improvements



(a) AMC (b) MATH-500

Figure 4: Accuracy of MAD (Qwen2.5-3B) on AMC and MATH-500

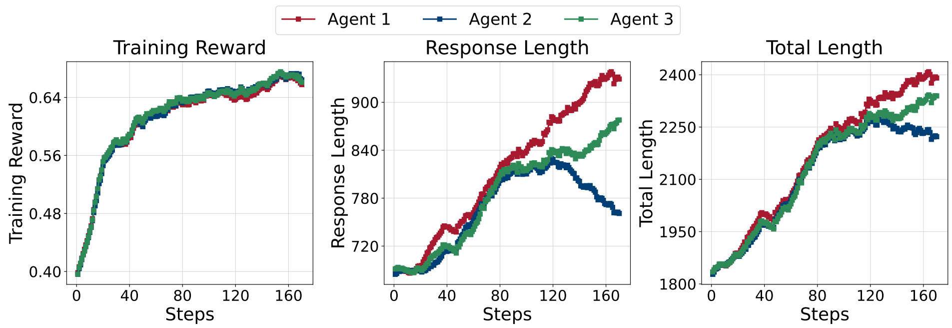


Figure 5: Training Dynamics of MAD (Qwen2.5-3B) with RL on MATH.

instead of only absolute correctness. Table 4 reports ablations on Qwen2.5-3B for MAD 2×2 and MoA 3×1. Removing reward shaping causes a clear drop in average performance: from 45.6 to 36.6 for MAD and from 43.1 to 38.1 for MoA. This shows that reward shaping is essential for stabilizing multi-agent RL; purely outcome-based rewards are more prone to instability and reward hacking in multi-turn interaction, whereas our shaping provides smoother optimization signals that better align with collaborative reasoning quality.

4 Discussions

4.1 Case Study: Multi-Agents Debate

Experimental Setup. We conduct multi-agent debate training using two model architectures: `Qwen2.5-3B` and `DeepScaleR-1.5B-Preview`. The `Qwen2.5-3B` model is trained using REINFORCE++ on Level 3 to 5 samples from the MATH-500 dataset, while `DeepScaleR-1.5B-Preview` employs TTRL on the AIME benchmark.

Training Dynamics. Model accuracy results are presented for both AMC and MATH-500 benchmarks in Figures 4a and 4b, respectively. Additionally, we analyze the training dynamics of RL in Figure 5 and TTRL optimization in Figure 6.

We present additional case studies analyzing training dynamics across various multi-agent architectures in Appendix C, including Mixture-of-Agents (MoA) (Appendix C.1), Chain-of-Agents (CoA) (Appendix C.2), and Judge-based Debate (Appendix C.3).

4.2 Statistics of Asynchronous Generations

Previous rollouts in RL frameworks are typically performed in batches for batch generation. However, this approach proves inefficient for multi-turn interactions, such as multi-turn tool calls and multi-agent interactions, due to significant discrepancies in time costs during generation. As a result, asynchronous generation has become a core feature in mainstream RL frameworks, particularly in

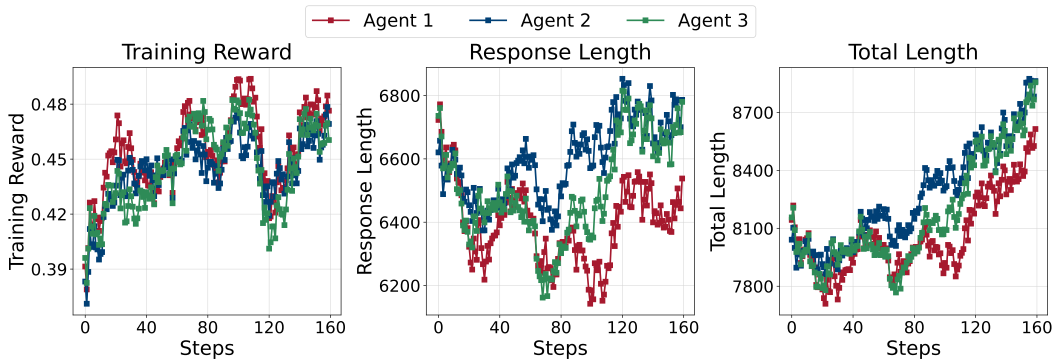


Figure 6: Training Dynamics of MAD (DeepScaleR-1.5B-Preview) with TTRL on AIME.

Table 5: Statistics of asynchronous rollouts in MARTI. (a) End-to-end rollout time vs. concurrency for Chain-of-Agents and MAD. (b) Total rollout time vs. number of interaction rounds for different concurrency settings.

(a) Time vs. concurrency (seconds)

Workflow	Sync	Async×32	Async×64	Async×128	Async×256	Async×512
Chain	612.6	615.5	553.5	508.5	515.4	498.4
MAD	593.5	732.9	616.8	592.4	569.4	561.2

(b) Time vs. interaction rounds (seconds)

round	Sync	Async×32	Async×64
2	916.6	1074.1	887.5
5	2194.0	2186.1	2022.8
8	3308.0	3004.4	2928.0

agentic RL systems such as OpenRLHF (Hu et al., 2024a), veRL (Sheng et al., 2024), and AReaL (Fu et al., 2025). To the best of our knowledge, MARTI is the first framework to support asynchronous generation for multi-turn, multi-agent scenarios.

We further analyze the effect of asynchronous rollouts in MARTI. Table 5 summarizes the time costs for synchronous and asynchronous execution in both Chain-of-Agents and Multi-Agent Debate workflows under different levels of concurrency and interaction depth. With moderate concurrency, asynchronous rollouts consistently reduce end-to-end inference time, but very large numbers of parallel workers become compute bound and yield diminishing returns. When the interaction depth is small (e.g., 2 rounds), trajectories are short and synchronization overhead is negligible, so Async×64 provides only a modest $\sim$3% speed-up over the synchronous baseline. As the number of rounds increases, rollout latency grows and the throughput advantage of asynchrony becomes more pronounced. Overall, asynchronous generation is most beneficial for deep, interaction heavy workflows, while shallow workflows are primarily limited by raw compute rather than synchronization.

5 Conclusion

We present MARTI, a unified framework integrating multi-agent reinforcement learning (RL) with inference for LLM-based systems. By combining scalable RL training (via OpenRLHF) with adaptive multi-agent workflows, MARTI outperforms single-agent TTRL in reasoning tasks, achieving advanced performance on AIME. Challenges like reward modeling and real-world deployment persist, but MARTI advances MAS capabilities through built-in credit assignment and support for diverse reward models. Our work demonstrates that multi-agent RL elevates performance ceilings beyond single-agent approaches, offering a pathway to enhance reasoning in practical applications. Future work will focus on optimizing MAS training for broader adoption.

Ethics Statement

This work presents MARTI, a framework for LLM-based multi-agent reinforcement learning and inference. We use established public benchmarks to ensure transparent and unbiased evaluation, while minimizing computational waste through efficient configurations.

Reproducibility Statement

We provide comprehensive details to ensure reproducibility, including implementation specifics in Section 3.1 (models, inference details, training procedures, and evaluation metrics). Additionally, the anonymous MARTI codebase is provided in https://anonymous.4open.science/r/marti-anoy-C76F.

References

AI-MO. Aime 2024, 2024a. URL https://huggingface.co/datasets/AI-MO/aimo-validation-aime.

AI-MO. Amc 2023, 2024b. URL https://huggingface.co/datasets/AI-MO/aimo-validation-amc.

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. Make your llm fully utilize the context. *Advances in Neural Information Processing Systems*, 37:62160–62188, 2024.

Antonis Antoniades, Albert Örwall, Kexun Zhang, Yuxi Xie, Anirudh Goyal, and William Wang. Swe-search: Enhancing software agents with monte carlo tree search and iterative refinement. *arXiv preprint arXiv:2410.20285*, 2024.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Mert Cemri, Melissa Z. Pan, Shuyi Yang, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Why do multi-agent llm systems fail?, 2025. URL https://arxiv.org/abs/2503.13657.

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. Chateval: Towards better LLM-based evaluators through multi-agent debate. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=FQepisCUWu.

Guangyao Chen, Siwei Dong, Yu Shu, Ge Zhang, Jaward Sesay, Börje Karlsson, Jie Fu, and Yemin Shi. Autoagents: a framework for automatic agent generation. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 22–30, 2024a.

Pei Chen, Shuai Zhang, and Boran Han. CoMM: Collaborative multi-agent, multi-reasoning-path prompting for complex problem solving. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 1720–1738, Mexico City, Mexico, June 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.112. URL https://aclanthology.org/2024.findings-naacl.112/.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong

Sun, and Jie Zhou. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations*, 2024c. URL https://openreview.net/forum?id=EHg5GDnyq1.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. Universal self-consistency for large language model generation. *arXiv preprint arXiv:2311.17311*, 2023.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

Haikang Deng and Colin Raffel. Reward-augmented decoding: Efficient controlled text generation with a unidirectional reward model. *arXiv preprint arXiv:2310.09520*, 2023.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 11733–11763. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/du24e.html.

Ahmed El-Kishky, Alexander Wei, Andre Saraiva, Borys Minaev, Daniel Selsam, David Dohan, Francis Song, Hunter Lightman, Ignasi Clavera, Jakub Pachocki, et al. Competitive programming with large reasoning models. *arXiv preprint arXiv:2502.06807*, 2025.

Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, et al. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. *arXiv preprint arXiv:2505.24298*, 2025.

Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, Nathan Lile, and Noah D Goodman. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars. *arXiv preprint arXiv:2503.01307*, 2025.

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

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

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges, 2024. URL https://arxiv.org/abs/2402.01680.

Shanshan Han, Qifan Zhang, Yuhang Yao, Weizhao Jin, Zhaozhuo Xu, and Chaoyang He. Llm multi-agent systems: Challenges and open problems, 2024. URL https://arxiv.org/abs/2402.03578.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

Brendan Hogan. Debate framework for language model training, 2024. URL https://github.com/brendanhogan/debate-framework.

Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, et al. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 3(4):6, 2023.

Sirui Hong, Mingchen Zhuge, Jiaqi Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. Metagpt: Meta programming for a multi-agent collaborative framework, 2024. URL https://arxiv.org/abs/2308.00352.

Jian Hu. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.

Jian Hu, Xibin Wu, Zilin Zhu, Weixun Wang, Dehao Zhang, Yu Cao, et al. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*, 2024a.

Jian Hu, Xibin Wu, Zilin Zhu, Xianyu, Weixun Wang, Dehao Zhang, and Yu Cao. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework, 2024b. URL https://arxiv.org/abs/2405.11143.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=IkmD3fKBPQ.

Dom Huh and Prasant Mohapatra. Multi-agent reinforcement learning: A comprehensive survey. *arXiv preprint arXiv:2312.10256*, 2023.

Che Jiang, Biqing Qi, Xiangyu Hong, Dayuan Fu, Yang Cheng, Fandong Meng, Mo Yu, Bowen Zhou, and Jie Zhou. On large language models' hallucination with regard to known facts. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1041–1053, 2024.

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

junyou li, Qin Zhang, Yangbin Yu, QIANG FU, and Deheng Ye. More agents is all you need. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=bgzUSZ8aeg.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

Sayash Kapoor, Benedikt Stroebl, Zachary S Siegel, Nitya Nadgir, and Arvind Narayanan. Ai agents that matter. *arXiv preprint arXiv:2407.01502*, 2024.

Maxim Khanov, Jirayu Burapacheep, and Yixuan Li. Args: Alignment as reward-guided search. *arXiv preprint arXiv:2402.01694*, 2024.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: communicative agents for" mind" exploration of large language model society. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pp. 51991–52008, 2023.

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

Yunxuan Li, Yibing Du, Jiageng Zhang, Le Hou, Peter Grabowski, Yeqing Li, and Eugene Ie. Improving multi-agent debate with sparse communication topology. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 7281–7294, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.427. URL https://aclanthology.org/2024.findings-emnlp.427/.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17889–17904, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.992. URL https://aclanthology.org/2024.emnlp-main.992/.

Xinbin Liang, Jinyu Xiang, Zhaoyang Yu, Jiayi Zhang, Sirui Hong, Sheng Fan, and Xiao Tang. Openmanus: An open-source framework for building general ai agents, 2025. URL https://doi.org/10.5281/zenodo.15186407.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Runze Liu, Junqi Gao, Jian Zhao, Kaiyan Zhang, Xiu Li, Biqing Qi, Wanli Ouyang, and Bowen Zhou. Can 1b llm surpass 405b llm? rethinking compute-optimal test-time scaling. *arXiv preprint arXiv:2502.06703*, 2025a.

Siqi Liu, Guy Lever, Josh Merel, Saran Tunyasuvunakool, Nicolas Heess, and Thore Graepel. Emergent coordination through competition. *arXiv preprint arXiv:1902.07151*, 2019.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025b.

Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. Inference-time scaling for generalist reward modeling, 2025c. URL https://arxiv.org/abs/2504.02495.

Ziyu Liu, Zeyi Sun, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Haodong Duan, Dahua Lin, and Jiaqi Wang. Visual-rft: Visual reinforcement fine-tuning. *arXiv preprint arXiv:2503.01785*, 2025d.

Sikai Lu, Yingfeng Cai, Ze Liu, Yubo Lian, Long Chen, and Hai Wang. A preference-based multi-agent federated reinforcement learning algorithm framework for trustworthy interactive urban autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 2025.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. Notion Blog, 2025. Notion Blog.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594, 2023.

Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. Generative reward models. *arXiv preprint arXiv:2410.12832*, 2024.

Sumeet Ramesh Motwani, Chandler Smith, Rocktim Jyoti Das, Rafael Rafailov, Ivan Laptev, Philip HS Torr, Fabio Pizzati, Ronald Clark, and Christian Schroeder de Witt. Malt: Improving reasoning with multi-agent llm training. *arXiv preprint arXiv:2412.01928*, 2024.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Melissa Z Pan, Mert Cemri, Lakshya A Agrawal, Shuyi Yang, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Kannan Ramchandran, Dan Klein, et al. Why do multiagent systems fail? In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*, 2025.

Chanwoo Park, Seungju Han, Xingzhi Guo, Asuman Ozdaglar, Kaiqing Zhang, and Joo-Kyung Kim. Maporl: Multi-agent post-co-training for collaborative large language models with reinforcement learning. *arXiv preprint arXiv:2502.18439*, 2025.

Chen Qian, Zihao Xie, Yifei Wang, Wei Liu, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Scaling large language model-based multi-agent collaboration. In *International Conference on Learning Representations (ICLR)*, 2025.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

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

Darsh J Shah, Peter Rushton, Somanshu Singla, Mohit Parmar, Kurt Smith, Yash Vanjani, Ashish Vaswani, Adarsh Chaluvaraju, Andrew Hojel, Andrew Ma, et al. Rethinking reflection in pre-training. *arXiv preprint arXiv:2504.04022*, 2025.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv:2409.19256*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, pp. 1279–1297. ACM, March 2025. doi: 10.1145/3689031.3696075. URL http://dx.doi.org/10.1145/3689031.3696075.

David Silver and Richard S Sutton. Welcome to the era of experience. *Google AI*, 2025.

Oliver Slumbers, David Henry Mguni, Kun Shao, and Jun Wang. Leveraging large language models for optimised coordination in textual multi-agent reinforcement learning. *arXiv*, 2023.

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.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021, 2020.

Yunhao Tang, Taco Cohen, David W Zhang, Michal Valko, and Rémi Munos. Rl-finetuning llms from on-and off-policy data with a single algorithm. *arXiv preprint arXiv:2503.19612*, 2025.

Kimi Team, Angang Du, Bohong Yin, Bowei Xing, Bowen Qu, Bowen Wang, Cheng Chen, Chenlin Zhang, Chenzhuang Du, Chu Wei, et al. Kimi-vl technical report. *arXiv preprint arXiv:2504.07491*, 2025.

Qwen Team. Qwen3, April 2025. URL https://qwenlm.github.io/blog/qwen3/.

Raghav Thind, Youran Sun, Ling Liang, and Haizhao Yang. Optimai: Optimization from natural language using llm-powered ai agents. *arXiv preprint arXiv:2504.16918*, 2025.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl, 2020.

Ziyu Wan, Yunxiang Li, Yan Song, Hanjing Wang, Linyi Yang, Mark Schmidt, Jun Wang, Weinan Zhang, Shuyue Hu, and Ying Wen. Rema: Learning to meta-think for llms with multi-agent reinforcement learning. *arXiv preprint arXiv:2503.09501*, 2025.

Junlin Wang, Jue WANG, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-agents enhances large language model capabilities. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=h0ZfDIrj7T.

Lei Wang, Jianxun Lian, Yi Huang, Yanqi Dai, Haoxuan Li, Xu Chen, Xing Xie, and Ji-Rong Wen. CharacterBox: Evaluating the role-playing capabilities of LLMs in text-based virtual worlds. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *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. 6372–6391, Albuquerque, New Mexico, April 2025b. Association for Computational Linguistics. ISBN 979-8-89176-189-6. URL https://aclanthology.org/2025.naacl-long.323/.

Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*, 2023a.

Qineng Wang, Zihao Wang, Ying Su, Hanghang Tong, and Yangqiu Song. Rethinking the bounds of LLM reasoning: Are multi-agent discussions the key? In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6106–6131, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.331. URL https://aclanthology.org/2024.acl-long.331/.

Xihuai Wang, Zheng Tian, Ziyu Wan, Ying Wen, Jun Wang, and Weinan Zhang. Order matters: Agent-by-agent policy optimization. *arXiv preprint arXiv:2302.06205*, 2023b.

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.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, et al. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025c.

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.

Sean Welleck, Amanda Bertsch, Matthew Finlayson, Hailey Schoelkopf, Alex Xie, Graham Neubig, Ilia Kulikov, and Zaid Harchaoui. From decoding to meta-generation: Inference-time algorithms for large language models. *arXiv preprint arXiv:2406.16838*, 2024.

Bosi Wen, Pei Ke, Xiaotao Gu, Lindong Wu, Hao Huang, Jinfeng Zhou, Wenchuang Li, Binxin Hu, Wendy Gao, Jiaxing Xu, et al. Benchmarking complex instruction-following with multiple constraints composition. *Advances in Neural Information Processing Systems*, 37:137610–137645, 2024.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation, 2023a. URL https://arxiv.org/abs/2308.08155.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023b.

Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P Lillicrap, Kenji Kawaguchi, and Michael Shieh. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*, 2024.

Zhenran Xu, Senbao Shi, Baotian Hu, Jindi Yu, Dongfang Li, Min Zhang, and Yuxiang Wu. Towards reasoning in large language models via multi-agent peer review collaboration. *arXiv preprint arXiv:2311.08152*, 2023.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*, 2025.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Sen Yang, Yafu Li, Wai Lam, and Yu Cheng. Multi-llm collaborative search for complex problem solving. *arXiv preprint arXiv:2502.18873*, 2025.

Yaodong Yang. *Many-agent reinforcement learning*. PhD thesis, UCL (University College London), 2021.

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

Hai Ye, Mingbao Lin, Hwee Tou Ng, and Shuicheng Yan. Multi-agent sampling: Scaling inference compute for data synthesis with tree search-based agentic collaboration. *arXiv preprint arXiv:2412.17061*, 2024.

Zhangyue Yin, Qiushi Sun, Cheng Chang, Qipeng Guo, Junqi Dai, Xuanjing Huang, and Xipeng Qiu. Exchange-of-thought: Enhancing large language model capabilities through cross-model communication. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 15135–15153, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.936. URL https://aclanthology.org/2023.emnlp-main.936/.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*, 2024.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025a.

Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyan Qi. Masrouter: Learning to route llms for multi-agent systems, 2025b. URL https://arxiv.org/abs/2502.11133.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild, 2025. URL https://arxiv.org/abs/2503.18892.

Guibin Zhang, Kaijie Chen, Guancheng Wan, Heng Chang, Hong Cheng, Kun Wang, Shuyue Hu, and Lei Bai. Evoflow: Evolving diverse agentic workflows on the fly. *arXiv preprint arXiv:2502.07373*, 2025a.

Guibin Zhang, Luyang Niu, Junfeng Fang, Kun Wang, Lei Bai, and Xiang Wang. Multi-agent architecture search via agentic supernet. *arXiv preprint arXiv:2502.04180*, 2025b.

Hangfan Zhang, Zhiyao Cui, Xinrun Wang, Qiaosheng Zhang, Zhen Wang, Dinghao Wu, and Shuyue Hu. If multi-agent debate is the answer, what is the question? *arXiv preprint arXiv:2502.08788*, 2025c.

Kaiyan Zhang, Jiayuan Zhang, Haoxin Li, Xuekai Zhu, Ermo Hua, Xingtai Lv, Ning Ding, Biqing Qi, and Bowen Zhou. Openprm: Building open-domain process-based reward models with preference trees. In *The Thirteenth International Conference on Learning Representations*, 2025d.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. In *International Conference on Learning Representations (ICLR)*, 2025e. URL https://openreview.net/forum?id=Ccwp4tFEtE.

Shao Zhang, Xihuai Wang, Wenhao Zhang, Yongshan Chen, Landi Gao, Dakuo Wang, Weinan Zhang, Xinbing Wang, and Ying Wen. Mutual theory of mind in human-ai collaboration: An empirical study with llm-driven ai agents in a real-time shared workspace task. *arXiv preprint arXiv:2409.08811*, 2024a.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, et al. Which agent causes task failures and when? on automated failure attribution of llm multi-agent systems. *arXiv preprint arXiv:2505.00212*, 2025f.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. Siren's song in the ai ocean: a survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*, 2023.

Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Sercan Arik. Chain of agents: Large language models collaborating on long-context tasks. *Advances in Neural Information Processing Systems*, 37:132208–132237, 2024b.

Jian Zhao, Runze Liu, Kaiyan Zhang, Zhimu Zhou, Junqi Gao, Dong Li, Jiafei Lyu, Zhouyi Qian, Biqing Qi, Xiu Li, and Bowen Zhou. Genprm: Scaling test-time compute of process reward models via generative reasoning. *arXiv preprint arXiv:2504.00891*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*, 2025.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning acting and planning in language models. *arXiv preprint arXiv:2310.04406*, 2023.

Hengguang Zhou, Xirui Li, Ruochen Wang, Minhao Cheng, Tianyi Zhou, and Cho-Jui Hsieh. R1-zero's" aha moment" in visual reasoning on a 2b non-sft model. *arXiv preprint arXiv:2503.05132*, 2025.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. GPTSwarm: Language agents as optimizable graphs. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62743–62767. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/zhuge24a.html.

Yuxin Zuo, Kaiyan Zhang, Shang Qu, Li Sheng, Xuekai Zhu, Biqing Qi, Youbang Sun, Ganqu Cui, Ning Ding, and Bowen Zhou. Ttrl: Test-time reinforcement learning, 2025. URL https://arxiv.org/abs/2504.16084.

A Related Works

A.1 Multi-Agent LLM Systems

Previous researches explore multi-agent LLM workflows, focusing on multi-agent debate (Du et al., 2024; Liang et al., 2024; Xu et al., 2023; Yin et al., 2023; Wang et al., 2024; Chen et al., 2024b; Zhang et al., 2025c), communication topology (Chan et al., 2024; Chen et al., 2024c; Zhuge et al., 2024; Qian et al., 2025; Li et al., 2024; Yue et al., 2025b; Zhang et al., 2025b;a), and test-time scaling (junyou li et al., 2024; Wang et al., 2025a; Antoniades et al., 2024; Ye et al., 2024; Yang et al., 2025). These works demonstrate the potential of multi-agent systems to enhance collaborative problem-solving and scalability, effectively managing complex interactions across diverse tasks.

Multi-agent frameworks improve collaborative task-solving effectively (Li et al., 2023; Chen et al., 2024a; Liang et al., 2025). CAMEL (Li et al., 2023) uses a role-playing framework where a user agent decomposes tasks and an assistant agent executes them, guided by an inception prompt. MetaGPT (Hong et al., 2024) simulates the collaboration of a software company by assigning distinct roles for handling complex tasks. However, fixed agent roles restricts the adaptability of multi-agent frameworks. To address this, AutoAgents (Chen et al., 2024a) dynamically generates specialized agents and coordinates them through a central planning module for complex tasks. AutoGen (Wu et al., 2023a) focuses on developing LLM applications through layered and extensible multi-agent design. Similarly, OpenManus (Liang et al., 2025) provides a modular framework with agents, flows, prompts, and tools, adopting a tool-centric ReAct (Yao et al., 2023) paradigm to support plan-then-act decision-making, effectively handling tasks requiring extended reasoning.

Despite the wide variety of existing multi-agent LLM systems, significant performance gains at the emergent level remain elusive (Liu et al., 2019; Chen et al., 2024c). In some tasks, multi-agent frameworks exhibit only marginal gains over single-agent approaches (Pan et al., 2025). These limitations often stem from the inherent constraints of single LLM agents. When handling context, a single agent may fail to follow task or role instructions (Wen et al., 2024; Wang et al., 2025b), or lose focus in long-context scenarios (Zhang et al., 2024b; An et al., 2024). Additionally, the model itself may generate outputs with factual hallucinations or misinterpret contextual cues (Zhang et al., 2023; Jiang et al., 2024). On the other hand, the design of the workflow and inter-agent coordination mechanisms often plays a critical role in MAS failures. Common issues include overly complex system design (Kapoor et al., 2024), disorganized memory management (Han et al., 2024), and failure of verify-refine mechanisms (Huang et al., 2024). Our proposed MARTI framework provides a platform for testing, observing, and mitigating such failures through training.

A.2 Reinforcement Learning for LLMs

Test-time scaling (TTS) is designed to enhance the capabilities of LLMs in handling complex tasks by increasing computational resources at the time of testing. Prior research (Snell et al., 2024; Liu et al., 2025a) indicates that TTS is more efficient than scaling during pre-training (Kaplan et al., 2020); thus, reallocating the same computational resources from pre-training to test-time could yield greater improvements in model performance. Current studies on TTS fall into two categories (Welleck et al., 2024): parallel generation and sequential generation. Parallel generation entails LLMs producing multiple candidate responses (self-consistency (Wang et al., 2022; Chen et al., 2023), best-of-N (Stiennon et al., 2020; Nakano et al., 2021)), decision steps (Monte Carlo Tree Search (Zhou et al., 2023; Xie et al., 2024)), or tokens (Reward-guided Search (Deng & Raffel, 2023; Khanov et al., 2024)) during inference. Subsequently, an aggregation strategy is applied to integrate these candidates, commonly utilizing process reward models (Lightman et al., 2024; Wang et al., 2023a; Zhang et al., 2025d). Concurrently, sequential generation focuses on extending the LLMs' output to include longer responses with reflective and chain-of-thought processes (Wei et al., 2022; Madaan et al., 2023). Although prompting techniques are widely adopted, they are often constrained by the capabilities of the underlying models. Notably, DeepSeek-R1 (Guo et al., 2025) represents a significant advancement in this area, achieving extended reasoning capabilities in pre-trained language models through outcome-based RL, like group relative policy optimization (Shao et al., 2024). Compared to the first approach, which requires intensive process-level supervision (Yuan et al., 2024), the second approach is more scalable due to its reliance on rule-based rewards.

A.3 Multi-Agent Reinforcement Learning

Multi-agent reinforcement learning has emerged as a powerful framework for modeling strategic interactions, guided by game-theoretic principles that shape both learning dynamics and reasoning processes (Yang, 2021; Huh & Mohapatra, 2023). Recent research has focused on addressing its unique challenges such as non-stationarity, credit assignment, and scalability. Wang et al. (2023b) introduce a sequential agent-wise update scheme with off-policy correction, ensuring monotonic improvement and enhancing performance in cooperative tasks. Slumbers et al. (2023) leverage shared policies, centralized training, and natural language communication to enhance performance in text-based environments. Zhang et al. (2024a) shows that LLM-driven agents with Theory of Mind improve perceived coordination in human-AI teams, though bidirectional communication can hinder performance. Wan et al. (2025) separates meta-thinking and reasoning into distinct agents, achieving improved generalization and performance on complex reasoning tasks. Park et al. (2025) jointly trains multiple LLMs via inference-aware rewards to foster effective, transferable collaboration in multi-turn tasks. Lu et al. (2025) proposes a preference-guided multi-agent federated framework that integrates rule-based models and human preference signals in urban autonomous driving scenarios. Thind et al. (2025) translates natural language optimization problems into executable solvers through role-specialized agents.

B Workflows

B.1 Workflow Code

Listing 1: The pseudo-code of the Abstract Workflow.

```python

async def workflow(
    prompt: str,
    label: str,
    agents: List[Dict[str, Any]],
    tool_manager,
    task: str,
    metadata: Optional[Dict] = None,
    **kwargs
) -> Dict[str, Any]:
    # Customized Interactions
    trajectory = [
        {
            "turn_id": 0,
            "agent_index": 0,
            "agent_name":  "agent0",
            "agent_role": "generator",
            "agent_input": "input_example",
            "agent_output": "output_example",
            "metadata": {}
        },
        # Add more turns
    ]
    rewards = [0]
    # Add reward for each turn if exist

    return {
        "prompt": prompt,
        "label": label,
        "trajectory": trajectory,
        "final_reward": rewards[-1]
    }
```

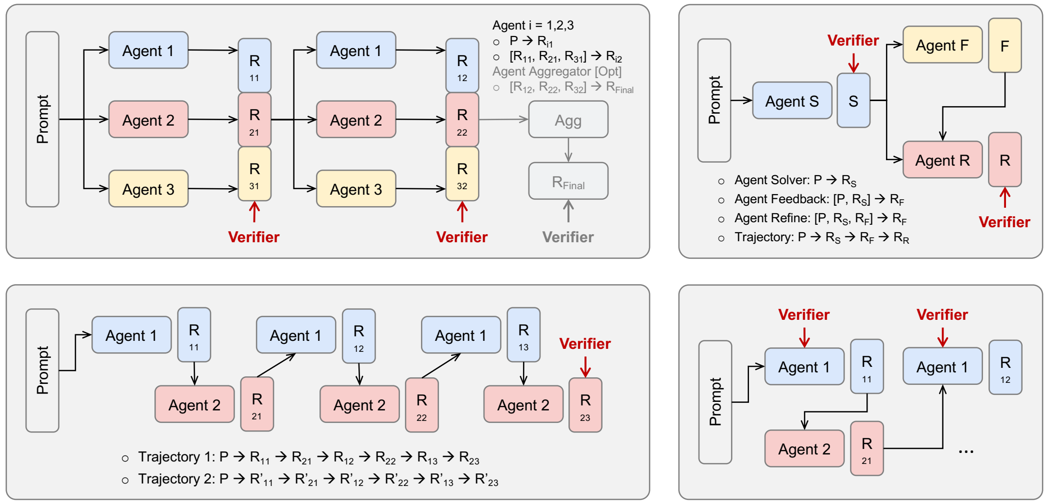
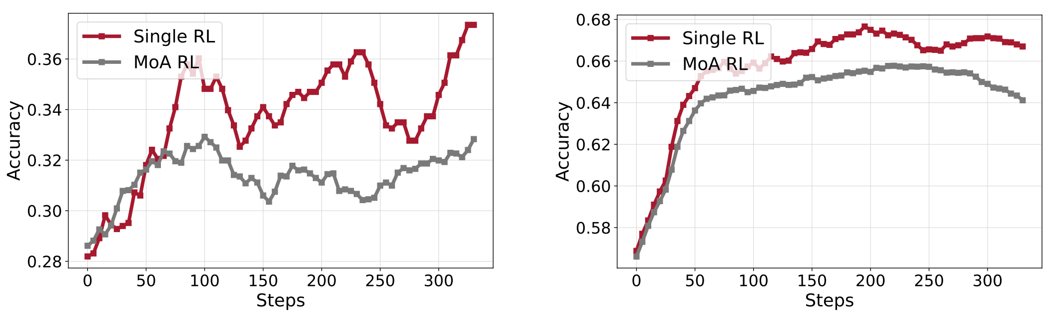


Figure 7: MAS Examples for Typical Multi-Agent Workflows.



(a) AMC (b) MATH-500

Figure 8: Accuracy of MoA (Qwen2.5-3B) on AMC and MATH-500

B.2 Workflow Example

We introduce and compare several workflows in Figure 7, including mixture-of-agents and chain-of-agents. For the chain-of-agents workflow, two typical credit assignment strategies are considered: (1) assigning verifiable rewards at each turn or (2) assigning the final reward at the end, with the final reward distributed across the intermediate turns. These workflows are fully supported in the MARTI framework for further experimentation.

B.3 Code Example

A full example for MathChat with three agents is provided in Figure 2.

C Case Study

C.1 Case 1: Mixture-of-Agents

Experimental Setup. We evaluate a mixture-of-agents approach using the `Qwen2.5-3B` model, trained on Levels 3 through 5 of the MATH-500 training dataset.

Training Dynamics. The model's accuracy results are presented for both AMC and MATH-500 benchmarks in Figures 8a and 8b, respectively. Furthermore, we analyze the complete training dynamics in Figure 9, including training rewards, response length, and total length.

Figure 9: Training Dynamics of MoA (Qwen2.5-3B) with RL on MATH.

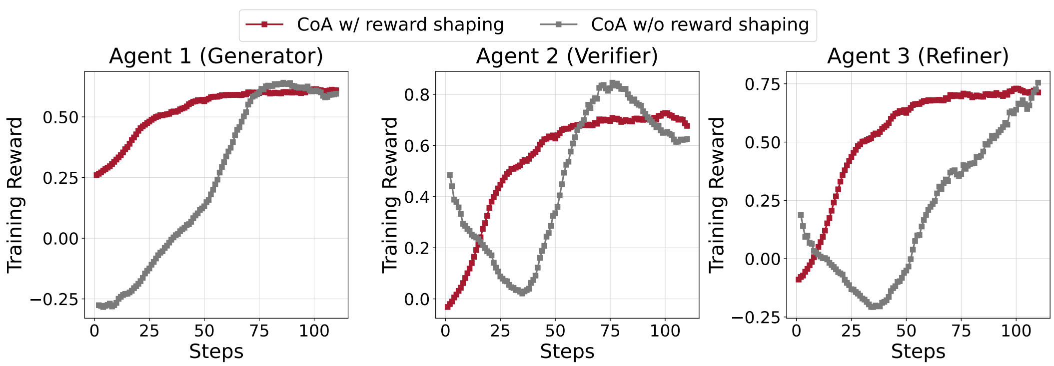


Figure 10: Training Rewards of CoA (Qwen2.5-3B) with RL on MATH.

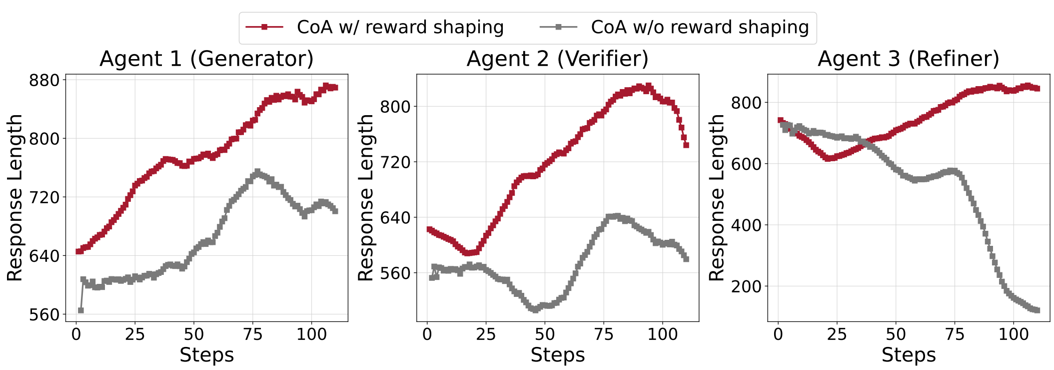


Figure 11: Training Response Length of CoA (Qwen2.5-3B) with RL on MATH.

C.2 Case 2: Chain-of-Agents

Experimental Setup. We investigate chain-of-agents reinforcement learning (RL) using Levels 3–5 of the MATH-500 training set. Our evaluation compares standard RL training with a quality-aware reward shaping variant to assess performance improvements.

Training Dynamics. The training process is characterized by three key metrics:

- Training reward curve in Figure 10.
- Response length dynamics in Figure 11.

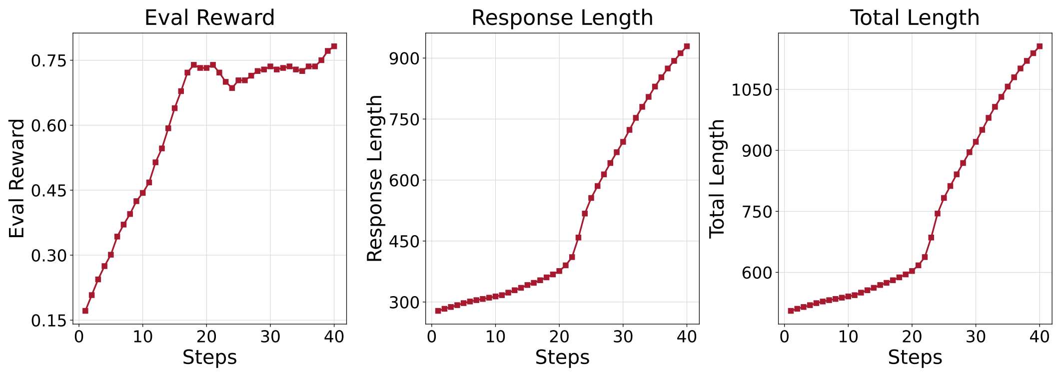


Figure 12: Training Dynamics of Judge-based Two-player Debate.

C.3 Case 3: Judge-based Debate

Experimental Setup. We conduct LLM training using GenRM-generated feedback under a debate setting following (Hogan, 2024). Specifically, we use the round-robin tournament data from (Hogan, 2024) and train a Llama-3.2-1B-Instruct model to debate against a Llama-3.1-8B-Instruct model. The reward is the win rate in the round-robin tournament judged by a Qwen2.5-14B-Instruct model. **Training Dynamics.** We present the evaluation rewards, response length, and total trajectory length during training in Figure 12.

D The Use of Large Language Models

All core research ideas, theoretical derivations, experimental designs, and algorithmic innovations were developed by the authors without LLM assistance. Additionally, all paragraphs in the paper were originally written by humans. LLMs were used solely to fix bugs in the MARTI framework, under human review, and to polish sections of the paper.

Listing 2: The pseudo-code of MathChat workflow.

```python
from typing import Dict, List, Any

async def workflow(
    prompt: str,
    label: str,
    agents: List[Dict[str, Any]],
    tool_manager: Any,
    task: str,
    **kwargs
) -> Dict[str, Any]:
    """
    Orchestrates an asynchronous multi-agent workflow and collects data for training.

    This example defines a three-step interaction:
    1. A 'generator' agent proposes a solution.
    2. A 'coder' agent implements the solution in code, which is then executed.
    3. A 'refiner' agent verifies all outputs to provide a final answer.

    The collected 'trajectory' retains all inputs, outputs, and rewards,
    forming a complete data sample for reinforcement learning.
    """
    # 1. Initialize workflow and identify agents by their predefined roles
    trajectory = []
    generator_agent, coder_agent, refiner_agent = agents[0], agents[1], agents[2]

    # --- Turn 1: Generator proposes a solution ---
    generator_input = f"Problem: {prompt}\nPlease reason step by step..."
    generator_response = await generator_agent["llm"].generate_async.remote(
        generator_input, generator_agent["sampling_params"]
    )
    generator_output = generator_response.outputs[0].text
    trajectory.append({
        "agent_role": "generator", "agent_input": generator_input, "agent_output": generator_output
    })

    # --- Turn 2: Coder writes and executes code based on the generator's solution ---
    coder_input = f"Problem: {prompt}\nSolver Output: {generator_output}\nWrite Python code..."
    coder_response = await coder_agent["llm"].generate_async.remote(
        coder_input, coder_agent["sampling_params"]
    )
    coder_output = coder_response.outputs[0].text

    # Use the tool manager to execute the generated code
    code_to_execute = extract_code(coder_output)
    execution_result, _ = await tool_manager.execute_tool(
        "code_interpreter", {"code": code_to_execute}
    )

    trajectory.append({
        "agent_role": "coder", "agent_input": coder_input, "agent_output": coder_output,
        "metadata": {"tool_output": execution_result}
    })

```

```python
    # --- Turn 3: Refiner verifies all outputs to produce a final answer ---
    refiner_input = (f"Problem: {prompt}\nSolver Output: {generator_output}\n"
                     f"Code Output: {execution_result}\nVerify and provide the final answer...")
    refiner_response = await refiner_agent["llm"].generate_async.remote(
        refiner_input, refiner_agent["sampling_params"]
    )
    refiner_output = refiner_response.outputs[0].text
    trajectory.append({
        "agent_role": "refiner", "agent_input": refiner_input, "agent_output": refiner_output
    })

    # 2. Evaluate the completed trajectory to assign rewards for RL training
    all_outputs = [turn["agent_output"] for turn in trajectory]
    all_rewards = auto_verify(task, all_outputs, [label] * len(all_outputs))

    for turn, reward in zip(trajectory, all_rewards):
        turn["agent_reward"] = reward

    # 3. Return the structured data sample in the required format
    return {
        "prompt": prompt,
        "label": label,
        "trajectory": trajectory,
        "final_reward": all_rewards[-1]
    }
```

Table 6: Average number of output tokens per instance on Qwen2.5-3B across tasks and workflows. Multi-agent workflows operate under a comparable token budget to Majority@4, and can even be more efficient.

Output Tokens	AIME	AMC	MATH500	Avg
Single-Agent (Avg@4)	4532	3706	2322	3520
MAD 2×2	2698	4268	2698	3221
MoA 3×1	3083	2146	1518	2249

E Compute Accounting and Inference Budget

Unified Definition of Inference Budget. To ensure fair comparisons between single- and multi-agent workflows, we standardize the *inference budget* in terms of the number of model rollouts under a fixed sampling configuration (temperature, top-p, maximum length, etc.). Concretely, a single-agent Majority@4 evaluation and a MAD 2×2 session both generate four trajectories and therefore consume an equivalent rollout budget. Similarly, MoA 3×1 produces three trajectories plus one final aggregation step, which in practice is comparable to Majority@4 in terms of compute.

We also report untrained multi-agent baselines (with the backbone kept frozen) to disentangle the benefit of test-time compute from that of learned collaboration. These baselines only change the interaction topology (e.g., voting, MAD, MoA) while keeping the rollout budget fixed, and thus highlight that the gains of MARTI mainly come from reinforcement learning on collaborative behaviours rather than simply sampling more trajectories.

Token-Level Statistics Across Workflows. In addition to rollout counts, we measure the average number of output tokens per instance for different workflows on Qwen2.5-3B. Table 6 summarizes the results. MAD 2×2 consumes a similar number of tokens to the single-agent Majority@4 baseline (within roughly 10% on average), while MoA 3×1 is even more token-efficient. This confirms that the accuracy gains reported in the main paper do not arise from substantially increased generation cost.